Siebel Analytics Applications Installation and Administration Guide

Version 7.8.2

June 2005

Siebel Systems, Inc., 2207 Bridgepointe Parkway, San Mateo, CA 94404

Contents

Chapter 5: Installing and Configuring Siebel Analytics Applications on Windows

Chapter 9: DAC Functional Reference

Chapter 10:Customizing the Siebel Data Warehouse

Chapter 11:Using Siebel Data Warehouse Exception Reports

1 What's New in This Release

What's New in Siebel Analytics Applications Installation and Administration Guide, Version 7.8.2

Table 1 lists changes described in this version of the documentation to support Release 7.8.2 of the software.

Table 1. What's New in Siebel Analytics Applications Installation and Administration Guide, Version 7.8.2

Topic	Description
Book title change	The name of the *Siebel Data Warehouse Installation and Administration Guide* changed to *Siebel Analytics Applications Installation and Administration Guide*.
Overview of Analytics Product Licensing on page 15	Added new chapter that describes the licensing options available for Siebel Analytics Platform and Siebel Analytics Applications.
Running the Siebel Analytics Applications Installation Wizard (Windows) on page 59	Added new instructions for installing Siebel Analytics Applications.
Running the Informatica PowerCenter Installation Wizard (Windows) on page 60	Added new instructions for installing Informatica PowerCenter 7.1.2.
Setting Up the Informatica Server on page 78	Revised procedure for new version of Informatica.
Setting Up the Informatica Repository Server on page 82	Revised procedure for new version of Informatica.
Pointing to the Informatica Repository on page 87	Revised procedure for new version of Informatica.
Updating Siebel Transactional Database Schema Definitions on page 98	Updated SIF file information.
Teradata-Specific Installation and Configuration Checklist (Windows) on page 102	Added new checklist for Teradata installations.
Installing and Configuring the Informatica Repository Server on UNIX on page 111	Revised procedure for new version of Informatica.
Installing and Configuring the Informatica Server on UNIX on page 113	Revised procedure for new version of Informatica.
Updating the Informatica License Files on UNIX on page 119	Added procedure required for new version of Informatica.

Table 1. What's New in Siebel Analytics Applications Installation and Administration Guide, Version 7.8.2

Topic	Description
Teradata-Specific Installation and Configuration Checklist for UNIX on page 128	Added new checklist for Teradata installations.
Upgrading Siebel Analytics Applications and Data Warehouse on page 129	Revised information on upgrading to Siebel Analytics Applications version 7.8.2.
Siebel Analytics Bridge Applications on page 255	Revised information on the Siebel Analytics Bridge applications.

The features listed in Table 2 were introduced in Version 7.8.1, the Limited Distribution release of the software.

Table 2. What's New in Siebel Analytics Applications Installation and Administration Guide, Version 7.8.1

Topic	Description
Setting Up DAC System Properties on page 67	Added description of new DAC system properties: ■ Verify and Create Non-Existing Indices ■ Drop and Create Change Capture Views Always
Creating Transactional and Data Warehouse Database Connections on page 73	Added new topic.
Comparing and Merging DAC Repositories on page 163	Added new topic.
Replacing an Informatica Workflow with a Custom SQL File on page 166	Added new topic.
Determining the Informatica Server Maximum Sessions Parameter Setting on page 166	Added new topic.
Determining the Number of Transactional and Data Warehouse Database Connections on page 167	Added new topic.
Running Two DAC Servers on the Same Machine on page 168	Added new topic.
Tracking Deleted Records on page 169	Added new topic.
Customizing Index and Analyze Table Syntaxes on page 171	Added new topic.
About the DAC Indices Tab on page 181	Added description of new Always Drop & Create column on the Indices tab. Added description of new Index Override column on the Databases subtab of the Indices tab.
Siebel Data Warehouse for Life Sciences Data Considerations on page 269	Extensively revised chapter.

2 Overview of Analytics Product Licensing

This chapter provides an overview of Siebel Analytics product licensing. It includes the following topic: "Analytics Product Licensing" on page 15.

Analytics Product Licensing

Siebel Analytics has the following licensed products.

Siebel Analytics Platform

The Siebel Analytics platform product is a set of servers and tools used to build Analytics applications. Using the Siebel Analytics platform, you can develop your own integrated business intelligence application solution. The Siebel Analytics platform is described in Table 3.

Table 3. Siebel Analytics Platform Components

Siebel Analytics Platform Components
Siebel Analytics Server
Siebel Analytics Web Server
Siebel Analytics Scheduler Server
Siebel Data Warehouse Administration Console (Client and Server)
Related component programs: ■ Siebel Analytics Administration Tool ■ Siebel Analytics Catalog Manager ■ Siebel Analytics ODBC Client ■ Siebel Analytics Client Tools

The following documentation is applicable to the Siebel Analytics Platform:

- ***Siebel Analytics Installation and Configuration Guide.*** This guide provides information about the installation and configuration of the Siebel Analytics platform.
- ***Siebel Analytics Web Administration Guide.*** This guide provides post-installation configuration and administration procedures for Siebel Answers, Siebel Delivers, Siebel Intelligence Dashboards, the Analytics Web Catalog, and the Analytics Web Server.
- ***Siebel Analytics Server Administration Guide.*** This guide provides information about administration of the Analytics Server, Analytics repository files, and Mobile Analytics.

Siebel Analytics Applications

Siebel Analytics Applications are prebuilt business intelligence solutions. Siebel Analytics applications are of two types:

- Siebel Analytics Applications (CRM)
- Siebel Analytics Applications (Enterprise)

Siebel Analytics Applications (CRM)

These Analytics applications support Siebel CRM applications, such as Siebel Sales, Siebel Service, and Siebel Marketing. If you already own a Siebel CRM application, you can purchase Siebel Analytics platform and Siebel Analytics applications to work with the Siebel CRM application.

The Analytics CRM applications consist of the components shown in Table 4.

Table 4. Siebel Analytics Applications (CRM) Components

Component	Description
DAC metadata repository files	This content includes repository objects such as tables, subject areas, execution plans, and tasks, and is contained in XML files.
Embedded Informatica ETL Tool	This is a third-party application that performs the extract, transform, and load operations for the Data Warehouse.
Prebuilt Informatica content	This content includes Extract-Transform-Load (ETL) repository objects, such as mappings, sessions, and workflows, and is contained in the Informatica repository file (Siebel_DW_Rep.rep).
Prebuilt metadata content	This metadata content is contained in the Siebel Analytics repository file (SiebelAnalytics.rpd).
Prebuilt reports and dashboard content	This content is contained in the Siebel Analytics Web Catalog file (SiebelAnalytics.webcat).
Prebuilt Siebel Data Warehouse	The Siebel Relationship Management Warehouse (RMW) is also referred to as the Siebel Data Warehouse.

NOTE: The Siebel Data Warehouse is also referred to as the Relationship Management Warehouse (RMW).

The *Siebel Analytics Applications Installation and Administration Guide* provides installation and configuration information about the Analytics Applications components referred to above. It also covers the configuration and administration of the Data Warehouse Administration Console (DAC).

Siebel Analytics Applications (Enterprise)

Siebel Analytics Applications (Enterprise) are analytics applications that provide complete support for Enterprise data, including financial, supply chain, and workforce sources. These enterprise applications typically source from non-Siebel data sources (such as SAP, Oracle, or PeopleSoft). They are separate products and licensed separately from the Siebel Analytics Applications (CRM). Siebel Analytics Enterprise Applications do not interoperate directly with Siebel Business Applications, but they can be integrated with Siebel CRM solutions.

Siebel Enterprise-related Analytics Applications components are shown in table Table 5.

Table 5. Siebel Analytics Applications (Enterprise) Components

Component	Description
Embedded Informatica ETL Tool	This is a third-party application that performs the extract, transform, and load operations for the Data Warehouse.
Prebuilt Informatica content	This content includes Extract-Transform-Load (ETL) repository objects, such as mappings, sessions, and workflows. This content is sourced from SAP R/3, Oracle, PeopleSoft applications, and other data sources. This content is contained in the Informatica repository file (Shell.rep).
Prebuilt metadata content	This metadata content is contained in the Siebel Analytics repository file (SiebelAnalytics.rpd).
Prebuilt reports and dashboard content	This content is contained in the Siebel Analytics Web Catalog file (SiebelAnalytics.webcat).
Prebuilt Siebel Customer-Centric Enterprise Warehouse	**NOTE:** This data warehouse is not to be confused with the Siebel Relationship Management Warehouse (RMW) or Siebel Data Warehouse.

The *Siebel Customer-Centric Enterprise Warehouse Installation and Configuration Guide* provides installation and configuration information about the Siebel Analytics Applications (Enterprise) product.

3 Overview of Siebel Data Warehouse

This chapter provides an overview of the Siebel Data Warehouse. It includes the following sections:

Siebel Data Warehouse Overview

The Siebel Data Warehouse is a unified data repository for all customer-centric data. The purpose of the Siebel Data Warehouse is to support the analytical requirements of Siebel Business Applications.

NOTE: The Siebel Data Warehouse is also referred to as the Siebel Relationship Management Warehouse (RMW).

The Siebel Data Warehouse includes the following:

- A data integration engine that combines data from the Siebel transactional database and other data sources to build a data warehouse.
- An open architecture to allow organizations to use third-party analytical tools in conjunction with the Siebel Data Warehouse using the Siebel Analytics Server.
- Optional prebuilt data extractors to incorporate data from external applications into the Siebel Data Warehouse (licensed separately).
- A set of ETL (extract-transform-load) processes that takes data from the Siebel Business Applications transactional data (OLTP) and creates the Siebel Data Warehouse tables.
- The Data Warehouse Application Console (DAC), a centralized console for the set up, configuration, administration, loading, and monitoring of the Siebel Data Warehouse.

Figure 1 provides an overview of the Siebel Data Warehouse.

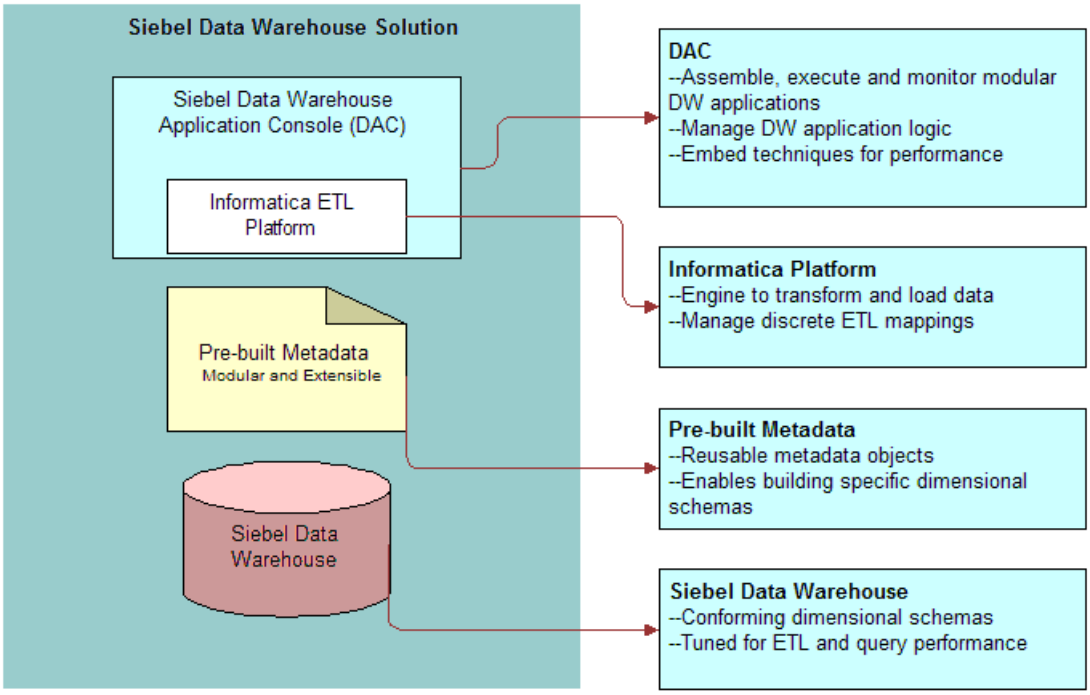


Figure 1. Data Warehouse Overview

Siebel Data Warehouse Architecture

High-level analytical queries, like those commonly used in Siebel Analytics, scan and analyze large volumes of data using complex formulas. This process can take a long time when querying a transactional database, which impacts overall system performance.

For this reason, the Siebel Data Warehouse was constructed using dimensional modeling techniques to allow for fast access to information required for decision making. The Siebel Data Warehouse derives its data from Siebel operational applications, and uses Informatica's data integration technology to extract, transform, and load data from the Siebel transactional database into the Siebel Data Warehouse.

Figure 2 illustrates the Siebel Data Warehouse architecture.

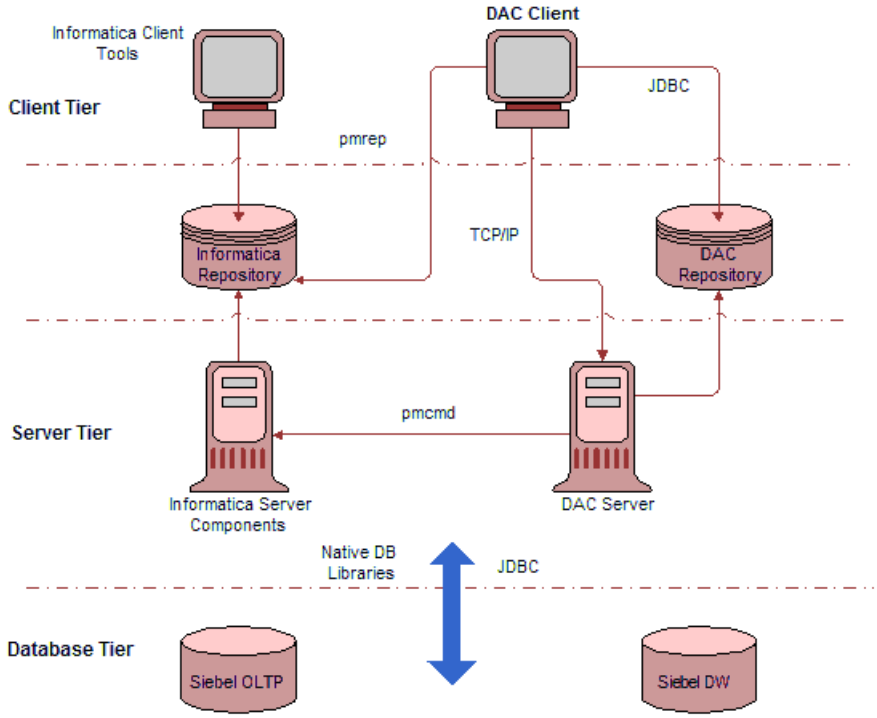


Figure 2. Siebel Data Warehouse Architecture

Siebel Data Warehouse Architecture Components

The Siebel Data Warehouse architecture comprises the following components:

- **DAC client.** A command and control interface for the data warehouse to allow for set up, configuration, administration, and monitoring of data warehouse processes.
- **DAC server.** Executes the instructions from the DAC client. The DAC server manages data warehouse processes, including scheduling, loading of the ETL, and configuring the subject areas to be loaded. It dynamically adjusts its actions based on information in the DAC repository. Depending on your business needs, you might incrementally refresh the Siebel Data Warehouse once a day, once a week, once a month, or on another similar schedule.
- **DAC repository.** Stores the metadata (semantics of the Siebel Data Warehouse) that represents the data warehouse processes.
- **Informatica Server.** When the server is loading or refreshing the Siebel Data Warehouse, it is recommended that the server be dedicated to that activity. However, when the ETL server is not loading or refreshing the Siebel Data Warehouse, you can use it for other purposes.
- **Informatica Repository Server.** Manages the Informatica repository.
- **Informatica Repository.** Stores the metadata related to Informatica workflows.

- **Informatica client utilities.** Tools that allow you to create and manage the Informatica repository.

About the Data Warehouse Application Console

The Data Warehouse Application Console (DAC) provides a framework for the entire life cycle of data warehouse implementations. It allows you to create, configure, and execute modular data warehouse applications in a parallel, high-performing environment. For information about the DAC process life cycle, see "About the DAC Process Life Cycle" on page 27.

The DAC complements the Informatica ETL platform. It provides *application-specific* capabilities that are not pre-built into ETL platforms. For example, ETL platforms are not aware of the semantics of the subject areas being populated in the data warehouse nor the method in which they are populated. The DAC provides these application capabilities at a layer of abstraction above the ETL execution platform, as illustrated in Figure 3.

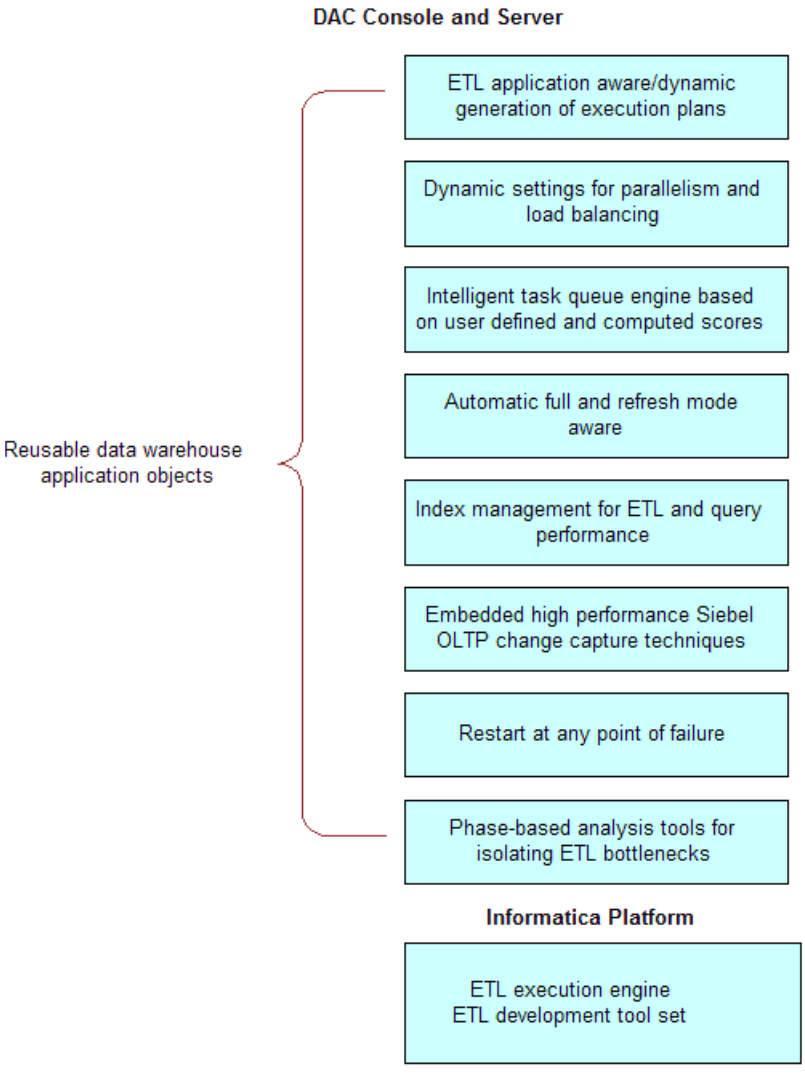


Figure 3. DAC Application-Specific Capabilities

Important DAC Features

Important DAC features allow you to do the following:

Minimize install, setup, and configuration time

- Create physical data model in the data warehouse
- Accelerate Informatica configuration
- Set language, currency, and other settings
- Develop data warehouse applications

Manage metadata driven dependencies and relationships

- Generate custom ETL execution plans
- Automate change capture for the Siebel transactional database
- Capture deleted records
- Assist in index management
- Perform metadata validation checks
- Perform dry-run development and testing
- Provide in-context documentation

Provide reporting and monitoring to isolate bottlenecks

- Perform error monitoring and email alerting
- Perform structured ETL analysis and reporting

Utilize performance execution techniques

- Automate full and refresh mode optimization rules
- Set the level of Informatica session concurrency
- Load balance across multiple Informatica servers
- Restart from point of failure

- Queue execution tasks for performance (See Figure 4.)

  The DAC manages the task execution queue based on metadata driven priorities and scores computed at runtime. This combination allows for flexible and optimized execution.

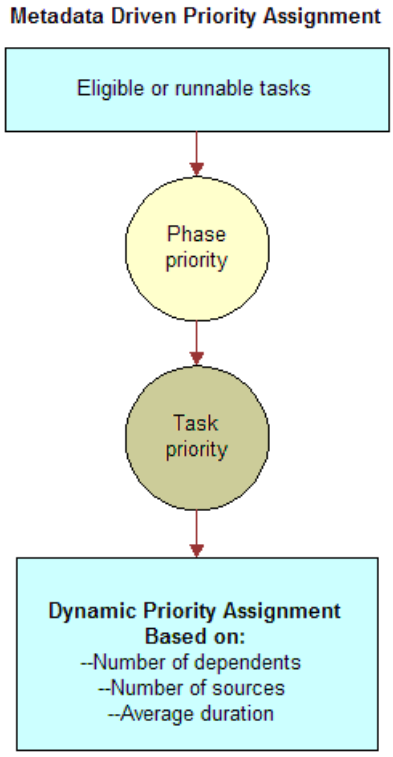


Figure 4. Task Execution Queue

About DAC Repository Objects

The DAC repository stores application objects in a hierarchical framework that defines a data warehouse application.

A data warehouse application comprises the following repository objects:

- **Tables.** Physical database tables defined in the database schema. Can be transactional database tables or data warehouse tables. Table types can be fact, dimension, hierarchy, aggregate, and so on.
- **Table group.** A logical grouping of tables that defines a logical relationship between its component tables. The table group definition is based on a central or a main table that drives the relationship between itself and the other tables in the group. For example, table groups can be star schemas based on a central fact and its dimensions and aggregates.

- **Subject area.** A logical grouping of table groups that is related to a particular subject or application context. The subject area derives its component tables indirectly from the table group. Subject Area definitions are used for scheduling. A subject area or set of subject areas can be scheduled for full or incremental loads on a single time schedule.
- **Execution plan.** A data transformation plan defined on subject areas that need to be transformed at certain frequencies of time. An execution plan is defined based on business requirements for when the data warehouse needs to be loaded. Execution plans are scheduled in sequence, not in parallel.
- **Task.** A Task can be related to data transformation or database objects. Tasks typically operate on a set of database tables, but they can be independent of the database, for example, creating a file or executing an OS command. Tasks can be of different types, such as extract, load, or batch script, and so on, and can be classified as pre-ETL or post-ETL.
- **Dependency Rule.** A rule that determines the order in which tasks are executed. Dependency rules are defined to determine the sequence of when tasks can run, ensuring that data consistency is maintained for the warehouse. Dependency rules are defined at the task level.
- **Schedule.** A schedule specifies when and how often an execution plan runs. An execution plan can be scheduled for different frequencies or recurrences by defining multiple schedules.

Figure 5 shows the hierarchical relationship among the repository objects.

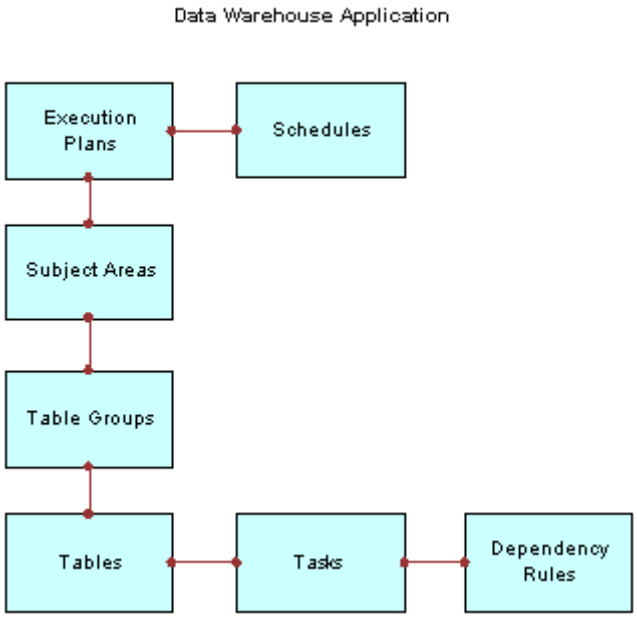


Figure 5. DAC Object Hierarchy

About the DAC Process Life Cycle

The DAC is used by different user groups to design, execute, monitor, and diagnose data warehouse applications. These phases together make up the DAC process life cycle, as shown in Figure 6.

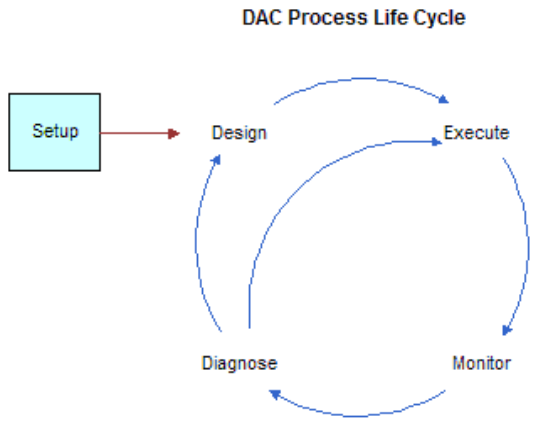


Figure 6. DAC Process Life Cycle

The phases of the process and the actions associated with them are as follows:

- Setup (Data warehouse developer)
  - Set up database connections
  - Set up application and database servers
  - Set up email recipients
- Design (Data warehouse developer)
  - Design data warehouse applications
  - Define application objects
- Execute (Data warehouse/ETL administrator)
  - Define scheduling parameters to execute data warehouse applications
  - Deploy data warehouse applications
  - Access run-time controls to restart or stop currently running schedules
- Monitor (Data warehouse/ETL administrator)
  - Monitor run-time execution of data warehouse applications
  - Monitor users, DAC repository, and application maintenance jobs
- Diagnose (Data warehouse/ETL administrator)
  - View the schedule diagnostics at run time or offline through log files
  - Run Exception Report execution plans

4 Preinstallation Considerations for Siebel Data Warehouse

This chapter provides information about preparing to deploy the Siebel Data Warehouse. You should review this information before you begin the installation process.

NOTE: To install the server components, the computers need to meet the conditions specified in *Siebel System Requirements and Supported Platforms, Siebel Analytics*.

This chapter contains the following topics:

Siebel Data Warehouse Deployment Configuration

Figure 7 shows the recommended Siebel Data Warehouse deployment configuration.

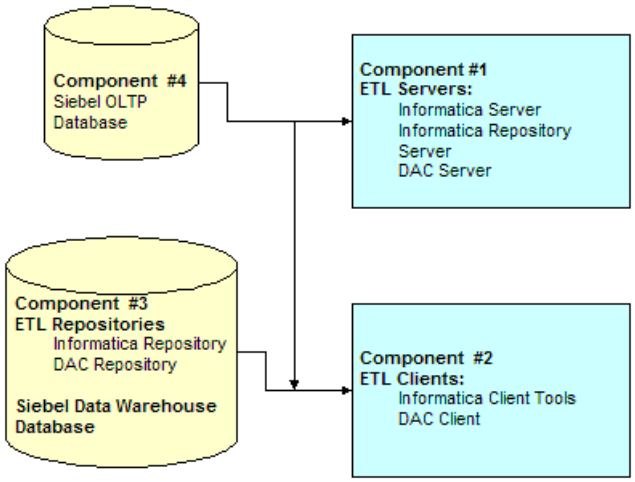


Figure 7. Siebel Data Warehouse Deployment Configuration

- Component 1 hosts all the ETL servers, that is, the Informatica Server, Informatica Repository Server, and the DAC server.

  NOTE: You can install the Informatica Server on other machines as well to increase performance. The other ETL servers can also be hosted on other machines.

- Component 2 hosts all the ETL clients, that is, the Informatica client tools and the DAC client.
- Components 3 and 4 are database instances that can be hosted on one or more machines. The hardware requirements are entirely dependent on your usage and performance requirements. It is highly recommended that each of these components be defined in their own database instance to allow for instantiating parameters that are optimized for each component's usage.

NOTE: For more detailed information about system requirements, see *Siebel System Requirements and Supported Platforms, Siebel Analytics*.

Operating System, Driver, and Connectivity Requirements for Siebel Data Warehouse

Table 6 provides the operating system, driver, and connectivity software requirements for the Siebel Data Warehouse components.

Table 6. OS, Driver and Connectivity Requirements for Siebel Data Warehouse

Component	Operating System	Software	Connectivity and Driver
1 ETL Servers	■ Solaris ■ AIX ■ HP ■ Windows	■ Informatica Server 6.2.1 (6.2.2 for Siebel Data Warehouse version 7.7.1.4 and higher) ■ Informatica Repository Server 6.2.1 (6.2.2 for Siebel Data Warehouse version 7.7.1.4 and higher) ■ DAC Server	■ Java SDK 1.4.2 ■ JDBC drivers ■ Javamail
2 ETL Clients	■ Windows	■ Informatica client tools ■ DAC client	■ Java SDK 1.4.2 ■ JDBC drivers ■ ODBC drivers
3 (A) Siebel DW	■ Solaris ■ AIX ■ HP ■ Windows ■ OS/390	Database software with Siebel Data Warehouse	Not applicable
3 (B) ETL Repositories	■ Solaris ■ AIX ■ HP ■ Windows	Database software with ETL repositories	Not applicable

General Guidelines for Setting Up Siebel Data Warehouse

The Siebel Data Warehouse is a database that contains dimensional schemas. Although it is technically possible to put the Siebel Data Warehouse in the same database as the Siebel transactional database, it is not recommended for performance reasons. The Siebel transactional database is structured as an online transaction processing (OLTP) database, whereas the Siebel Data Warehouse is structured as an online analytical processing database. Each is optimized for its own purpose. The reasons for not combining the two databases are:

- The analytical queries interfere with normal use of the transactional database, which is entering and managing individual transactions.
- The data in a transactional database is normalized for update efficiency. Transactional queries join several normalized tables and will be slow (as opposed to prejoined, denormalized analytical tables).
- Historical data cannot be purged from a transactional database, even if not required for current transaction processing, because you need it for analysis. (By contrast, the analytical database is the warehouse for historical as well as current data.) This causes the transactional database to further slow down.
- Transactional databases are tuned for one specific application, and it is not productive to use these separate transactional databases for analytical queries that usually span more than one functional application.
- The analytical database can be specifically tuned for the analytical queries and Extract-Transform-Load (ETL) processing. These are quite different from transactional database requirements.
- On the transactional database, you should place the S_ETL tables in a separate tablespace. These ETL tables are used by the Siebel Data Warehouse and should not be part of the routine backup processes.

  A complete listing of these tables is available in *Siebel Data Warehouse Data Model Reference.*

The Informatica repository stores all of the Informatica object definitions for the ETL mappings that populate the Siebel Data Warehouse. It is a series of repository tables that are stored in a database, which can be transactional, analytical, or a separate database.

The Siebel Data Warehouse works with relational database management systems. In addition to the general requirements, there are additional database management systems (DBMS)-specific requirements depending on the DBMS you are using.

The following general guidelines will help you set up the data warehouse physical database for performance and growth:

- At a minimum, separate the data and index tablespaces. Create more tablespaces to separate heavily used tables and their indices.
- Use the 32K block/page size for tablespaces, because it provides good overall performance and also does not impose low limits to the maximum size to which the tablespace can grow, as compared to 4K, 8K,16K sizes.

- If you are using multiple disk storage systems, stripe the tablespace containers and files across as many disks as possible.
- Raw devices for tablespaces provide better performance as compared to cooked file systems.
- RAID-5 is known to give a good balance of performance and availability.
- Size the bufferpools based on content and size (number of tables and their sizes) of tablespaces.
- Allocate about 75 percent of the total available server memory to the database, assuming no other application is running on the same server.

During the Siebel Data Warehouse configuration process, when you create the data warehouse tables using the procedure "Creating Data Warehouse Tables" on page 76, you can create tables in one tablespace and indices in another tablespace. However, for performance reasons, it is recommended that you create tablespaces as described in Table 7.

Table 7. Recommended Tablespace Configuration

Tablespace Name	List of Tables
DIM_STG	W_*DS
FACT_STG	W_*FS
DIM	W_*D and W_*MD
FACT	W_*F
AGG	W_*A
OTHER	Remaining W* tables
DIM_INDX	Indices of W_*D tables
FACT_INDX	Indices of W_*F tables
OTHER_INDX	Remaining W* tables

IBM DB2 UDB-Specific Database Guidelines for Siebel Data Warehouse

Table 8 provides guidelines for parameter settings for DB2 relational database management system (RDBMS) usage. Use these guidelines as a starting point. You will need to make changes based on your specific database sizes, data shape, server size (CPU and memory), and type of storage. The database administrator should make changes to the settings based on performance monitoring and tuning considerations.

Table 8. Recommended DB2 Parameter Settings

Parameter	DB2 UDB V7	DB2 UDB V8	Notes
SHEAPTHRES	400000	400000	
ASLHEAPSZ	15	15	
RQRIOBLK	65535	65535	
QUERY_HEAP_SZ	16384	16384	
JAVA_HEAP_SZ	2048	2048	
MAXAGENTS	400	400	
NUM_INITAGENTS	10	10	
NUM_POOLAGENTS	200	200	
INTRA_PARALLEL	YES	YES	
FCM_NUM_BUFFERS	12288	12288	
SHEAPTHRES_SHR	N/A	=SHEAPTHRES	
DBHEAP	16384	16384	
CATALOGCACHE_SZ	5558	5558	
LOGBUFSZ	2048	2048	
UTIL_HEAP_SZ	10000	10000	
NUM_ESTORE_SEGS	16	NIL	Estore is not needed in DB2 V8 64-bit because the V7 limit of 1.75 GB addressable memory has been lifted.
ESTORE_SEG_SZ	65536	NIL	
LOCKLIST	25000	25000	
APP_CTL_HEAP_SZ	5000	5000	
SORTHEAP	40000	40000	
STMTHEAP	40960	40960	

Table 8. Recommended DB2 Parameter Settings

Parameter	DB2 UDB V7	DB2 UDB V8	Notes
APPLHEAPSZ	2560	2560	
PCKCACHESZ	2560	2560	
STAT_HEAP_SZ	20000	20000	
DLCHKTIME	10000	10000	
MAXLOCKS	50	50	
LOCKTIMEOUT	1200	1200	
MAXAPPLS	500	500	
AVG_APPLS	10	10	
MAXFILOP	500	500	
GROUPHEAP_RATIO	N/A	70	New in V8
APPGROUP_MEM_SZ	N/A	30000	New in V8
DATABASE_MEMORY	N/A	AUTOMATIC	New in V8

IBM DB2 UDB zOS and OS/390-Specific Database Guidelines for Siebel Data Warehouse

The following requirements apply to IBM DB2 RDBMS usage for zOS and OS/390:

- The Siebel Analytics applications communicate with IBM DB2 UDB for z/OS and OS/390 (running on zSeries servers) through IBM DB2 Connect middleware.

  The following editions of DB2 Connect are supported:

  - **DB2 Connect Enterprise Edition (EE).** This edition is installed on a midtier server such as an Informatica Server/Client, DAC, and Siebel Analytics.
  - **DB2 Connect Unlimited Edition (UE).** This edition provides the functionality of DB2 Connect Enterprise Edition but is priced differently.

- The ODBC driver for all connections must use the IBM DB2 ODBC Driver.
- Make the appropriate connections using the DB2 Client Configuration Assistant.

- Use the variable settings shown in Table 9.

Table 9. Variable Settings for IBM DB2 UDB zOS and OS/390 Databases

Parameter	Recommended Setting	Notes
IDTHTOIN	1800	
CDSSRDEF	Any	
STARJOIN	1	This setting indicates that star join is enabled. The one table with the largest cardinality is the fact table. However, if there is more than one table with this cardinality, star join is not enabled.

Oracle-Specific Database Guidelines for Siebel Data Warehouse

Table 10 provides parameter guidelines based on the rule-based optimizer for Oracle 8i and the cost-based Optimizer for Oracle 9i. Use these guidelines as a starting point. You will need to make changes based on your specific database sizes, data shape, server size (CPU and memory), and type of storage. The database administrator should make changes to the settings based on performance monitoring and tuning considerations.

Table 10. Recommended Variable Settings for Oracle Databases

Parameter	Oracle 8i	Oracle 9i	Notes
Always_anti_join	HASH	N/A	
Cursor_space_for_time	TRUE	TRUE	Use only if you do not have many concurrent connections.
Db_block_lru_latches	Set to number of CPUs	N/A	
Db_block_size	32 k	32 k	
Db_cache_size		8.92 MB	
Db_file_direct_io_count	64	N/A	
Db_file_multiblock_read_count	32	32	
DB_files		100	
db_writer_processes	Number of CPUs	Number of CPUs	
Default Pctincrease	0	0	
Dml_locks		1000	

Table 10. Recommended Variable Settings for Oracle Databases

Parameter	Oracle 8i	Oracle 9i	Notes
Hash_area_size	20 MB	N/A	Setting WORKAREA_SIZE_POLICY = AUTO and PGA_AGGREGATE_TARGET to about one-third of the available memory takes care of HASH and SORT areas in Oracle 9i.
Hash_multiblock_io_count	0.5 * (db_file_multiblock_Read_count)	N/A	Make sure that you are *not* using MTS.
Initial Extent	20 MB	20 MB	
Log_buffer	16 MB	16 MB	
Log_checkpoint_interval		100000	
Next Extent	20 MB	20 MB	
Open_cursors	2000	2000	
Optimizer_index_caching	Unset		Use Unset as the value for this parameter to avoid nested loop index joins (as favored by the optimizer). Unset is not an allowable value for Oracle 9i. For Oracle 9i, the allowable values are in the range 1 to 100.

Table 10. Recommended Variable Settings for Oracle Databases

Parameter	Oracle 8i	Oracle 9i	Notes
Optimizer_index_cost_adjustment	Choose appropriate value based on data shape and performance monitoring	Choose appropriate value based on data shape and performance monitoring	This parameter can be used to tune the performance of a system in cases in which the optimizer may be choosing too few or too many index access paths. For example, setting it to 50 percent will make the index access path look half as expensive as normal. The legal range of values for this parameter is 1 to 10000 percent. The default for this parameter is 100 percent, which makes the optimizer cost index access paths at the regular cost. Also see "Additional Suggestions for Optimizing Oracle Performance in Siebel Data Warehouse" on page 40.
Optimizer_max_permutations	2000	2000	May be applicable to other schemas in the database. This means longer parsing times for the access paths, but once these are parsed, they should remain in the shared_pool, provided they do not get aged out. See the notes for Cursor_space_for_time.
Optimizer_mode	ALL_ROWS	CHOOSE	
Pga_aggregate_target	N/A	About one-third of available memory	
Processes		510	

Table 10. Recommended Variable Settings for Oracle Databases

Parameter	Oracle 8i	Oracle 9i	Notes
Query_rewrite_integrity	USE_STALE	TRUSTED	USE_STALE is not an allowable value for Oracle 9i. For Oracle 9i, the allowable values are: ■ STALE_TOLERATED ■ ENFORCED ■ TRUSTED
Query_rewrite_enabled		True	
Rollback_segments	■ Initial Extent 50 MB ■ Next Extent 50 MB ■ Optimum Extent 50 MB	N/A	For Oracle 9i, use undo_management = AUTO undo_tablespace = UNDO undo_retention = 60000
SGA_max_size		1024 MB	Allocate one-third to one-half of available memory and adjust as needed based on performance monitoring.
Shared_pool_size	150 MB	512 MB	
Shared_pool_reserved_size		200 MB	
Sort_area_retained_size	10 MB	N/A	Setting WORKAREA_SIZE_POLICY = AUTO and PGA_AGGREGATE_TARGET to about one-third of the available memory takes care of HASH and SORT areas in Oracle 9i.
Sort_area_size	10 MB	N/A	Setting WORKAREA_SIZE_POLICY = AUTO and PGA_AGGREGATE_TARGET to about one-third of the available memory takes care of HASH and SORT areas in Oracle 9i.
Sort_multiblock_read_count	4	N/A	
Star_transformation_enabled	TRUE	TRUE	

Table 10. Recommended Variable Settings for Oracle Databases

Parameter	Oracle 8i	Oracle 9i	Notes
Statistics_level		Normal	
Undo_management	N/A	Auto	
Undo_retention	N/A	10000	
Undo_tablespace	N/A	Undo	
Workarea_size_policy	N/A	Auto	

Additional Suggestions for Optimizing Oracle Performance in Siebel Data Warehouse

- Siebel eBusiness Applications under Oracle support only binary sorting. If you are running an Oracle client, do one of the following:
  - Set the NLS_SORT parameter to BINARY.
  - Choose a NLS_LANG setting that includes binary.

  These settings are required for adequate performance from the dedicated Web client.
- Make sure that cost-based optimization is enabled in the Oracle development, test, and production databases and that statistics are kept up to date. Otherwise, the rule-based optimizer may be used.
- Create foreign keys in the Oracle database, but configure Oracle to not enforce the foreign key relationship. The existence of foreign keys will allow Oracle to better optimize certain queries. By turning off enforcement, the database load should not be negatively affected.
  - In addition, you may also configure the Oracle star-join transformation. This requires non-enforced foreign keys in Oracle and the creation of necessary bitmap indices. This task is optional. It may not be necessary, as ongoing tuning may reach the desired performance goals.
- Analyze application for occurrences of highly skewed data that is indexed. Create histogram statistics for these indices to enable the optimizer to better perform queries.
- To increase data throughput between Siebel Analytics Server and Oracle, change SDU and TDU settings in listener.ora. The default is 2 KB and can be increased to 8 KB.
- On the server side, edit the listener.ora file. Under the particular SID_LIST entry, modify SID_DESC as follows:

```
SID_LIST_LISTENER =

   SID_LIST =

      SID_DESC = (SDU=16384)(TDU=16384)

         ORACLE_HOME = /.....)
```

```
            SID_NAME = SOLAP)
        )
    )
```

- Set the tablespace to at least the same as the Siebel transactional database size. Make sure the temporary tablespace has adequate space.
- Set the number of log file groups to 4.
- Set the size of each log file to 10 MB.
- The sga_max_size to 700 MB.
- On the client side, edit the tnsnames.ora file. Modify the TNS alias by adding SDU= and TDU= as follows:

```
myhost_orcl.world=
    DESCRIPTION=(SDU=16384)(TDU=16384)
    ADDRESS = (PROTOCOL = TCP)(HOST=myhost)(PORT=1521))
CONNECT_DATA=(SID=ORCL))
```

SQL Server-Specific Database Guidelines for Siebel Data Warehouse

This section lists guidelines for SQL Server database usage.

NOTE: SQL Server databases with binary sort order or case-sensitive dictionary sort order are supported. Case-insensitive dictionary sort order is not supported.

SQL Server databases should be created with ANSI NULL option selected.

To set the ANSI NULL option

1. In Enterprise Manager, select Database.
2. Right-click and choose Database properties.
3. Click the Options tab and select the box for ANSI NULL default.

In a SQL Server 2000 environment, when loading Analytics tables with international data, or loading more than one language, modify the DB Library Options setting.

To modify the DB Library Options setting

1. In the program menu Microsoft SQL Server, launch the Client Network utilities.
2. Select the DB Library Options tab.

3 Clear the option Automatic ANSI to OEM.

NOTE: SQL Server 2000 automatically tunes many of the server configuration options; therefore, an administrator is required to do little, if any, tuning. Although these configuration options can be modified, the general recommendation is that these options be left at their default values, allowing SQL Server to automatically tune itself based on run-time conditions.

Recommended SQL Server Database Parameters

If necessary, SQL Server components can be configured to optimize performance, as shown in Table 11.

Table 11. Recommended Variable Settings for SQL Server Databases

Parameter	Recommended Setting	Notes
Affinity mask	0	
Allow updates	0	
Awe enabled	0	
C2 audit mode	0	
Cost threshold for parallelism	5	
Cursor threshold	–1	
Default full-text language	1033	
Default language	0	
Fill factor	95%	For insert-intensive transactions, set Fill Factor between 90 and 95%. For better query performance, set Fill factor to 95 or even 100%.
Index create memory	1024 KB	Default is 0.
Lightweight pooling	0	
Locks	0	
Max degree of parallelism	0	Default is 0. This turns off parallelism. Max degree of parallelism should be left at 0, which means use parallel plan generation. It should be set to 1 (use only 1 process) if you run multithreaded components (for example, several EIM threads).
Max server memory	2000 MB	Default is 2147483647.
Max text repl size	65536 B	
Max worker threads	100	Default is 255.

Table 11. Recommended Variable Settings for SQL Server Databases

Parameter	Recommended Setting	Notes
Media retention	0	
Min memory per query	1024 KB	
Min server memory	500 MB	Default is 0.
Nested triggers	1	
Network packet size	8192 B	Default is 4096.
Open objects	0	
Priority boost	0	
Query governor cost limit	0	Modify to 60 only if CPU is high.
Query wait	–1 sec	
Recovery interval	0 min	
Remote access	1	
Remote login timeout	20 sec	
Remote proc trans	0	
Remote query timeout	600 sec	
Scan for startup procs	0	
Set working set size	0	
Two-digit year cutoff	2049	
User connections	0	
User options	0	

- **SQL Server memory:** Make sure adequate memory is available.
- **Transaction logs and TempDB:** Reside on a separate disk from those used by database data.
- **Full load:** Full Recovery model for the database.
- **Incremental (Refresh) load:** Change from Full to Bulk-Logged Recovery model.

Teradata-Specific Database Guidelines for Siebel Data Warehouse

The following requirements apply to Teradata database usage:

- Install the FastLoad and TPump Teradata utilities on the same machine where the Informatica Server is installed.

- When installing Informatica PowerCenter, make sure there are no spaces in the Informatica Server directory name or the directory path. The default directory contains spaces that you must remove manually.

Deploying Siebel Data Warehouse in Latin-1 General, Unicode and Non-English Environments

This section describes the different settings for Informatica servers and databases when deploying the Siebel Data Warehouse in Latin-1 General, Unicode, and non-English environments. When you configure Informatica, the Informatica repository, and the databases, you will need to refer to this section.

The Siebel Data Warehouse can be deployed in various code page environments to support global deployments. The following source and data warehouse configurations are supported:

- Latin-1 General (ASCII) to Latin-1 General (ASCII). See "Source and Data Warehouse Code Pages for Latin-1 General (ASCII) to Latin-1 General (ASCII)" on page 45.
- Unicode to Unicode. See "Source and Data Warehouse Code Pages for Unicode to Unicode" on page 46.
- Code page (multi- or single-byte) to Unicode. See "Source and Data Warehouse Code Pages for Code Page to Unicode" on page 48.
- Code page to code page (where the code pages are the same). See "Source and Data Warehouse Code Pages for Code Page to Code Page" on page 50.

NOTE: If you are installing the Siebel Data Warehouse in a UNIX environment, you must use the Unicode character data mode.

Code Page Overview

A code page contains the encoding to specify characters in a set of one or more languages. An encoding is the assignment of a number to a character in the character set. You use code pages to identify data that might be in different languages. For example, if you are importing Japanese data into a mapping, you must select a Japanese code page for the source data.

When you set a code page, the application or program for which you set the code page refers to a specific set of data that describes the characters the application recognizes. This influences the way that application stores, receives, and sends character data.

Choosing Code Pages

Choose code pages based on the character data you are using in mappings. Character data can be represented by character modes based on the character size.

Character size is measured by the amount of storage space a character requires in the database. Database character can be one of the following: single byte, double byte or multibyte.

Code Page Compatibility

Compatibility between code pages is essential for accurate data movement when the Informatica Server runs in the Unicode data movement mode. When two code pages are compatible, the characters encoded in the two code pages are virtually identical.

For accurate data movement, the data warehouse code page must be a superset of the source code page. If the source code page is a superset of the data warehouse code page, the Informatica Server cannot process the characters because they cannot be encoded in the data warehouse code page. The data warehouse then results in incorrect or missing data.

Source and Data Warehouse Code Pages for Latin-1 General (ASCII) to Latin-1 General (ASCII)

This section provides the code pages for Latin-1 General (ASCII) to Latin-1 General (ASCII) configurations.

Informatica Server and Repository Server Running on Windows with OS ENU

Table 12 provides the code pages for the Informatica Server and Repository Server running on Windows with OS ENU.

Table 12. Code Pages for Informatica Server and Repository Server on Windows with OS ENU

Component Code Page	Code Page
Source code page	MS Windows Latin 1 (ANSI), superset of Latin 1
Data warehouse code page	MS Windows Latin 1 (ANSI), superset of Latin 1
Informatica Repository code page	MS Windows Latin 1 (ANSI), superset of Latin 1
Informatica Server code page	MS Windows Latin 1 (ANSI), superset of Latin 1

Informatica Server and Repository Server Running on UNIX with OS ENU

The following environment variables need to be set before you begin the installation process:

- NLS_LANG (for Oracle). For instructions, see "To set the NLS_LANG environment variable for Oracle databases" on page 52.
- DB2CODEPAGE (for DB2). For instructions, see "To set the DB2CODEPAGE environment variable for DB2 databases" on page 52.

The following environment variable needs to be set during the configuration process:

- Informatica Server Data Movement. For instructions, see "To set up the Informatica Server" on page 78.

Table 13 provides the code pages for the Informatica Server and Repository Server running on UNIX with OS ENU.

Table 13. Code Pages for Informatica Server and Repository Server on UNIX with OS ENU

Component Code Page	Code Page
Source code page	MS Windows Latin 1 (ANSI), superset of Latin 1
Data warehouse code page	MS Windows Latin 1 (ANSI), superset of Latin 1
Informatica Repository code page	ISO 8859-1 Western European
Informatica Server code page	ISO 8859-1 Western European

Informatica Server on UNIX, Repository Server on Windows, both with OS ENU

The following environment variables need to be set before you begin the installation process:

- NLS_LANG (for Oracle). For instructions, see "To set the NLS_LANG environment variable for Oracle databases" on page 52.
- DB2CODEPAGE (for DB2). For instructions, see "To set the DB2CODEPAGE environment variable for DB2 databases" on page 52.

The following environment variable needs to be set during the configuration process:

- Informatica Server Data Movement. For instructions, see "To set up the Informatica Server" on page 78.

Table 14 provides the code pages for the Informatica Server running on UNIX, and the Repository Server on Windows, both with OS ENU.

Table 14. Code Pages for Informatica Server on UNIX, Repository Server on Windows with OS ENU

Component Code Page	Code Page
Source code page	MS Windows Latin 1 (ANSI), superset of Latin 1
Data warehouse code page	MS Windows Latin 1 (ANSI), superset of Latin 1
Informatica Repository code page	MS Windows Latin 1 (ANSI), superset of Latin 1
Informatica Server code page	ISO 8859-1 Western European

Source and Data Warehouse Code Pages for Unicode to Unicode

The following code pages are supported for source and data warehouse databases:

- UTF8 and AL32UTF8 are supported for Oracle.

- UCS-2 is supported for DB2.

The following environment variables need to be set before you begin the installation process:

- NLS_LANG (for Oracle). For instructions, see "To set the NLS_LANG environment variable for Oracle databases" on page 52.
- DB2CODEPAGE (for DB2). For instructions, see "To set the DB2CODEPAGE environment variable for DB2 databases" on page 52.

The following environment variables need to be set during the configuration process:

- Informatica Server Data Movement. For instructions, see "To set up the Informatica Server" on page 78.
- SiebelUnicodeDB. For instructions on Windows, see "Setting the SiebelUnicodeDB Environment Variable on Windows" on page 80. For instructions on UNIX, see "To set SiebelUnicodeDB on the Informatica Server if source and target databases are Unicode on UNIX" on page 122.

Informatica Server and Repository Server Running on Windows with OS ENU

Table 15 provides the code pages for the Informatica Server and Repository Server running on Windows with OS ENU.

Table 15. Code Pages for Informatica Server and Repository Server on Windows with OS ENU

Component Code Page	Code Page
Source code page	UTF-8 encoding of Unicode
Data warehouse code page	UTF-8 encoding of Unicode
Informatica Repository code page	MS Windows Latin 1 (ANSI), superset of Latin 1
Informatica Server code page	MS Windows Latin 1 (ANSI), superset of Latin 1

Informatica Server and Repository Server Running on UNIX with OS ENU

Table 16 provides the code pages for the Informatica Server and Repository Server running on UNIX with OS ENU.

Table 16. Code Pages for Informatica Server and Repository Server on UNIX with OS ENU

Component Code Page	Code Page
Source code page	UTF-8 encoding of Unicode
Data warehouse code page	UTF-8 encoding of Unicode
Informatica Repository code page	ISO 8859-1 Western European
Informatica Server code page	ISO 8859-1 Western European

Informatica Server on UNIX, Repository Server on Windows, both with OS ENU

Table 17 provides the code pages for the Informatica Server running on UNIX, the Repository Server on Windows with OS ENU.

Table 17. Code Pages for Informatica Server on UNIX, Repository Server on Windows with OS ENU

Component Code Page	Code Page
Source code page	UTF-8 encoding of Unicode
Data warehouse code page	UTF-8 encoding of Unicode
Informatica Repository code page	MS Windows Latin 1 (ANSI), superset of Latin 1
Informatica Server code page	ISO 8859-1 Western European

Source and Data Warehouse Code Pages for Code Page to Unicode

The following code pages are supported for source databases:

- JPN16JIS (for Oracle).
- 943 (for DB2).

The following code pages are supported for data warehouse databases:

- UTF8 and AL32UTF8 are supported for Oracle.
- UCS-2 is supported for DB2.

The following environment variables need to be set before you begin the installation process:

- NLS_LANG (for Oracle). For instructions, see "To set the NLS_LANG environment variable for Oracle databases" on page 52.
- DB2CODEPAGE (for DB2). For instructions, see "To set the DB2CODEPAGE environment variable for DB2 databases" on page 52.

The following environment variables need to be set during the configuration process:

- Informatica Server Data Movement. For instructions, see "To set up the Informatica Server" on page 78.
- SiebelUnicodeDB. For instructions on Windows, see "Setting the SiebelUnicodeDB Environment Variable on Windows" on page 80. For instructions on UNIX, see "To set SiebelUnicodeDB on the Informatica Server if source and target databases are Unicode on UNIX" on page 122.
- If the Informatica Server is running on Unix, then you also need to set the PMREPCODEPAGE environment variable appropriately. For example: `PMREPCODEPAGE=MS932`.

NOTE: Informatica Server can only run <LANG> OS based on the source code page. For example, if the source code page is Japanese then the Informatica Server should be running on JPN OS.

The following information uses <LANG> = JPN as an example. If you are using a language other than Japanese, replace the code page with the appropriate language.

Informatica Server and Repository Server Running on Windows and OS <LANG>

Table 18 provides the code pages for the Informatica Server and Repository Server running on Windows with OS <LANG>.

Table 18. Informatica Server and Repository Server on Windows with OS <LANG>

Component Code Page	Code Page
Source code page	MS Windows Japanese, superset of shift JIS
Data warehouse code page	MS Windows Japanese, superset of shift JIS
Informatica Repository code page	MS Windows Japanese, superset of shift JIS
Informatica Server code page	MS Windows Japanese, superset of shift JIS

Informatica Server and Repository Server Running on UNIX with OS <LANG>

Table 19 provides the code pages for the Informatica Server and Repository Server running on UNIX with OS <LANG>.

Table 19. Code Pages for Informatica Server and Repository Server on UNIX with OS <LANG>

Component Code Page	Code Page
Source code page	MS Windows Japanese, superset of shift JIS
Data warehouse code page	MS Windows Japanese, superset of shift JIS
Informatica Repository code page	MS Windows Japanese, superset of shift JIS
Informatica Server code page	MS Windows Japanese, superset of shift JIS

Informatica Server on UNIX, Repository Server on Windows, both with OS <LANG>

Table 20 provides the code pages for the Informatica Server running on UNIX and the Repository Server running on Windows, both with OS <LANG>.

Table 20. Code Pages for Informatica Server on UNIX, Repository Server on Windows, OS <LANG>

Component Code Page	Code Page
Source code page	MS Windows Japanese, superset of shift JIS
Data warehouse code page	MS Windows Japanese, superset of shift JIS

Table 20. Code Pages for Informatica Server on UNIX, Repository Server on Windows, OS <LANG>

Component Code Page	Code Page
Informatica Repository code page	MS Windows Japanese, superset of shift JIS
Informatica Server code page	MS Windows Japanese, superset of shift JIS

Source and Data Warehouse Code Pages for Code Page to Code Page

The following code pages are supported for source databases:

- JPN16JIS (for Oracle).
- 943 (for DB2).

The following code pages are supported for data warehouse databases:

- UTF8 and AL32UTF8 are supported for Oracle.
- UCS-2 is supported for DB2.

The following environment variables need to be set before you begin the installation process:

- NLS_LANG (for Oracle). For instructions, see "To set the NLS_LANG environment variable for Oracle databases" on page 52.
- DB2CODEPAGE (for DB2). For instructions, see "To set the DB2CODEPAGE environment variable for DB2 databases" on page 52.

The following environment variables need to be set during the configuration process:

- Informatica Server Data Movement. For instructions, see "To set up the Informatica Server" on page 78.
- SiebelUnicodeDB. For instructions on Windows, see "Setting the SiebelUnicodeDB Environment Variable on Windows" on page 80. For instructions on UNIX, see "To set SiebelUnicodeDB on the Informatica Server if source and target databases are Unicode on UNIX" on page 122.
- If the Informatica Server is running on UNIX, then you also need to set the PMREPCODEPAGE environment variable appropriately. For example: `PMREPCODEPAGE=MS932`.

The following information uses <LANG> = JPN as an example. If you are using a language other than Japanese, replace the code page with the appropriate language.

Informatica Server and Repository Server Running on Windows and OS <LANG>

Table 21 provides the code pages for the Informatica Server and the Repository Server running on Windows, both with OS <LANG>.

Table 21. Code Pages for Informatica Server and Repository Server on Windows, OS <LANG>

Component Code Page	Code Page
Source code page	MS Windows Japanese, superset of shift JIS
Data warehouse code page	MS Windows Japanese, superset of shift JIS
Informatica Repository code page	MS Windows Japanese, superset of shift JIS
Informatica Server code page	MS Windows Japanese, superset of shift JIS

Informatica Server and Repository Server Running on UNIX and OS <LANG>

Table 22 provides the code pages for the Informatica Server and the Repository Server running on UNIX, both with OS <LANG>.

Table 22. Code Pages for Informatica Server and Repository Server on Windows, OS <LANG>

Component Code Page	Code Page
Source code page	MS Windows Japanese, superset of shift JIS
Data warehouse code page	MS Windows Japanese, superset of shift JIS
Informatica Repository code page	MS Windows Japanese, superset of shift JIS
Informatica Server code page	MS Windows Japanese, superset of shift JIS

Informatica Server on UNIX, Repository Server on Windows, Both with OS <LANG>

Table 23 provides the code pages for the Informatica Server running on UNIX and the Repository Server running on Windows, both with OS <LANG>.

Table 23. Code Pages for Informatica Server on UNIX, Repository Server on Windows, OS <LANG>

Component Code Page	Code Page
Source code page	MS Windows Japanese, superset of shift JIS
Data warehouse code page	MS Windows Japanese, superset of shift JIS
Informatica Repository code page	MS Windows Japanese, superset of shift JIS
Informatica Server code page	MS Windows Japanese, superset of shift JIS

To set the NLS_LANG environment variable for Oracle databases

1 Determine the NLS_LANG value.

   a In the data warehouse database, run the command

      `SELECT * FROM V$NLS_PARAMETERS`

   b Make a note of the NLS_LANG value, which is in the format [NLS_LANGUAGE]_[NLS_TERRITORY].[NLS_CHARACTERSET].

      For example: `American_America.UTF8`

2 For Windows:

   a Navigate to Control Panel > System and click the Advanced tab. Click Environment Variables.

   b In System variables section, click New.

   c In the Variable Name field, enter `NLS_LANG`.

   d In the Variable Value field, enter the NLS_LANG value that was returned in Step 1.

      The format for the NLS_LANG value should be [NLS_LANGUAGE]_[NLS_TERRITORY].[NLS_CHARACTERSET].

      For example: `American_America.UTF8`.

3 For UNIX, set the variable as shown below:

   `setenv NLS_LANG <NLS_LANG>`

   For example: `setenv NLS_LANG American_America.UTF8`.

   If your data is ASCII and the Informatica Server is running on UNIX, then set

   `NLS_LANG <NLS_LANGUAGE>_<NLS_TERRITORY>.WE8ISO8859P1`

4 Reboot the machine after creating the variable.

To set the DB2CODEPAGE environment variable for DB2 databases

1 Determine the DB2CODEPAGE value.

   a In the data warehouse database, run the command

      `SELECT CODEPAGE FROM SYSCAT.DATATYPES WHERE TYPENAME = 'VARCHAR'`

   b Make a note of the result.

      For example: 1208

2 For Windows:

   a Navigate to Control Panel > System and click the Advanced tab. Click Environment Variables.

   b In System variables section, click New.

   c In the Variable Name field, `DB2CODEPAGE`.

   d In the Variable Value field, enter the value that was returned in Step 1.

3 For UNIX, set the variable as shown below:

```
setenv DB2CODEPAGE <DB2CODEPAGE value>
```

For example: `setenv 1208`.

4 Reboot the machine after creating the variable.

5 Installing and Configuring Siebel Analytics Applications on Windows

This chapter contains the following main topics:

- Siebel Analytics Applications Installation Overview on page 55. Describes the components that are installed during a Siebel Analytics Applications installation.
- Siebel Analytics Applications Installation and Configuration Flow on page 57. Provides a process flow diagram that shows the installation and configuration steps for Siebel Analytics Applications.
- Siebel Analytics Applications Installation and Configuration Process Task List on page 58. Provides a sequential list of all the installation and configuration tasks required for Siebel Analytics Applications.

Siebel Analytics Applications Installation Overview

A Siebel Analytics Applications installation has three main parts. You should perform the installation in the sequence listed below.

1 Siebel Analytics Platform

You must install the Siebel Analytics platform before you install Siebel Analytics Applications. For information about installing the Siebel Analytics platform, see *Siebel Analytics Installation and Configuration Guide*.

2 Siebel Analytics Applications

Instructions for installing Siebel Analytics Applications are contained in "Running the Siebel Analytics Applications Installation Wizard (Windows)" on page 59. When you run the installation wizard, the following components are installed:

- Prebuilt metadata content

  This metadata content is contained in the Siebel Analytics repository file (SiebelAnalytics.rpd).

- Prebuilt reports and dashboard content

  This content is contained in the Siebel Analytics Web Catalog file (SiebelAnalytics.webcat).

- Prebuilt Informatica content

  This content includes Extract-Transform-Load (ETL) repository objects, such as mappings, sessions, and workflows, and is contained in the Informatica repository file (Siebel_DW_Rep.rep).

- DAC metadata repository files

  This content includes repository objects, such as tables, subject areas, execution plans, and tasks, and is contained in XML files.

3 Informatica PowerCenter

Instructions for installing Informatica PowerCenter are contained in "Running the Informatica PowerCenter Installation Wizard (Windows)" on page 60. When you run the installation wizard, you will be instructed to select the following components for installation:

- PowerCenter Server
- PowerCenter Repository Server
- PowerCenter Client

NOTE: The computers on which you install Siebel Analytics need to meet the conditions specified in *Siebel System Requirements and Supported Platforms, Siebel Analytics*.

Siebel Analytics Applications Installation and Configuration Flow

Figure 8 shows the sequence for the Siebel Analytics Applications installation and configuration.

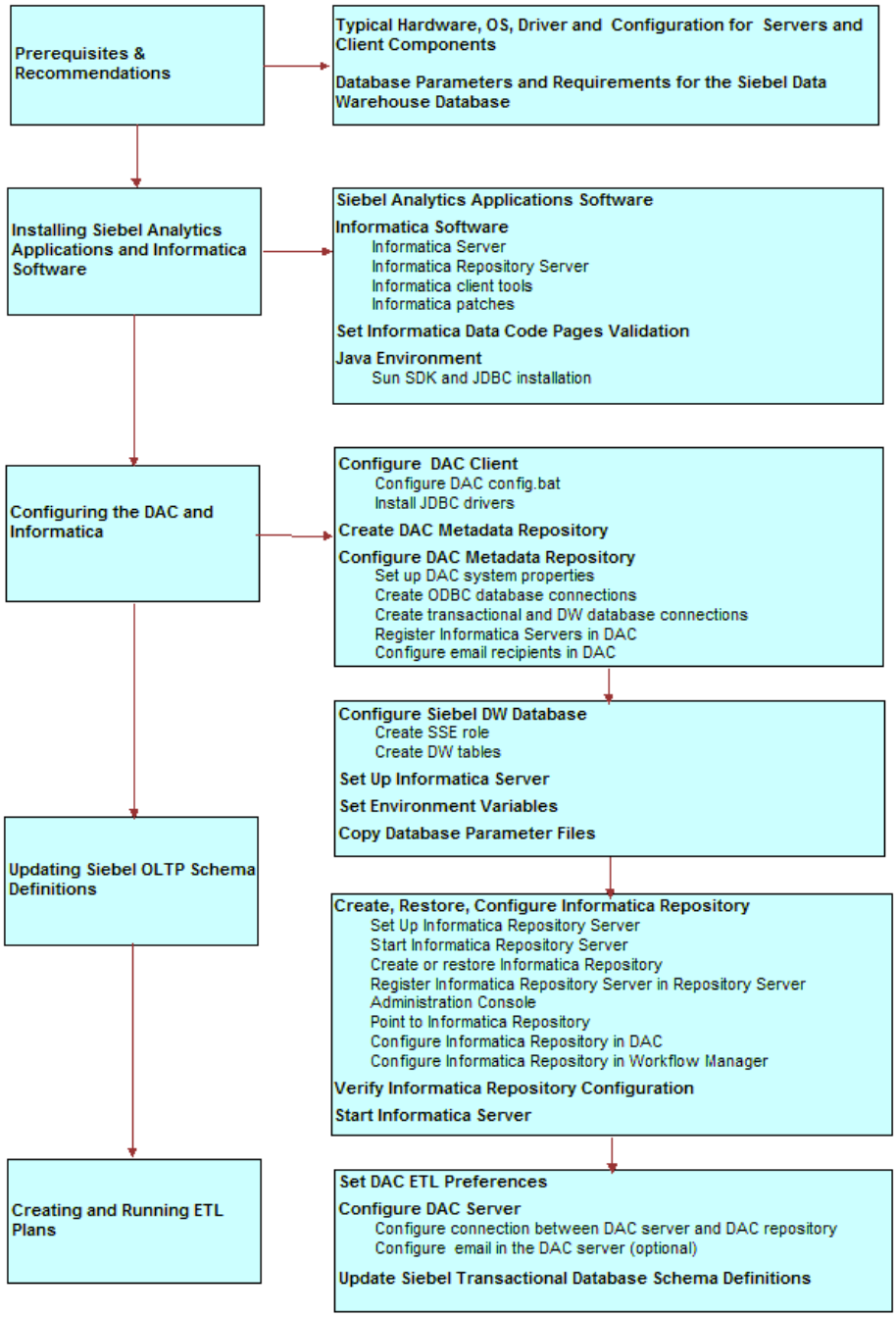


Figure 8. Siebel Analytics Applications Installation and Configuration Flow

Siebel Analytics Applications Installation and Configuration Process Task List

The Siebel Analytics Applications installation and configuration process consists of the following tasks. Complete each of these tasks in the order listed below.

1. Running the Siebel Analytics Applications Installation Wizard (Windows) on page 59.
2. Running the Informatica PowerCenter Installation Wizard (Windows) on page 60.
3. Installing the Informatica Patches on Windows on page 62.
4. For Teradata installations, install the Teradata patch. See "Installing the Teradata Patch on Windows" on page 63.
5. Setting Informatica Data Code Pages Validation on page 63.
6. Installing the Java SDK on page 63.
7. Configuring the DAC Client on page 64.
   a. Configuring the DAC Config.bat File on page 64.
   b. Installing JDBC Drivers for DAC Database Connectivity on page 64.
8. Creating the DAC Metadata Repository on page 65.
9. Configuring the DAC Metadata Repository on page 67.
   a. Setting Up DAC System Properties on page 67.
   b. Creating ODBC Database Connections on page 72.
   c. Creating Transactional and Data Warehouse Database Connections on page 73
   d. Registering Informatica Servers in the DAC Client on page 74.
   e. Configuring Email Recipients in the DAC Client on page 75.
   f. Activating Join Indices for Teradata Databases on page 75.
10. Configuring the Siebel Data Warehouse Database on page 76.
    a. Creating the SSE Role on page 76.
    b. Creating Data Warehouse Tables on page 76.
    c. Creating and Dropping Data Warehouse Tables on a Teradata Database on page 77
11. Setting Up the Informatica Server on page 78.
12. Setting Environment Variables on Windows on page 80.
13. Copying Database Parameter Files on page 81.
14. Copying Teradata Loader Control Files on page 82.
15. Creating, Restoring, and Configuring the Informatica Repository on page 82.
    a. Setting Up the Informatica Repository Server on page 82.

Running the Siebel Analytics Applications Installation Wizard (Windows)

NOTE: Before you install Siebel Analytics Applications, you must first install Siebel Analytics Platform. For instructions on installing Siebel Analytics Platform, see *Siebel Analytics Installation and Configuration Guide*.

To install the Siebel Analytics Applications on Windows

1 Access the installation files on the installation CD-ROM, and then run the program setup.exe.

2 The installation wizard window appears and prompts you through each screen, as shown in the following table.

To continue to the next screen, click Next. To return to a previous screen, click Back.

Screen	Your Action	Notes
Welcome	Click Next.	
License Agreement	Select “I accept...” and click Next.	
License File	Browse for and select the appropriate XML file, and then click Next.	
Summary Information	Click Next.	Siebel Analytics Applications will be installed in the same directory where you have Siebel Analytics Platform installed.
Summary Information	Click Finish.	

Running the Informatica PowerCenter Installation Wizard (Windows)

Follow this procedure to install Informatica PowerCenter.

NOTE: You must install the Informatica Server on the same machine where the DAC server is installed, because the DAC server requires the Informatica utility pmcmd.

To install Informatica PowerCenter on Windows

1 Access the installation files on the installation CD-ROM, and run the program setup.exe.

2 The installation wizard window appears and prompts you through each screen, as shown in the following table.

To continue to the next screen, click Next. To return to a previous screen, click Back.

Screen	Your Action	Notes
Welcome	Click Next.	
Customer Information	Enter the user name, company name, and Product license key. Do not enter the Option or Connectivity license key during the product installation.	Informatica provides three different license keys: ■ **Product.** Allows you to install the Informatica Server and Repository Server. Enter during the product installation and the procedures "Setting Up the Informatica Server" on page 78 and "Pointing to the Informatica Repository" on page 87. ■ **Connectivity.** Allows you to access sources and targets. Enter during the procedures "Setting Up the Informatica Server" on page 78 and "Pointing to the Informatica Repository" on page 87. ■ **Options.** Allows you to access specific options, such as Team-Based Development, Server Grid, and Partitioning. Enter during the procedures "Setting Up the Informatica Server" on page 78 and "Pointing to the Informatica Repository" on page 87. For more information about Informatica license keys, see *Siebel Business Third-Party Bookshelf*.
License Agreement	Select "I accept..." and click Next.	

Screen	Your Action	Notes
Select Components	■ Select the following components: ■ Informatica PowerCenter 7.1.2 OEM for Siebel - Client ■ Informatica PowerCenter 7.1.2 OEM for Siebel - Server ■ Informatica PowerCenter 7.1.2 OEM for Siebel - Repository Server ■ Browse for and select the appropriate destination folder, and then click Next.	
Program Folder	Select a program folder or accept the default, and then click Next.	**NOTE:** If you are using a Teradata database, make sure there are no spaces in the Informatica Server directory name or the directory path. The default directory contains spaces that you must remove manually.
Review Settings	Review the selected components and destination directory information, and then click Next.	
Setup Type	■ Remove the check marks from the Configure Informatica Server and Configure Informatica Repository Server check boxes. ■ (Optional) Select the View Readme check box to view the Read Me file. ■ Click Next.	You will configure the Informatica Server and Repository Server in later steps.
Wizard Complete	Click Finish.	

Installing the Informatica Patches on Windows

After you install Informatica PowerCenter version 7.1.2, you need to install two Informatica patches.

To install the Informatica patches on Windows

1. Access the patch files on the installation CD-ROM.
2. Locate the folder named Patch1, and install the patch by following the instructions in the readme file.
3. Locate the folder named Patch2, and install the patch by copying the files from the appropriate operating system folder to \Informatica PowerCenter 7.1.2\Server\bin.

Installing the Teradata Patch on Windows

If you are using a Teradata database, you need to install the Teradata patch.

To install the patch for Teradata databases on Windows

1. Locate the Patch_for_Teradata subdirectory in the Informatica directory on the installation CD-ROM.
2. Copy the contents from the Patch_for_Teradata\Windows directory to the Informatica Server bin directory.

   It is recommended that you first rename the existing files in the bin directory.
3. Copy the resword.txt file from the Patch_for_Teradata directory to the Informatica Server bin directory.

Setting Informatica Data Code Pages Validation

After installing the Informatica Server, Repository Server, and client tools, add the following section to the powrmart.ini file located in the Informatica client directory:

```
[Code Pages]

ValidateDataCodePages=No
```

Installing the Java SDK

You need to install the Java SDK on the machines where both the DAC server and DAC client are installed.

NOTE: The DAC client and server require the SDK for the Java 2 Platform, Standard Edition, Version 1.4.2.

To install Java SDK 1.4.2

- Download the Java SDK from the Sun Web site, and install it on the machines where both the DAC server and DAC client are installed. Make sure there are no spaces in the directory path.

  For example on Windows: D:\j2sdk142

Configuring the DAC Client

This section includes the following topics:

Configuring the DAC Config.bat File

Follow this procedure to configure the DAC config.bat file.

To configure the DAC config.bat file

1. In the DAC directory, find and open the config.bat file.

   This directory is usually in the SiebelAnalytics root directory.

2. Edit the JAVA_HOME variable to point to the directory where you installed the Java SDK.

   Make sure there are no spaces in the path reference.

   For example:

   ```
   set JAVA_HOME=d:\j2sdk1.4.2
   ```

3. Edit the DAC_HOME variable to point to the directory where you installed the DAC.

   For example:

   ```
   set DAC_HOME=d:\SiebelAnalytics\DAC
   ```

Installing JDBC Drivers for DAC Database Connectivity

This section provides instructions for installing the appropriate JDBC driver in the DAC\lib directory to enable DAC database connectivity. You need to install the JDBC driver on the machines where both the DAC server and DAC client are installed.

To install JDBC drivers in the SiebelAnalytics\DAC\lib directory

- **Oracle.** If you are using an Oracle database (other than 8.x), find the directory where Oracle is installed. Copy the file named ojdbc14.jar in the jdbc\lib directory and paste it in the SiebelAnalytics\DAC\lib directory.

- If you are using Oracle 8.x, copy the file named classes12.zip and paste it in the SiebelAnalytics\DAC\lib directory. Also, edit the ORACLELIB setting in the config.bat file as follows:

  ```
  set ORACLELIB=%DAC_HOME%\lib\classes12.zip
  ```

- **DB2.** If you are using a DB2 database, find the directory where DB2 is installed. In the Java subdirectory copy the file named db2java.zip and paste it in the SiebelAnalytics\DAC\lib directory.

  NOTE: If your source or target database is DB2-UDB, you also need to create Siebel stored procedures to be used during the ETL process. For instructions, see "Creating Siebel Stored Procedures for DB2-UDB" on page 101.

- **MSSQL.** If you are using an MSSQL database, download the SQL Server JDBC drivers from Microsoft's Web site. Copy the files msbase.jar, mssqlserver.jar, and msutil.jar to the SiebelAnalytics\DAC\lib directory.

- **Teradata.** If you are using a Teradata database, find the directory where Teradata is installed. Copy the files terajdbc4.jar, log4j.jar, and tdgssjava.jar and paste them in the SiebelAnalytics\DAC\lib directory. Depending on the Teradata JDBC version, you may not have log4j.jar and tdgssjava.jar.

Creating the DAC Metadata Repository

This procedure provides instructions for creating a repository connection and importing seed data.

NOTE: The DAC client can connect to only one DAC repository at a time.

NOTE: Before you start this procedure, you need to create a database for the DAC metadata repository. This database can be the same as the data warehouse database.

To create the DAC metadata repository connection

1. Launch the DAC client by double-clicking the icon on your desktop or navigating to the SiebelAnalytics\DAC directory and double-clicking the startclient.bat file.
2. In the Login... dialog box, select Configure.
3. In the Configuring... dialog box, select Create Connection, and then click Next.
4. Enter the appropriate connection details.

Field	Description
Name	A unique database name.
Connection Type	Type of database.

Field	Description
Connection String	If you are using... ■ Oracle (OCI8) use the tnsnames entry. ■ Oracle (Thin) use the instance name. ■ SQL server use the database name. ■ DB2-UDB use the connect string as defined in the DB2 configuration.
Database Host	Machine name where the database resides.
Database Port	The port number where the database listens.

5 Test the connection to make sure it works.

If the DAC repository schema does not exist in the database to which you are connecting, the schema will be automatically created. When a repository is created on Oracle or DB2 databases, you have the option of specifying a tablespace. For MSSQL and DB2 databases, you can indicate whether the environment is Unicode.

6 (Optional for Oracle or DB2 databases) Specify a tablespace.

7 (Optional for MSSQL or DB2 databases) Check the Unicode check box if the environment is Unicode.

8 Click Yes when asked if you want to have the repository created automatically.

NOTE: The DAC repository is not supported on DB2-390 databases. If your source or target database is DB2-390, you need to use a DB2-UDB, MSSQL, or Oracle database for both the DAC repository and the Informatica repository.

To import seed data into the DAC metadata repository

1 From the DAC menu bar, choose Tools > DAC Repository Management > Import.

2 Select the appropriate import option, and click OK.

Option	Description
Import All Tables	Imports all database tables.
Import Categories > Logical	Imports all information contained in the DAC Design view and database connection information.
Import Categories > Run Time	Imports all information contained in the DAC Execute view.
Import Categories > System	Imports all information contained in the DAC Setup view, except passwords for servers and database connections.
Import Selected Tables	Allows you to select tables for importing.

3 If your source or target database is a DB2-390 database, run DAC_DB2390_Source.sql or DAC_DB2390_Target.sql immediately after importing the seed data by connecting to the database where the DAC repository resides.

Configuring the DAC Metadata Repository

This section includes the following topics:

NOTE: You will configure the DAC server to point to this repository later in the installation process, in the procedure "Configuring the DAC Server" on page 96.

Setting Up DAC System Properties

Follow this procedure to set up the DAC system properties, which determine the behavior of the DAC server.

To set up the DAC system properties

1 Launch the DAC client.

2 Click Setup on the DAC toolbar.

The DAC System Properties tab is active.

3 Set up the system properties appropriately.

Property	Description
Analyze Frequency (in days)	For DAC metadata tables, the frequency (in days) the DAC client automatically updates the table and index statistics for the DAC repository. The value must be numerical.
Analyze Tables	For the data warehouse schema tables, indicates whether during the ETL process the DAC server automatically issues analyze table statements when a table is truncated and loaded. Possible values are True and False.

Property	Description
Auto Restart ETL	Possible values are True and False. When set to True: An ETL that is running when the DAC server abnormally terminates will continue running when the DAC server is restarted. When set to False: An ETL that is running when the DAC server abnormally terminates will not automatically restart when the DAC server restarts. The ETL status will be updated to Failed. An administrator will have to manually restart the ETL.
CreateQueryIndexesAtTheEnd	Possible values are True and False. During the ETL process, the DAC server automatically drops and creates indices. When set to True, this property groups all indices of the Query type and creates them after the ETL is complete. **NOTE:** The DropAndCreateIndexes property takes precedence over this property. Therefore, if the DropAndCreateIndexes property is set to False, you cannot set the property CreateQueryIndexesAtTheEnd to True to have indices of the Query type created at the end of the ETL process. Also, be aware that when this property is set to True, tables will be analyzed twice. If any indices are marked as Query type indices, and are used by ETL processes, it can adversely affect the performance of the ETL process.
DAC Server Host	Host name of the machine where the DAC server resides. You cannot use an IP address for this property. **NOTE:** The DAC server and a given DAC repository have a one-to-one mapping. That is, you can only run one DAC server against any given DAC repository. Thus, in the repository you must specify the network host name of the machine where the DAC sever is to be run. **NOTE:** This property also takes the value *localhost*. However, this value is provided for development and testing purposes and should not be used in a production environment.

Property	Description
DAC Server OS	Operating system of the machine where the DAC server resides. Possible values are Windows, Solaris, HP, or AIX. **NOTE:** If you move the DAC server from another operating system to AIX, you need to do the following: change the DAC server host to the appropriate value; restart the DAC client; reenter all the password fields for the Informatica servers and database connections; and reconfigure the DAC server on the AIX machine by running serverSetupPrompt.sh.
DAC Server Port	Network port to which the DAC server binds in order to listen to client requests. The default value is 3141. If this port has been assigned to another process, you can enter any numerical port value greater than 1024.
DropAndCreateIndexes	Possible values are True and False. Indicates whether, during the ETL process, the DAC server automatically drops and creates indices. **NOTE:** This property takes precedence over the CreateQueryIndexesAtTheEnd. Therefore, if the DropAndCreateIndexes property is set to False, you cannot set the property CreateQueryIndexesAtTheEnd to True to have indices of the Query type created at the end of the ETL process.
Dryrun	Possible values are True and False. Indicates whether tasks are executed without invoking Informatica workflows. The following processes are executed: change capture, truncation of tables, drop and creation of indices, and analyze statements. Possible values are True and False. **NOTE:** This option should be used for debugging purposes only and not used in a production environment.
Generic Task Concurrency Limit	Determines how many tasks with execution types other than Informatica can be run concurrently. The value must be numerical. To set this value, you should consider what the external tasks do. For example, if the tasks open connections to a database, you should consider how this would affect the preconfigured tasks.
HeartBeatInterval	Frequency (in seconds) the DAC server checks on the health of the database connections. The value must be numerical. For example, a value of 300 (the default value) indicates the system will perform subsystem diagnostics and recovery procedures every 300 seconds.

Property	Description
Output Redirect	Indicates whether logging information and standard output and errors are redirected to files in the log directory (when property is set to True). The file containing standard output starts with out_ and ends with the .log extension. The standard error messages are in the file starting with err_ and ending with the .log extension. If this property is set to False, the logging information is directed to the machine's standard output and error files, which typically defaults to the console from which the DAC server was launched if the server was launched in a visible console mode. If the server is launched as a Windows service, the logging information is directed to the service log. If the server is launched with the command shell not visible, all logging information is deleted.
Repository DB Pool Size	Maximum number of connections to the DAC repository the DAC server will maintain. The value must be numerical. **NOTE:** As the DAC server runs, it needs to perform transactions and queries on the DAC repository. In order for it to do so efficiently, it internally maintains a pool of preexisting database connections. This pool of database connections reduces latency and leads to better use of resources through connection sharing. However, you need to specify the maximum size of the connection pool. This setting should be determined in consultation with the database administrator of the database where the DAC repository resides. **NOTE:** This property was depracated as of version 7.7.1.5.
Repository Name	Unique name for the repository.
Scheduler.Poll.Interval	Frequency (in seconds) the DAC server polls for changes in the schedule configuration.
Script After Every ETL	The name of the script or executable to be run after every execution plan. For more information, see the description of the property Script Before Every ETL.
Script Before Every ETL	The name of the script or executable to be run before every execution plan. For example, before running an execution plan, you might want to run a process or perform certain tasks. These can be contained in a script or executable. This file should be placed in the scripts subdirectory of the DAC server. The execution plan runs only after the external process has finished. Therefore, it is important that the script or executable does not fail.

Property	Description
Server Log Level	Output logging level. Possible values are Finest, Finer, Fine, Config, Info, Warning, and Severe. The Severe value produces minimal log details, and Finest produces the most extensive amount of reporting.
SQL Trace	Possible values are True and False. Indicates whether the SQL statements to the DAC repository and database connections are added to the log file. Possible values are True and False. The True value sends a hint to the database connectivity layer of the DAC server to enable SQL tracing; thus, every SQL statement that is run by the DAC server is spooled to the appropriate output log file. It is recommended that you set this property to False.
Verify and Create Non-Existing Indices	Possible values are True and False. Indicates whether indices defined in the DAC repository will be automatically created in the data warehouse database during an incremental load. **NOTE:** When this system property is set to True, the DAC server verifies whether indices defined in the DAC repository are also defined in the data warehouse database. This verification process can delay the execution of an execution plan.

Drop and Create Change Capture Views Always	Possible values are True and False.
	
	When set to True (the default value), the DAC server drops and creates change capture views every time it performs a change capture process, including for both full and incremental loads.
	
	Setting this property to True can create system catalog lock up for DB2-UDB and DB2-390 databases. Therefore, by setting the property to False, the DAC server will drop and create views selectively, using the following rules:
	
	■ In full mode:
	  ■ During the change capture phase, views will be dropped and created as full views.
	  ■ During the change capture sync process, incremental views will be generated.
	■ In incremental mode:
	  ■ If the view exists, it will not be dropped and created.
	  ■ If the view does not exist, the incremental view will be created.

Creating ODBC Database Connections

The Informatica Server and Repository Server require native connections to the Siebel Data Warehouse and Siebel transactional database. The Informatica client needs both an ODBC connection and a native connection to the Siebel Data Warehouse and transactional database.

NOTE: If the Informatica Server is installed in MSSQL in non-ASCII mode, it will need an ODBC connection to the Siebel Data Warehouse and transactional database.

To create database connections for DB2 installations

1. Using the DB2 Client Configuration Assistant, create a database connection to the Siebel Data Warehouse, the Siebel transactional database, and the Informatica repository.
2. In Windows, in the System DSN tab of the ODBC Data Source Administrator, create an ODBC connection to the Siebel Data Warehouse and transactional database using an ODBC driver.
3. Test the connections to make sure they work.

To create database connections for Oracle installations

1. Using Net8 Assistant or Oracle Net Service, create a native connect string (net service name) to the Siebel Data Warehouse, the Siebel transactional database, and the Informatica repository.
2. In Windows, in the System DSN tab of the ODBC Data Source Administrator, create an ODBC connection to the Siebel Data Warehouse using the Siebel Merant ODBC driver that is supplied with Siebel Enterprise Server.
3. In Windows, in the System DSN tab of the ODBC Data Source Administrator, create an ODBC connection to the Siebel Data Warehouse and transactional database using the Informatica-supplied Merant Closed driver. This ODBC driver was installed during the Informatica client installation (the exact name depends on the specific release of Informatica being used).
4. Test the connections to make sure they work.

To create database connections for SQL Server installations

1. In Windows, in the System DSN tab of the ODBC Data Source Administrator, create an ODBC connection to the Siebel Data Warehouse and Siebel transactional database using a SQL Server ODBC driver.
2. Test the connections to make sure they work.

To create database connections for Teradata installations

1. In Windows, in the System DSN tab of the ODBC Data Source Administrator, create an ODBC connection for the Teradata database.

2 Set the following parameters:

Field	Enter
DateFormat	AAA
SessionMode	ANSI
NoScan	Yes

3 For Unicode environments, in the Teradata ODBC Driver Advanced Options dialog box, set the Character Set parameter to UTF8.

Creating Transactional and Data Warehouse Database Connections

Follow this procedure to create the transactional and data warehouse database connections.

To create transactional and data warehouse database connections

1 Launch the DAC client.

2 Click Setup on the toolbar, and then click the Database Connections tab.

3 In the Edit subtab, enter the following:

Field	Description
Logical Name	Logical name for the database connection.
Type	Database functional type. Possible values are: ■ Source ■ Warehouse ■ Informatica Repository ■ DAC Repository ■ Other
Connection Type	Type of database. Possible values are: ■ Oracle (OCI8) ■ Oracle (Thin) ■ DB2 ■ DB2-390 ■ MSSQL ■ Teradata

Field	Description
Database Name	Name of database.
Table Owner	Valid database user.
Table Owner Password	Valid database user password.
Max Num Connections	Maximum number of database connections this connection pool can contain. For information about determining how to set this value, see "Determining the Number of Transactional and Data Warehouse Database Connections" on page 167.
DB Host	Machine name where the database resides.
Port	Port number where the database listens.

4 Click Test Connection to make sure the connection works.

Registering Informatica Servers in the DAC Client

This section describes registering the Informatica servers in the DAC and specifying the number of workflows that can be executed in parallel. The DAC server automatically load balances across the severs and does not run more than the value specified for each of them.

You need to register at least one Informatica server. You should register one Informatica server for each Informatica repository.

NOTE: It is recommended that the Informatica Server be installed on the machine where the DAC server is installed. If a machine is running only the DAC client, it does not need an Informatica server installed.

To register Informatica servers in the DAC client

1 Launch the DAC client.

2 Click Setup on the DAC toolbar, and then click the Informatica Servers tab.

3 Enter the following information:

Field	Enter
Name	Logical name for the Informatica server.
Server Host Name	Informatica server host name or IP address.
Port	Port where the Informatica server connects for requests. (The default port number is 4001).
Login	Informatica user name who has appropriate privileges to execute workflows.

Field	Enter
Password	Informatica user password.
Maximum Sessions	Maximum number of workflows that can be executed in parallel on the Informatica server. If the number of sessions is zero or is not specified, the DAC server assigns the default value of five for the DAC Lite version and 10 for the full DAC version. The DAC Lite version cannot run more than five sessions in parallel. For information about how to determine this setting, see "Determining the Informatica Server Maximum Sessions Parameter Setting" on page 166.
Inactive	Indicates whether the Informatica server is active or inactive.

Configuring Email Recipients in the DAC Client

Follow this procedure to configure email recipients.

To configure email recipients in the DAC client

1. Launch the DAC client.
2. Click Setup on the DAC toolbar, and then click the Email Recipients tab.
3. Enter the following information:

Field	Description
Name	Logical name of the user to be notified.
Email Address	Email address where the notification is sent.
Notification Level	The notification levels are as follows: ■ 10 -- Notifies recipient of success or failure of each task. ■ 5 -- Notifies recipient of success of failure of the entire ETL process. ■ 1 -- Notifies recipient that ETL completed successfully.
Inactive	Indicates whether the record is active or inactive.

Activating Join Indices for Teradata Databases

For Teradata databases, the preconfigured tasks for creating and dropping join indices are inactive. You need to activate these tasks in order to run ETL processes.

To activate join indices for Teradata databases

1. In the DAC, navigate to the Task Definitions tab in the Design view.
2. Query for all tasks beginning with Teradata.
3. Remove the check from the Inactive check box for all the tasks.
4. Click Save.

Configuring the Siebel Data Warehouse Database

This section includes the following procedures for configuring the Siebel Data Warehouse database:

- Creating the SSE Role on page 76
- Creating Data Warehouse Tables on page 76
- Creating and Dropping Data Warehouse Tables on a Teradata Database on page 77

NOTE: Before you start this procedure, you need to create a database for the data warehouse. This database can be the same as the DAC metadata repository database.

Creating the SSE Role

Follow this procedure to create the SSE role.

To create the SSE Role

- Create a database role named SSE_ROLE (SSEROLE for DB2 390 databases) and make sure this role is granted the same privileges as the SSE_ROLE on the Siebel transactional database. Assign this role to the database user. For instructions on creating roles, see the documentation provided with your database.

  NOTE: If you are using a Teradata database, you do not need to create the SSE role.

Creating Data Warehouse Tables

Use this procedure to create the data warehouse tables. Before you perform this procedure make sure the SSE role has been created for the Siebel Data Warehouse and Siebel transactional database and that the database user has been associated with the role.

If you are using a Teradata database, follow the instructions in the section "Creating and Dropping Data Warehouse Tables on a Teradata Database" on page 77.

NOTE: Additional work by the database administrator is required if the data warehouse tables need to be moved to different tablespaces for performance or manageability reasons.

To create data warehouse tables

1. From the DAC menu bar, click Tools > ETL Management > Configure.
2. In the Sources dialog box, select the database platform for the target data warehouse, source transactional database, and the Informatica repository.

   The Data Warehouse Configuration Wizard opens.
3. Select the Create Siebel Data Warehouse Tables check box, and click Next.

   The Data Warehouse tab is active.
4. Enter the following information:

Field	Description
Database Type	Type of database. If the database type is DB2 390, then check the 390 Database check box.
Table Owner	Valid database user.
Password	Valid database user password.
ODBC Data Source	Data Source Name (DSN) for the Siebel Data Warehouse.
Data Area	(Optional) Tablespace where data warehouse tables are created.
Index Area	(Optional) Indexspace where data warehouse indices are created (applicable only to Oracle and DB2 databases).
Is Unicode	Specifies whether the data warehouse database is Unicode.

Creating and Dropping Data Warehouse Tables on a Teradata Database

This section includes the procedures for creating and dropping data warehouse tables on a Teradata database.

NOTE: For Unicode environments, perform these procedures using a login that has the default character set UTF-8.

To create data warehouse tables on a Teradata database

1. Navigate to SiebelAnalytics\dwrep.
2. Copy the script create_dw_tera.sql to SQL Assistant and execute it.

To drop data warehouse tables on a Teradata database

1. Navigate to SiebelAnalytics\dwrep.
2. Copy the script drop_dw_tera.sql to SQL Assistant and execute it.

Setting Up the Informatica Server

After the Informatica Server is installed, you must configure it before you can start it. Make note of the settings you enter in the Server tab and Repository tab of the Configure Informatica Service window; you will need to enter the same settings in other configuration steps.

To set up the Informatica Server

1. Navigate to Programs > Informatica PowerCenter 7.1.2 > Informatica PowerCenter 7.1.2 OEM for Siebel - Server > Informatica Server Setup.
2. In the Informatica Server Setup Application - Options window, select Configure Informatica Service, and then click Continue.
3. In the Server tab, enter the following:

Field	Description
Server Name	A logical name for the Informatica Server. It is recommended that you enter SIEBEL_DW_SERVER.
TCP/IP Host Address	The IP address or name of the machine on which you plan to run the Informatica Server.
Max no. of concurrent sessions	From 1 to 15 sessions but not greater than 15.
Shared Memory	A number not greater than 3000000.

4. In the Repository tab, enter the following:

Field	Description
Repository Name	A logical name for the Informatica repository. It is recommended that you enter Siebel_DW_Rep. For Teradata databases, it is recommended that you enter Siebel_DW_Rep_Teradata.
Repository User	The account used to access the repository. It is recommended that you use Administrator as the username.
Repository Password	The password you use to connect to the Repository Server from the Repository Manager. It is recommended that you use Administrator as the password.
Repository Server Host Name	The host machine name where the Informatica Repository Server is installed.
Repository Port Number	The port number the Repository Server uses to connect to repository client applications. By default, this value is 5001. It is recommended that you use the default value.

5 In the Licenses tab:

a Leave the default License File Name as `pm.lic`.

b If you have an option or connectivity license key, enter it in the "with Key" field, and click Update. Do this for each option and connectivity license key.

6 In the Compatibility and Database tab:

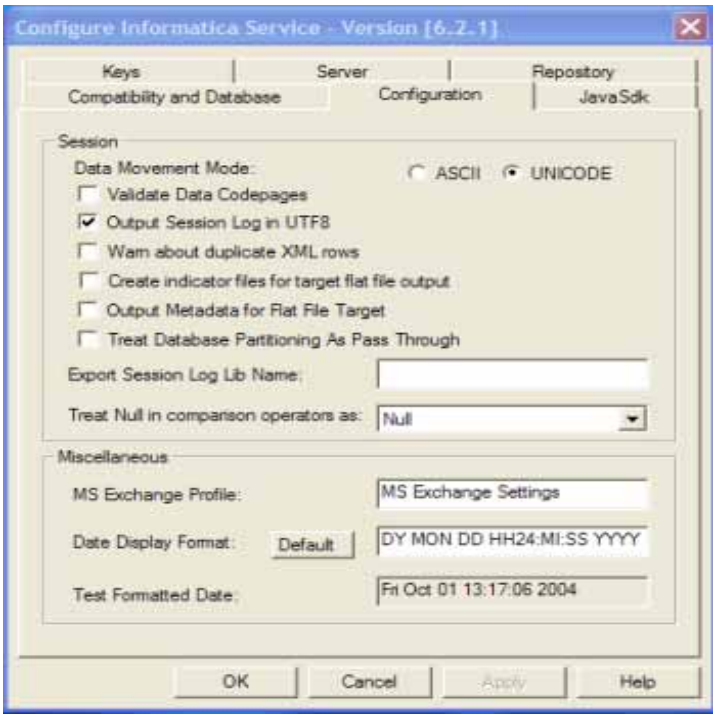
a Make sure the "Treat CHAR as CHAR on read" option is selected.

b Make sure the maximum number of database connections is 100.

7 In the Configuration tab, set the Data Movement Mode.

a Select the ASCII or UNICODE option.

b If you select the UNICODE option, remove the check from the Validate Data Codepages check box, and check the Output Session Log in UTF8 check box.



c Leave the default values in the remaining fields.

8 In the JavaSdk tab, leave the default values in all fields.

9 Click OK in the Configure Informatica Service dialog box to complete the configuration of the Informatica Server service.

NOTE: You cannot start the Informatica Server until you have completed the procedures in the section "Creating, Restoring, and Configuring the Informatica Repository" on page 82.

Setting Environment Variables on Windows

This section includes instructions for setting the following environment variables:

NOTE: You need to reboot your machine after setting the environment variables for the change to take effect.

Setting the PATH Environment Variable for the Informatica Server

The DAC server uses pmcmd to run the Informatica workflows on the Informatica servers. This requires the path to the bin directory for the Informatica server to be added to the Windows system PATH environment variable.

To set the PATH environment variable for the Informatica server

- In the Windows System Properties window, add to the PATH environment variable the path for the Informatica Server bin directory.

Setting the SiebelUnicodeDB Environment Variable on Windows

If you are using the Unicode character data mode, you need to set the SiebelUnicodeDB environment variable.

To set SiebelUnicodeDB for Windows

1. In Windows, click Start > Run.
2. In the Open field, enter `regedit`.
3. Navigate to HKEY_LOCAL_MACHINE > SYSTEM > CurrentControlSet > Services > PowerMart > Parameters > Configuration.
4. On the right window panel, right-click and select New > String Value.
5. Rename the new string value `SiebelUnicodeDB`.
6. Double-click SiebelUnicodeDB.
7. In the Value data field, enter your [user_OLTP]@[connectString_OLTP] [user_OLAP]@[ConnectString_OLAP].

   For example, siebel@db204007 siebel@db204008.

   - For Oracle and DB2, leave a space between siebel@db204007 and siebel@db204008.

- For MSSQL, use the system DNS name for [connectString_OLTP] and [connectString_OLAP].

NOTE: You must enter the user names and connect strings in the same case as you used in the Workflow Manager > Connections > Relational settings.

Setting the Bulk Load Registry for DB2/390 Databases

DB2/390 databases do not support bulk loading; therefore, you need to disable bulk loading in the Windows registry.

To disable bulk loading for DB2/390 databases

- For DB2/390 databases, enter `DisableDB2BulkMode = Yes` in the Windows registry.

Copying Database Parameter Files

Follow this procedure to copy the database parameter files to the Informatica server.

- Go to the appropriate source database directory in SiebelAnalytics\dwrep\Informatica\ParameterFiles and copy the parameterfileOLTP.txt to the SrcFiles directory on the Informatica Server machine.
- Go to the appropriate target database directory in SiebelAnalytics\dwrep\Informatica\ParameterFiles and copy the parameterfileDW.txt to the SrcFiles directory on the Informatica Server machine.
- For Siebel eBusiness and Industry Applications, go to SiebelAnalytics\dwrep\Informatica\ParameterFiles and copy ApplicationVariables.csv and Day_FiscalCalendar.csv to the SrcFiles directory on the Informatica Server machine.
- For Siebel eBusiness and Industry Applications, go to the appropriate source database directory in SiebelAnalytics\dwrep\Informatica\ParameterFiles and copy parameterfileOLTP_UA.txt to the SrcFiles directory on the Informatica Server machine.
- For Siebel eBusiness and Industry Applications, go to the appropriate target database directory in SiebelAnalytics\dwrep\Informatica\ParameterFiles and copy parameterfileDW_UA.txt to the SrcFiles directory on the Informatica Server machine.
- For Siebel Industry Applications, go the appropriate source database directory in SiebelAnalytics\dwrep\Informatica\ParameterFiles and copy both parameterfileOLTP.txt and sia_parameterfileOLTP.txt to the SrcFiles directory on the Informatica Server machine.
- For Siebel Industry Applications, go the appropriate target database directory in SiebelAnalytics\dwrep\Informatica\ParameterFiles and copy both parameterfileDW.txt and sia_parameterfileDW.txt to the SrcFiles directory on the Informatica Server machine.
- For Siebel Industry Applications, go to the appropriate source database directory in SiebelAnalytics\dwrep\Informatica\ParameterFiles and copy both parameterfileOLTP_UA.txt and sia_parameterfileOLTP_UA.txt to the SrcFiles directory on the Informatica Server machine.
- For Siebel Industry Applications, go to the appropriate target database directory in SiebelAnalytics\dwrep\Informatica\ParameterFiles and copy both parameterfileDW_UA.txt and sia_parameterfileDW_UA.txt to the SrcFiles directory on the Informatica Server machine.

Copying Teradata Loader Control Files

If you are using a Teradata database, follow this procedure to copy the loader control files to the Informatica Server.

To copy Teradata loader control files

1. Navigate to the directory SiebelAnalytics\dwrep\Informatica\teradata_loader_control_files or teradata_loader_control_files_UNIX.
2. Copy all of the control files in the directory to the TgtFiles directory under the Informatica Server directory.
3. Modify the control files to use the appropriate database, login, password, table owner, and directory path for the input data files and loader logs.
4. Change the permissions setting on the control files to read only.

Creating, Restoring, and Configuring the Informatica Repository

This section contains instructions for creating, restoring, and configuring the Informatica repository. It also includes instructions for dropping the Informatica repository, to be used when you want to drop and then restore the repository.

The following topics are included:

Setting Up the Informatica Repository Server

After installing the Informatica Repository Server, you must perform the set up step before you can start it.

To set up the Informatica Repository Server

1 Navigate to Programs > Informatica PowerCenter 7.1.2 > Informatica PowerCenter 7.1.2 OEM for Siebel - RepServer > Informatica Repository Server Setup.

2 In the Configure Repository Server dialog box, enter the following information:

Field	Description
Server Port Number	**NOTE:** The port number the Repository Server uses to connect to repository client applications. By default, this value is set to 5001. It is recommended that you use the default value. This value must match the value you entered in "Setting Up the Informatica Server" on page 78.
Administrator Password	The password you use to connect to the Repository Server from the Repository Manager. It is recommended that you use Administrator as the password. **NOTE:** This value must match the value you entered in "Setting Up the Informatica Server" on page 78.

3 Leave the default values in the remaining fields, and click OK.

Starting the Informatica Repository Server

Follow this procedure to start the Informatica Repository Server.

To start the Informatica Repository Server

1 Navigate to Start > Settings > Control Panel > Administrative Tools > Services.

2 Double-click Informatica Repository Server.

3 In the Informatica Repository Server Properties dialog box, click Start.

Creating or Restoring the Informatica Repository

If you are using an MSSQL, Oracle, or DB2 database, follow this procedure to create or restore the Informatica repository using the DAC.

If you are using a Teradata database, you need to create or restore the Informatica repository using the Informatica Repository Manager. You cannot create or restore the Informatica repository using the DAC. For instructions, see "To create or restore the Informatica repository for Teradata installations" on page 85.

NOTE: Before you start this procedure, you need to create a database to contain the Informatica Repository. This database can be the same as the data warehouse database.

To create or restore the Informatica repository

1 From the DAC menu bar, click Tools > ETL Management > Configure.

2 Select the database platforms for the following:

- Target data warehouse
- Source transactional database
- Informatica repository

3 In the Data Warehouse Configuration wizard, select the Create Informatica Repository check box, and then click Next.

The Informatica Repository tab is active.

4 Enter the following information:

Field	Description
Database Type	Type of database.
Table Owner	Valid database user.
Database Connection String	If you are using... ■ Oracle (OCI8) use the tnsnames entry. ■ Oracle (Thin) use instance name. ■ SQL Server use the database name. ■ DB2-UDB/DB2-390 use the connect string as defined in the DB2 configuration. ■ Teradata use the database name.
Host	Applicable only to SQL Server. Machine host name where the server resides.
Database Table Owner	Name of table owner.
Database Password	Password for table owner.
MSSQL Database Name	Applicable to SQL Server only. The database name where you want to restore the repository.
Informatica Administrator Name/Password	Not applicable while restoring the repository. Used only to configure the repository. **NOTE:** These values should match the values you entered in "Setting Up the Informatica Server" on page 78.
Repository Server	Name of machine hosting the Informatica Repository Server. **NOTE:** This value should match the value you entered in "Setting Up the Informatica Server" on page 78.

Field	Description
Repository Server Port	Port number used by Informatica Repository Server to listen to requests. **NOTE:** This value should match the value you entered in "Setting Up the Informatica Server" on page 78.
Email Address	Not applicable while restoring the repository. Used only to configure the repository.

To create or restore the Informatica repository for Teradata installations

1. Launch the Informatica Repository Manager.
2. Register the Informatica Repository Server.
   a. In the left pane, select the Informatica Repository Servers node, which appears under Console Root.
   b. From the menu bar, select Action > New Server Registration.
   c. Enter the host name (the machine where the repository resides).
   d. Accept the default port number 5001 or enter the appropriate port number. Click OK.

      The Repository Server host name appears in the right pane under Hostname.
   e. Right-click the server name and choose Connect.
   f. In the Connection Information section, enter the Administrator password. Click OK.

      These values should match the values you entered in "Setting Up the Informatica Server" on page 78.
3. In the left pane, select the Repositories folder, which appears under the Repository Server name.
4. In the menu bar, click Action > Add Repository.
5. Add the repository to the cache.
   a. Enter the Database Type.
   b. Enter the Repository Name. This value should match the value you entered in "Setting Up the Informatica Server" on page 78.
   c. Enter the Connect String Name.
   d. Enter the Code Page based on the operating system platform.
   e. Enter a User and Password. These values should match the values you entered in "Setting Up the Informatica Server" on page 78. Click OK to save the information.

      A message box states the repository has been added to the Repository Server's cache, and asks if you want to create a repository using this connection information.
6. In the message box, click No.

   The repository will be restored from the .rpd file.

7 Copy the file Siebel_DW_Rep_Teradata.rep from the directory \SiebelAnalytics\dwrep\Informatica\Repository to the directory \Informatica Repository Server\bin\Backup.

For UNIX installations, copy the file to the directory /Informatica Repository Server/Backup.

NOTE: You should now proceed to the procedure "To configure the Informatica repository in the Workflow Manager for Teradata installations" on page 90.

You do not need to perform the procedures "Registering the Informatica Repository Server in Repository Server Administration Console" on page 86 or "Pointing to the Informatica Repository" on page 87.

Dropping the Informatica Repository (Optional)

This section includes instructions for dropping the Informatica repository. After you drop the Informatica repository, you can restore it by following the procedure "Creating or Restoring the Informatica Repository" on page 83.

To drop the Informatica repository

1 From the DAC menu bar, click Tools > ETL Management > Configure.

2 Select the Drop Informatica Repository check box.

3 In the Informatica Repository tab, enter the appropriate information, and then click Start.

Registering the Informatica Repository Server in Repository Server Administration Console

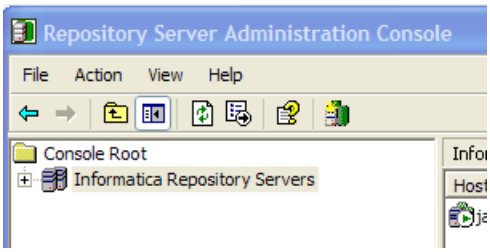
Follow this procedure to register the Informatica Repository Server in the Informatica Repository Server Administration Console.

NOTE: Make sure the Informatica Repository Server is running before you begin this procedure.

To register the Informatica Repository Server

1 Launch the Repository Server Administration Console.

2 In the left pane, select the Informatica Repository Servers node, which appears under Console Root.



3 From the menu bar, select Action > New Server Registration.

a Enter the host name (the machine where the repository resides).

b Accept the default port number 5001 or enter the appropriate port number. Click OK.

The Repository Server host name appears in the right pane under Hostname.

c Right-click the server name and choose Connect.

d In the Connection Information section, enter the Administrator password. Click OK.

These values should match the values you entered in "Setting Up the Informatica Server" on page 78.

Pointing to the Informatica Repository

Follow this procedure to point to the Informatica repository using the Repository Server Administration Console.

To point to the Informatica repository

1 Launch the Repository Server Administration Console.

2 In the left pane, select the Repositories node.

3 From the menu bar, select Action > New Repository.

The New Repository dialog box appears.

4 In the General tab, enter the following information:

Field	Description
Repository Name	The name for the Informatica repository. This value should match the value you entered in "Setting Up the Informatica Server" on page 78.
Creation Mode	Select "Do not create any content. A repository already exists under the specified database connection."

5 In the Database Connection tab, enter the following information:

Field	Description
DatabaseType	The type of database storing the repository.
CodePage	The code page based on the operating system platform.
ConnectString	The native connect string the Repository Server uses to access the database containing the repository. For MSSQL Server databases, use <SERVERNAME@DATABASENAME>.
DBUser	The account used to access the repository. This value should match the value you entered in "Setting Up the Informatica Server" on page 78.
DBPassword	The password for the repository user. This value should match the value you entered in "Setting Up the Informatica Server" on page 78.

6 Click the Licenses tab.

You may see a message stating the license file is empty.

7 Click OK to close the message dialog box.

8 In the License Key field, enter the Product license key, and then click Update.

The license key information appears in the lower window of the Licenses tab.

9 Repeat the step above for the Option and Connectivity license keys.

10 Click OK to save the configuration options.

11 Select the repository, right-click and then click Start to start the repository.

NOTE: The Informatica repository tables need to be analyzed for better performance. This procedure must be done manually by the database administrator.

Configuring the Informatica Repository in the DAC

Follow this procedure to configure the Informatica repository in the DAC's Data Warehouse Configurator.

If your source or target database is on SQL Server and you have non-ASCII data, you must manually create database connections in the Informatica repository and replace the connections. For instructions, see "To replace database connections for SQL Server databases that have non-ASCII data" on page 89.

If you are using a Teradata database, you need to configure the source and data warehouse database relational connections in the Informatica Workflow Manager. You cannot use the DAC's Data Warehouse Configurator to configure relational connections. For instructions, see "To configure the Informatica repository in the Workflow Manager for Teradata installations" on page 90.

To configure the Informatica repository in the Data Warehouse Configurator

1. From the DAC menu bar, click Tools > ETL Management > Configure.
2. Select the database platform for the target, source, and Informatica repository databases.
3. In the Data Warehouse Configurator wizard, check the Configure Informatica Repository check box, and click Next.
4. Enter the appropriate information in the Data Warehouse, Transactional Database, and Informatica Repository tabs, and then click Start.

NOTE: The Informatica repository cannot be configured using the Data Warehouse Configurator if the source or target database is Unicode. Therefore, make sure the Is Unicode check box is not checked in the Data Warehouse tab and in the Transactional Database tab of the Data Warehouse Configurator. You then need to configure the Informatica repository in the Informatica Workflow Manager to specify the source and target databases as Unicode. For instructions, see "To set the database connection code page in Workflow Manager" on page 90.

To replace database connections for SQL Server databases that have non-ASCII data

1. Launch the Informatica Workflow Manager.
2. Connect to the repository.
3. Select Connections >Relational.

   A new window appears.
4. Click Add, and then select ODBC.
5. If your source database is on SQL Server, create a connection using the following values:

Field	Value
Name	ODBC_OLTP
Code Page	UTF-8 encoding of Unicode

6. If your target database is on SQL Server, create a connection using the following values:

Field	Value
Name	ODBC_OLAP
Code Page	UTF-8 encoding of Unicode

   The connections you created appear in the Replace Connections list.

7 Select Connections > Replace.

8 Click Add a New Connection Pair.

9 If your source database is SQL Server, enter `MSSQL_OLTP` in the From field, and `ODBC_OLTP` in the To field.

10 If your target database is SQL Server, click Add a New Connection Pair, and enter `MSSQL_OLAP` in the From field, and `ODBC_OLAP` in the To field.

11 Click Replace.

The database connection code page also has to be set manually through the Informatica Workflow Manager for Unicode/Non-Latin code pages.

To set the database connection code page in Workflow Manager

1 Launch the Informatica WorkFlow Manager.

2 Connect to the repository.

3 Select Connections > Relational.

The Relational Connection Browser window appears.

4 Configure the code page for the source connection:

a Double-click the appropriate OLTP connection (DB2_OLTP for DB2, ORA_OLTP for Oracle, ODBC_OLTP for MSSQL).

The Connection Object Definition dialog box opens.

b For the Code Page, select UTF-8 encoding of Unicode or the appropriate code page, and click OK.

5 Configure the code page for the target connection:

a Double-click the appropriate OLAP connection (DB2_OLAP for DB2, ORA_OLAP for Oracle, ODBC_OLAP for MSSQL).

b For the Code Page, select UTF-8 encoding of Unicode or the appropriate code page, and click OK.

To configure the Informatica repository in the Workflow Manager for Teradata installations

1 Launch the Informatica Workflow Manager.

2 In the Repositories list, select the Informatica repository. The recommended repository name is Siebel_DW_Rep_Teradata for Teradata databases. This value should match the value you entered in "Setting Up the Informatica Server" on page 78.

3 In the menu bar, select Repository > Connect.

4 Log in using the username and password you entered in "Setting Up the Informatica Server" on page 78. Click Connect.

5 Configure the relational connections.

a In the menu bar, select Connections > Relational.

b In the Objects window, select Olap_Teradata, and then click Edit.

c Edit the User Name, Password, Database Name, and Data Source Name, and then click OK.

d If you cannot use the existing source connection (the default source connection is for an Oracle database), you need to create a new relational connection and then use the Replace Connections functionality (Connections > Replace) to replace the relational database connection.

e Click Close to close the Relational Connection Browser.

6 Configure the external loader connection.

a In the menu bar, select Connections > Loader.

b In the Objects window, select Teradata_Tpump_Upsert, and then click Edit.

c Edit the User Name, Password, TDPID, Database Name, and other attributes as necessary.

d Repeat these steps for the following objects:

- Teradata_Tpump_Update
- Teradata_Tpump_Insert
- Teradata_Tpump_Delete
- Teradata_FastLoad

e For Unicode environments, append `-c UTF8` to the value for the External Loader Executable attribute for each external loader.

For example:

```
fastload -c UTF8
```

```
tpump -c UTF8
```

f Click Close to close the Relational Connection Browser.

Configuring the Informatica Repository in Workflow Manager

Follow this procedure to configure the Informatica repository in the Informatica Workflow Manager.

To configure the Informatica repository in Workflow Manager

1 Launch the Informatica Workflow Manager.

2 In the Repositories list, select the Informatica repository. The recommended repository name is Siebel_DW_Rep (Siebel_DW_Rep_Teradata for Teradata databases). This value should match the value you entered in "Setting Up the Informatica Server" on page 78.

3 In the menu bar, select Repository > Connect.

The Connect to Repository window opens.

4 Log in using the username and password you entered in "Setting Up the Informatica Server" on page 78. Click Connect.

5 Once connected, double-click the icon of the server name (SIEBEL_DW_SERVER is the recommended server name) under the repository icon.

The Server dialog box opens.

6 In the Host Name field, enter the computer name (as shown in the Network control panel) of the machine running the Informatica Server.

7 Optionally, configure the Code Page setting.

- If the Informatica Server uses ENU Windows, leave the setting unchanged. For localized environments, choose the appropriate setting.

8 Click Advanced to expose the Server Variables list.

9 Make sure the $PMRootDir variable points to the directory where the Informatica Server is installed.

NOTE: If you install the Informatica Server on a machine different from the recommended configuration, make sure to update the $PMRootDir variable with the new location.

10 Close the Server window, and exit from Workflow Manager.

Setting the Code Page File Property for Unicode Environments on Teradata Databases

If you have a Unicode environment on a Teradata database, you need to set the code page file property of sessions that use the Teradata external loaders to use the code page UTF8. You need to follow this procedure to set the code page file property for each session that uses a Teradata external loader.

To set the code page file property for a Unicode environment on a Teradata database

1 Launch the Informatica Workflow Manager.

2 Open the appropriate session.

3 In the Edit Tasks dialog box, select the Targets tab.

4 Click Set File Properties.

5 In the Flat Files - Targets dialog box, in the File Properties area, click Advanced.

6 In the Fixed Width Properties - Targets dialog box, in the Code Page area, select UTF-8 encoding of Unicode.

Setting Up the HOSTS File for Teradata Installations on Windows

If you are using a Teradata database, you need to set the TDPID parameter on the machine where the Informatica Server is installed by adding an entry in the HOSTS file.

To set up the HOSTS file for Teradata installations

1. On the machine where the Informatica Server is installed, go to the \<SystemRoot>\system32\drivers\etc directory and open the HOSTS file.
2. In the HOSTS file, enter a line in the following format:

   `<IP address of remote server> <remote server> <remote server>COP<n>`

   For example:

   172.20.176.208 tdatsvr tdatsvrCOP1

   where `tdatsvrCOP1` is the alias for the remote server. The alias must begin with an alphabetic string and end with the `COP n` suffix, where n is a number between 1 and the total number of applications processors that are associated with the Teradata communications processor.
3. Save the HOSTS file.

For more information about setting the TDPID parameter, see the Teradata documentation.

Verifying the Informatica Repository Configuration

Follow this procedure to verify the Informatica repository configuration.

To verify the Informatica repository configuration

1. Launch the Informatica Workflow Manager.
2. Double-click the SIEBEL_DW_SERVER icon that appears under the Siebel_DW_Rep icon.

   The Server dialog box opens.
3. Click Advanced to expose the Server Variables list.
4. Verify the following settings:
   - Host Name
   - Port
   - Code Page
   - $PMRootDir
5. Close the Server dialog box.

6 In the menu bar, select Connections > Relational.

The Relational Connection Browser window opens.

7 Verify the OLAP and OLTP connection settings.

Starting the Informatica Server

At this point in the installation and configuration process, you can start the Informatica Server. It should now be able to communicate with the Informatica Repository Server and the Informatica repository.

To start the Informatica Server

1 Navigate to Start > Settings > Control Panel > Administrative Tools > Services.

2 Double-click Informatica.

The Informatica Properties dialog box opens.

3 In the General tab, in the Startup Type drop-down list, select Automatic.

4 In the Log On tab, if you plan on using email notification, enter the appropriate username and password. Select System Account if you do not plan on using email notification.

NOTE: If the Informatica Repository Manager and Workflow Manager are not pointing to the Siebel Data Warehouse, the Informatica service cannot start.

5 In the General tab, in the Server Status area, click Start.

6 Close the Services window.

Setting DAC ETL Preferences

Follow this procedure to set the appropriate ETL parameters in the DAC.

To set the DAC ETL Preferences

1 Launch the DAC client.

2 From the DAC menu bar, choose Tools > ETL Management Preferences.

The available parameters are described in Table 24.

Table 24. ETL Preferences

Parameter	Description
Analysis Start Date	The start date used to build the day dimension and to flatten exchange rates and costs lists.
Analysis End Date	The end date used to build the day dimension and to flatten exchange rates and costs lists.
Prune Days	The LAST_UPD column in Siebel transactional database tables is used for incremental change capture. This timestamp reflects the actual event time. It is therefore possible for a data row to be committed to the transactional database with a LAST_UPD date that is older than the date on which the last refresh was executed. This will cause the data row to be missed in the subsequent extract (if based purely on LAST_UPD date). However, the LAST_UPD date column still provides an opportunity to improve the change capture process by overlapping the extraction date window by the number of days set in this parameter. The records extracted in the overlapped window are filtered by comparing this information with information in the Image table. The Prune Days setting ensures that the rows that had values in LAST_UPD older than values in LAST_REFRESH_DATE are not missed. This is a parameter that can be set based on experience with processes, such as remote sync, that potentially can cause records to be missed. This parameter cannot be less than 1. For example: Assume the table W_PERSON_D was refreshed on January 15th by querying the table S_CONTACT. And, the Prune Days setting was set to 5. The next time S_CONTACT is queried to load W_PERSON_D, the records that have a LAST_UPD value since January 10 are compared with the ROW_ID of the Image table to cover for any missing records between January 15 and January 10 (the overlap period).
Default OLTP Currency	During the ETL process, if a record's transaction currency is null, the ETL process uses the currency specified here as the default transaction currency.
DW Currency	The currency to which all financial transaction are converted during the load process.
Default Exchange Rate	The exchange rate used during a data warehouse currency conversion if certain currency exchange rates are not available.
Max Days in a Sales Stage	Applicable to opportunities only. The maximum number of days an opportunity can spend in a sales stage.

Table 24. ETL Preferences

Parameter	Description

Default Language	The language into which columns' display strings are translated.
	
	NOTE: Language-independent strings are stored in the _l columns. If you do not know what the default language setting is, issue the following query against the transactional database:
	
	`select VAL from S_SYS_PREF`
	`where SYS_PREF_CD='ETL Default Language';`

Default DW Index Space	Applicable to Oracle databases only. The indexspace where indices are created. The indexspace can be specified at the index level or at the global level (ETL Preferences dialog box). When indices are dropped and created, they are created in the default indexspace for the table if an indexspace is not specified in the indexspace property of the index itself. If you do not want to specify indexspaces for every index, you can specify a default indexspace in the ETL Preferences dialog box. **NOTE:** This indexspace is used only when indices are dropped and created. After the indices are created, changing the value does not move them over.
Alignment Version	Applicable to vertical applications only. Seed data needs to be updated in certain lists of values to make sure that the ETL process correctly populates the alignment type and product category relationships in the Siebel Data Warehouse. Remove the check from this check box if you want to make changes to the existing alignment. Check this check box if you want to create a new alignment version. This setting is used by Siebel Life Sciences to control loading of changes in assignment rules in Siebel Assignment Manager.

Configuring the DAC Server

This section provides the procedures for configuring the connection between the DAC server and the DAC metadata repository and for setting up email recipients. You must perform this task on the machine hosting the DAC server. There should be one installation of the DAC server for each DAC repository.

This section includes the following procedures:

- Configuring the Connection between the DAC Server and DAC Repository on page 97
- Configuring Email in the DAC Server (Optional) on page 97

Configuring the Connection between the DAC Server and DAC Repository

Follow this procedure to configure the connection between the DAC server and the DAC repository.

To configure the connection between the DAC server and the DAC repository

1. Launch the DAC client.
2. From the menu bar, click Tools > DAC Server Management > DAC Server Setup.

   A confirmation dialog box asks you to confirm that you want to configure the DAC repository connection.
3. Click Yes.

   The Server Configuration dialog box appears.
4. In the Repository Connection Information tab, enter the appropriate information.
5. Click Test Connection to make sure the connection works.

Configuring Email in the DAC Server (Optional)

Follow this procedure to configure the email administrator account in the DAC.

The DAC server has a built-in login-authentication based email (SMTP) client, so it talks to any SMTP login-authenticating server to send emails. You can set up the DAC server to notify an administrator by email about the status of various on-going activities, such as the status of an ETL execution.

NOTE: For the email functionality to work, you must be using an SMTP server in which the SMTP authentication mode LOGIN is enabled. For example, if you are using Microsoft Exchange Server, you must enable the SMTP module and enable authentication in the Basic Mode. The SMTP server outbound email authentication must be turned on for the DAC server to be able to send email notifications.

To configure the email administrator account in the DAC server

1. Launch the DAC client.
2. From the menu bar, click Tools > DAC Server Management DAC Server Setup.

   A confirmation dialog box asks you to confirm that you want to configure the DAC repository connection.
3. Click Yes.

   The Server Configuration dialog box appears.
4. In the Email Configuration tab, enter the appropriate information, and click Save.

Once the configuration has been completed, you can start the DAC server.

Updating Siebel Transactional Database Schema Definitions

If you are installing Siebel Analytics Applications for the first time or if you are upgrading to Siebel Analytics Applications version 7.8.2, you need to update the schema definitions in the Siebel transactional database (OLTP).

If you are using a Siebel Analytics Bridge application and need to update the schema definitions in the Siebel transactional database, see Appendix A, "Siebel Analytics Bridge Applications."

NOTE: If your Siebel transactional database is version 7.5.3, and you are upgrading to Siebel Analytics 7.7 or later, use the performance-related .sif file provided in *Technical Note 406*, located on SupportWeb.

The process of updating schema definitions in the Siebel transactional database consists of the following tasks:

Backing Up the Siebel Repository

You need to back up the repository regardless of whether the SIF file is being applied to a development or to a production environment.

To back up a repository

1. Open a DOS window and navigate to the siebsrvr\bin directory.
2. Run the command siebenv to set the Siebel environment variables.
3. Export the repository to a .dat file:

   ```
   repimexp /a e /u username /p password /c odbc /d table_owner /r repository /f
   dat_file /v y /l logfile
   ```

Importing the SIF File into the Siebel Transactional Database

Follow this procedure to import new schema definitions into the Siebel transactional database.

To import new schema definitions into the Siebel transactional database

1. Copy the file named SRW_OLTP_78.sif from SiebelAnalytics\dwrep to \tools\objects.
2. Open Siebel Tools and point to the development database.
3. Lock the projects named Table ETL, Table Nquire, and Table CCM.
4. In the menu, click Tools > Import from Archive.
5. Navigate to the directory containing the SIF files and open SRW_OLTP_78.sif.

   The Import Wizard—Preview screen appears.
6. Click Next.
7. Select the option Merge the object definition from the archive file with the definition in the repository, and click Next.

   The Import Wizard reviews all the object conflicts between what is in the repository and what is in the SIF file.
8. When the Import Wizard process is done, expand the tree under Conflicting Objects to verify the changes being applied from the SIF file to the repository. When you are done, click Next.
9. A warning appears:

   `The operation is going to modify your repository ...`

   Click Yes.
10. When the process is finished, click Finish.

    After you import the new schema into Siebel Tools, you need to apply the schema to the physical database.

To apply schema definitions to the Siebel transactional database

1. In Siebel Tools, in the Object Explorer, click on the Table tab and query for S_ETL_*.

   A list of tables appears.
2. Select all the tables in the list and click Apply.

   The Apply Schema wizard appears.
3. In the Tables drop-down menu, select Current Query. Enter the appropriate connection parameters, and click Apply.

Verifying the SIF Patch Application

After the SIF import is complete, verify the following tables were created in the Siebel transactional database.

- For Siebel Business Applications:
  - S_ETL_R_IMG_1 through S_ETL_R_IMG_88

- S_ETL_I_IMG_1 through S_ETL_I_IMG_88
- S_ETL_D_IMG_1 through S_ETL_D_IMG_88
- S_ETL_R_IMG_133 through S_ETL_R_IMG_147
- S_ETL_I_IMG_133 through S_ETL_I_IMG_147
- S_ETL_D_IMG_133 through S_ETL_D_IMG_147
- S_ETL_PARAM
- S_ETL_PRD_ATTR
- S_ETL_PRD_REL

- For Siebel Industry Applications:
  - S_ETL_R_IMG_84 through S_ETL_R_IMG_132
  - S_ETL_I_IMG_84 through S_ETL_I_IMG_132
  - S_ETL_D_IMG_84 through S_ETL_D_IMG_132
  - S_ETL_R_IMG_148 through S_ETL_R_IMG_151
  - S_ETL_I_IMG_148 through S_ETL_I_IMG_151
  - S_ETL_D_IMG_148 through S_ETL_D_IMG_151

Creating Delete Triggers

The change capture process in Siebel Data Warehouse uses delete triggers to capture deleted records. The preconfigured ETL process captures deleted records for tables W_ORG_D and W_PERSON_D. The source tables for W_ORG_D and W_PERSON_D are S_ORG_EXT, S_CONTACT, and S_PRSP_CONTACT; these tables need to have delete triggers created in the Siebel transactional database in order to track deleted records. Such records are identified by setting the ACTIVE_FLG column to D.

For vertical applications, the following additional source tables need to have delete triggers created in the Siebel transactional database: S_MDF_TXN, S_ASGN_GRP_POSTN, S_ASGN_RULE_ITEM. The preconfigured ETL process captures deleted records for W_FUND_F and W_ALIGNMT_DH.

In Siebel Data Warehouse, preconfigured visibility tables are inactivated. If your organization is going to use any of the visibility-related data warehouse tables, you need to activate them in the Tables tab of the Design view of the DAC client. If you activate visibility tables, you should also create delete triggers on the optional tables.

The preconfigured SIA Account and Contact visibility tables are activated by default for vertical applications. If your organization is not going to use any of the visibility tables, you need to inactivate them in the Tables tab of the Design view in the DAC client.

When creating delete triggers using the Data Warehouse Configurator, you have the option of including the optional tables for which you want to enable the delete triggers. You also have the option of having the trigger statements be directly executed or being written to a file, which can then be executed by a database administrator.

To create a delete trigger

1. From the DAC menu bar, click Tools > ETL Management Configure.
2. Select the Create Delete Triggers in Transaction Database check box.

   The Delete Triggers tab is active.
3. Select Create Triggers.
4. Select the transactional database where the triggers will be created.
5. (Optional) Select the Include optional triggers check box to include optional tables.
6. Click Start.

Creating Siebel Stored Procedures for DB2-UDB

The DAC uses siebstat and siebtrun stored procedures when running ETL processes. Typically, these stored procedures will be available in your Siebel transactional database; they may not be available on the data warehouse database.

To verify the existence of these procedures, follow the steps below.

- From a DB2 command prompt or control center, issue the following SQL scripts:

```
db2 => create table test_siebproc(id int);
DB20000I  The SQL command completed successfully.

db2 => insert into test_siebproc(id) values (1);
DB20000I  The SQL command completed successfully.

db2 => call siebtrun('TEST_SIEBPROC');
SQL0805N  Package "NULLID.SYSSH200" was not found.  SQLSTATE=51002
```

NOTE: If you get an error message, you do not have the Siebel procedures.

Creating stored procedures must be performed by the database administrator.

To create DB2 stored procedures

1. Refer to folder SiebelAnalytics\dwdb\siebproc. It contains compiled Siebel stored procedures for all platforms.

2 Copy the stored procedures into the appropriate directory on the DB2 server side, for example, d:\Program Files\SQLLIB\function directory for Windows.

If your DB2 environment is 32-bit, copy siebproc. If it is 64-bit, copy siebproc64 and rename it to siebproc.

This directory also contains the files siebproc.sql and sqlproc.ksh file, which are used to create the function.

Once these procedures are created, you can verify that they exist. After the test is complete, you can drop the table TEST_SIEBPROC.

Teradata-Specific Installation and Configuration Checklist (Windows)

Table 25 provides a list of Teradata-specific installation and configuration steps that are performed during the Siebel Analytics Applications installation and configuration process. You should review this list to make sure that you have performed all of the required Teradata-specific steps.

Table 25. Teradata-Specific Installation Checklist for Windows

Action	Link to Topic
Review Teradata-specific database guidelines.	Teradata-Specific Database Guidelines for Siebel Data Warehouse on page 43.
Install Teradata patch.	Installing the Teradata Patch on Windows on page 63.
Activate join indices.	Activating Join Indices for Teradata Databases on page 75.
Create or drop data warehouse tables.	Creating and Dropping Data Warehouse Tables on a Teradata Database on page 77.
Use recommended name for Informatica repository.	Setting Up the Informatica Server on page 78.
Copy Teradata loader control files.	Copying Teradata Loader Control Files on page 82.
Create or restore the Informatica repository using the Informatica Repository Manager. You cannot create or restore the Informatica repository using the DAC.	To create or restore the Informatica repository for Teradata installations on page 85.
Configure the source and data warehouse database relational connections in the Informatica Workflow Manager. You cannot use the DAC's Data Warehouse Configurator to configure relational connections.	To configure the Informatica repository in the Workflow Manager for Teradata installations on page 90.

Table 25. Teradata-Specific Installation Checklist for Windows

Action	Link to Topic
Set the Code Page File Property for Unicode Environments.	Setting the Code Page File Property for Unicode Environments on Teradata Databases on page 92.
Set up the HOSTS file.	Setting Up the HOSTS File for Teradata Installations on Windows on page 93.

6 Installing and Configuring Siebel Analytics Applications on UNIX

This chapter provides instructions for installing and configuring the Siebel Analytics Applications in UNIX environments. It contains the following main topics:

NOTE: If you are installing Siebel Analytics Applications in a UNIX environment, you must use the Unicode character data mode. For more information about Unicode, see "Deploying Siebel Data Warehouse in Latin-1 General, Unicode and Non-English Environments" on page 44

Overview of Deploying Siebel Analytics Applications on UNIX

Before you begin the installation and configuration process on UNIX, you should review the following:

- Chapter 4, "Preinstallation Considerations for Siebel Data Warehouse"
- Chapter 5, "Installing and Configuring Siebel Analytics Applications on Windows"

Figure 9 shows the deployment configuration for a UNIX installation. Note that the Informatica client tools and the DAC client are installed on Windows only. You will complete some of the configuration tasks in the Windows environment. Table 26 lists each task in the installation and configuration process and tells you in which environment to complete the task.

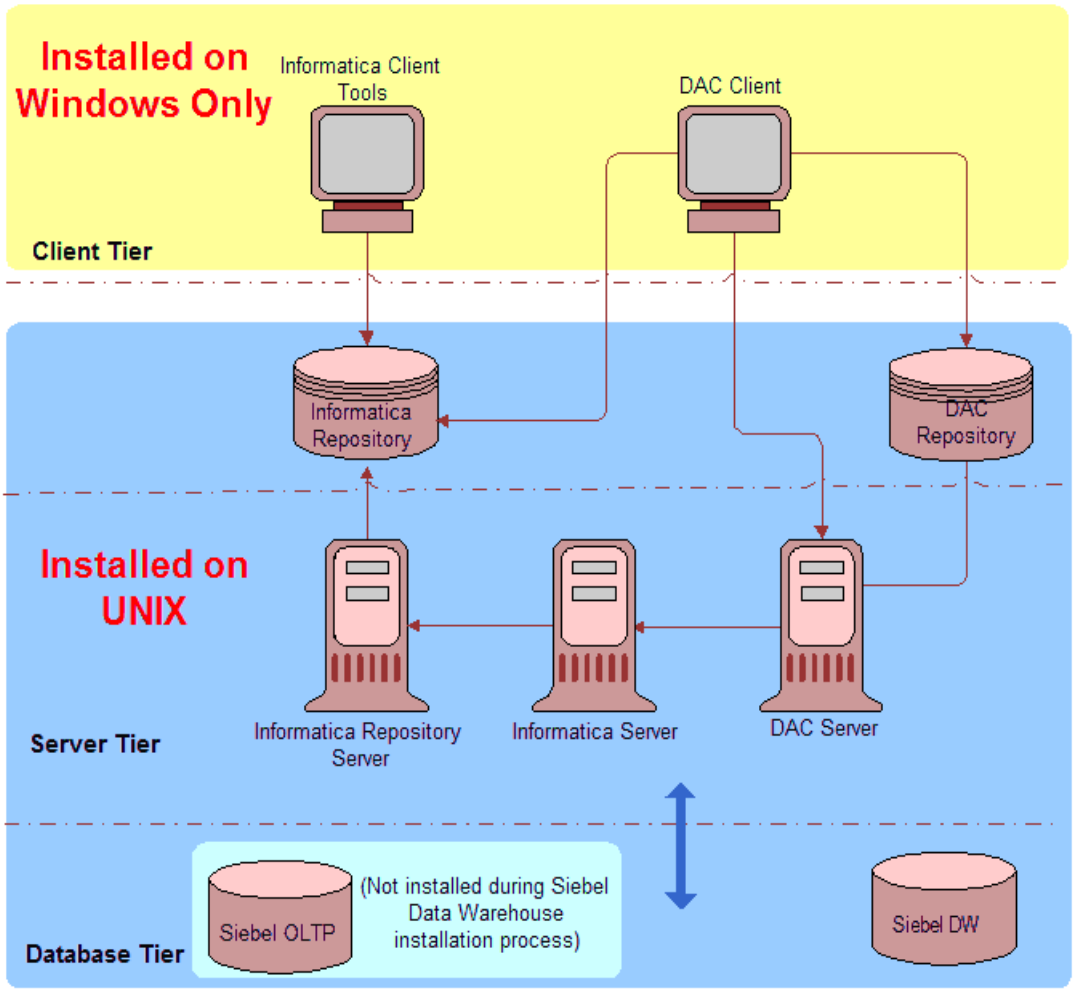


Figure 9. UNIX Installation of Siebel Data Warehouse

Siebel Analytics Applications Installation and Configuration Process on UNIX

The Siebel Analytics Applications installation and configuration process on UNIX consists of the following steps. Complete each of the following tasks in the order listed below.

Table 26. UNIX Installation and Configuration Process

Task	Instructions	Environment
1 Install Siebel Analytics Applications on Windows.	Running the Siebel Analytics Applications Installation Wizard (Windows) on page 59	Windows
2 Install Siebel Analytics Applications on UNIX.	Installing Analytics Applications on UNIX on page 110	UNIX
3 Install and Configure the Informatica Repository Server on UNIX.	Installing and Configuring the Informatica Repository Server on UNIX on page 111.	UNIX
4 Install and Configure the Informatica Server on UNIX.	Installing and Configuring the Informatica Server on UNIX on page 113.	UNIX
5 Update the Informatica license files.	Updating the Informatica License Files on UNIX on page 119.	UNIX
6 Install the Informatica patch.	Installing the Informatica Patches on UNIX on page 120	UNIX
7 For Teradata databases, install the Teradata patch.	Installing the Teradata Patch on UNIX on page 120.	UNIX
8 Set Informatica Data Code Pages Validation.	Setting Informatica Data Code Pages Validation on page 63.	Windows
9 Install the Java SDK on the machines where both the DAC client and DAC server are installed.	Installing the Java SDK on page 63.	■ Windows (DAC client) ■ UNIX (DAC server)
10 Configure the DAC config.bat file.	Configuring the DAC Config.bat File on page 64.	Windows
11 Create the DAC Metadata Repository.	Creating the DAC Metadata Repository on page 65.	Windows

Table 26. UNIX Installation and Configuration Process

Task	Instructions	Environment
12 Configure the DAC Metadata Repository.	Setting Up DAC System Properties on page 67.	Windows
	Creating ODBC Database Connections on page 72.	Windows
	Creating Transactional and Data Warehouse Database Connections on page 73	Windows
	Creating Database Connections for Teradata Installations on page 121.	UNIX
	Registering Informatica Servers in the DAC Client on page 74.	Windows
	Configuring Email Recipients in the DAC Client on page 75.	Windows
	Activating Join Indices for Teradata Databases on page 75.	Windows
13 Configure the Siebel Data Warehouse.	Creating the SSE Role on page 76.	Windows
	Creating Data Warehouse Tables on page 76.	Windows
	Creating and Dropping Data Warehouse Tables on a Teradata Database on page 77.	Windows
14 Set environment variables for the Informatica Server.	Setting Environment Variables for the Informatica Server on UNIX on page 121.	UNIX
15 Set the Windows PATH environment variable for the Informatica Server.	Setting Environment Variables on Windows on page 80.	Windows
16 Copy database parameter files.	Copying Database Parameter Files on page 81.	Windows
17 Copy Teradata Loader Control Files.	Creating, Restoring, and Configuring the Informatica Repository on page 82.	Windows

Table 26. UNIX Installation and Configuration Process

Task	Instructions	Environment
18 Start the Informatica Repository Server.	Starting the Informatica Repository Server on UNIX on page 123.	UNIX
19 Create or restore and configure the Informatica repository.	Creating or Restoring the Informatica Repository on page 83.	Windows
	Dropping the Informatica Repository (Optional) on page 86.	Windows
	Registering the Informatica Repository Server in Repository Server Administration Console on page 86.	Windows
	Pointing to the Informatica Repository on page 87.	Windows
	Configuring the Informatica Repository in the DAC on page 88.	Windows
	Configuring the Informatica Repository in Workflow Manager on page 91.	Windows
20 Set the Code Page File Property for Unicode Environments on Teradata Databases.	Setting the Code Page File Property for Unicode Environments on Teradata Databases on page 92	Windows
21 Set Up the HOSTS File for Teradata Installations.	Setting Up the HOSTS File for Teradata Installations on UNIX on page 122.	UNIX
22 Verify the Informatica Repository configuration.	Verifying the Informatica Repository Configuration on page 93.	Windows
23 Start the Informatica Server.	Starting the Informatica Server on UNIX on page 126.	UNIX
24 Set the DAC ETL Preferences.	Setting DAC ETL Preferences on page 94.	Windows
25 Configure the DAC server.	Configuring the DAC Server on UNIX on page 124.	UNIX

Table 26. UNIX Installation and Configuration Process

Task	Instructions	Environment
26 Update the Siebel transactional database schema definition.	Backing Up the Siebel Repository on page 98.	Windows
	Importing the SIF File into the Siebel Transactional Database on page 98.	Windows
	Verifying the SIF Patch Application on page 99.	Windows
	Creating Delete Triggers on page 100.	Windows

Installing Analytics Applications on UNIX

Before you install Siebel Analytics Applications on UNIX you must first complete the following installations:

- Install the Siebel Analytics platform on Windows.

  For instructions, see *Siebel Analytics Installation and Configuration Guide*.

- Install the Siebel Analytics platform on UNIX.

  For instructions, see *Siebel Analytics Installation and Configuration Guide*.

- Install Siebel Analytics Applications on Windows.

  For instructions, see "Installing and Configuring Siebel Analytics Applications on Windows" on page 55.

To install Siebel Analytics Applications on UNIX

1. Copy the folder SiebelAnalytics\dwrep\Informatica from the Siebel Analytics Applications installation directory in the Windows environment to the UNIX environment. Use the same directory structure in the UNIX environment as the Windows environment.

   This action copies pmrep, the Informatica repository file (Siebel_DW_Rep.rep), and the parameter files to the UNIX environment.

2. Copy the following files from the Windows environment into the UNIX environment. Use the same directory structure in the UNIX environment as the Windows environment.

File Type	File Name	File Location
Data warehouse schema	ddlsme.ctl	SiebelAnalytics\dwrep
SIF files	*.sif	SiebelAnalytics\dwrep

File Type	File Name	File Location
DAC repository	*.*	SiebelAnalytics\DAC\export
DAC application related	*.*	SiebelAnalytics\DAC\CustomSQLs
Delete trigger	DeleteTriggers.list	SiebelAnalytics\DAC\conf
Siebel Analytics repository	SiebelAnalytics.rpd	SiebelAnalytics\Repository
Web catalog	SiebelAnalytics.webcat	SiebelAnalyticsData\Web\Catalog
Language	*.*	SiebelAnalyticsData\Web\Res

Installing and Configuring the Informatica Repository Server on UNIX

If you are using a non-ENU operating system, make sure you have all the required locale files installed.

Install the Informatica software directly from the CD-ROM. If you cannot install the software directly from the CD-ROM, you must use FTP in binary mode; otherwise, the installation will fail. After the FTP process, use the chmod command to change the permission setting for all install files to read, write, and execute.

To install and configure the Informatica Repository Server

1. Log onto the UNIX machine.
2. Change to the appropriate directory and run the installer by typing `./install`.
3. Select the language in which you want to perform the installation.
4. Enter the Product license key.
5. Accept the license agreement.
6. Select Informatica Repository Server from the list of components available to install.
7. Specify the directory where the Repository Server will be installed.
8. Enter the appropriate values for the options listed below.

Option	Required/ Optional	Description
Configuration file name	Required	The name of the Repository Server configuration file.
ServerPort	Required	The port number the Repository Server uses for repository client connections. The default is 5001.

Option	Required / Optional	Description
AdminPasswd	Required	The administrative password used to access the Repository Server. It is recommended that you use Administrator as the password.
RaMinPort	Required	The minimum port number the Repository Server can assign to the Repository Agent process. The default is 5002.
RaMaxPort	Required	The maximum port number the Repository Server can assign to the Repository Agent process. The default is 65535.
ConfigDir	Required	The name of the directory in which the Repository Server stores repository configuration information files. You can specify either a relative path or an absolute path. The default is Config.
BackupDir	Required	The name of the directory in which the Repository Server stores repository backup files. You can specify either a relative path or an absolute path. The default is Backup.
PluginDir	Required	The name of the directory in which the Repository Server stores repository plugin files. You can specify either a relative path or an absolute path. The default is Plugin.
ErrorSeverityLevel	Required	The level of error messages written to the Repository Server log. Specify one of the following message levels: ■ Error. Writes ERROR code messages to the log. ■ Warning. Writes WARNING and ERROR code messages to the log. ■ Information. Writes INFO, WARNING, and ERROR code messages to the log. ■ Tracing. Writes TRACE, INFO, WARNING, and ERROR code messages to the log. ■ Debug. Writes DEBUG, TRACE, INFO, WARNING, and ERROR code messages to the log. The default is Information
LogFileName	Required	The name of the Repository Server log file. The default is pmrepserver.log.

9 Type Y if you want to view the readme file, or N to read it later.

10 Press enter to continue to install the Informatica Server.

Do not exit from the installer. Proceed to "Installing and Configuring the Informatica Server on UNIX" on page 113.

Installing and Configuring the Informatica Server on UNIX

The instructions in this section for installing and configuring the Informatica Server are meant to be followed immediately after the Informatica Repository Server installation, without exiting the installer. If you exited the installer after installing the Repository Server, you may want to review "Installing and Configuring the Informatica Repository Server on UNIX" on page 111.

To install and configure the Informatica Server

1 Select Informatica PowerCenter Server from the list of components available to install.

2 Specify the directory where the Informatica Server will be installed.

3 Enter the appropriate values for the options listed below.

Option	Required / Optional	Description
Configuration file name	Required	The name of the Informatica Server configuration file.
ServerName	Required	The name of the Informatica Server registered with the repository. It is recommended that you use SIEBEL_DW_SERVER.
RepositoryName	Required	The repository name entered when creating or upgrading a repository. It is recommended that you use Siebel_DW_Rep.
PMUser	Required	The repository user name used to connect to the repository. It is recommended that you use Administrator
PMPassword	Required	The password corresponding to the repository user (PMUser). It is recommended that you use Administrator.
RepServerHostName	Required	The host name of the machine hosting the Repository Server.
RepServerPortNumber	Required	The port number the Repository Server uses to communicate with repository client applications.
RepServerTimeout	Required	The maximum number of seconds the Informatica Server tries to establish a connection to the Repository Server.

Option	Required / Optional	Description
LogFileName	Optional	The filename of the Informatica Server log file.
DataMovementMode	Required	Possible values are ASCII and Unicode. The default data movement mode is ASCII, which passes 7-bit ASCII character data. To pass 8-bit ASCII and multibyte character data from sources to targets, use the Unicode mode. **NOTE:** If you use the Unicode mode, you should set the ValidateDataCodePages option as No and the SessionLogInUTF8 option as Yes.
ValidateDataCodePages	Required	If set to Yes, the Informatica Server enforces data code page compatibility. If set to No, the Informatica Server lifts restrictions for source and target data code page selection, stored procedure and lookup database code page selection, and session sort order selection. **NOTE:** If you set the DataMovementMode option to Unicode, you should set the ValidateDataCodePages option as No.
SessionLogInUTF8	Required	If set to Yes, the Informatica Server writes to the session log using the UTF-8 character set. If set to No, the Informatica Server writes to the session log using the Informatica Server code page. **NOTE:** If you set the DataMovementMode option to Unicode, you should set the SessionLogInUTF8 option as Yes.
MaxSessions	Required	The maximum number of sessions stored in shared memory. This sets the maximum number of sessions the Informatica Server can run in parallel.
LMSharedMem	Required	The amount of shared memory available for use by the Informatica Server Load Manager process. For every 10 sessions in MaxSessions, you need at least 2,000,000 bytes reserved in LMSharedMem. The default is 2,000,000 bytes.
TimeStampWorkflowLogMessages	Optional	Set to Yes if you want to append a time stamp to messages written to the workflow log.
FailSessionIfMaxSessionsReached	Optional	Set to Yes to have the Informatica Server fail the session if the number of sessions already running is equal to the value configured for MaxSessions. If you set this option to No, the Informatica Server places the session in a ready queue until a session slot becomes available.

Option	Required / Optional	Description
ExportSessionLogLibName	Optional	If you want the Informatica Server to write session log messages to an external library, enter the name of the library file.
ErrorSeverityLevel	Optional	The level of error messages written to the Informatica Server log. Specify one of the following message levels: ■ Error. Writes ERROR code messages to the log. ■ Warning. Writes WARNING and ERROR code messages to the log. ■ Information. Writes INFO, WARNING, and ERROR code messages to the log. ■ Tracing. Writes TRACE, INFO, WARNING, and ERROR code messages to the log. ■ Debug. Writes DEBUG, TRACE, INFO, WARNING, and ERROR code messages to the log.
DateHandling40Compatibility	Optional	Set to Yes if you want the Informatica Server to handle dates as in PowerCenter 1.0/PowerMart 4.0. Set to No if you want the Informatica Server to handle dates as defined in the current version of PowerCenter. Date handling significantly improved in PowerCenter 1.5 and PowerMart 4.5. If you need to revert to PowerCenter 1.0 or PowerMart 4.0 behavior, you can configure the Informatica Server to handle dates as in PowerCenter 1.0 and PowerMart 4.0.
PMServer3XCompatibility	Optional	Set to Yes if you want the Informatica Server to handle dates as in PowerCenter 1.0/PowerMart 4.0. Set to No if you want the Informatica Server to handle dates as defined in the current version of PowerCenter. Date handling significantly improved in PowerCenter 1.5 and PowerMart 4.5. If you need to revert to PowerCenter 1.0 or PowerMart 4.0 behavior, you can configure the Informatica Server to handle dates as in PowerCenter 1.0 and PowerMart 4.0.
AggregateTreatNullAsZero	Optional	Set to 1 to have the Informatica Server treat nulls as zero in Aggregator transformations. Set to 0 to have the Informatica Server treat nulls as nulls in aggregate calculations.

Option	Required / Optional	Description
AggregateTreatRowAsInsert	Optional	Set to 1 to have the Informatica Server perform aggregate calculations before flagging records for insert, update, delete, or reject in Update Strategy expressions. Set to 0 to have the Informatica Server perform aggregate calculations based on the Update Strategy transformation.
DateDisplayFormat	Required	If specified, the Informatica Server validates the date display format and uses it in session log and server log entries. If the date display format is invalid, the Informatica Server uses the default date display format. The default date display format is DY MON DD HH 24:MI:SS YYYY.
JoinerSourceOrder6xCompatibility	Optional	If set to Yes, the Informatica Server processes master and detail pipelines sequentially as it did in versions prior to 7.0. The Informatica Server processes all data from the master pipeline before starting to process the detail pipeline. Also, if you enable this option, you cannot specify the Transaction level transformation scope for Joiner transformations. If set to No, the Informatica Server processes the master and detail pipelines concurrently.
TreatNullInComparisonOperatorAs	Required	Determines how the Informatica Server evaluates null values in comparison operations. Specify one of the following options: ■ Null. The Informatica Server evaluates null values as null in comparison expressions. If either operand is null, the result is null. This is the default behavior. ■ High. The Informatica Server evaluates null values as greater than non-null values in comparison expressions. If both operands are null, the Informatica Server evaluates them as equal. When you choose High, comparison expressions never result in null. ■ Low. The Informatica Server evaluates null values as less than nonnull values in comparison expressions. If both operands are null, the Informatica Server treats them as equal. When you choose Low, comparison expressions never result in null.

Option	Required / Optional	Description
WriterWaitTimeOut	Optional	In target-based commit mode, the amount of time in seconds the writer remains idle before it issues a commit when the following conditions are true: ■ The Informatica Server has written data to the target. ■ The Informatica Server has not issued a committed. The Informatica Server may commit to the target before or after the configured commit interval. Default is 60 seconds. If you configure the timeout to be 0 or a negative number, the Informatica Server defaults to 60 seconds.
LoadManagerAllowDebugging	Optional	If set to Yes, you can run the Debugger.
CreateIndicatorFiles	Optional	If set to Yes, the Informatica Server creates indicator files when you run a workflow with a flat file target.
XMLWarnDupRows	Optional	If set to Yes, the Informatica Server writes duplicate row warnings and duplicate rows for XML targets to the session log.
OutputMetaDataForFF	Optional	If you specify this option, the Informatica Server writes column headers to flat file targets. It writes the target definition port names to the flat file target in the first line, starting with the # symbol. By default, this option is disabled.
MaxLookupSPDBConnections	Optional	Allows you to specify a maximum number of connections to a lookup or stored procedure database when you start a session. If the number of connections needed exceeds this value, session threads must share connections. This can result in a performance loss. If you do not specify a value, the Informatica Server allows an unlimited number of connections to the lookup or stored procedure database. If the Informatica Server allows an unlimited number of connections, but the database user does not have permission for the number of connections required by the session, the session fails.

Option	Required / Optional	Description
MaxMSSQLConnections	Optional	Allows you to specify a maximum number of connections to a MSSQL Server database when you start a session. If the number of connections required by the session is greater than this value, the session fails.
NumOfDeadlockRetries	Optional	Allows you to specify the number of times the Informatica Server retries a target write on a database deadlock.
DeadlockSleep	Optional	Allows you to specify the number of seconds before the Informatica Server retries a target write on database deadlock. If set to 0 seconds, the Informatica Server retries the target write immediately.
LicenseFile	Required	The name of the file that contains the license keys. The default file name is pm.lic. It is recommended that you use this file name. During the installation, the product license key is automatically added to this file. After the installation, you must add any additional Option and Connectivity license keys to this file. For instructions, see "Updating the Informatica License Files on UNIX" on page 119.
JVMDllPath	Optional	The absolute path to the JVM library file.
JVMClassPath	Optional	You can set the CLASSPATH to any JAR files you need to run a session using a web service source, target, or transformation. The Informatica Server appends the values you set to the system CLASSPATH.
JVMMinMemory	Optional	Set this value to increase the minimum amount of memory for JVM to use during an Informatica Server session. The default value is 32 MB. If the session fails due to a lack of memory, you may want to increase this value.
JVMMaxMemory	Optional	Set this value to increase the maximum amount of memory for JVM to use during an Informatica Server session. The default value is 64 MB. If the session fails due to a lack of memory, you may want to increase this value.

Option	Required / Optional	Description
TrustStore	Optional	Enter the value for TrustStore using the following syntax: <path>/<filename>
ClientStore	Optional	Enter the value for ClientStore using the following syntax: <path>/<filename>
JvmOptionName[n]	Optional	You can enter names and values for up to 10 custom VM options. Use the following syntax: JvmOption[n]=-D<VMOption_prefix>=<VMOption_value> n can range from 1 to 10 and must be unique for each option.
JCEProvider	Optional	Enter the JCEProvider class name to support NTLM authentication.
HttpProxyServer	Required	Name of the HTTP proxy server.
HttpProxyPort	Required	Port number of the HTTP proxy server. This must be a number.
HttpProxyUser	Optional	Authenticated username for the HTTP proxy server.
HttpProxyPassword	Optional	Password for the authenticated user.
HttpProxyDomain	Optional	Domain for authentication.

4 When the installation is complete, proceed to "Updating the Informatica License Files on UNIX" on page 119.

Updating the Informatica License Files on UNIX

When you install the Informatica PowerCenter, you enter a product license key. The installation process adds the product license key to a license file called pm.lic. Copies of this license file are stored in the installation directories for the Informatica Server and the Repository Server. You must add to the license file any additional Option and Connectivity license keys you have.

To update the Informatica license files

1 Navigate to the Informatica Server installation directory.

2 Update the license file with a license key using the following syntax:

```
pmlic update -key <keystring_to_add>
```

For example, if the license key is AAAA-AAAA-AAAA-AAAA-AAAA, enter the following:

```
pmlic update -key AAAA-AAAA-AAAA-AAAA-AAAA
```

3 Repeat Step 2 for every license key you need to add.

4 Navigate to the Repository Server installation directory, and repeat Step 2 for every license key you need to add.

Installing the Informatica Patches on UNIX

After you install Informatica PowerCenter version 7.1.2, you need to install two Informatica patches.

To install the Informatica patches

1 Access the patch files on the installation CD-ROM.

2 Locate the folder named Patch1, and install the patch by following the instructions in the readme file.

3 Locate the folder named Patch2, and install the patch by copying the files from the appropriate operating system folder to <INFA_home>/informatica/server.

Installing the Teradata Patch on UNIX

If you are using a Teradata database and your Informatica Server is installed on a UNIX machine, you need to install a Teradata patch.

To install the Teradata patch

1 Locate the Patch_for_Teradata subdirectory in the Informatica directory on the installation CD-ROM.

2 Copy the contents from the Patch_for_Teradata\<UNIX> directory to the Informatica Server directory.

It is recommended that you first rename the existing files in the bin directory.

3 Copy the resword.txt file from the Patch_for_Teradata directory to the Informatica Server directory

Creating Database Connections for Teradata Installations

If you are using a Teradata database, follow this procedure to create database connections.

To create database connections for Teradata installations

1. Using the Teradata ODBC driver, create an ODBC connection for the Teradata database.
2. Set the following variables in the ODBC.INI file:

   ```
   DateFormat=AAA

   SessionMode=ANSI

   NoScan=Yes
   ```

   For Unicode environments, add the following:

   ```
   CharacterSet=UTF8
   ```

Setting Environment Variables for the Informatica Server on UNIX

This section includes the following procedures:

To add setenv variables to .cshrc

Add setenv variables to .cshrc. Otherwise, you have to set the environment variables every time you log into the UNIX machine before configuring, starting, or stopping the Informatica servers.

- For Solaris, in the root directory, add these command lines to .cshrc:

  ```
  setenv PATH /<INFA_home>/informatica/pm:.:${PATH}

  setenv LD_LIBRARY_PATH /<INFA_home>/informatica/pm:${LD_LIBRARY_PATH}

  setenv PATH /<INFA_home>/informatica/repositoryserver:.:${PATH}

  setenv LD_LIBRARY_PATH /<INFA_home>/informatica/
  repositoryserver:${LD_LIBRARY_PATH}
  ```

- For AIX, in the root directory, add these command lines to .cshrc:

```
setenv PATH ${PATH}:/<INFA_home>/informatica/pc:.

setenv LIBPATH ${LIBPATH}:/<INFA_home>/informatica/pc

setenv PATH ${PATH}:/<INFA_home>/informatica/repositoryserver:.

setenv LIBPATH ${LIBPATH}:/<INFA_home>/informatica/repositoryserver
```

- For HP-UX, in the root directory, add these command lines to .cshrc:

```
setenv PATH /<INFA_home>/informatica/pm:.:${PATH}

setenv SHLIB_PATH /<INFA_home>/informatica/pm:${SHLIB_PATH}

setenv PATH /<INFA_home>/informatica/repositoryserver:.:${PATH}

setenv SHLIB_PATH /<INFA_home>/informatica/repositoryserver:${SHLIB_PATH}
```

To set the code page for non_ENU operating systems on UNIX

- If you are using a non-ENU operating system, you must set the environment variable PMREPCODEPAGE appropriately. For example: `PMREPCODEPAGE=MS932`.

To set SiebelUnicodeDB on the Informatica Server if source and target databases are Unicode on UNIX

- For all UNIX platforms, if both the transactional and data warehouse data sources are Unicode, add the following line in /<INFA_home>/informatica/pm/pmserver.cfg file.

```
SiebelUnicodeDB=<OLTP_TBO>@<OLTP_connectionString>
<OLAP_TBO>@<OLAP_ConnectionString>
```

For example:

```
SiebelUnicodeDB=ora16221@sdcdl580i016_qa16b ora16222@sdcdl580i016_qa16b
```

To disable the bulk loader for DB2/390 databases

- For DB2/390 databases, add the following line in /<INFA_home>/informatica/pm/pmserver.cfg file.

```
DisableDB2BulkMode = Yes
```

Setting Up the HOSTS File for Teradata Installations on UNIX

If you are using a Teradata database, you need to set the TDPID parameter on the machine where the Informatica Server is installed by adding an entry in the HOSTS file.

To set up the HOSTS file for Teradata installations

1. On the machine where the Informatica Server is installed, go to the /etc directory and open the HOSTS file.
2. In the HOSTS file, enter a line in the following format:

   `<IP address of remote server> <remote server> <remote server>COP<n>`

   For example:

   172.20.176.208 tdatsvr tdatsvrCOP1

   where `tdatsvrCOP1` is the alias for the remote server. The alias must begin with an alphabetic string and end with the `COP n` suffix, where n is a number between 1 and the total number of applications processors that are associated with the Teradata communications processor.
3. Save the HOSTS file.

For more information about setting the TDPID parameter, see the Teradata documentation.

Starting the Informatica Repository Server on UNIX

Follow this procedure to start the Informatica Repository Server on UNIX.

To start the Informatica Repository Server

- Go to /<INFA_home>/informatica/repositoryserver and enter ./pmrepserver.

  Check the log file to see if the server started. If it started correctly, the log file will state: “Server Initialization Completed.”

Configuring the DAC Server on UNIX

Shell scripts are provided in the *.sh and *.csh formats. You must use the appropriate scripts in the appropriate shells, because their syntaxes are different. Table 27 lists the available shell scripts and their usage. These files contain comments that provide information about how to configure the scripts.

Table 27. Siebel Data Warehouse Scripts for Bash and C Shells

Script	Purpose
config.sh config.csh	Configures the environment variables for DAC_HOME and JAVA_HOME.
dacCmdLine.sh dacCmdLine.csh	Invokes DAC commands on the DAC server. See the file for usage information.
dac_env.sh dac_env.csh	Configures environment variables, including Informatica's pmcmd utility, which enables the DAC server to invoke workflows. Other variables included are NLS_LANG and ORA_HOME.
serverSetupPrompt.sh serverSetupPrompt.csh	Configures DAC metadata repository connection information and DAC administrator email account information, which enables the DAC to send email notification upon the success or failure of execution plans.
serverinfo.sh serverinfo.csh	Displays the DAC server version.
startserver.sh startserver.csh	Starts the DAC server on machines running Solaris and HP.
startserver_aix.sh startserver_aix.csh	Starts the DAC server on AIX machines.
stopserver.sh stopserver.csh	Shuts down the DAC server.

NOTE: All the files listed in Table 27 need to have read, write, and execute permissions. If the files do not have these permissions, modify them using the chmod command.

To configure the DAC server on UNIX

1 Edit the dac_env.sh or dac_env.csh file to add the path of the directories for the Informatica Server and Informatica Repository Server to the appropriate environment variable:

- For Solaris: LD_LIBRARY_PATH
- For AIX: LIBPATH

- For HP: SHLIB_PATH

For example:

```
setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:/<INFA_HOME>/informatica621/pm
```

2. If you are using an Oracle database and the OCI driver, configure the following variables and uncomment them by removing the # sign at the beginning of the line:
   - TNS_ADMIN: To set the directory for Oracle to use tnsnames.ora files
   - LD_LIBRARY_PATH (Solaris), LIBPATH (AIX), or SHLIB_PATH (HP): To set the Oracle library path.

     NOTE: Informatica does not support 64-bit libraries.
3. Edit the config.sh or config.csh file to set the JAVA_HOME directory.

   CAUTION: Do not edit any other parameter in this file.
4. Copy the database-specific .jar files to the $DAC/lib directory:
   - **Oracle.** If you are using an Oracle database (other than 8.x), find the directory where Oracle is installed. Copy the file named ojdbc14.jar in the jdbc/lib directory and paste it in the $DAC/lib directory.
     - If you are using Oracle 8.x, copy the file named classes12.zip and paste it in the $DAC/lib directory. Also, edit the ORACLELIB setting in the config.sh or config.csh file as follows:

       ```
       setenv ORACLELIB=$DAC_HOME/lib/classes12.zip
       ```
   - **DB2.** If you are using a DB2 database, find the directory where DB2 is installed. In the Java subdirectory copy the file named db2java.zip and paste it in the $DAC/lib directory.

     NOTE: If your source or target database is DB2-UDB, you also need to create Siebel stored procedures to be used during the ETL process. For instructions, see "Creating Siebel Stored Procedures for DB2-UDB" on page 101.
   - **MSSQL.** If you are using an MSSQL database, download the SQL Server JDBC drivers from Microsoft's Web site. Copy the files msbase.jar, mssqlserver.jar, and msutil.jar to the $DAC/lib directory.
5. Run serverSetupPrompt.sh or serverSetupPrompt.csh to set the repository connection information and the email settings:
   a. At the initial prompt, enter 1 to set the repository connection information.
   b. Enter 3 to test the repository connection.
   c. Enter 4 to save the information you entered.
   d. Enter 2 to set up the administrator email account.

      NOTE: For the email functionality to work, you must be using an SMTP server in which the SMTP authentication mode LOGIN is enabled. For example, if you are using Microsoft Exchange Server, you must enable the SMTP module and enable authentication in the Basic Mode. The SMTP server outbound email authentication must be turned on for the DAC server to be able to send email notifications.

e Enter 5 to exit.

6 Copy content-specific XML files from the SiebelAnalytics\DAC\CustomSQLs folder in the Windows installation to the SiebelAnalytics/DAC/CustomSQLs directory in the UNIX installation.

Starting the Informatica Server on UNIX

Follow this procedure to start the Informatica Server.

Requirements

- From the UNIX machine, make sure you can connect to the Siebel transactional database and to the data warehouse database.
- The Informatica repository must have been configured using the Informatica Workflow Manager.
  - For instructions, see "Configuring the Informatica Repository in Workflow Manager" on page 91.
- The environment variables for the Informatica Server on UNIX must be set.
- The appropriate parameter files should have been copied to the SrcFiles directory.
- For Unicode, the SiebelUnicodeDB variable must be set.

To start the Informatica Server

- Go to the Informatica Server directory, and enter ./pmserver.

  For Teradata installations, enter ./pmserver ./pmserver.cfg>./pmserver.out

  Check the log file to see if the server started. If it started correctly, the log file will state: "Server Initialization Completed."

Stopping the Informatica Server on UNIX

Follow this procedure to stop the Informatica Server on UNIX.

To stop the Informatica Server

- Enter

  ```
  ./pmcmd stopserver <repository username> <repository password> <port #>.
  ```

  For example:

```
./pmcmd stopserver Administrator Administrator 4001
```

You can check whether the Informatica Server is running by entering the command `ps -elf |grep pmserver`.

If it lists pmserver process, the server is running. If it lists grep pmserver, that is the process of the ps command.

Starting the DAC Server on UNIX

Follow this procedure to start the DAC server on UNIX. Before you can start the DAC server, you must have already configured the config.csh or config.sh and dac_env.csh or dac_env.sh files.

To start the DAC server on UNIX

- In csh-related shells, issue the following command:

  ```
  ./startserver.csh
  ```

- In bash-related shells, issue the following command:

  ```
  ./startserver.sh
  ```

To run the DAC server in the background

- In csh-related shells, issue the following command:

  ```
  nohup startserver.csh 2>&1 &
  ```

- In bash-related shells, issue the following command:

  ```
  nohup startserver.sh 2>&1 &
  ```

The nohup command allows the DAC server to run in the background on UNIX even if the DAC client is disconnected.

Stopping the DAC Server on UNIX

Follow this procedure to stop the DAC server on UNIX.

To stop the DAC server on UNIX

- In csh related shells, issue the following command:

  ```
  ./stopserver.csh
  ```

- In bash-related shells, issue the following command

```
./stopserver.sh
```

NOTE: When you execute ./stopserver.csh or ./stopserver.sh, the server will issue a warning about the shutdown request. When the server receives the request, it will shut down even if there is an ETL in progress. The statuses for the ETL run will not be set correctly. The next time the DAC server starts, it will set the status to Failed for any uncompleted run.

Teradata-Specific Installation and Configuration Checklist for UNIX

Table 28 provides a list of Teradata-specific installation and configuration steps that are performed during the Siebel Analytics Applications installation and configuration process on UNIX. You should review this list to make sure that you have performed all of the required Teradata-specific steps. You should also review "Teradata-Specific Installation and Configuration Checklist (Windows)" on page 102 for Teradata-specific information related to the Windows environment.

Table 28. Teradata-Specific Installation Checklist for UNIX

Action	Link to Topic
Review Teradata-specific database guidelines	Teradata-Specific Database Guidelines for Siebel Data Warehouse on page 43.
Install Teradata patch	Installing the Teradata Patch on UNIX on page 120.
Create database connections	Creating Database Connections for Teradata Installations on page 121.
Set up the HOSTS file	Setting Up the HOSTS File for Teradata Installations on UNIX on page 122.

7 Upgrading Siebel Analytics Applications and Data Warehouse

This chapter provides information about upgrading Siebel Analytics Applications and the Siebel Data Warehouse.

To upgrade Siebel Analytics Applications and the Siebel Data Warehouse, you need to perform the following tasks in the order listed below:

Verifying the DAC Platform Upgrade

Follow this procedure to verify the DAC platform was upgraded to the current version of Siebel Analytics.

To verify the DAC platform was upgraded to the current version of Siebel Analytics

- Verify the following files are in the appropriate folder in the Siebel Analytics installation directory. If no files appear in the folders listed below, then the DAC platform was not upgraded.

File Type	File Name	File Location
Data warehouse schema	ddlsme.ctl	SiebelAnalytics\dwrep
SIF files	*.sif	SiebelAnalytics\dwrep
DAC repository	*.*	SiebelAnalytics\DAC\export
DAC application related	*.*	SiebelAnalytics\DAC\CustomSQLs
Delete trigger	DeleteTriggers.list	SiebelAnalytics\DAC\conf
Siebel Analytics repository	SiebelAnalytics.rpd	SiebelAnalytics\Repository
Web catalog	SiebelAnalytics.webcat	SiebelAnalyticsData\Web\Catalog
Language	*.*	SiebelAnalyticsData\Web\Res

Upgrading the DAC and Informatica Repositories

NOTE: Some of the tasks in this procedure are contained in Chapter 5, "Installing and Configuring Siebel Analytics Applications on Windows." If you already performed a task when you installed the current version of Siebel Analytics Applications, you do not need to perform it again.

1. Back up your existing DAC repository.
2. Upgrade the DAC metadata to the Analytics 7.8.2 format.
   a. Navigate to the SiebelAnalytics\DAC directory.
   b. Run the script UpgradeFromAN77ToAN771.sql.
   c. Merge your previous DAC repository with the new DAC repository. For instructions, see "Comparing and Merging DAC Repositories" on page 163.
3. Restore the Informatica repository.
   a. Back up your existing Informatica repository.
   b. Restore the Informatica repository. For instructions, see "Creating or Restoring the Informatica Repository" on page 83.

   NOTE: This step assumes you have already upgraded to the current version of Informatica PowerCenter.
4. Copy your custom folder from your previous Informatica repository to the newly restored Informatica repository.
   a. Launch Informatica Repository Manager and connect to both your previous and newly restored Informatica repositories.
   b. Select the custom folder in your previous repository and copy it (Edit > Copy).
   c. Select the newly restored Informatica Repository and paste it (Edit > Paste). Rename the folder if necessary to paste it in this location.

      This action will create the custom folder and all its contents (mappings, workflows and so on).
   d. Create an individual workflow for each of the mappings in the custom folder.
   e. Register the mappings in the DAC.

Upgrading Transactional Database and Data Warehouse Schemas

Follow this procedure to upgrade the Siebel transactional database and the Siebel Data Warehouse schemas.

To upgrade the transactional database and data warehouse schemas

1 Upgrade the transactional database schema definitions. For instructions, see "Updating Siebel Transactional Database Schema Definitions" on page 98.

2 Upgrade the data warehouse schema. For instructions, see "Creating Data Warehouse Tables" on page 76.

Upgrading Data in the Siebel Data Warehouse

Follow this procedure to upgrade the data in the Siebel Data Warehouse.

To upgrade data in the Siebel Data Warehouse

1 Configure the DAC:

   a In the DAC, navigate to Setup > Database Connections.

   b In the Tables subtab, set the Refresh Date as null for the source tables S_SRC and S_SRC_DCP.

2 Execute the Upgrade 7.7 DW to 7.8 DW execution plan:

   a Navigate to the Execute view.

   b In the Execution Plans tab, select Upgrade 7.7 DW to 7.8.2 DW.

   c Click Run Now.

3 In Informatica Workflow Manager, navigate to the appropriate folder and run the upgrade workflows in the Workflows folder.

   ■ If you are upgrading from version 7.5.3 to version 7.8.2, run the workflows in the folder Upgrade_7.5.XtoINCA and then run the workflows in the folder Upgrade_INCA_to_7.8.

   ■ If you are upgrading from version 7.7. to version 7.8.2, run the workflows in the folder Upgrade_INCA_to_7.8.

   ■ If you are upgrading from version 7.7.1 to version 7.8.2, run the workflows in the folder Upgrade_7.7.1_to_7.8.

4 If the Usage Accelerator subject area is part of the data warehouse, run the Upgrade_From77_to78_for_UsageAccelerator workflow that is in the Upgrade_INCA_to_7.8 folder.

5 In the DAC, run the Complete ETL execution plan as a refresh (to populate new data warehouse entities).

To upgrade Siebel Data Warehouse for Siebel Industry Applications

1 Configure the DAC.

   a In the DAC, navigate to Setup > Database Connections.

   b In the Tables subtab, set the Refresh Date as null for the source tables S_SRC and S_SRC_DCP.

2 Execute the Upgrade 7.7 DW to 7.8 DW execution plan.

   a Navigate to the Execute view.

   b In the Execution Plans tab, select Upgrade 7.7 DW to 7.8 DW.

   c Click Run Now.

3 In Informatica Workflow Manager, navigate to the appropriate folder and run the upgrade workflows in the Workflows folder.

   - If you are upgrading from version 7.5.3 to version 7.8.2, run the workflows in the folder Upgrade_7.5.XtoINCA and then run the workflows in the folder Upgrade_INCA_to_7.8.
   - If you are upgrading from version 7.7. to version 7.8.2, run the workflows in the folder Upgrade_INCA_to_7.8.
   - If you are upgrading from version 7.7.1 to version 7.8.2, run the workflows in the folder Upgrade_7.7.1_to_7.8.

4 If the Usage Accelerator subject area is part of the data warehouse, run the workflows in the folder Upgrade_From77_to78_for_UsageAccelerator.

5 In the Informatica Workflow Manager, navigate to the folder Upgrade_INCA_to_7.7.1_2 and run the SIA_Upgrade_From7_7_to7_7_1 workflow.

6 In the DAC, run the Complete ETL execution plan as a refresh (to populate new data warehouse entities).

8 DAC Quick Start

This chapter contains the following topics:

DAC Licensing Options in Siebel Data Warehouse 7.7.1

Siebel Data Warehouse 7.7.1 includes two licensing options for the Data Warehouse Application Console (DAC).

- The DAC Lite version:
  - Is included with any Analytics application
  - Allows you to build and execute ETL plans
  - Is suitable for small data warehouses with little or no customization
- The DAC version:
  - Is an optional license
  - Is suitable for large data warehouses with multiple customizations
  - Minimizes ETL execution time by using optimization techniques

Table 29 summarizes the differences between the DAC and DAC Lite versions.

Table 29. Differences between DAC and DAC Lite

Feature	DAC Lite	DAC
Develop data warehouse applications		
Manage metadata driven dependencies and relationships	✓	✓
Provide metadata validation checks	✓	✓
Generate additional change capture DWQL scripts		✓
Directly import database and Informatica objects		✓
Additional change capture filtering		✓
Utilize high-performance execution techniques		
Task queue optimization engine		✓
Load balancing across multiple ETL servers		✓
Unlimited ETL session concurrency		✓
Automate full and refresh mode optimization rules	✓	✓
Restart from point of failure	✓	✓
Provide analysis to isolate bottlenecks		
Error monitoring and email alerting		✓
Structured ETL analysis and reporting		✓

DAC Server Features Not Available with DAC Lite

The following DAC server features are not available with the DAC Lite version of the Siebel Data Warehouse 7.7.1:

- Email
- Optimized queuing
- Change capture filtering
- Support for multiple Informatica servers.

NOTE: The DAC Lite version supports one Informatica server and five tasks running in parallel.

DAC Client Features Not Available with DAC Lite

The following DAC client features are not available with the DAC Lite version of the Siebel Data Warehouse 7.7.1:

- Output to File (right-click menu)
- Update Rows (right-click menu)
- Output Task Description (right-click menu on Task Definitions tab)
- Generate View Scripts (right-click menu on Tables tab)
- Generate Change Capture SQL (right-click menu on Tables tab)
- Import Database Tables (right-click menu on Task Definitions tab)
- Import Indices (right-click menu on Task Definitions)
- Auto Refresh (right-click menu and toolbar on Current Run tab)
- Analyze Run (right-click menu on Run History tab)
- Get Chart (right-click menu and toolbar on Current Run tab)
- Get Phase Chart (right-click menu and toolbar on Current Run tab)
- Get Graph (right-click menu and toolbar on Current Run tab)
- Import Informatica Sessions dialog box
- Server Log
- Email Configuration tab on Server Configuration dialog box
- Email Recipients tab
- Ability to edit the Priority field on the Task Definitions form
- External Program, SQL File, and Stored Procedure execution types on Task Definitions form
- New and delete buttons on Task Phases dialog box
- Ability to edit the Priority field in the Task Phases dialog box
- Multiple Informatica Server records and sessions

  NOTE: The DAC Lite version supports one Informatica Server and five sessions.

NOTE: *Siebel Analytics Applications Installation and Administration Guide, Version 7.7.1, Rev. C* and higher versions document the features of the full DAC version and not the DAC Lite version. The installation instructions in this guide for Windows apply to both the DAC and DAC Lite versions. For installation instructions on UNIX, see the latest revision of *Siebel Analytics Installation and Configuration Guide*.

Navigating the DAC User Interface

Figure 10 shows the main elements of the DAC window.

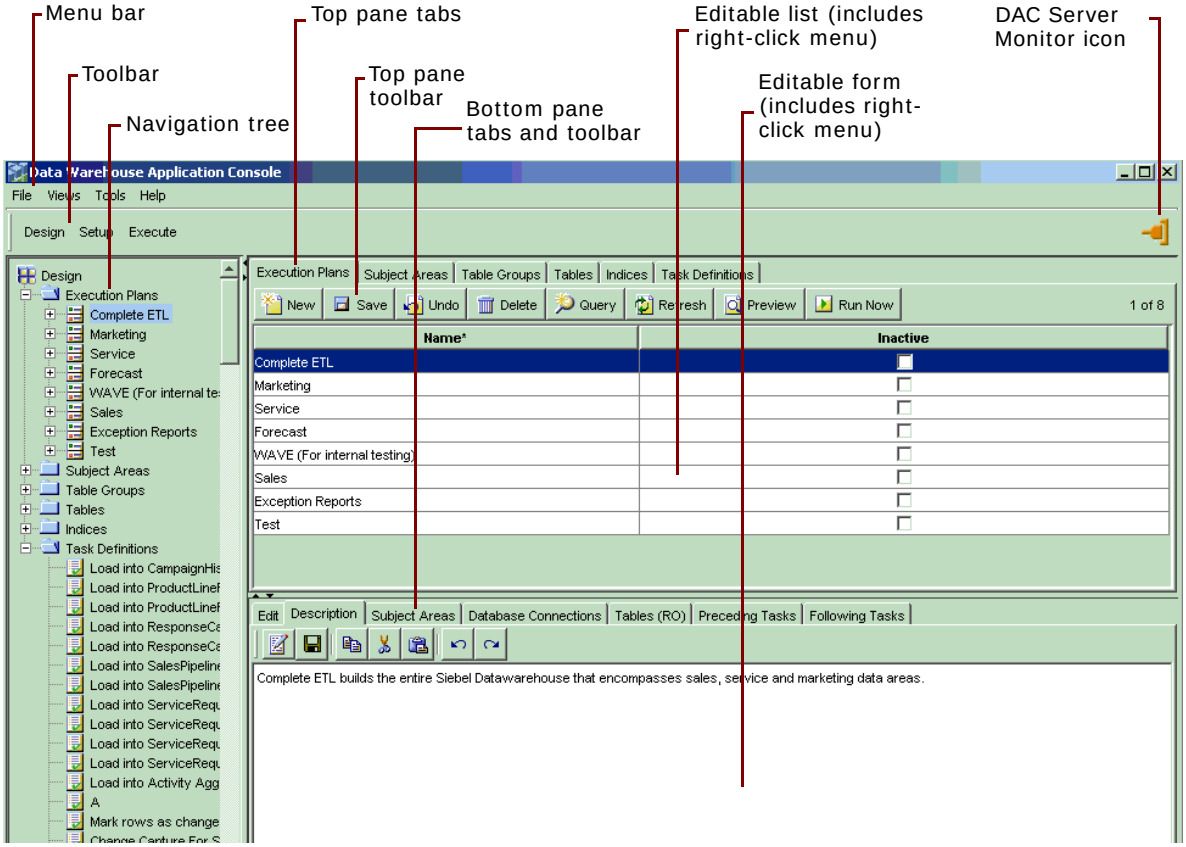


Figure 10. Example of DAC Window

Main Toolbar of the DAC

The main toolbar of the DAC is located directly under the menu bar. It provides access to the following views:

- **Design.** The Design view provides access to functionality related to creating and managing execution plans. For more information, see "About the DAC Design View" on page 173.
- **Setup.** The Setup View provides access to functionality related to setting up DAC system properties, Informatica servers, database connections, and email notification. For more information, see "About the DAC Setup View" on page 190.
- **Execute.** The Execute view provides access to functionality related to running, monitoring, and scheduling execution plans. For more information, see "About the DAC Execute View" on page 193.

DAC Toolbars and Right-Click Menu Commands

Table 30 lists commands available in some of the top and bottom pane toolbars and in the right-click menus. The commands that are available depend on the tab that is active.

Table 30. DAC Toolbar and Right-Click Menu Commands

Command	Location in DAC	Description
New	Top and bottom pane toolbars, right-click menu	Creates a placeholder for a new record in the selected list.
Save	Top and bottom pane toolbars, right-click menu	Saves the current record.
Undo	Top and bottom pane toolbars, right-click menu	Undoes changes made to the current record after the last save.
Delete	Top and bottom pane toolbars, right-click menu	Deletes the selected record. If you delete a parent record, the child records are also deleted.
Query	Top and bottom pane toolbars, right-click menu	Opens a blank query.
Refresh	Top and bottom pane toolbars, right-click menu	Retrieves the data from the repository with the last used query.
Preview	Top pane toolbar in Design view tabs: Execution Plans, Subject Areas, and Table Groups	Generates a preview of all tasks and their dependencies.
Edit	Bottom pane toolbar in Design view	Opens a dialog box in which you can add and remove objects associated with the selected repository object.
Details	Bottom pane toolbar in Execute view, Task tab	Provides a list of the task details that are executed when the selected task is run.
Run Now	Top pane toolbar in Design view, Execution Plans tab	Starts a new ETL process.

Table 30. DAC Toolbar and Right-Click Menu Commands

Command	Location in DAC	Description
Start	Top pane toolbar and right-click menu in Execute View tabs: Current Run, Run History	Restarts the selected ETL, after the ETL has failed, stopped, or been aborted.
Stop	Top pane toolbar and right-click menu in Execute View tabs: Current Run, Run History	Stops an ETL in progress. All currently running tasks will complete, and queued tasks will stop. The status of the ETL changes to Stopped.
Abort	Top pane toolbar and right-click menu in Execute View tabs: Current Run, Run History	Causes an ETL in progress to abort. All currently running tasks will be aborted. The status of queued tasks and the ETL itself will change to Stopped.
Auto Refresh	Top pane toolbar and right-click menu in Execute View tabs: Current Run, Run History	Allows you to turn on and off the automatic screen refresh functionality and set the refresh interval.
Copy String	Right-click menu	Copies the contents of a cell (editable and read-only) to the clipboard
Paste String	Right-click menu	Pastes a string from the clipboard into a selected cell that supports a string data type.
Copy Record	Right-click menu	Creates a copy of the selected record, with a unique record ID and *(2)* appended to its name. The new record is committed to the database when you click the Save button or click outside the cell. In the Design view tabs (except for the Indices tab), Copy Record copies the selected record and the record's child records. In the Design view Indices tab and Setup and Execute views, Copy Record copies only the selected record.
Output to File	Right-click menu	Outputs to a text file in the DAC root folder the contents of the current tab's record list.
Record Info	Right-click menu	Displays the record's unique ID and the timestamp for when it was last updated.

Table 30. DAC Toolbar and Right-Click Menu Commands

Command	Location in DAC	Description
Generate Index Scripts	Right-click menu on Execution Plan and Tables tabs	Generates drop index, create index, and analyze table scripts for all tables that participate in the ETL process. The results are stored in the log\scripts directory.
Generate Change Capture Scripts > Generate Image and Trigger Scripts	Right-click menu on Tables tab	Generates change capture scripts for tables with defined image suffixes. The scripts may include delete triggers, create and drop statements for delete triggers, and image tables and their indices.
Generate Change Capture Scripts > Generate View Scripts	Right-click menu on Tables tab	Generates change capture view scripts for full or incremental mode for tables that participate in the change capture process. This command can be used for unit testing.
Generate Change Capture Scripts > Generate Change Capture SQL	Right-click menu on Tables tab	Generates change capture SQL scripts for full or incremental mode for tables that participate in the change capture process. This command can be used for unit testing.
Update Rows	Right-click menu	For some columns, allows you to update the column value for each row to a single value. When you select this command, a dialog box allows you to select the column, and then a second dialog box allows you to enter a value for the column.
Import Database Tables	Right-click menu	Allows you to import table definitions from a selected database.
Import Indices	Right-click menu	Allows you to import index definitions from the database for one or more tables as listed in the result of the query.
Output Task Description	Right-click menu	Saves to an HTML file the description for a selected task definition or for all task definitions.
Run Now	Right-click menu	Starts a new ETL process.
Go To	Right-click menu	Allows you to drill down to a particular record that is associated with the selected record. Available when the Task Definitions, Indices, Tables, and Table Groups tabs are active (Design View).

Table 30. DAC Toolbar and Right-Click Menu Commands

Command	Location in DAC	Description
Mark as Completed	Right-click menu	Changes the status of a stopped or failed ETL to Completed. In the audit trail for this ETL, the status is Marked as Completed. Use this command with caution; it can cause the data warehouse to be inconsistent.
Requeue Task	Right-click menu	Changes the status of the selected, completed task and its dependents to Queued. Available only for failed ETLs.
Get Run Information > Get Log File	Right-click menu	Fetches the log file for this run from the DAC server and saves it in the ServerLog folder.
Get Run Information > Analyze Run	Right-click menu	Saves a description of the run as an HTML file in the Log/Statistics folder.
Get Run Information > Get Chart	Right-click menu	Displays a chart showing changes in task statuses over time in a separate window.
Get Run Information > Get Graph	Right-click menu	Displays a graph showing changes in task statuses over time in a separate window.

DAC Tools Menu

The Tools menu provides access to various tools you can use to manage and monitor the DAC repository, DAC server, and ETL process.

Table 31 provides a description of the tools available in the Tools menu.

Table 31. DAC Tools Menu Commands

Tools Menu Command	Description
DAC Repository Management > Export	Allows you to export all or part of the DAC metadata in order to back up the data or to reproduce the environment elsewhere. The seed data is exported in XML format. In the Export dialog box, you can specify a directory in which to store the XML file or accept the default directory, which is DAC\export. In the Export dialog box, you can select the following options: ■ Export All Tables. ■ Export Categories. The Logical option exports all information contained in the Design view and database connection information. The Run Time option exports all information contained in the Execute view. The System option exports all information contained in the Setup view, except passwords for servers and database connections. ■ Export Selected Tables. Allows you to select tables for exporting.
DAC Repository Management > Import	Allows you to import all or part of the DAC metadata. When you import data, any existing data in the repository is overwritten. If you want to import new DAC metadata while still working on the same data warehouse, you must keep the table refresh dates intact. In the Import dialog box, you can specify a directory from which to import the data or accept the default directory, which is DAC\export. In the Import dialog box, you can select the following options: ■ Import All Tables. ■ Import Categories. The Logical option imports all information contained in the Design view and database connection information. The Run Time option imports all information contained in the Execute view. The System option imports all information contained in the Setup view, except passwords for servers and database connections. ■ Import Selected Tables. Allows you to select tables for importing.
DAC Repository Management > Validate Repository	Allows you to start the repository validation process.

Table 31. DAC Tools Menu Commands

Tools Menu Command	Description
DAC Repository Management > Import Informatica Sessions	Allows you to import a list of Informatica workflow names and folders into a staging table in the DAC repository.
DAC Repository Management > Purge Run Details	Allows you to purge completed runs from the run history. You can purge all runs (except the last run) or specify particular runs to be purged. Note: The last run cannot be purged. In the Purging Runs... dialog box, the following options are available: ■ Entire Run History. Allows you to purge all runs and their related information. The data warehouse must be reset for this option to complete successfully. ■ All. Purges all completed runs except for the last run. ■ By Execution Plan. Allows you to select an execution plan whose associated runs you want to purge. ■ By Run Name. Allows you to select an individual run for purging. ■ Before Specified Date. Allows you to select a date before which all runs except the last run will be purged. ■ Details Only. Purges all related information about a run but leaves the run header information.
DAC Repository Management > Analyze Repository Tables	Allows you to run analyze table commands for all the DAC repository tables.
DAC Repository Management > Default Index Properties	Allows you to specify which databases will be associated with newly created indices.
DAC Repository Management > Drop DAC Repository	Allows you to drop all the DAC repository tables. This action deletes all data in the repository.
DAC Server Management > Get Server Log	When the DAC server is running an ETL process, this command opens a text box that displays streaming data related to the process.
DAC Server Management > DAC Server Setup	Allows you to configure the DAC server connections and server email settings. This action should be performed on the machine where the DAC server is running.
ETL Management > Configure	Opens the Data Warehouse Configuration wizard, which you can use to configure an ETL process.
ETL Management > Reset Data Warehouse	Clears the refresh dates for all source and target tables. This action forces a full load to occur.

Table 31. DAC Tools Menu Commands

Tools Menu Command	Description
ETL Management > Preferences	Allows you to configure ETL preferences, such as start and end dates for building the day dimension, currency, language, exchange rate, prune days for disconnected users, maximum number of days an opportunity can spend in a sales stage, and the index space for Oracle databases.
Task Properties > Task Phases	Allows you to add or edit task phases.
Task Properties > Task Folders	Allows you to add or edit task folders.
UI Styles > Windows (MFC)	Changes the user interface to the Windows style.
UI Styles > UNIX (MOTIF)	Changes the user interface to the UNIX style.
UI Styles > Java (METAL)	Changes the user interface to the Java style.

DAC Server Monitor Icon

The Server Monitor is located in the upper-right corner of the DAC client. Its color and shape change based on the DAC server status. When the DAC client cannot establish a connection to the DAC server, the Server Monitor icon resembles a red electrical plug. When the client is connected to the server and the server is idle, the icon resembles an orange electrical plug in a socket. Finally, if the client is connected to a server that is running an ETL process, the icon resembles a green electrical plug with a lightning sign superimposed on it. In addition, clicking on the icon when there is a connection to the server opens a text box that displays data related to the ETL process.

Table 32. DAC Server Monitor Icons

Icon	Description
	DAC client cannot establish a connection to the DAC server.
	DAC client is connected to the server, and the server is idle.
	DAC client is connected to a server that is running an ETL process.

DAC Navigation Tree

The navigation tree appears on the left side of the DAC window, as shown in Figure 11. The tree root nodes correspond to the tabs in the top pane of the DAC window. When a plus sign (+) appears before a node, you can expand the node to view the records belonging to the node. You can double-click a record in the tree to have it display in the top pane in a single-record mode (New, Delete, Copy Record, and Query commands are unavailable), and double-click the root node to return to the list mode.

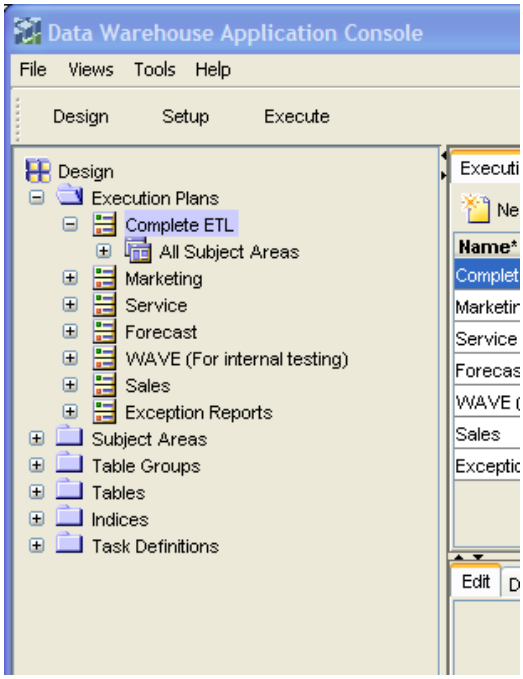


Figure 11. DAC Navigation Tree

DAC Editable Lists

The top and bottom panes of the DAC window display records in a list format. Some of the columns in the list are editable, and others are read-only. The toolbar at the top of each pane allows you to perform various tasks associated with a selected record in the list. For a description of the toolbar commands, see and "DAC Toolbars and Right-Click Menu Commands" on page 137.

A right-click menu is also accessible from the lists in both the top and bottom panes. For a description of these commands, see "DAC Tools Menu" on page 140.

The list format allows you to do the following:

- Edit the data in place and save the record by either clicking another record in the list or clicking the Save button.
- Reorder the columns.
- Sort the data in the list by clicking on the column name.

- Select predefined values from picklists.
- For fields that refer to values from other entities, use the query functionality in pop-up dialog boxes.
- Use Ctrl+C to copy an editable string to the clipboard (not available for read-only strings).
- Ctrl+V to paste a string from the clipboard into a selected cell that supports a string data type.

Logging In to the DAC

The DAC application connects to a database called the DAC repository, which contains definitions of DAC application objects. When you log in to the DAC for the first time, you need to configure the database connection to the DAC repository. You can also connect to multiple DAC repositories by configuring multiple database connections, but only one DAC repository is active at a time.

To log in to DAC for the first time

1. Double-click the DAC Client icon on the desktop.
2. In the Login ... dialog box, select Configure.
3. In the Configuring ... dialog box, select Create Connection, and then click Next.
4. In the Connection Details area of the Configuring ... dialog box, enter the appropriate information for the database to which you want to connect.
5. Click Test Connection to make sure you configured the database correctly.

   When you connect to a database for the first time, a DAC repository schema is created if it does not already exist.

Version Detection

In Siebel Data Warehouse version 7.7.1.4 and higher, if the schema, repository, or seed data are of lower versions than the DAC client or do not exist, you will be prompted to upgrade the DAC repository. This upgrade process is automatic. If the upgrade process is interrupted, it will restart from the point it was last saved the next time you log on to the DAC repository.

Information about the upgrade process is logged in SiebelAnalytics\DAC\log\upgrade.txt.

Tailoring Execution Plans for Your Needs

Execution plans are groups of subject areas that are used to execute ETL processes. For more information about the repository objects that are the components that make up execution plans, see “About DAC Repository Objects” on page 25. For information about DAC functionality, see “DAC Functional Reference” on page 173. For detailed information about customizing execution plans, see “Customizing the Siebel Data Warehouse” on page 201.

In designing an execution plan, you should consider the following questions:

- **Tables.** Which tables need to be populated for the data warehouse. From which tables does your organization source data?
- **Indices.** Do the target tables have the correct indices defined?
- **Tasks.** Are the tasks that load this table defined?
- **Dependencies.** What does a task require to be run?
- **Table groups.** Do table groups need to be built?
- **Subject areas.** Do the subject areas cover all the relevant table groups?
- **Execution plans.** Which subject areas need to be refreshed and how often and at what time frequency?

When an execution plan is run, the DAC server uses the following logic to identify the tasks that need to be run and their order of execution:

1. Initial selection of tables.
   a. Find all the subject areas.
   b. Find all the table groups that belong to the subject areas.
   c. Find all the tables that belong to the table groups.
2. Recursive selection of related tables.

   Recursively find all the table groups and their tables with the above list of tables. For a description of this process, see "About the DAC Table Groups Tab" on page 177.
3. Initial selection of tasks.

   Find all the tasks that load into the tables selected above, that is, tasks whose target table is one of the selected tables (the tasks that are non-optional).
4. Recursive selection of all tasks.
   a. Depending on the non-optional dependency rules, recursively figure out the prerequisite tasks.
   b. Depending on the optional dependency rules, figure out the optional tasks.
5. Filter tasks based on the database connections that will participate in the ETL.
6. Selection of dependencies.

   Load dependencies to identify the execution order.
7. Identify the task details.

   Iterate through all the tasks selected and compute the task details based on whether the target table is loaded in full mode or incremental mode.
8. Identify the list of source tables for change capture, create tasks for change capture, and sync for Siebel source tables.
9. At the end of the ETL process, the DAC server updates refresh timestamps for all source tables (primary or auxiliary) and target tables. The refresh timestamps are the database timestamps. The ETL history timestamps (Start and End timestamps) are the DAC server timestamps.

In the DAC, the Design view allows you to view and modify the relationships among the repository objects discussed above. For more information about the Design view, see "About the DAC Design View" on page 173.

Pruning Execution Plans in the DAC

The Siebel Data Warehouse includes preconfigured execution plans. The Complete ETL execution plan contains a single subject area, named All Subject Areas, which includes all the table groups of the star type. There are also preconfigured execution plans for marketing, sales, forecast, and service. If none of the preconfigured execution plans meet the needs of your organization, you can define your own subject areas and build stars as needed. You can also modify an existing execution plan to meet your needs by inactivating its child objects.

Pruning Subject Areas in the DAC

A subject area is a group of table groups (stars). For example, the subject area named Sales Subject Area may consist of the following stars: Order, Opportunity, and Quote. Including a subject area in an execution plan means that all the tables that belong to the table groups and the related table groups will be populated during the ETL process.

You can add and remove table groups from the preconfigured subject areas, or you can remove a table group from a subject area by *inactivating* the table group.

You can also create a new subject area by associating it with the appropriate table groups. You can create a subject area with any kind of table group, but it is recommended that you use table groups of the star type. A star usually consists of a fact table, dimensions, and aggregates (if any).

Pruning Table Groups in the DAC

A table group is a logical grouping of tables that defines a logical relationship among all the component tables. The table group definition is based on a main table that drives the relationship between itself and the other tables in the group.

A table can be a main table for only one table group. When you design a table group, you associate it with a list of tables that need to be populated whenever the main table participates in an ETL. The tables can be main tables for other table groups; in such cases, when an ETL is run, the DAC server recursively includes all necessary tables. This process in which the tables are recursively populated is called *chaining*.

You can prune a table group by inactivating one or more of the tables associated with it. When designing an execution plan, inspect the table groups that you are interested in. If there are dimensions in a fact table that you are not interested in populating during the ETL process, inactivate them or remove the table association. This might also require you to modify the workflows and dependencies as well. Use the Preview functionality to figure out if your design is correct at the table group, subject area, and execution plan level.

CAUTION: If a table is removed from one table group but is still active in others that belong to the execution plan, you will see it appearing in the preview. If the table is not required for the entire ETL, consider inactivating the table. For more information about inactivating tables, see "The Chaining Process" on page 148.

The Chaining Process

When multiple target (data warehouse) tables share the same source table, they must be populated together. For example, the Person Fact table group contains the tables listed in Table 33. The main table for this table group is W_PERSON_F. When W_PERSON_F participates in an ETL, the other tables listed below also participate. W_PERSON_D is the main table for the Person Dimension table group; whenever it participates in an ETL, the other tables that are part of the Person Dimension table group also participate in the ETL.

Table 33. Person Fact Table Group Example

Name	Main Table for Group	Relationship Type
W_GEO_D	GEO DIMENSION GORUP	Dimension
W_PERSON_F	PERSON FACT GROUP	Self
W_PERSON_D	PERSON DIMENSION GROUP	Dimension
W_ORG_D	ORGANIZATION DIMENSION GROUP	Dimension
W_DAY_D	DAY DIMENSION GROUP	Dimension
W_INDUSTRY_D	INDUSTRY IDMENSION GROUP	Dimension

A similar kind of chaining concept is used for entities that have parent-child relationships. When a parent table participates in an ETL, its child tables also get populated. For example, the Order Dimension and Order Item Fact tables have a parent-child relationship. When the Order Dimension table is part of a table group that participates in an ETL, the Order Item Fact table will also be populated.

Therefore, when pruning the data warehouse, you should start with the table groups of type star and verify that all the related tables are required. If a table is not required and it is a parent table, you need to disable the link between it and the child tables that you do not want to populate. For information about disabling entities, see "Disabling DAC Repository Entities" on page 149.

The chaining process is used for the following reasons:

- To maintain data integrity

  For example, when the Person Fact table group is built, the Person Dimension group is also populated.

- Two or more data warehouse tables can share the same source table

  When multiple data warehouse tables (target tables) share the same source table, they must be populated together. For example, the tables W_PERSON_F and W_PERSON_D have the same main source table, S_CONTACT. Suppose you want to populate the Revenue Fact table group, which has W_REVN_F as its main table and W_PERSON_D as one of the dimension tables. Whenever the Revenue Fact table group is chosen for an ETL execution, W_PERSON_F is also populated because W_PERSON_F is chained to W_PERSON_D.

- To maintain parent-child relationships

For example, when the Order Dimension table is built, the Order Item Fact table also must be populated, otherwise the data will not be in sync.

Pruning Tasks and Dependencies in the DAC

A task is a command that is used to populate a certain table or tables. A task can have prerequisite tasks that need to be executed prior to its own execution. These are defined as dependency rules. There are mandatory and optional dependencies. For example, Load into Activity Fact has a mandatory dependency on Extract into Activity Fact, but has optional dependency on all the tasks that load the dimensions. This is because loading of the Activity Fact table cannot be accomplished until its extract is done. However, if the dimension tables are not absolutely necessary, that is, if part of the pruning process eliminates a certain table from the initial list of tables selected, then the load task will not wait for that optional tasks.

Disabling DAC Repository Entities

You can disable entities at different levels for the sake of pruning.

- If certain tables are not required by a table group, you can remove the tables from the table group or inactivate them. You might also need to review and modify the Informatica workflows.
- If a task that is loading/updating a certain table is deemed not necessary, you can inactivate it.
- If a task is depending on another task, and is deemed not necessary, you can remove the dependency or inactivate the link.
- If a table is not required in the whole of the ETL process, you can disable the table, and none of the tasks that write to it will be chosen for execution.

Once the dependencies are correctly defined, the DAC server will execute them in proper order during ETL execution.

Using the DAC Preview Functionality to View Tasks and Dependencies

You can verify the design of an execution plan using the Preview functionality, which is accessible from the top pane toolbars of the Execution Plans, Subject Areas, and Table Groups tabs, and in the right-click menu on these tabs.

The Preview functionality generates a preview of all tasks and their dependencies using the DAC server logic described in "Tailoring Execution Plans for Your Needs" on page 145. This process reviews an execution plan's associated subject areas, table groups, and tables to identify tasks and their dependencies.

The preview shows the approximate order in which tasks will be executed, and a list of source tables, target tables, and tables for which an image will be built. Only the tasks that source or write to database connections as defined in the execution plan will be included. The preview also shows a detailed analysis of the dependencies. By reviewing the preview, you can validate the design of your execution plan without having to run the ETL.

The preview has three parts. The first part is titled, "Initial Selection of Tables going through the related Table Groups." This part shows the selection of tables to be populated by table group and is created by iterating through the table groups. The selection process starts with the initial, distinct list of all the table groups (iteration 0) and goes through the related table groups (iteration 1, 2, and so on), selecting tables that are not already on the list of chosen tables.

The second part of the preview is titled, "Task Selection Analysis." This part shows the task program. It starts with the tasks that write to any of the tables selected during the table selection process. The DAC server logic goes through the dependency definitions, selecting all the mandatory (non-optional) tasks and then selecting all the optional tasks. Eventually, the dependencies define the order of execution.

The third part of the preview is titled, "Step Dependency Analysis." This part shows, for all given tasks, the following information:

- **Depends On.** The immediate tasks that a given task depends on.
- **Dependents.** The tasks that are dependent on a given task.
- **Number of All Depends.** All the Depends On tasks.
- **Number of All Dependents.** All the Dependent tasks.

This part of the preview is useful for understanding how a task relates to all other tasks, which can be helpful for debugging.

Figure 12 shows the dependencies of various tasks: task A depends on tasks B and C, task B depends on tasks D and E, and task D depends on tasks F and G.

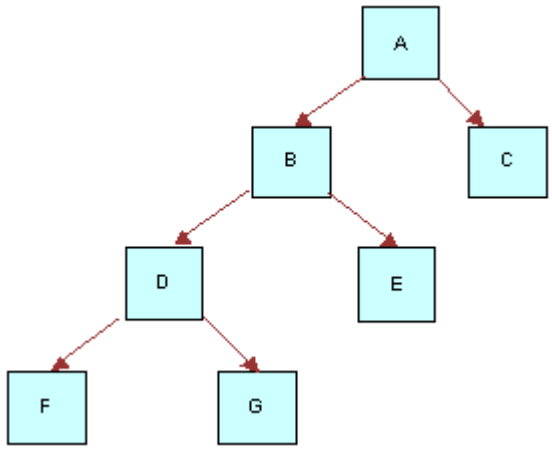


Figure 12. Task Dependencies

In Figure 13 the tasks are shown as grouped by the DAC server based on the level of dependency. (The level of dependency is referred to as a *depth*.) Tasks that have no dependencies are depth 0. Tasks that depend on other tasks of depth 0 are depth 1. Tasks that depend on other tasks of depth 1 or 0 are depth 2, and so on.

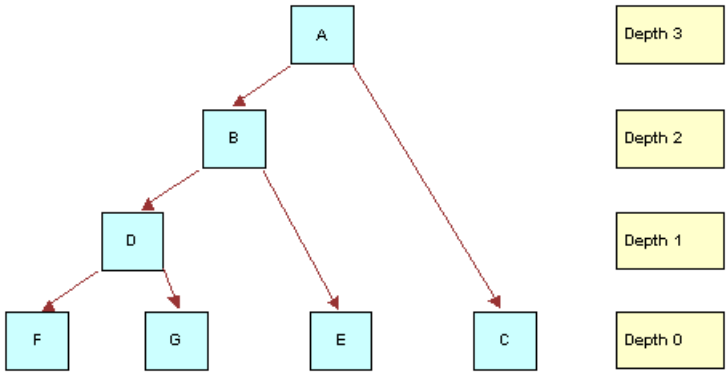


Figure 13. Task Dependencies by Depth

Because the DAC server dynamically prioritizes the tasks that are eligible to run based on the number of resources available, it uses the information generated by the preview to decide which tasks to execute. When more than one task is eligible to be run, a series of prioritization rules is applied.

Distributing DAC Metadata

Typically, you may have multiple environments, such as development, QA, production, and so on. When you make changes to the development environment, you test it, and then deliver it, exporting the whole environment and distributing it to the other environments. The data is exported as XML files, which are stored in the DAC\export directory on the client machine where the export is done.

To apply changes from the development environment to any other, you copy all of the XML files into the DAC\export folder and then import the data.

To distribute DAC metadata from the development environment to another environment

1. In the DAC, select Tools > DAC Repository Management > Export.

   The Export dialog box appears.

2. Select one of the following options:

   - **Export All Tables.** Select this option to duplicate the entire environment. All tables in the DAC repository and the information in the Design view associated with them are exported as XML files and stored in the DAC\export folder. Exercise caution when selecting this option, because when you import this data, it replaces all run-time information that exists on the database where the import is performed.
   - **Export Categories.** Select this option to export part of the data from the development environment. The data is stored as XML files in the DAC\export folder. Select one of the following categories:
     - **Logical.** Exports all information contained in the Design view and the database connection information. This option preserves all the run data in the new environment, including refresh dates.
     - **Run Time.** Exports the run-time information but preserves all the logical definitions in the new environment. You may seldom have to select this option.
     - **System.** Exports system preferences, ETL preferences, and email recipient information. Passwords for servers and database connections are not exported.

3. Copy the XML files from the DAC\export folder in the development environment to the DAC\export folder on the local machine in the new environment.

4. In the DAC, in the new environment, select Tools > DAC Repository Management > Import.

   The Import dialog box appears.

5. Select the import option that corresponds to the export option you selected in step 2. The following options are available:

   - **Import All Tables.** For security reasons, this option does not populate the password fields. Make sure you update the passwords before attempting to run an ETL process.
   - **Import Categories.** Select one of the following categories:
     - Logical

- Run Time
- System

Using the DAC Query Functionality

Querying is a way to locate one or more records that meet your specified criteria. Query functionality is available in every DAC screen. When you enter query mode, the Edit and Description child tabs in the bottom pane are not available.

DAC Query Commands and Operators

You can use the query commands and operators in Table 34 to define your query criteria.

For Siebel Data Warehouse version 7.7.1.4 and higher, queries are case insensitive.

Table 34. DAC Query Commands and Operators

Operator	Description
=	Placed before a value, returns records containing a value equal to the query value.
<	Placed before a value, returns records containing a value less than the query value.
>	Placed before a value, returns records containing a value greater than the query value.
< >	Placed before a value, returns records containing a value that is not equal to the query value.
< =	Placed before a value, returns records containing a value less than or equal to the query value.
> =	Placed before a value, returns records containing a value greater than or equal to the query value.
*	Wildcard that can be placed in the middle, or at the beginning or end of a text string.
!	Used for negation.
“ “	Surrounds a string that, unless modified by a wildcard, must be matched exactly.
\	Escape symbol is used when double quotes should not be processed as a special symbol. For example, *!(“*null text”or(\”*\”))* is a value expression for a text field. The query returns values that do not end with a string *null text* and that are not surrounded by double quotes.
()	Surrounds the values and operators that will be processed first.
NULL	Returns records for which the query field is blank.

Table 34. DAC Query Commands and Operators

Operator	Description
AND	Placed between values, returns only records for which all the given conditions are true. (Not case sensitive.)
OR	Placed between values, returns records for which at least one condition is true. (Not case sensitive.)

DAC Query Examples

The following examples show different ways you can query on the Name column of the Tasks tab.

- *Extract** lists all tasks whose name starts with Extract.
- **Extract** lists all tasks whose name contains the word *Extract*.
- *!Extract** lists all tasks whose name does not start with the word *Extract.*
- *!null* lists all tasks whose name is not null.
- *Extract* or Aggregate** lists all tasks whose name starts with *Extract* or *Aggregate*.
- *Load* and *Aggregate** lists all tasks whose name starts with *Load* and also contains the word *Aggregate.*
- *"Extract for Wave Dimension" or "Load into Wave Dimension"* lists tasks whose name is either *Extract for Wave Dimension* or *Load into Wave Dimension*.

NOTE: When using spaces within strings, you need to surround the string with quotes ("").

Common DAC Query Procedures

This section includes instructions for common query procedures.

To create and execute a query in the DAC

1. In the top or bottom pane of the DAC, click Query on the toolbar or in right-click menu.

   A blank row in a list appears.

2. Enter the query criteria in the appropriate fields.
3. Click Run Query on the toolbar.

The query is executed and the records appear.

To enter a query value in a date field

1. In the date field, click the calendar icon on the right side of the cell.

   The Date dialog box appears.

2 Enter the date and time for which you want to search, and select the appropriate query condition.

You can use the query commands and operators listed in "DAC Query Commands and Operators."

Executing an ETL with the DAC

Execution plans are subject areas that are used to execute ETL processes. Before you attempt to run an ETL, do the following:

- Set ETL preferences. For more information, see "Setting DAC ETL Preferences" on page 94.
- Set database connections to the Siebel transactional database and the data warehouse. For more information, see "Creating ODBC Database Connections" on page 72.
- Register the Informatica Servers. For more information see, "Creating and Dropping Data Warehouse Tables on a Teradata Database" on page 77.

To execute an ETL with the DAC

1 Launch the DAC client by double-clicking the startclient.bat file in the DAC directory or by running startclient from a command prompt.

2 Navigate to the Execution Plans tab in the Design view.

3 Select the execution plan you want to execute.

4 Click Run Now.

Once the ETL starts running you can monitor its progress in the Current Run tab of the Execute view. For more information about monitoring an ETL, see "About the DAC Current Run Tab" on page 194 and "About the DAC Run History Tab" on page 197.

Running the DAC Server Automatically

Follow this procedure to set up the DAC server to be run automatically when your machine reboots.

To set up the DAC server to run automatically upon rebooting the machine

1 Navigate to Programs > Accessories > System Tools > Scheduled Tasks.

2 Double-click Add Scheduled Task.

3 In the Scheduled Task Wizard, browse to the startserver.bat file, and click Open.

4 Select the option "When my computer starts," and click Next.

5 Enter the domain user account to start the DAC server and a password, and click Finish.

The startserver task appears in the Scheduled Task window.

6 Right-click the task and select Properties.

7 In the Settings tab, remove the check from the "Stop the task if it runs for 72 hours" check box.

To start the DAC server as a scheduled task

1. Navigate to Programs > Accessories > System Tools > Scheduled Tasks.
2. Right-click startserver, and then click Run.

To stop the DAC server as a scheduled task

1. Navigate to Programs > Accessories > System Tools > Scheduled Tasks.
2. Right-click startserver, and then click End Task.

To check if the DAC server is running

1. Navigate to Programs > Accessories > System Tools > Scheduled Tasks.
2. Select the startserver task.
3. In the Windows menu bar, select View > Details.

Overview of Command Line Access to the DAC Server

NOTE: This feature is only available if you have installed Siebel Data Warehouse version 7.7.1.1 and higher.

This chapter covers accessing the DAC server through a command line. It includes the following topics:

You can access the Siebel DAC server through a command line to start and stop execution plans and to get status information for servers, databases, and execution plans. This feature allows you to access the DAC server using third-party administration or management tools, without using the DAC client.

Command Line Operations

The command line feature allows you to start an execution plan and stop the operation of a running execution plan.

Starting an Execution Plan

When the DAC server receives a request to start an execution plan, it performs a series of checks to verify that the execution plan can be started. It first checks that an execution plan with the requested name exists and that the execution plan is active. Next, it checks the status of the execution plan that last ran. If an execution plan is still running and the DAC server receives a request to start another execution plan, the request will be rejected. If an execution plan failed, a request to run the same execution plan again will be executed; however, a request to run a different execution plan will be rejected. If the execution plan that last ran completed successfully, a request to run a new execution plan will be executed.

When the DAC server receives a request to start an execution plan, it will issue a warning if any of the following conditions are true. (A warning is for informational purposes and does not mean the execution plan will not start.)

- The Generic task concurrency limit (set in the system properties) is not a positive number.
- There are no active Informatica servers in the server list.
- One or more Informatica servers do not have the passwords defined.
- One or more Informatica servers do not have a maximum number of sessions properly set.
- One or more data sources do not have the passwords defined.
- One or more data sources do not have a maximum number of connections properly set.

Stopping the Operation of a Running Execution Plan

When the DAC server receives a request to stop the operation of a running execution plan, the request will fail in the following cases:

- The name of the execution plan that is running is different from the name in the request.
- There is no execution plan currently running.

Command Line Status Monitoring Queries

The command line feature allows you to get the following status information:

- Summary of the requested execution plan. If there are multiple instances of the same execution plan, a summary of the instance that last ran is returned. Below is an example of the information contained in the summary.

```
(c) 2003 Siebel Systems, Inc.

Siebel DAC Server comprising the etl execution-management,
scheduler, logger, and network server.

ETL details for the last run:

ETL Process Id : 255
ETL Name : Complete ETL
Run Name : DRY RUN OF Complete ETL: ETL Run - 2004-06-17 18:30:13.201
DAC Server : (aqamarD510)
DAC Port : 3141
```

```
Status: Stopped
Log File Name: Complete_ETL.255.log
Database Connection(s) Used :

   OLTP jdbc:microsoft:sqlserver://vranganaw8:1433;DatabaseName=OLTP
   Data Warehouse jdbc:microsoft:sqlserver://vranganaw8:1433;DatabaseName=olap

Informatica Server(s) Used  :

   InformaticaServer4-vranganaw8:(4)
   InformaticaServer2-vranganaw8:(2)
   InformaticaServer3-vranganaw8:(3)
   InformaticaServer1-vranganaw8:(10)

Start Time: 2004-06-17 19:00:06.885
Message: ETL was interrupted
Actual Start Time: 2004-06-17 18:30:13.357
End Time: 2004-06-17 19:05:56.781
Total Time Taken: 35 Minutes

Start Time For This Run: 2004-06-17 19:00:06.885
Total Time Taken For This Run: 5 Minutes

Total steps: 212
Running steps: 0
Complete steps: 142
Failed/Stopped steps:70
```

- Summary of connection status to all active databases and Informatica servers.

Setting Up Command Line Access to the DAC Server

Follow this procedure to set up the command line functionality.

To set up command line access to the DAC server

1. Make sure you have the Java SDK 1.4.2 installed.
2. Copy the following files from the SiebelAnalytics\DAC directory to a local directory:
   - DAWSystem.jar
   - dac.properties
   - dacCmdLine.bat
3. In the dacCmdLine.bat file, edit the JAVA_HOME variable to point to the directory where the Java SDK is installed.

   Make sure there are no spaces in the path reference.
4. Edit the DAC_HOME variable to point to the directory where DAC-REMOTE is installed.

5 In the dac.properties file, edit the following parameter values.

Parameter	Value
ServerHost=	Host name of the DAC server.
ServerPort=	Port of the DAC server. The default is 3141.
RepositoryStampVal=	Repository stamp that appears in the DAC client Login Details screen. To find this value, in the DAC client navigate to Help > Login Details.

Your dac.properties file should look similar to the following:

```
ServerHost=vranganaw8
ServerPort=3141
RepositoryStampVal=851E0677D5E1F6335242B49FCCd6519
```

Using the Command Line to Access the DAC Server

Follow this procedure to use the command line to access the DAC server.

To use the command line to access the DAC server

- At the command prompt, enter the following:

dacCmdLine <method name> <optional execution plan name>

where method name is one of the following:

Method Name	Description
StartETL	Starts an execution plan. You must specify an execution plan name.
StopETL	Stops the operation of an execution plan. You must specify an execution plan name.
ETLStatus	If you do not specify an execution plan name, the status of the execution plan that last ran is returned. If you specify an execution plan name, the status of the specified execution plan is returned.
DatabaseStatus	Verifies whether the DAC server can connect to all active database connections. You do not need to specify an execution plan name.
InformaticaStatus	Verifies whether the DAC server is able to ping all active Informatica servers.
NOTE: The method names are case insensitive. Execution plan names are case sensitive. Also, if the execution plan name contains spaces, place beginning and ending double quotes around the name.	

For example:

Command Line	Description
dacCmdLine EtlStatus	Returns the status of the execution plan that last ran.
dacCmdLine EtlStatus Forecast	Returns the status of the last instance of the Forecast execution plan.
dacCmdLine StopEtl Forecast	If the execution plan currently running is Forecast, the operation will be terminated. Otherwise, the request is ignored.
dacCmdLine databasestatus	Returns the health status of all the database connections as defined in the DAC repository from the DAC server.
dacCmdLine InformaticaStatus	Returns the health status of all the Informatica server connections as defined in the DAC repository from the DAC server.

Handling ETL Failures with the DAC

This section includes the following topics:

When the Execution of an Execution Plan Fails

When an execution plan is executed, if a task fails, the status of the tasks that are dependent on the failed task is changed to Stopped. While tasks are still running, the execution plan's status is Running. When all the tasks have been run, and if one or more tasks have failed, the execution plan's status is changed to Failed.

You can check the tasks that have failed in the Current Run tab of the Execute view, fix the problems, and then requeue the failed tasks by changing the status to Queued. You can then restart the ETL. All the tasks will then be rerun. You can also manually run a task, change its status to Completed, and then restart the ETL. Tasks with a Completed status are skipped.

CAUTION: The DAC server does not validate tasks that have been run manually.

To restart a Failed ETL, click Run Now from the Current Run tab of the Execute view.

In Case of Abnormal Termination of the DAC Server

If the DAC server fails during the execution of the ETL, the status of the ETL execution will remain as Running. When the DAC server is started again, it will automatically run the ETL if the Auto Start ETL system preference is set to True. If the same system preference is set to False, when the server restarts, it will set the correct status as Failed. In order to execute the ETL from the point of failure, submit the request to the server again.

The DAC server will automatically terminate if it looses connection to the DAC repository.

Discarding the Current Run Execution Plan

You can discard an execution plan that failed by navigating to the Current Run tab, right-clicking on the execution plan and changing its status to Mark as Completed. This will force the run status to be updated as Completed. When you submit a request for another run, the DAC server creates another instance of it.

CAUTION: Perform this procedure in a development or testing environment only, since it might leave the data in an inconsistent state, causing you to have to reload all of the data.

Failure of Aggregator Transformation Tasks with Sorted Input

Tasks that use Informatica Aggregator transformation can fail when the Sorted Input option is active. The tasks SDE_DTLFORECASTFACT and SDE_COSTLIST are examples of tasks that can fail in such a situation.

To prevent such tasks from failing, in Informatica Designer, navigate to Mapping Designer, open the corresponding mapping, and in the Aggregator transformation, remove the check from the Sorted Input check box.

Performing a Full Load Again

In test environments, after running several ETLs, you might want to perform a full load again. You do this by accessing Tools > ETL Management > Reset Data Warehouse.

CAUTION: This procedure will cause the DAC server to perform a full load the next time an ETL is run, which will truncate data from the target tables. Therefore, this action should be used only in development and testing environments. You should not perform this procedure in a production environment unless you plan to rebuild the data warehouse in its entirety.

Using the Change Capture Filter

The change capture filter allows you to selectively load data from the Siebel transactional database into the data warehouse. You can set a condition in the ChangeCaptureFilter.xml file to filter data from specific tables. This file is located in the SiebelAnalytics\DAC\CustomSQLs directory. It provides an XML sample that you can copy and alter to fit your needs. Detailed instructions for using this feature are included at the top of the file.

Comparing and Merging DAC Repositories

The DAC Repository Merge feature allows you to identify differences between a source DAC repository and a target DAC repository and to move new objects from the source to the target repository. You can use this feature to upgrade repository customizations of prior releases to a new release or to compare and merge two repositories, such as development and production repositories.

The DAC Repository Merge feature identifies the following differences between the source and target repositories. The identified differences are listed in the DAC Repository Difference report. This report is automatically generated during the compare and merge process and saved in the SiebelAnalytics\DAC\diff\html directory.

- **Modified objects.** The row ID and user key of an object type in the source repository is compared with the row ID and user key of the same object type in the target repository. If either the row ID and user key are the same for the two objects or if both the row ID and user key are the same, the objects are considered to be a match, and the any difference between the attributes of these objects will appear in the Difference report.
- **Objects added to the target repository.** Objects that appear in the target repository but do not appear in the source repository.
- **Objects missing from the target repository.** Objects that appear in the source repository but do not appear in the target repository.

Table 35 lists the objects that are compared and the table and user key associated with the object.

When you perform the compare process, using the Compare and Merge Repositories dialog box, you can choose to import from the source repository into the target the objects that are missing. Records that were modified in the source repository cannot be imported into the target. You must enter the modifications manually in the target repository.

Table 35. Objects Compared in DAC Repository Compare Process

Entity	Main Table	User Key
Database Connections (source/target)	W_ETL_DBCONN	NAME
Database Indices	W_ETL_DB_INDEX	W_ETL_DB_INDEX.DB_TYPE, W_ETL_INDEX.NAME
Database Tables	W_ETL_TABLE_DT	W_ETL_DBCONN.NAME, W_ETL_TABLE.NAME
Execution Plan	W_ETL_DEFN	W_ETL_NAME
Execution Plan—Database Connections	W_ETL_DEFN_DB	W_ETL_DBCONN.NAME, W_ETL_DEFN.NAME
Execution Plan—Subject Area	W_ETL_DEFN_SA	W_ETL_DEFN.NAME, W_ETL_SA.NAME

Table 35. Objects Compared in DAC Repository Compare Process

Entity	Main Table	User Key
Execution Plan—Pre-Post Steps	W_ETL_DEFN_STEP	W_ETL_DEFN.NAME, W_ETL_STEP.NAME
Group	W_ETL_GROUP	NAME
Group Tables	W_ETL_GRP_TBL	W_ETL_GROUP.NAME, W_ETL_TABLE.NAME
Index Columns	W_ETL_INDEX_COL	W_ETL_INDEX.NAME, W_ETL_INDEX_COL.NAME
Indices	W_ETL_INDEX	NAME
Informatica Folder	W_ETL_FOLDER	NAME
Tables	W_ETL_TABLE	NAME
Tasks	W_ETL_STEP	NAME
Task Dependencies	W_ETL_STEP_DEP	W_ETL_STEP.NAME, W_ETL_STEP.NAME
Task Phase	W_ETL_PHASE	NAME
Task Tables	W_ETL_STEP_TBL	W_ETL_STEP.NAME, W_ETL_TABLE.NAME
Subject Area	W_ETL_SA	NAME
Subject Area/Group	W_ETL_SA_GROUP	W_ETL_GROUP.NAME, W_ETL_SA.NAME
System Properties	W_ETL_SYSPROP	NAME

To compare and merge DAC repositories

1. Restore the target DAC repository on a separate database instance.

   If you are upgrading from a previous version, the target repository is the new version.

2. Export the source DAC repository to back up the data.

3. Set up the database connections for the source and target repositories.

   a. In the DAC, navigate to Setup > Database Connections.

   b. Enter the following:

Field	Enter
Logical Name	A logical name for the connection.
Type	DAC Repository.
Connection Type	Type of database connection.

Field	Enter
Dynamically changes based on Connection Type	If you are using... ■ Oracle (OCI8) enter the TNS name. ■ Oracle (Thin) enter the instance name. ■ SQL Server enter the database name. ■ DB2-UDB enter the connect string as defined in the DB2 configuration. ■ Teradata enter the database name.
Table Owner	Name of table owner.
Table Owner Password	Password for table owner.
Max Num Connections	The maximum number of connections the DAC server will maintain to the database.
DB Host	Host machine where the database resides. Required for Oracle (Thin) and MSSQL databases. Not required for Oracle (OCI8), DB2, and DB2-390.
Port	Port where the database receives requests. Required for Oracle (Thin) and MSSQL databases. Not required for Oracle (OCI8), DB2, and DB2-390.

4 From the DAC toolbar, select Tools > DAC Repository Management > Compare and Merge Repositories.

5 In the Compare and Merge Repositories dialog box, select the source and target repositories that you want to compare.

6 To import from the source repository into the target the objects that are missing, select the Import Missing Objects check box.

NOTE: Modifications made to objects in the source repository must be entered manually in the target repository.

The Difference report is automatically created.

7 To view the Difference report, navigate to SiebelAnalytics\DAC\diff\html, and double-click index.html.

DAC Common Tasks

This section contains the following topics:

Replacing an Informatica Workflow with a Custom SQL File

You can improve the performance of loads by replacing Informatica workflows with custom SQL files.

For Teradata databases, SQL files are provided in the SiebelAnalytics\utility\Teradata directory.

To replace an Informatica workflow with a custom SQL file

1. Create a SQL file to be used to load the table, and unit test it.
2. Create an XML file with one or more SQL statements in the format that the DAC can understand.

   For more information about creating an XML file, see "Using SQL Files as an Execution Type in the DAC" on page 188.

   You can create one file for a full load and one for an incremental load, or you can use the same file for both full and incremental loads.
3. Save the file in the SiebelAnalytics\DAC\CustomSQL directory.
4. In the Task Definitions tab of the DAC Design view, query for the task for which you want to replace the Informatica workflow.
5. Replace the workflow name in the Command for Incremental Load or Command for Full Load fields with the XML file.
6. Change the Execution Type to SQL.

   The DAC uses the target database connection to execute these commands.

Determining the Informatica Server Maximum Sessions Parameter Setting

You set the Maximum Sessions parameter value when performing the procedure "Registering Informatica Servers in the DAC Client" on page 74. This parameter specifies the maximum number of workflows that can be executed in parallel on the Informatica server. If the number of sessions is zero or is not specified, the DAC server assigns the default value of five for the DAC Lite version and 10 for the full DAC version. The DAC Lite version cannot run more than five sessions in parallel.

You should consider the following factors when determining the Maximum Sessions parameter value:

- How powerful the machine is that hosts the Informatica Server.
- How many Informatica Server servers are available.

- The number of Runnable tasks in the queue. A Runnable task is a task for which the Depends On tasks have completed and is ready to be run but is waiting for an Informatica slot to be available. For information about the different task run statuses, see "About the DAC Current Run Tab" on page 194.

  For an optimal run, the runnable queue should be at zero or should reach zero in a short time. For example, Figure 14 shows an ideal performance run when 15 sessions were run in parallel. There were many runnable tasks before the process began, but the queue soon reached zero.

  You can generate a run analysis such as Figure 14 from the right-click menu (Get Run Information > Get Graph) on the DAC Current Run and Run History tabs. If you find that the runnable curve does not reach zero in a short time, you should increase the Maximum Sessions parameter value to make more Informatica slots available.

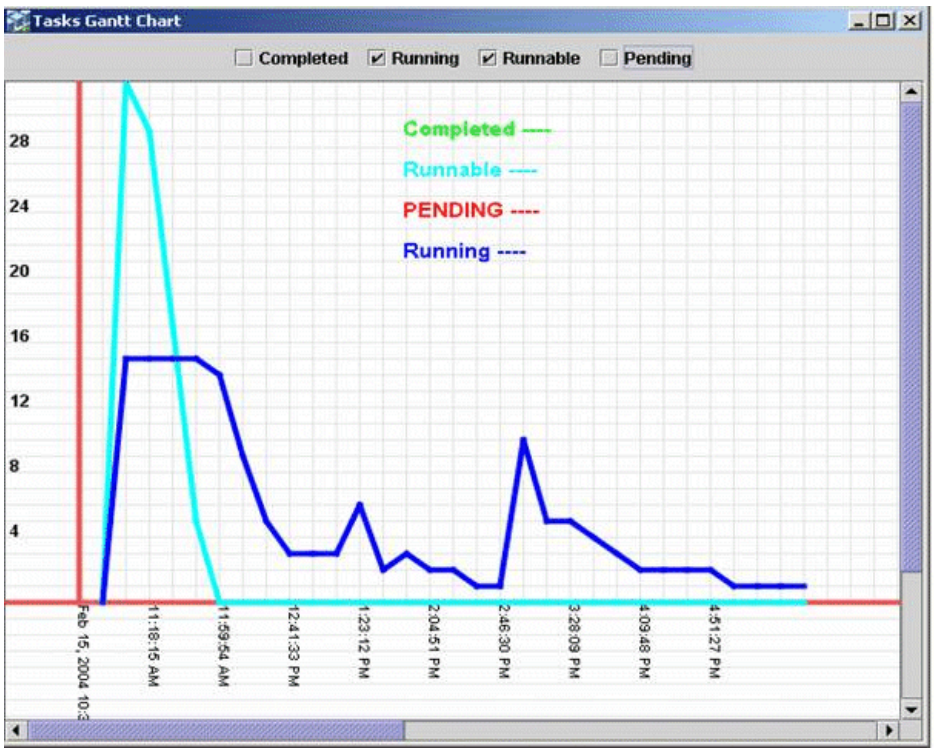


Figure 14. Sample Performance Run

Determining the Number of Transactional and Data Warehouse Database Connections

This section describes how to determine the maximum number of database connections you need between the DAC server and the transactional database and the DAC server and the data warehouse database. You set the Max Num Connections parameter when performing the procedure, "Creating Transactional and Data Warehouse Database Connections" on page 73.

For the transactional database, the DAC server uses these connections to perform change capture. The number of connections you set for this connection pool determines how many change capture processes can run concurrently. If you have a powerful transactional database server and are going to run ETL processes during off-peak times, you can increase the Max Num Connections setting to 15 or 20 (10 is the default). If you have a less powerful transactional database server, you should not overload the operational system with ETL processes. Therefore, you should set the value below 10.

For the Data Warehouse database, the DAC server uses these connections to perform processes such as truncate tables, drop and create indices, and analyze tables. You should not set the Max Num Connections value higher than the Maximum Sessions parameter value (the maximum number of workflows that can be executed in parallel on the Informatica server) because these values have a one to one relationship.

Running Two DAC Servers on the Same Machine

You can run two DAC servers on the same machine as long as they are listening on different ports and pointing to two different repositories.

To run two DAC servers on the same machine

1. Install two instances of the DAC server in two different directories.

   For installation instructions, see "Installing and Configuring Siebel Analytics Applications on Windows" on page 55.

2. For each instance, edit the config.bat file to set the DAC_HOME variable appropriately.

   For instructions, see "Configuring the DAC Config.bat File" on page 64.

3. Launch each of the DAC clients by navigating to the DAC directories and double-clicking the startclient.bat file.

4. For each instance, configure the DAC repository connection.

   a. Navigate to Tools > DAC Server Management > DAC Server Setup.

      An informational dialog box states this operation should be performed on the machine running the DAC server. It asks whether you want to continue.

   b. Click Yes.

   c. In the Repository Connection Information tab, enter the appropriate information for each instance. The Database Host should be the same for each instance, and the Database Port should be different.

5. For each instance, set up the DAC server system properties.

   a. Navigate to Setup > DAC System Properties.

   b. Set the DAC Server Host, OS, and Port properties.

6. Start each DAC server from its directory.

Tracking Deleted Records

The Siebel Data Warehouse change capture process uses delete triggers to identify records for deletion on the Siebel transactional database. The deleted records are stored in S_ETL_D_IMG tables. During the change capture process, the DAC server moves the data from the S_ETL_D_IMG tables to the S_ETL_I_IMG tables, where D appears in the OPERATION column to show the records were deleted. During the change capture sync process, the records in the S_ETL_D_IMG tables that were moved to the S_ETL_I_IMG tables are flushed. In the DAC, you can view the SQL that runs during the change capture and change capture sync processes by navigating in the DAC to Design > Task Definitions > Description.

The preconfigured ETL process captures deleted records for the target tables W_ORG_D and W_PERSON_D, the source tables for which are S_ORG_EXT, S_CONTACT, and S_PRSP_CONTACT. These source tables need to have delete triggers created in the Siebel transactional database in order for deleted records to be tracked.

For vertical applications, the preconfigured ETL process captures deleted records for W_FUND_F and W_ALIGNMT_DH. You need to create delete triggers in the transactional database for the following additional tables: S_MDF_TXN, S_ASGN_GRP_POSTN, S_ASGN_RULE_ITEM.

In the Siebel Data Warehouse, preconfigured visibility tables are inactivated. If you activate visibility tables, you should also create delete triggers on the optional tables. You can activate visibility tables in the DAC by navigating to Design > Tables.

The preconfigured SIA Account and Contact visibility tables are activated by default for vertical applications. If your organization is not going to use any of the visibility tables, you need to inactivate them in the Tables tab of the Design view in the DAC client.

On the target tables for which deleted records are tracked, a D appears in the INACTIVE_FLG column to show the records as deleted when the source records are deleted. This method of flagging a record as deleted is known as a *soft delete,* as compared to a *hard delete* when the record is physically deleted. When deleted records are tracked on visibility-related data warehouse tables, the records are physically deleted. The general rule is that soft deletes should be used for tables that are referenced by other tables. If a table is not referenced by any other table, then you can use hard deletes.

Aggregate tables are rebuilt during each ETL process. Therefore, records can be physically deleted from the base tables without consequence. If you want to use the soft delete method, you should consider changing the aggregate building mappings so that the deleted records are omitted.

NOTE: The Analytics Server does not recognize soft deletes. Therefore, you have to modify the .rpd file so that it does not pick up soft-deleted records for reporting.

To create delete triggers for preconfigured ETL change capture

1. From the DAC menu bar, click Tools > ETL Management > Configure.
2. In the Sources dialog box, select the database platform for the target and transactional databases, and click OK.

3 In the Data Warehouse Configuration Wizard, select the Create Delete Triggers in Transaction Database check box, and click Next.

The Delete Triggers tab is active.

4 Select one of the following:

Option	Description
Create Triggers	Executes the trigger statements directly.
Write Script to File	Writes the trigger statements to a file, which can be executed by a database administrator.

5 Select the database type as defined in the DAC.

6 For DB2 zSeries databases, enter the base table owner.

7 (Optional) Select the Include Optional Triggers check box to create triggers for the optional tables.

8 Click Start.

To create delete triggers for new source tables

1 In the DAC, navigate to Design > Tables.

2 Select the table for which you want to track deleted records.

Make sure the table has an image suffix.

3 Right-click the table and select Change Capture Scripts > Generate Image and Trigger Scripts.

4 In the Triggers and Image Tables dialog box, select the database type of the source database.

5 Make sure the Generate Image Table Scripts and Generate Trigger Script(s) options are selected.

6 Execute the script on the database.

To track deleted records

1 Make sure the delete triggers are enabled for the appropriate tables.

2 Write custom Informatica workflows with a clause WHERE operation = 'D' to the appropriate I_IMG table to take them across to the dimension and fact tables.

3 In the DAC, register the workflows as tasks.

4 Define the appropriate dependencies.

For an example of such a workflow, see the preconfigured task SDE_OrganizationDimension_LoadDeletedRows.

Customizing Index and Analyze Table Syntaxes

The customsql.xml file, located in the SiebelAnalytics\DAC\CustomSQLs directory, contains the syntax for dropping and creating indexes and analyzing tables. You can edit the customsql.xml file to change the behavior of these operations.

To edit the Analyze Table syntax

1. Open the customsql.xml file located in the SiebelAnalytics\DAC\CustomSQLs directory.
2. Locate the Analyze Table syntax for the appropriate database type.

   For example, the syntax for an Oracle database is as follows:

   ```
   <SqlQuery name = "ORACLE_ANALYZE_TABLE" STORED_PROCEDURE = "TRUE">
   DBMS_STATS.GATHER_TABLE_STATS(ownname => '@TABLEOWNER', tabname => '%1',
   estimate_percent => 30, method_opt => 'FOR ALL COLUMNS SIZE AUTO',cascade => true )
   </SqlQuery>
   ```

3. Edit the syntax.

   For example, to gather statistics for only the indexed columns, edit the syntax as follows:

   ```
   <SqlQuery name = "ORACLE_ANALYZE_TABLE" STORED_PROCEDURE = "TRUE">
   DBMS_STATS.GATHER_TABLE_STATS(ownname => '@TABLEOWNER', tabname => '%1',
   estimate_percent => 30, method_opt => 'FOR ALL INDEXED COLUMNS',cascade => true )
   </SqlQuery>
   ```

   NOTE: The variables @TABLEOWNER, %1, %2, and so on, will be substituted appropriately by the DAC when the statement is executed.

To edit the Create Index syntax

1. Open the customsql.xml file located in the SiebelAnalytics\DAC\CustomSQLs directory.
2. Locate the Create Index syntax for the appropriate database type, and edit the syntax.

9 DAC Functional Reference

This chapter describes the functionality available in the Siebel Data Warehouse Application Console (DAC). It contains the following topics:

About the DAC Design View

The DAC Design view provides access to functionality related to creating and managing execution plans. The top pane of the Design view contains tabs, each of which represents a component of the execution plan. The bottom pane of the Design view contains subtabs that relate to the tab selected in the top pane. The navigation tree allows you to see the hierarchical relationship among all the DAC repository objects.

This section includes the following topics:

About the DAC Execution Plans Tab

The Execution Plans tab displays a list of execution plans in the top pane. This list includes the execution plan name and an Inactive column, which indicates whether the execution plan is active or inactive. Inactive execution plans do not participate in the ETL process.

Table 36 provides a description of the subtabs in the bottom pane.

Table 36. DAC Execution Plans Subtabs

Tab	Description
Edit	Allows you to edit the execution plan selected in the top pane.
Description	Displays a description of the execution plan selected in the top pane.
Subject Areas	Displays a list of subject areas associated with the execution plan selected in the top pane.
Database Connections	Displays a list of database connections associated with the execution plan selected in the top pane. In order for a task to participate in the ETL process, the task's source and target must match the list of database connections displayed here. You can inactivate a database connection by selecting the Inactive check box. If you inactivate a database connection, all tasks that read or write to it are eliminated.
Tables (Read Only)	Displays a read-only list of the initial tables that participate in an ETL. For a complete list of tables participating in the ETL, use the Preview functionality. The list of tables contains the following column headings: ■ **Name.** Name of the target table. ■ **Main Table for Group.** If the table is a main table for a table group, the group name is displayed in this column. ■ **Table Type.** Table type as defined in the table properties.
Preceding Tasks	Displays a list of tasks that must be completed before an ETL is executed. The list of tasks contains the following column headings: ■ **Name.** Name of the task. ■ **Priority.** Indicates the order in which the task runs. If two or more tasks have the same priority, the order occurs randomly. ■ **Inactive.** Allows you to inactivate a task by selecting the check box. ■ **Command.** Command associated with the task.
Following Tasks	Displays a list of tasks that must be completed after an ETL is executed. The list of tasks contains the following column headings: ■ **Name.** Name of the task. ■ **Priority.** Indicates the order in which the task runs. If two or more tasks have the same priority, the order occurs randomly. ■ **Inactive.** Allows you to inactivate a task by selecting the check box. ■ **Command.** Command associated with the task.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Executions Plan tab is active.

Preview Command

This command generates a preview of all tasks and their dependencies by reviewing the subject areas, table groups, tables, and tasks. Information provided includes the approximate order in which tasks are executed, a list of source tables, target tables, and tables for which an image will be built. Only the tasks that source or write to database connections as defined in the execution plan are included. For more information about the Preview functionality, see "Using the DAC Preview Functionality to View Tasks and Dependencies" on page 150.

Copy Record

This command creates a copy of the selected execution plan and also copies the references to subject areas and database connections.

Generate Index Scripts Command

This command generates drop index, create index, and analyze table scripts for all the tables that participate in the ETL process. The results are stored in the log\scripts directory. The index scripts are separated by ETL indices and query indices. The DAC server automatically drops and creates indices and analyzes the tables. This option can be turned off by setting the system preferences. It is recommended that this option be turned off only for DB2-390 environments.

Run Now Command

This command submits a request to the DAC server to execute the execution plan. Inactive execution plans are not included in the process.

When an execution plan is run, the following logic is used to identify the tasks that need to be run and their order of execution.

1. Check on the previous run status.
   a. If the previous run executed a different execution plan from the one currently submitted (either manually or through the scheduler), abort the run.
   b. If the previous run executed is the same as the current execution plan and did not complete, continue from where it left off.
   c. If the previous run has completed, create a new run.
2. Check resources.
   a. If the DAC server cannot establish connections to the databases defined for that execution plan, abort the run.
   b. If there are no Informatica servers registered, abort the run.
   c. If the Informatica servers are registered and the DAC server is unable to ping them, abort the run.
3. Select initial tables.

a Find all the subject areas.

b Find all the table groups that belong the subject areas.

4 Recursively find all the table groups and their tables with the above list of tables.

5 Select initial tasks: Find all the tasks that load into the above selected tables. (Tasks whose target table is one of the selected tables and that are non-optional.)

6 Recursively select all tasks. Depending on the non-optional dependency rules, figure out the prerequisite tasks recursively.

7 Select optional tasks (with the tasks already chosen).

8 Select dependencies: Load dependencies and compute for each task the number of all Depends On tasks. This number will be decremented for tasks higher up in the hierarchy as each of the tasks complete. When the number of Depends On tasks reaches zero, the task itself will become eligible to be run.

9 Identify the task details. Iterate through all the tasks selected, compute the task dependencies.

10 Identify the list of source tables for change capture and create tasks for change capture and sync for Siebel sources. Iterate through all the tasks and find the primary or auxiliary tables and group them by source database connection.

11 Execute pre-ETL tasks. If there are errors, stop the run.

12 Execute the change capture process. If there are errors, stop the process.

13 Execute the dependency graph. If there are errors, stop all the tasks that depend on the failed tasks. Continue to execute all others, until there are no more to run, and then stop.

14 Execute the change capture sync process. If there are errors, stop the process.

15 Execute the post-ETL process. If there are errors, stop the process.

16 At the end of the ETL process, the DAC server updates refresh timestamps for all source tables (primary or auxiliary) and target tables. The refresh timestamps are the database timestamps. The ETL history timestamps (Start and End timestamps) are the DAC server timestamps.

About the DAC Subject Areas Tab

A subject area is a logical grouping of table groups that is related to a particular subject or application context.

The Subject Areas tab displays a list of subject areas in the top pane. This list includes the subject area name and an Inactive column, which indicates whether the subject area is active or inactive. Inactive subject areas do not participate in the ETL process.

Table 37 provides a description of the subtabs in the bottom pane.

Table 37. DAC Subject Areas Subtabs

Tab	Description
Edit	Allows you to edit the subject area selected in the top pane.
Description	Displays a description of the subject area selected in the top pane.
Table Groups	Displays a list of table groups associated with the subject area selected in the top pane. You can inactivate a table group by selecting the Inactive check box.
Tables (Read Only)	Displays a read-only list of the initial tables that participate in an ETL. For a complete list of tables participating in the ETL, use the Preview functionality. The list of tables contains the following column headings: ■ **Name.** Name of the target table. ■ **Main Table for Group.** If the table is a main table for a table group, the group name is displayed in this column. ■ **Table Type.** Table type as defined in the table properties.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Subject Areas tab is active.

Preview Command

This command generates a preview of all tasks and their dependencies by reviewing the table groups, tables, and tasks. Information provided includes the approximate order in which tasks are executed, a list of source tables, target tables, and tables for which an image will be built. The preview will include all tasks regardless of the source or target. Unlike the preview generated from the Execution Plans tab, this preview includes all the tasks, regardless of the source or target.

Copy Record

This command creates a copy of the selected subject area and also copies the references to table groups.

About the DAC Table Groups Tab

The Table Groups tab displays a list of table groups in the top pane. This list includes the name and type of the table group, the main table of the table group, and an Inactive column, which indicates whether the table group is active or inactive. Inactive table groups do not participate in the ETL process.

Table 38 provides a description of the tabs in the bottom pane.

Table 38. DAC Table Groups Subtab

Tab	Description
Edit	Allows you to edit the table group selected in the top pane.
Description	Displays a description of the table group selected in the top pane.
Tables	Displays a list of the initial tables that participate in an ETL. For a complete list of tables participating in the ETL, use the Preview functionality. The list of tables contains the following column headings: ■ **Name.** Name of the target table. ■ **Main Table for Group.** A read-only column showing if the table is a main table for another table group. ■ **Relationship Type.** Relationship type describing how the table relates to the main table. For documentation purposes only. ■ **Inactive.** Indicates whether the table's relationship to the table group is active or inactive.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Table Groups tab is active.

Copy Record

This command creates a copy of the selected table group and also copies the references to tables.

Preview Command

This command generates a preview of all tasks and their dependencies by reviewing the tables and tasks. Information provided includes the approximate order in which tasks are executed, a list of source tables, target tables, and tables for which an image will be built. Unlike the preview generated from the Execution Plans tab, this preview includes all the tasks, regardless of the source or target.

About the DAC Tables Tab

The Tables tab displays a list of tables in the top pane. This list includes the following information shown in Table 39.

Table 39. DAC Tables Tab List Columns

Column	Description
Name	Name of table.
Table Type	Indicates the type of target table. There is no differentiation for source tables.
Image Suffix	Applicable to Siebel source tables only. For more information, see "Generate Image and Trigger Scripts Command" on page 180.
Inactive	Indicates whether a table is active or inactive. Inactive tables do not participate in the ETL process.

Table 40 provides a description of the tabs in the bottom pane.

Table 40. DAC Tables Subtabs

Tab	Description
Edit	Allows you to edit the table selected in the top pane.
Description	Displays a description of the table selected in the top pane.
Database Connections	Displays a list of data sources associated with the table selected in the top pane. The Edit button allows you to add or remove data sources. The Table Refresh Date column displays the start time of the last ETL process that read from or wrote to the table selected in the top pane. (The time is that of the database server and not the DAC server.)
Indices	Displays a read-only list of indices that belong to the table selected in the top pane.
Source for Tasks	Displays a read-only list of tasks that use the table selected in the top pane as a source.
Target for Tasks	Displays a read-only list of tasks that use the table selected in the top pane as a target.
Conditional for Tasks	Displays a read-only list of tasks that are optional tasks for the table selected in the top pane.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Tables tab is active.

Copy Record

This command creates a copy of the selected table and also copies the references to database connections.

Generate Index Scripts Command

This command generates drop index, create index, and analyze table scripts for all the tables that participate in the ETL process. You can query for one or more tables, and the command will generate all the scripts for all the tables. The results are stored in the log\scripts directory. Files are generated that provide the following information: drop indices of type ETL, create indices of type ETL, drop indices of type Query, create indices of type Query, and analyze tables.

Generate Image and Trigger Scripts Command

This command generates change capture scripts for tables with defined image suffixes. The scripts include creation of the image tables to assist in the change capture process and their indices. You can also create the triggers for deletes. However, you should do so with caution because preconfigured tables for the change capture are created by applying SIF files on the Siebel transactional database (OLTP). Triggers for deletes should be used only for new source tables for which you want to use change capture. The scripts will be generated for the selected table or all the tables shown in the list applet. You can narrow down the tables of interest by querying. The results are shown in a text box, which can be supplied to the DBA for executing.

Generate View Scripts

This command generates change capture view scripts for full or incremental loads for any table that participates in the change capture process. This command can be useful when the change capture fails and you want to run a task manually. It can also be used for unit testing.

Generate Change Capture SQL

This command generates change capture SQL scripts for full or incremental mode for tables that participate in the change capture process. This command can be useful when the change capture fails and you want to run a task manually. It can also be used for unit testing

Import Database Tables Command

This command allows you to import table definitions from a selected database. You will be prompted to choose a database connection from a list that has already been defined in the DAC repository. When this command is executed, a list of tables in the data source is shown in a list applet. Check the Import flag in the list applet to select the tables that you want to import. The tables are then brought into the DAC repository. The default table type associated with the newly imported tables is Other. After you import the tables, change the table type to the appropriate type and associate the table with a database connection.

Import Indices Command

This command allows you to import index definitions from the database for one or more tables as listed in the result of the query. You will be prompted to select a data source. The client connects to the database and retrieves the index definitions for the table. The list includes new indices that may have been created in the database, indices whose definitions are modified in the database, or indices that do not exist in the database. To accept the changes, check the Merge? check box. When you select the list to be merged, one of the following actions will take place: If an index definition does not exist in the DAC repository, it will be created; if an index definition exists but the definition is the same, the new definition will be brought in; or if an index as defined in the repository does not exist is the database, it will be inactivated.

NOTE: In MSSQL Server databases, when you import indices with columns whose sort order type is "Desc," the sort order type appears in the DAC as "Asc." You have to set the sort order manually to "Desc."

About the DAC Indices Tab

The Indices tab displays a list of all the indices of all the data warehouse tables. It is recommended that you do not register any indices for source tables. During the ETL process, when a table is going to be truncated, all the indices as defined in the repository will be dropped before the data is loaded and will be created after the data is loaded automatically. While this improves the ETL performance, the preconfigured workflows have the bulk load option turned on. The bulk load will fail if there are indices on the table. Therefore, it is important to keep the index definitions in sync with the database. For example, if you create an index on the database, and it is not registered in the repository, the index will not be dropped and the load will fail.

For Teradata databases, only secondary indices should be registered in the DAC. You should not register primary indices or the more complex indices, such as single- and multi-table indices, because they cannot be dropped and recreated. You can use SQL commands to drop and create such tasks in the DAC.

The drop-down list to the right of the top pane toolbar allows you to filter the set of displayed indices based on the database type. To display all indices regardless of database type, select the option All.

Table 41 provides a description of the index properties displayed in the list.

Table 41. DAC Indices Tab List Columns

Column	Description
Name	Unique name with which the index is created.
Table Name	Table for which an index is created.
Index Usage	Usage of index: ETL or Query. An ETL index is typically used during the ETL process. A Query index is an index is used only during the reporting process. It is recommended that you have a clear understanding of when and where the index will be used at the time of registering the index.
Databases	Allows you to associate an index with a database type.

Table 41. DAC Indices Tab List Columns

Column	Description
# Unique Columns	For unique indices, the number of columns that will be unique.
Is Unique	Indicates whether the index is unique.
Is Clustered	Indicates whether the index is clustered. There can be only one clustered index per table.
Is Bitmap	Indicates whether the index is of the bitmap type.
Allow Reverse Scan	Applicable only for DB2-UDB databases. The index will be created with the Allow Reverse Scan option.
Always Drop & Create	Indicates whether the index will be dropped and created regardless of whether the table is being loaded using a full load or incremental load.
Table Space Name	Applicable only for Oracle databases. If a tablespace is mentioned, an index is created in that particular tablespace. If all indices go to a tablespace different from the data area, you can define it in the ETL Preferences (Tools > ETL Management> Preferences).
Inactive	Indicates whether an index is active or inactive. Inactive indices do not participate in the ETL process.

Table 42 shows which index properties are available for the different database types that are supported.

Table 42. Index Properties and Database Types Matrix

Column / Database	Is Unique	Is Clustered	Is Bitmap	Allow Reverse Scan	# Unique Columns
Oracle	X		X		
DB2	X	X		X	X
DB2-390	X				
MSSQL	X	X			

Table 43 provides a description of the tabs in the bottom pane.

Table 43. Indices Subtabs

Tab	Description
Edit	Allows you to edit the table selected in the top pane. The “Display database-specific properties” drop-down list allows you to specify a database type in order to display database-specific properties in the Edit window.
Description	Displays a description of the table selected in the top pane.
Columns	Displays the list of columns the index is made of. The list of columns contains the following column headings: ■ **Name.** Name of the column. ■ **Position.** Position of the column in the index. ■ **Sort Order.** Indicates whether the sort order is ascending or descending. ■ **Inactive.** Indicates whether the column is active or inactive.
Databases	For Siebel Data Warehouse version 7.7.1.3 and lower, indicates whether an index is applicable only to a certain type of database. If no database is indicated, the index will always be created. For Siebel Data Warehouse version 7.7.1.4 and higher, indicates the database types that apply to the index. If no database type is indicated, the index will not be created. The Index Override column allows you to define custom index creation syntax for a particular database type. The DAC server will use the syntax you define instead of generating an index statement based on the metadata. To enter the custom syntax, click in the Index Override column to open the text editor. The Generate Default Index Statement icon on the text editor toolbar allows you to generate the default statement, which you can edit. You can inactivate a database type by checking the Inactive check box.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Indices tab is active.

Copy Record

This command creates a copy of the selected index.

Advanced Custom Index Management

The DAC allows you to drop and recreate indices during the load process, which reduces the overall load time during a full load. The DAC drops and recreates indices based on the index definitions stored in the DAC metadata.

NOTE: More advanced index management needs to be handled outside of the DAC, such as creating partitioned indices on Oracle databases, which requires specific syntax, and creating single-table and multi-table join indices on Teradata databases. In such cases you can use DAC SQL tasks placed appropriately in the task dependencies, or you can use a pre-session or post-session script in Informatica. You must drop the partitioned index before a task runs to load data using the bulk loader, because the bulk loader may fail if there are indices on the target database.

If you need to modify the preconfigured indices with extra options or syntax, you must inactivate them in the DAC metadata so that the DAC server does not try to drop and recreate them. You can then manage these indices in the same manner as the advanced index management described above.

About the DAC Task Definitions Tab

The Task Definitions tab displays a list of tasks in the top pane. Table 44 provides a description of the task properties displayed in the list.

Table 44. Task Definitions List Columns

Column	Description
Name	A logical, unique name for the task.
Command for Incremental Load	A table can be loaded in Full Mode or Incremental Mode. Full Mode refers to data loaded for the first time or data that is truncated and then loaded. Incremental Mode refers to new or changed data being added to the existing data. The DAC maintains a last refresh timestamp whenever a table is changed during the ETL process. (You can view this timestamp under Design > Tables > Database Connections or Setup > Database connections > Tables.) If a table has a timestamp, the command appearing in this column is executed. If a table does not have a timestamp, the command for a full load is executed. If the execution type is Informatica, the workflow name is used as the command.
Command for Full Load	If a table has no last refresh timestamp, this command is executed.
Folder Name	For execution type of Informatica only. The folder in which the workflow resides. Note: The name cannot contain spaces.
Source DB	Source database that is queried.
Target DB	Target database that is populated.

Table 44. Task Definitions List Columns

Column	Description
Task Phase	Task phase of the ETL process. The DAC server uses the task phase to prioritize tasks and to generate a summary of the time taken for each of the phases.
Execution Type	Tasks are executed based on their execution type. The following types are supported: ■ **Informatica.** Task is invoked on an Informatica server using pmcmd. ■ **External Program.** Task is an operable program on the operating system where the DAC server is running. This program can be a batch file, shell script, or any other program that can be run like a bulk loader. ■ **SQL File.** Task is a SQL script in .xml or .sql format. For more information, see "Using SQL Files as an Execution Type in the DAC" on page 188. ■ **Stored Procedures.** Task is a stored procedure that has been defined on the databases. In addition, there are several internal execution types that you will not be able to select when creating new tasks. These tasks are categorized as either internal change capture tasks or internal data warehouse tasks; all of these tasks are color-coded in pink in the Task Definitions tab. ■ IMG_BUILD - internal change capture. If you are using multiple Siebel transactional sources, you cannot change the behavior of the change capture process. This task requires change capture tables to be created on the other sources also. When adding additional Siebel sources, go to Setup > Database Connections and select the appropriate Siebel source, right-click and select Change Capture Tasks. This action generates change capture tasks. Use this same action to disable or delete change capture tasks. ■ IMG_SYNC - internal change capture. If you are using multiple Siebel transactional sources, you can create this task for the additional tasks for doing similar change capture sync processes. You cannot change the behavior of the change capture sync process. This task requires change capture tables to be created on the other sources also. This task should be used with discretion for Siebel sources only.

Table 44. Task Definitions List Columns

Column	Description
Execution Type	■ QUERY_INDEX - internal data warehouse. This task allows you to alter when the Query indices are created. The DAC server drops all indices before loading when the CreateQueyIndexesAtTheEnd setting is set to True. When this setting is set to False, all the indices, regardless of the index type, get created as part of the task that does the loading. ■ UPDATE_ETL_PARAM - internal data warehouse. This task is used for only to update W_PARAM_G from the DAC server. This task populates the preferences to the W_PARAM_G table in the Data Warehouse by querying values defined in the DAC repository. Because only one Data Warehouse per DAC repository is supported, this execution type should not be chosen for any task.
Build Image	Applicable for Siebel transactional sources only. When this check box is selected, the change capture for the primary/auxiliary source tables executes.
Truncate Always	When this check box is selected, the target tables are truncated regardless of whether a full or incremental load is occurring. Any indices registered for this table are dropped before the command is executed and are recreated after the command completes successfully. When indices are dropped and created, the table is analyzed so that the index statistics are up-to-date.
Truncate for Full Load	When this check box is selected, the target tables are truncated only when a full load is occurring. Any indices registered for this table are dropped before the command is executed and are recreated after the command completes successfully. When indices are dropped and created, the table is analyzed so that the index statistics are up-to-date. When the Truncate Always option is selected, this option is unnecessary.
Optional	Indicates that a task is optional and will be executed only when the table listed in the Conditional Tables subtab is included in an execution plan. For example, the UpdateRecencyCat in Person Dimension task is optional and is associated with the Order Item fact table, which is listed in the Conditional Tables subtab. The UpdateRecencyCat in Person Dimension task will only be executed when the Order Item fact table is included in an execution plan.
Continue on Error	When this check box is selected, if the command fails, the dependent tasks are not stopped. However, if any auto-generated tasks fail, the dependent tasks are stopped.
Inactive	When this check box is selected, the record is inactive. Inactive tasks do not participate in the ETL process.

Table 45 provides a description of the tabs in the bottom pane.

Table 45. Task Definitions Subtabs

Tab	Description
Edit	Allows you to edit the table selected in the top pane.
Description	Displays a description of the task selected in the top pane.
Source Tables	Displays the list of tables used for getting data for the task selected in the top pane. The list contains the following column headings: ■ **Name.** Name of the table. ■ **Primary.** Indicates whether the table is a primary source of data. ■ **Auxiliary.** Indicates whether the table is a secondary source of data. **Note:** If a table is marked as Primary or Auxiliary and the Build Image property of the task is selected, the change capture process is invoked. There are special tasks that force the base table data to be extracted when data in auxiliary tables change. A table can be neither Primary nor Auxiliary but still be used for getting some attributes to populate a dimension or fact table. The changes in these kinds of source tables are not reflected in the dimension or fact table once the data is populated.
Target Tables	Displays the list of tables where data is loaded for the task selected in the top pane. The list contains the following column headings: ■ **Name.** Name of the table. ■ **Primary.** Indicates whether the table is a primary target table. For documentation purposes only.
Conditional Tables	Displays a list of tables that, if included in an execution plan, cause the optional task selected in the top pane to be executed. For example, the Order Item fact table is a conditional table associated with the optional task called UpdateRecencyCat in Person Dimension. The UpdateRecencyCat in Person Dimension task is executed only when the Order Item fact table is included in an execution plan. The list contains the following column headings: ■ **Name.** Name of the conditional table. ■ **Inactive.** Indicates whether the conditional table is inactive.

Table 45. Task Definitions Subtabs

Tab	Description
Depends On Tasks	Displays a list of tasks that are prerequisite for the execution of the task selected in the top pane. If any of the prerequisite tasks fail, the task selected in the top pane will be marked as Stopped. The list contains the following column headings: ■ **Name.** Name of the prerequisite task. ■ **Task Type.** Type of task (read only). ■ **Optional Flag**. When this flag is selected, the prerequisite task optionally depends on the task selected in the top pane. For example, task 1 loads table A, and task 2 loads table B and optionally depends on task 1. If an execution plan requires both tables A and B to be loaded, task 2 will wait for task 1 to successfully complete. If an execution plan requires only table B to be loaded, task 2 will be executed immediately, and task 1 will not be executed.
Dependent Tasks	Displays a list of tasks that are dependent on the task selected in the top pane. These tasks wait for the successful completion of the current task before they are executed. The list contains the following column headings: ■ **Name.** Name of the dependent task. ■ **Task Type.** Type of task (read only). ■ **Optional Flag**. See the description for Depends On Tasks.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Task Definitions tab is active.

Copy Record

This command creates a copy of the selected task and also copies the references to the tasks it depends on and source and target tables.

Using SQL Files as an Execution Type in the DAC

There are two types of custom SQL files that can be executed through the DAC: XML formatted .xml files, and plain text .sql files. For examples of XML and SQL files, go to the DAC\CustomSQL folder.

XML Formatted Files

An XML file consists of a set of SQL statements for which the name, type, and Continue on Fail option are defined using XML attributes. The set of SQL statements is in a CDATA section which allows for special characters (like <, >, \) to be used without breaking the XML structure. A CDATA section looks like the following:

```
<![CDATA[this is a place for a SQL statement]]>
```

```
An example of an XML file follows:
```

```
<?xml version="1.0"?>
```

```
<CustomSQLs>
```

```
<sql name="Create Temp" type="SQL" continueOnFail="true">
```

```
<![CDATA[CREATE TABLE w_etl_temp (row_wid varchar(50))]]>
```

```
</sql>
```

```
<!-- This is how a comment section looks in XML -->
```

```
<!-- It will be ignored-->
```

```
<sql name="Update Temp" type="SQL">
```

```
<![CDATA[UPDATE w_etl_temp SET row_wid = 'qwe' WHERE row_wid LIKE '5*']]>
```

```
</sql>
```

```
<sql name="Drop Temp" type="SQL">
```

```
<![CDATA[DROP TABLE w_etl_temp]]>
```

```
</sql>
```

```
</CustomSQLs>
```

This example consists of three SQL statements: Create Temp, Update Temp, and Drop Temp. These names will be used in DAC run task detail description to give details of the errors encountered during execution (if any). In this particular example all three statements are of type SQL. Regular update SQL statements and the Stored Procedure type can also be used.

If the Continue on Fail option is set to True for a given statement, the execution will continue even if the task fails. If the Continue on Fail option is set to False for a given statement, or is omitted, the subsequent statements will not be executed and the Task Run status will be Failed.

Plain Text SQL Files

Plain text SQL files consist of a set of SQL statements (no stored procedure calls). The SQL statements are separated by a semicolon (;), and comment tags are allowed (//, /* comment */, --). If any of the SQL statements fail, the Task Run status will be Failed.

An example of a plain text SQL file follows:

```
CREATE TABLE w_etl_temp (name varchar(50))
```

```
;
UPDATE w_etl_temp
SET name = 'that's right' //this line demonstrates the use of ' in a text area
WHERE name LIKE 'gone fishing%';

/*
*some
*query
*statement
*/
 SELECT * FROM w_etl_temp
;
DROP TABLE w_etl_temp
;
/*end of file*/
```

About the DAC Setup View

The Setup View provides access to functionality related to setting up DAC system properties, Informatica servers, database connections, and email notification.

This section includes the following topics:

About the DAC System Properties Tab

The DAC System Properties tab allows you to configure various properties that determine the behavior of the DAC server. The top pane displays a list of the properties and their current values. For more information about the DAC system properties, see "Setting Up DAC System Properties" on page 67.

About the DAC Informatica Servers Tab

The Informatica Servers Tab allows you to register one or more Informatica servers and specify how many workflows can be executed in parallel on each server. The DAC server automatically load balances across the servers. For more information about Informatica server properties, see "Creating and Dropping Data Warehouse Tables on a Teradata Database" on page 77.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Informatica Servers tab is active.

Test Connection

This command allows you to test whether the Informatica server is running. The DAC client will attempt to delegate the test to the DAC server. If the DAC server is not running, you will be asked whether you want to test the connection on the local client machine. The results of a test on the local client machine may not be true for the DAC server environment.

Copy Record

This command creates a copy of the selected record.

About the DAC Database Connections Tab

This tab provides access to the database connection properties. The top pane Indices tab displays a list of the properties and their current values. You can edit the property values in the top pane by double-clicking in a cell.

Table 46 provides a description of the database connection properties.

Table 46. Database Connection Properties

Property	Description
Name	Logical name for the connection. (The metadata that comes with Siebel Analytics Data Warehouse has two data sources: OLTP, which points to the Siebel transactional database, and the data warehouse.)
Type	Possible values are: ■ Source ■ Warehouse ■ Informatica Repository ■ DAC Repository ■ Other

Table 46. Database Connection Properties

Property	Description
Connection Type	Type of database connection. Possible values are the following: ■ **Oracle (OCI8).** Connect to Oracle using the tnsnames entry. ■ **Oracle (Thin).** Connect to Oracle using thin driver. ■ **DB2**. DB@ UDB database. ■ **DB2-390.** DB2 390 database. ■ **MSSQL.** Microsoft SQL Server. ■ **Teradata.** Teradata database.
Connection String	If you are using... ■ Oracle (OCI8) use the tnsnames entry. ■ Oracle (Thin) use instance name. ■ SQL server use the database name. ■ DB2-UDB/DB2-390 use the connect string as defined in the DB2 configuration. ■ Teradata use the database name.
Table Owner	Name of table owner.
Password	Password for table owner.
Max Num Connections	Maximum number of database connections this connection pool can contain. For information about determining how to set this value, see "Determining the Number of Transactional and Data Warehouse Database Connections" on page 167
DBHost	Host machine where the database resides. This field is mandatory if you are using Oracle (Thin) or MSSQL but is not required if you are using Oracle (OCI8), DB2, or DB2-390.
Port	Port where the database receives requests. Required for Oracle (Thin) and MSSQL databases. Not required for Oracle (OCI8), DB2, or DB2-390 databases.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Database Connections tab is active.

Test Connection

This command allows you to test the connection to the database. The DAC client will attempt to delegate the test to the DAC server. If the DAC server is not running, you will be asked whether you want to test the connection on the local client machine. The results of a test on the local client machine may not be true for the DAC server environment.

Internal Tasks

This command allows you to manage internal tasks. The following options are available:

- **Source (Change Capture Tasks)**. Generates internal tasks for performing change capture for Siebel sources only.
- **Data Warehouse (W_PARAM_G update, Query Index Creation)**. The supported configuration is one data warehouse for each repository; however, you can create aliases for the same database to be able to filter tasks. To do so, create either of the following tasks for the aliases:
  - Update W_PARAM_G. This task updates the W_PARAM_G table with the ETL preferences defined in the DAC.
  - Create Query Indexes. If the system preference CreateQueryIndexesAtTheEnd is set to True, then this task will hold the details of all the created query indexes.

About the DAC Email Recipients Tab

This tab allows you to set up a list of email addresses that will be notified about the status of the ETL process. The top pane displays a list of the properties and their current values. You can edit the property values in the top pane by double-clicking in a cell. For more information about the Email Recipients properties, see "Configuring Email in the DAC Server (Optional)" on page 97.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Email Recipients tab is active.

Copy Record

This command creates a copy of the selected record.

About the DAC Execute View

The Execute View provides access to functionality that allows you to run, schedule, and monitor execution plans.

This section includes the following topics:

About the DAC Execution Plans (Read Only) Tab

The Execution Plans tab in the Execute view displays, in read-only mode, the available execution plans. This information is the same as that displayed in the Execution Plans tab in the Design view. For more information, see "About the DAC Execution Plans Tab" on page 173.

About the DAC Current Run Tab

This tab displays a list of queued, running, and failed current ETL processes in the top pane. This list includes comprehensive information about each process.

Table 47 provides a description of the information provided in the top pane.

NOTE: Once an ETL process completes, it is accessible from the Run History tab.

Table 47. Current Run Tab Columns

Column	Description
Name	Name of the ETL process
Execution Plan Name	The execution plan whose run-time instance is this record. This field is read only.
Process ID	ID for the process. This value is an integer that is incremented by 1 for every run. This value is stored as ETL_PROC_WID in all the Data Warehouse tables. This field is read only.

Table 47. Current Run Tab Columns

Column	Description
Run Status	Status of the run. The possible values are the following: ■ Queued. Tasks for which the Depends On tasks are not yet completed. Displayed in yellow in the Current Run list. ■ Runnable. Tasks for which the Depends On tasks have completed and are ready to be run but are waiting for an Informatica slot to be available. ■ Running. Tasks for which the Depends On tasks have been completed, have gotten an Informatica slot, and are being executed. Displayed in blue. ■ Failed. Tasks that were executed but encountered a problem. Displayed in red. ■ Stopped. Tasks for which one or more Depends On tasks have failed. ■ Completed. All tasks have completed without errors. Displayed in green.
Total Number of Tasks	Total number of tasks for this run. This field is read only.
Number of Failed Tasks	Sum total of tasks that have failed and that have stopped. This field is read only.
Number of Successful Tasks	Number of tasks whose status is completed. This field is read only.
Number of Tasks Still in Queue	Number of tasks whose prerequisite tasks have not completed, and the number of tasks whose prerequisite tasks are completed and are waiting for resources. This field is read only.
Start Timestamp	Start time of the ETL. Reflects the start time of every ETL attempt. For example, if the ETL fails and is run again, it gets a new start timestamp. The history of attempted runs is maintained in the audit trail for the run. This field is read only.
End Timestamp	End time of the ETL. Reflects the end time of every ETL attempt. For example, if the ETL fails and is run again, it gets a new end timestamp when the ETL is complete. The history of attempted runs is maintained in the audit trail for the run. This field is read only.
Duration	A calculated field that shows the difference between start and end time stamps.
Status Description	Displays messages generated during run time. You can add notes to this field for Completed runs.

Table 48 provides a description of the tabs in the bottom pane.

Table 48. Current Run Subtabs

Tab	Description
Edit	Displays the status of the ETL selected in the top pane.
Description	Provides a description of the ETL process, including status and environment information. This description is updated at the beginning of the ETL process and when the process is complete.
Tasks	Displays run-time instances of the tasks. As the execution proceeds, the tasks are executed based on the dependency rules and some prioritization. As tasks complete, the tasks that depend on the completed tasks are notified and once their dependencies are completed, they become eligible to run. If a task fails, the administrator can address the failure and then requeue the task or mark it as completed. The DAC server polls for any changes in the failed task's detail status. If a failed task detail is queued, the task itself gets back into the ready to run queue and all its dependent tasks get into the queued status. The rules of prioritization are as follows: ■ Tasks with no dependencies are executed first. ■ If a task has failed and has been requeued, it gets the maximum priority. ■ Tasks with greater phase priorities are executed next. When several tasks of the same phase are eligible to run, the tasks with greater task priorities are executed next.
Summary (Read Only)	Provides a summary (based on dynamic SQL) of the ETL selected in the top pane. A start and end time for each phase is included.
Task Details	Provides a execution details about each task associated with the ETL selected in the top pane.
Audit Trail (Read Only)	Provides the history of the ETL selected in the top pane.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Current Run tab is active.

Start

This command restarts the selected ETL, after the ETL has failed, stopped, or been aborted.

Abort

This command causes an ETL in progress to abort. All currently running tasks will be aborted. The status of queued tasks and the ETL itself will change to Stopped.

Auto Refresh

This command allows you to turn on and off the automatic screen refresh functionality and set the refresh interval.

Get Log File

This command fetches the log file for this run from the DAC server and saves it in the ServerLog folder.

Mark as Completed

This command changes the status of a stopped or failed ETL to Completed. In the audit trail for this ETL, the status is Marked as Completed. Use this command with caution; it can cause the data warehouse to be inconsistent.

Analyze Run

This command saves a description of the run as an HTML file in the Log/Statistics folder.

About the DAC Run History Tab

The Run History tab displays information about completed ETL processes. The information displayed in the top and bottom panes is the same as in the Current Run tab. For a description of the information displayed in the Run History tab and the actions available, see "About the DAC Current Run Tab" on page 194.

About the DAC Scheduler Tab

The Scheduler tab allows you to schedule ETL processes to be run either once at a later time or periodically. When you schedule an ETL or make changes to a schedule, the DAC server picks up the information from the DAC client. The DAC server polls the DAC repository for changes periodically at a frequency set in the DAC system properties. For information about DAC system properties, see "Setting Up DAC System Properties" on page 67.

The top pane of the Scheduler tab lists ETL processes that have been scheduled. The bottom pane allows you to schedule an ETL process.

Table 49 provides a description of the information displayed in the top pane list.

Table 49. DAC Scheduler Tab List Columns

Column	Description
Name	User defined name of the schedule.
Execution Plan	Execution plan scheduled.
Recurrence	Indicates how often the schedule will be run.
Inactive	Indicates whether the schedule is active or inactive.

Actions Available

The following actions are available in the top pane toolbar and in the right-click menu when the Scheduler tab is active.

Copy Record

This command creates a copy of the selected record.

Scheduling an ETL

Use this procedure to schedule an ETL in the DAC.

To schedule an ETL

1. In the DAC, navigate to the Scheduler tab.

   The current list of schedules appears in the top pane.

2. Click New in the top pane toolbar.

   The Edit tab in the bottom pane becomes active.

3. Enter a name for the schedule.
4. Select an execution plan.
5. If you want the schedule to run once, select the Run Only Once check box, and then select a start and end date and time.
6. To create a periodic schedule, select a recurrence pattern, and enter the appropriate date and time parameters.
7. Click Save.

About the DAC Validation Tab

The Validation tab allows you to view the results of a validation. The following kinds of validation are available:

- Resource Validation: Checks to see if the database connections are validated. This will be checked as a first step of the ETL process. If the database connections are not valid, the ETL will abort. This validation is also available on the DAC client through Setup > Database Connections > Test Connection functionality.
- Table Validation: Checks to see if the tables do exist in the database. If the tables do not exist, the ETL will fail. This is not checked during the ETL run. The check also includes verifying the existence of image tables where applicable.
- Index Validation checks the following:
  - a If all the columns in the index definition exist on the table in the database.
  - b If the index does exist in the database, the validation compares the index definition and points out the differences.
  - c If there are multiple indices with the same definitions (columns in the same order) for a table.
  - d If there are indices with the number of columns greater than 255.
  - e If '# Include columns' is greater than the number of columns provided in the index column list.
  - f Checks to see if the index column has no columns.
  - g If the number of columns is more than 16. Some databases allow for more than 16 columns. However, the user should inspect the warning and verify its validity.
  - h If an index on a table was created on the database but not defined in the repository. This may cause the ETL process to fail, because all the indices need to be dropped prior to the load, otherwise the bulk loaders may fail.
- Informatica Workflow Validation: Verify for the task definitions of type Informatica to match the 'Command for Full Load' and 'Command for Incremental Load' do exist in the Informatica repository. For this validation to work, configure the system preferences 'Main Informatica Repository' to point to the database where the Informatica repository resides with the logical name given to the database under Setup-> Database Connections.
- Check for circular dependencies: While defining task dependencies, care should be taken to not to define circular dependencies. For example task A depends on task B, and task B depends on task C, and task C depends on task A. Such occurrences will be identified. If these errors do exist, the ETL will fail.

10 Customizing the Siebel Data Warehouse

This chapter describes concepts and techniques for customizing the Siebel Data Warehouse. It also provides an overview of how incremental Siebel application operational data is captured and how it can be used for adding columns and tables to the Siebel Data Warehouse. It also describes customization techniques using extension tables and Informatica mapping templates.

This information applies only to Siebel Analytics applications versions 7.7 and higher.

NOTE: Customization should be done only by those experienced in Informatica and Siebel Analytics applications.

This chapter contains the following topics:

Siebel Data Warehouse Customization Scenarios

In customizing the Siebel Data Warehouse, various scenarios are available to you based on whether the data source is the Siebel transactional database (OLTP) or data from a non-Siebel source. The matrix in Figure 15 shows categories of customization scenarios. The kinds of customizations available for each category are described in Table 50. For instructions on how to perform the customizations, see "Customization Steps for Siebel Data" on page 210 and "Siebel Data Warehouse Customization Steps for Non-Siebel Data" on page 215."

For detailed information about ETL processes and tables, see *Siebel Data Warehouse Data Model Reference*.

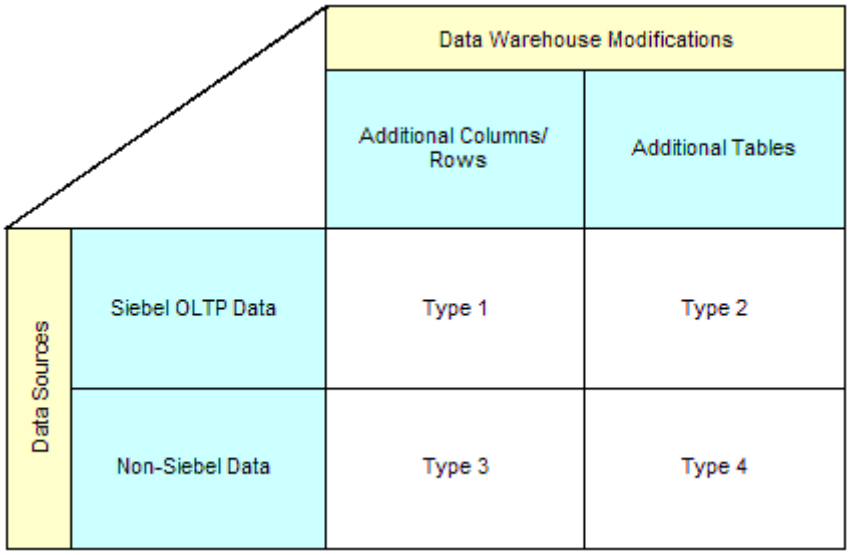


Figure 15. Data Warehouse Customization Scenarios

Table 50. Types of Data Warehouse Customization Scenarios

Type	Modification
Type 1	Add a column to the data warehouse from a primary source table in the Siebel transactional database (OLTP).
	Using Siebel transactional database data, add a column to the data warehouse from a related source table in the transactional database.
	Using Siebel transactional database data, modify an existing column in the data warehouse.
Type 2	Using Siebel transactional database data, add a new dimension table and use it with an existing fact table.
	Using Siebel transactional database data, add a new fact table.
	Using Siebel transactional database data, add a new dimension table for a new fact table.
Type 3	Using non-Siebel data, add new data as a whole row into an existing dimension table.
	Using non-Siebel data, add a column for an existing dimension table.
	Using non-Siebel data, add a column for an existing fact table.

Table 50. Types of Data Warehouse Customization Scenarios

Type	Modification
Type 4	Using non-Siebel data, add a new dimension table.
	Using non-Siebel data, add a new dimension table and use it with an existing fact table.
	Using non-Siebel data, add a new fact table.

Siebel Data Warehouse ETL Process Concepts

This section includes a detailed description of the ETL process. The following topics are covered:

About Siebel Data Warehouse Extension Tables

The extension tables are provided with the Siebel Data Warehouse exclusively to help customize the data warehouse tables.

CAUTION: Do not customize existing mappings or the database schema. If you need to customize the standard mappings, create a new mapping that loads the extension table instead.

For every extension table, there is an extension staging table that is populated by the custom SDE process. The custom SIL process sources data from this staging table and populates the final extension table. For information about how to use extension tables, see "Customization Steps for Siebel Data" on page 210 and "Siebel Data Warehouse Customization Steps for Non-Siebel Data" on page 215.

The extension tables have a 1:1 relationship with their corresponding dimension and fact tables.

The data warehouse extension tables are different from the Siebel transactional database extension tables in the following ways:

- Data warehouse extension tables are not maintained by Siebel Tools.
- There is no relationship between data warehouse extension columns and the corresponding transactional database extension columns. For example, attr01 in S_ORG_EXT_X is not the same as attr01 in W_ORG_DX.
- Mappings have to be implemented explicitly to populate the extension columns.

Using the Extension Tables

The extension tables have a 1:1 mapping between the dimension and fact records to the extension records. To insert a record into a dimension or fact table, choose the appropriate column and write a new mapping that uses the ETL server's incremental logic to load the staging table. When populating data from the staging table to the final dimension or fact tables using the INTEGRATION_ID and the DATASRC_NUM_ID, find the primary key of the dimension table and use that as the primary key of the extension table.

Naming New Extension Tables

When you create a new extension table, use the same naming convention as is used in the Siebel Data Warehouse by adding the prefix WC to the table name, for example, WC_*TABLENAME*_DS. This will simply future upgrades to the Siebel Data Warehouse.

NOTE: If a new table is created, you need to register the table and its indices in the DAC repository. Also, if you modify indices on existing tables, you need to register the indices.

NOTE: If you are creating tables on a DB2-UDB database, make sure the Not Logged Initially option is enabled.

Creating New Dimension or Fact Tables

You can create new tables using the same logic as is used for extension tables.

CAUTION: The data in any customized table will face upgrade issues.

If you are creating a new dimension or fact table, use the required system columns that are part of each of the data warehouse tables to maintain consistency and the ability to reference existing table structures. When you create a new table, you need to register the tables and its indices in the DAC repository.

Required Columns

For main and extension staging tables the following columns are required:

- **INTEGRATION_ID.** Stores the primary key or the unique identifier of a record as in the source table.
- **DATASRC_NUM_ID.** Stores the Data source from which the data is extracted. For example, Siebel OLTP is 1.

For data from any external data source assign a number greater than 1.

For dimension and fact tables, the required columns are the INTEGRATION_ID and DATASRC_NUM_ID columns as well as the following:

- **ROW_WID.** A sequence number generated during the ETL process, which is used as a unique identifier for the data warehouse.
- **ETL_PROC_WID.** Stores the ID of the ETL process information. The details of the ETL process are stored in the S_ETL_RUN table on the transactional database (OLTP) side and W_ETL_RUN_S table on the data warehouse side. This is also the Process ID on Current Run/Run History screen in the DAC.

The required columns for dimension and fact extension tables are ROW_WID and ETL_PROC_WID.

About the Siebel Data Warehouse DATASRC_NUM_ID Column

All the tables in the Siebel Data Warehouse schema have DATASRC_NUM_ID as part of their unique user key. This means that the rows can be loaded in the same data warehouse tables from different sources if this column is given a different value for each data source.

NOTE: The DATASRC_NUM_ID value of 1 is reserved for the main Siebel data source and cannot be used for other sources.

Working with the Custom Folder of the Siebel Data Warehouse

The Siebel Data Warehouse provides a folder within the repository specifically designed for customized changes to an implementation. This folder is named CUSTOM_DW_REP. Using the Informatica Designer, make modifications only in the CUSTOM_DW_REP folder.

Do not change objects in the SIEBEL_DW_REP folder unless it is explicitly mentioned. This folder may be overwritten in future upgrades. All changes should be placed in the CUSTOM_DW_REP folder within the Informatica Designer.

The easiest way to modify an object is to copy an existing object from SIEBEL_DW_REP into CUSTOM_DW_REP and to reuse existing business components, source and target definitions, transformations, mapplets, and mappings before you make any changes.

NOTE: If source tables are extended (after copying into the CUSTOM_DW_REP folder), then the source tables require manual editing in the Informatica Designer. Do not import the tables from the database, because it changes the source table definitions for the entire Siebel Data Warehouse.

When importing new tables from the database into the CUSTOM_DW_REP folder, use the Siebel Data Warehouse and Siebel Transaction Database ODBC database connections to connect to the source and target databases.

After importing new table definitions, change the database type to Oracle in the Informatica repository, regardless of the database platform you are using. This has no effect on the choice of a relational database. This step is very important because in Informatica if the database type of the source tables are not identical, all mappings and workflows that refer to the source tables will be invalid.

Creating Custom Informatica Workflows

You must create custom workflows for all customized mappings. The general requirements for creating custom workflows are as follows:

- Create the workflow such that each loads only one table. This helps to integrate the workflow with the DAC.
- The workflow name should match a session name that is used inside the workflow. This helps the DAC to collect certain statistics.
- Fail parent if this task fails should be checked for all the sessions within a workflow.
- Fail parent if this task does not run should be checked for all the sessions within a workflow.
- The Stop on Errors parameter should be set to 1 for all sessions with a workflow. This parameter is located in the Error Handling area of the Config Object tab in Informatica Designer.
- Set the appropriate source and target connection values in Informatica Designer.
- If the workflow is going to be used for full load command, you can choose to load in bulk mode (applicable only to Oracle and DB2-UDB databases). If the workflow is going to be used for the full load command in the DAC, then in the Properties tab of Informatica Designer, set the Target Load type to Bulk. This requires that the target table does not have any indices during the load. The DAC will drop the indices automatically; no action is required on your part.
- For all entities, such as dimension and fact tables, create two workflows, one to be used for a full load and the other to be used for an incremental load. This provides an opportunity to tune each of these load scenarios.
- If the workflow is designed for loading a dimension in a full mode, make sure the workflow also includes a session for the unspecified row creation.
- You can decide to truncate the target table in the workflow, but it is necessary that while defining a task in the DAC, you choose appropriate truncate option. This helps the DAC to decide whether to drop and create indices on the table.
- If some sessions have to be run in serial order and if all the sessions need to be rerun upon failure of the workflow, then you should consider designing a single workflow that has sessions run in serial. If upon failure, not all of the sessions need to be run again, then consider designing separate workflows and define dependencies in the DAC.
- Use views created by the DAC as the main source table in all extract processes.
- If a workflow uses a list of values, make it depend on the Fix List Of Values and Copy to OLTP task definition in the DAC.
- If a workflow uses exchange rates, make it depend on the Extract for ExchRate task definition in the DAC.
- If a workflow uses cost lists, make it depend on the Load Cost Lists task definition in the DAC.
- You can import the workflows into the DAC repository's temporary area while defining tasks.

Customization Categories

There are three categories of customizations:

- **Customizing preconfigured workflows.** Copy the mapping from the Siebel_DW_Rep folder into the Custom folder. Modify it as necessary. Create new workflows in the Custom folder. In the DAC, search for the task that executes the workflow and change the folder to the Custom folder. Update the command for full and incremental loads appropriately if the workflow names have been changed.
- **Creating new workflows to populate additional custom entities.** The custom workflows can be plugged into an ETL process by registering them in the DAC. A new task needs to be registered in the DAC, with the appropriate properties set. Also, you must define source and target tables, task definitions, and dependencies. In the Depends On Tasks tab, select the list of tasks that are prerequisite for the current task to execute. In the Dependent Tasks tab, define the tasks that need to wait for the successful completion of this task. Set the Optional flag to True.
- **Pruning preconfigured ETL processes.** Pruning is the process in which you modify a preconfigured execution plan to meet your needs. You can prune an execution plan by using the DAC to inactivate certain components of the execution plan, such as subject areas, table groups, tables, tasks, and dependencies. Inactivating the component means the component does not participate in the ETL process when the execution plan is run. For more information about pruning, see "Handling ETL Failures with the DAC" on page 160.

Important Notes About Siebel Data Warehouse Customizations

All custom work, unless stated otherwise, must be done in the Custom folder so that the custom work can be preserved during an Informatica repository upgrade. Doing work on the standard folder should be avoided whenever possible. Whenever custom work is done in the standard folder, a careful note of the custom work should be recorded so that the customizations can be reapplied whenever the Informatica repository is upgraded. An upgrade of the Informatica repository overrides any changes to the standard folder.

Additional Customization Instructions

- **Table definitions in Informatica.** Make sure that the SQL style is set to Oracle while importing the table definitions from external data sources. Even if the actual data source is of another database type, such as DB2 or MSSQL, it does not affect the logic of how data gets loaded.
- **Update strategy.** For loading new fact and dimension tables or loading data into the existing tables, design a custom process on the source side to detect the new and modified records. The SDE process should be designed to load only the changed data (new and modified). If the data is loaded without the incremental process, the data that was previously loaded will be updated again, which is a costly process. For example, the logic in the OTB SIL mappings looks up the destination tables based on the INTEGRATION_ID and DATASRC_NUM_ID and returns the ROW_WID if the combination exists, in which case it updates the record. If the lookup returns null, it inserts the record instead.
- **ETL process.** When using multiple sources for the data warehouse, you can decide to load from all of them at the same time or at different time frequencies using different execution plans.

- **Truncating target tables.** Truncating should be done through the DAC. A single task has place holders for a full load, and one for an incremental load.
  - For the SDE workflows, the commands for full load and incremental load are the same. They should have Truncate Always flag checked. For these kinds of tasks, the command for full load and incremental load are based on the same mapping.
  - For SIL workflows, the command can be different for full and incremental loads. They should have the Truncate For Full Load option checked. When a table gets truncated, the indices are automatically dropped and created after the data is loaded. And so, the workflow for full load command can have the Bulk Load option turned on, and, therefore, you can have an optimized version of the mapping that does not need to look up for finding if a record needs to be inserted or updated.
  - Auxiliary tasks need to be run only during incremental mode. So, for these tasks, the full load command is empty. No truncate options should be set.

	SDE Sessions	SIL Sessions	Aux Sessions
Truncate Option	Truncate Always	Truncate for Full Load	None

- **ETL_PROC_WID.** Use the same ETL_PROC_WID in W_PARAM_G table in custom mappings. ETL_PROC_WID is a reference key to Run History in the DAC. To use the same ETL_PROC_WID, copy the reusable lookup (called LKP_ETL_PROC_WID) defined in the Siebel_DW_Rep folder. The input to the lookup is a constant (hard coded to 1).
- **DATASRC_NUM_ID.** Even though preconfigured mappings look up W_PARAM_G for getting this value, for customizations that move data from other sources, you should hard code the value with a number other than 1, which is reserved for the main Siebel transaction database in custom mappings.
- **Creating indices and naming conventions.** Staging tables typically do not require any indices. Use care to determine if indices are required on staging tables. Create indices on all the columns that the ETL will use for dimensions and facts (for example, ROW_WIDs of Dimensions and Facts, INTEGRATION_ID and DATASRC_NUM_ID, and flags). Carefully consider which columns or combination of columns filter conditions should exist, and define indices to improve query performance. Inspect the OTB objects for guidance. Name all the newly created tables as WC_. This helps visually isolate the new tables from the OTB tables. Keep good documentation of the customizations done; this helps when upgrading your data warehouse. Once the indices are decided upon, they should be registered in the DAC, either manually or by right-clicking on the certain table and invoking the Import Indices command.
- **Currency.** For currency-related data, populate the table with the base currency and exchange date field (in order to convert the data appropriately). The data for currency conversion should be maintained in the main Siebel data source. (The currency data is maintained by converting all the currency information into a single Base Currency code specified in the DAC's ETL preferences.)
- **Day Dimension.** For the data that is related to W_DAY_D, use the reusable transformation EXP_DAY_DIMENSION_FK_RESOLUTION. Note that this transformation will take a date as input and return the foreign key to the Day dimension as output in the appropriate format (YYYYMMDD). Copy the reusable transformation and use it.

- **List Of Values.** This applies in particular to Type 1 and 2. The preconfigured columns that depend on a list of values have a language-dependent column and language-independent column. Use the mapplet MPLT_LOV_TRANSLATION to populate the language-dependent and independent columns in the dimension tables. For fact tables, use MPLT_LOV_D_ROW_WID to create a new foreign key to the LOV dimension. One could also deal with translations directly in SQL overrides to improve performance.

Standard Templates for Siebel Data Warehouse Customizations

This section describes the standard templates.

CustomTemplateSDE_WaveDimension

Using the SDE mapping, the ATTRIB_02 column of the extension stage table (W_PERSON_DSX) is populated with the value from S_CONTACT.RACE source column. This is a straightforward mapping. The different transformations are the following:

- **SQ_S_CONTACT.** The SQL in the SQL override section contains the link to the view created by the DAC so that only the incremental data is captured.
- **EXPTRANS.** The lookup (LKP_DATASOURCE_NUM_ID) obtains the DATASOURCE_NUM_ID from the W_PARAM_G table. The W_PARAM_G is populated in the beginning of the ETL by the DAC. Note that all the system columns are populated in addition to the attrib column. The ATTRIB_01 column is populated with the string Contact, because the combination of INTEGRATION_ID, DATASOURCE_NUM_ID, and CONTACT_TYPE is used to look up the ROW_WID value. In most of the dimensions, only the INTEGRATION_ID and DATASOURCE_NUM_ID values are used for this purpose.

CustomTemplateSIL_WaveDimension

Using the SIL mapping, the ATTRIB_02 column of the final extension table is populated with the person dimension (W_WAVE_DX). The different transformations are the following:

- **SQ_W_WAVE_DSX.** There is no SQL in the override. It is a straight query from W_PERSON_DSX table that has been populated with incremental rows.
- **LKP_W_WAVE_D.** Looks up the W_WAVE_D table to get the ROW_WID for the row.
- **LKP_W_WAVE_DX.** Looks up the target table to find out whether to insert or update the row. This information is used by the update strategy.
- **EXPTRANS.** Used to get the Insert/Update flag and ETL_PROC_WID by using the lookup LKP_ETL_PROC_WID from the W_PARAM_G table. This table is populated in the beginning of the Refresh Siebel DW session.
- **FILTRANS.** Passing only those rows that have entry in the parent dimension table.
- **UPDTRANS.** Insert or updates a row based on the flag.

Fact Templates for Siebel Data Warehouse Customizations

This template shows how to attach an existing dimension (account dimension) to a fact (asset fact).

CustomTemplateSDE_AssetFact

Using this SDE mapping, the staging extension table for the Asset fact is populated with incremental data. The different transformations are the following:

- **SQ_S_ASSET.** Selects the OWNER_ACCNT_ID that will ultimately be used to join with the account dimension in the fact. Note the extra *where* clause joining to the incremental image table.
- **EXPTRANS.** Gets DATASOURCE_NUM_ID using lookup LKP_DATA_SOURCE_NUM_ID from table W_PARAM_G. This table is populated in the beginning of the session by the DAC server.

Finally, the FK_01_ID field of the extension stage table W_ASSET_FSX is populated. All the system columns are also populated.

CustomTemplateSIL_AssetFact

This SIL map is used to populate the final extension table for the Asset fact. The different transformations are the following:

- **SQ_W_ASSET_FSX.** Note the SQL in the override. This joins with the dimension table W_ACCOUNT_D to get the ROW_WID of the dimension row to be populated in the foreign key id column of the extension table.
- **LKP_W_ASSET_F.** Retrieves the ROW_WID of the parent fact table W_ASSET_F.
- **LKP_W_ASSET_FX.** Gets the Insert/Update flag that is used by the update strategy to find out whether to insert or update this row.
- **EXPTRANS.** Gets ETL_PROC_WID using the lookup LKP_ETL_PROC_WID from table W_PARAM_G. This table gets populated in the beginning of Refresh Siebel DW. Also generates Insert/Update flag based on the previous transform LKP_W_ASSET_FSX.
- **UPDTRAN.** Updates or inserts a row based on the flag.

Customization Steps for Siebel Data

This section includes information about adding columns and tables to the Siebel Data Warehouse using data from the Siebel transactional database (OLTP). It includes the following topics:

- "Adding a New Dimension in the Siebel Data Warehouse" on page 213
- "Adding a New Fact Table in the Siebel Data Warehouse" on page 214
- "Adding a New Dimension Table for a New Fact Table in the Siebel Data Warehouse" on page 215

Adding a Column from a Main Source Table in the Siebel Transactional Database

Example:

- Adding *race* in W_PERSON_D because it is not in the supplied model. The value of race is sourced from S_CONTACT, which is already the main source table for W_PERSON_D.
- Adding measure *XYZ* in W_ORDERITEM_F because it is not in the supplied model. The value of XYZ comes from S_ORDERITEM, which is the main source table for W_ORDERITEM_F.

To add a column from a main source table in the Siebel transactional database

1. Choose one of the following options:
   - Use predefined columns in an existing extension table.
   - Create a new extension table with appropriate columns.
2. Create a new map (SDE and SIL) to load the extension table column, based on the provided templates (CustomTemplateSDE_PersonDimension).
3. Create new individual workflows for the new mappings in the Custom folder.
4. For extract mappings (SDE), use views as the main source of information.
5. Create new tasks in the DAC repository, one for SDE and the other for SIL.
6. Make the SDE task depend on any auxiliary workflows, so that it waits for all the auxiliary processing for the base table to complete.
7. For the CustomSIL mapping make SDE mapping and the SIL_PersonDimension as the Depends On tasks. This is because you cannot execute CustomSIL workflow until the CustomExtract is completed. Also, because the primary key lookup has to be performed, this task needs to wait on SIL_PersonDimension.

Adding a Column from a Related Source Table in the Siebel Transactional Database

- Adding *ABC* in W_PERSON_D because it is not in the supplied model. The value of ABC is sourced from S_CONTACT_X, which is not the main source table for W_PERSON_D.
- Adding measure *XYZ* in W_ORDERITEM_F because it is not in the supplied model. The value of XYZ comes from S_DOC_ORDER, which is not the main source table for W_ORDERITEM_F.

To add a column from a related source table in the Siebel transactional database

1. Choose one of the following options:
   - Use predefined columns in an existing extension table
   - Create a new extension table with appropriate columns.
2. Create new mappings and workflows (SDE and SIL) to load the extension table column based on the provided templates. The related table should be joined in the Source Qualifier's SQL. Even though additional information is being extracted from a related table in the SQL override, always use the view created on the base table to extract information.
3. If necessary (for the new table), write the auxiliary change capture mappings so that whenever a row changes in the source table for the new column, the corresponding row in the main table is also marked changed, by inserting rows in the Incremental Row Image table.
4. Create a new workflow for the new map and register the task in the DAC repository. Create the dependencies.

Including a Source Table for Change Capture Process

This procedure applies to Siebel source tables only.

To include a source table for the change capture process

1. Verify whether the source table is registered in the DAC.
   a. If an entry is not in the DAC, create a new record for the table and assign an image suffix to it.
   b. If the table is registered, make sure there is an image suffix assigned for this table.
2. If an image suffix does not exist for the source table, assign one.

   The image suffix should be three characters long. The recommended naming convention is to start with C. For example, use C1, C2, CA1, CA2, and so on.
3. Make sure the image suffix is not used by other tables by querying the tables list in the DAC for that image suffix.

   The DAC client does not validate this information when entering data.
4. Create Image tables in the Siebel transactional database (OLTP).
   a. Right-click on the table record in the DAC client, and choose Generate Change Capture Scripts.

      This will assist you in creating the image tables, necessary indices, and triggers if you plan on tracking deletes on the transactional database.
   b. Execute these scripts on the transactional database if you have the appropriate permission, otherwise ask the OLTP DBA to create it.
5. For the task that is created for the extract process, set the Build Image flag to True and select the new table as an auxiliary table or primary table.

   NOTE: At the end of the change capture process, the DAC creates a view on the actual source table. Use this view as the main source table in all the extract procedures.

Modifying an Existing Column in the Siebel Data Warehouse

- Modifying *Full Name* in W_PERSON_D with a name in a different format.
- Modifying *Tax Total* and *XYZ* in W_ORDERITEM_F with different tax calculation logic.

To modify an existing column in the data warehouse

1. Do not modify the existing column.
2. Use extension table columns for the new value.
3. Refer to the steps for adding an extra column.

Adding a New Dimension in the Siebel Data Warehouse

- Adding a new custom dimension WC_NEWDIM_D and linking it to W_REVN_F.

Reference:

- SDE and SIL_WaveDimension, SDE and SIL_AssetFact

To add a new dimension and use it with an existing fact table

1. Create a DDL for the new dimension based on the standard structure (with appropriate system columns). Create a staging table for this dimension.
2. Register the new source table (if it does not already exist) in the DAC repository and associate it with the appropriate database connection.
3. Assign it an image suffix if you plan on incrementally loading this table.
4. Create a new custom map SDE_XYZ to populate the dimension stage. Instead of the actual source table (for example S_ABC), use the view that will be generated by the change capture process (for example V_ABC) in the SQL so that it extracts only the incremental data. Use existing reference maps as examples of how to populate the system columns. Make sure you truncate the stage table in corresponding tasks.
5. Create a new custom map SIL_XYZ to populate the new dimension from the stage table. Use the above referenced map as example for how to populate the system columns.
6. Make the following changes to the fact-loading mapping to link the new dimension:

   Do one of the following:

   - Use the extension table's predefined fact table foreign keys that join to this new dimension. (In the above example, the extension table for W_REVN_F must be used to store the foreign key of this new dimension.)
   - Create a new extension table for the fact table with predefined structure to hold a new foreign key, if the foreign keys in existing the extension table have been exhausted.

- Create new mappings (SDE and SIL) to populate the extension table columns of the fact extension table.

7 Register the new dimension table in the DAC and associate it with the appropriate database connection.

If you are planning to build a new dimension incrementally, assign an image suffix to the source table. For instructions, see "Adding a Column from a Related Source Table in the Siebel Transactional Database" on page 211.

8 Add the new dimension to all the fact groups that may reference it.

9 Register the workflows as tasks in the DAC.

10 For SDE mapping of the dimension make sure you set the Build Image flag to True, and the Truncate Always option to True. And in the list of source tables, mark the primary/auxiliary source(s) of this dimension.

11 For SIL workflows of the dimension make sure you set only Truncate for Full Load option to True.

12 Make SIL_XYZ depend on SDE_XYZ.

13 Make CustomSIL_RevenueFact depend on load dimension task (SIL_XYZ) and load base fact task (SIL_RevenueFact).

Adding a New Fact Table in the Siebel Data Warehouse

- Creating a new custom WC_PROJECT_F by sourcing data from S_XYZ table.

Reference:

- SDE and SIL_AssetFact.

To add a new fact table

1 Create a DDL for the new fact based on the standard structure (with appropriate system columns). Create a staging table for this fact.

2 Register the new source table (if it does not already exist) in the DAC repository and associate it with a database connection.

3 Assign it an image suffix.

4 Create the change capture tables by right-clicking and selecting Generate Change Capture Scripts. For instructions, see "Including a Source Table for Change Capture Process" on page 212.

5 Create SDE mappings to populate the custom stage table. Use the view created by change capture as the main table in the SQL so that it extracts only the incremental data. Use the reference maps (above) as examples of how to populate the system columns. Be sure to truncate the stage table in corresponding workflows.

6 Create SIL mapping to populate the custom fact table. Use reference maps as examples of how to populate the system columns.

7 Use lookups or SQL override joins to dimension tables for populating dimension foreign keys (ROW_WIDs) pointing to the existing dimension.

8 In the DAC, register the target tables.

9 Because this is a fact table, create a new table group with this fact table as the main table.

10 Associate the fact table to the group with the relationship as Self.

11 Associate all the dimension tables.

12 Add the newly created group to the appropriate subject area.

13 Create new tasks for the workflows.

14 For the SDE task, make sure you have the Build Image flag set to True, and list all the source tables that it queries from. Choose one or more tables as primary or auxiliary. For the target tables choose the staging table. Set the Truncate Always flag to True.

15 For the SIL task, list all the dimensions that will be required under source tables, and make it depend on the SDE task of the custom fact table and SIL tasks of all the dimension tables.

Adding a New Dimension Table for a New Fact Table in the Siebel Data Warehouse

- Adding a new custom dimension WC_NEWDIM_D and linking it to new custom WC_BKLOG_F fact table.

The steps for creating a new dimension table are similar to the steps for incremental change capture.

To add a new dimension table for a new fact table

1 In the new custom fact loading mapping (SIL), use lookups for getting foreign keys to the new dimension.

2 Use existing maps as examples.

Siebel Data Warehouse Customization Steps for Non-Siebel Data

This section includes information about adding non-Siebel data to the data warehouse. It includes the following topics:

Adding New Data as a Whole Row into the Standard Dimension Table in Siebel Data Warehouse

Example: Loading Contact Data From an External Source

- New DDLs: None
- New Mappings: SDE Mappings for dimensions, SDE and SIL mappings for related fact tables
- Case: Bringing data into dimension tables
- Reference Mappings: SDE and SIL_WaveDimension mappings

To add new data as a whole row into the standard dimension table

1. Identify and understand the existing structure of staging tables. Refer to *Siebel Data Warehouse Data Model Reference* for the table structure. Non-system columns can include the null value.
2. Create a custom SDE mapping to load the data into the staging table in the Custom_DW_Folder for this purpose. The staging table needs to be populated with incremental data (rows that have been added or changed since the last Refresh ETL process), for performance reasons.
3. Populate the INTEGRATION_ID column with the unique identifier for the record.

   The combination of INTEGRATION_ID and DATASRC_NUM_ID is unique. When importing the data, make sure that a unique identifier for the external data source is inserted in the DATASRC_NUM_ID column. The DATASRC_NUM_ID is set to 1 for mappings that source data from the Siebel transactional database. This is a reserved value and is used in all standard mappings. For example, a value of 2 can be defined for DATASRC_NUM_ID in the custom SDE mapping. The standard SDE mappings populate the INTEGRATION_ID column of the dimension staging table (used for resolving the dimension's Siebel transactional database ROW_ID value). The custom process must be used to populate the same column with a unique identifier from the external data source.
4. After the data is populated in the staging table, use the standard SIL mappings to populate the dimension target tables.
5. Use one data source per dimension.

   This is a best practice. To consolidate data for a dimension row from multiple data sources, complete that process prior to the custom SDE mapping process. For example, to consolidate contacts from Siebel and Peoplesoft applications, complete the consolidation process (including removing duplicates, cleansing, and so on) before loading the contact list into the Siebel staging table.

6 Modify the SDE and SIL mappings of all the related fact tables (fact tables that need to be linked to this dimension).

The custom fact SDE mappings must populate the foreign key column of the changed dimension (using a custom map table process to convert from Siebel's row IDs to the external data source row IDs). The custom SIL mapping should be modified to use the appropriate DATASRC_NUM_ID, because the standard SIL mappings assume DATASRC_NUM_ID for the dimensions are the same as the fact table's DATASRC_NUM_ID.

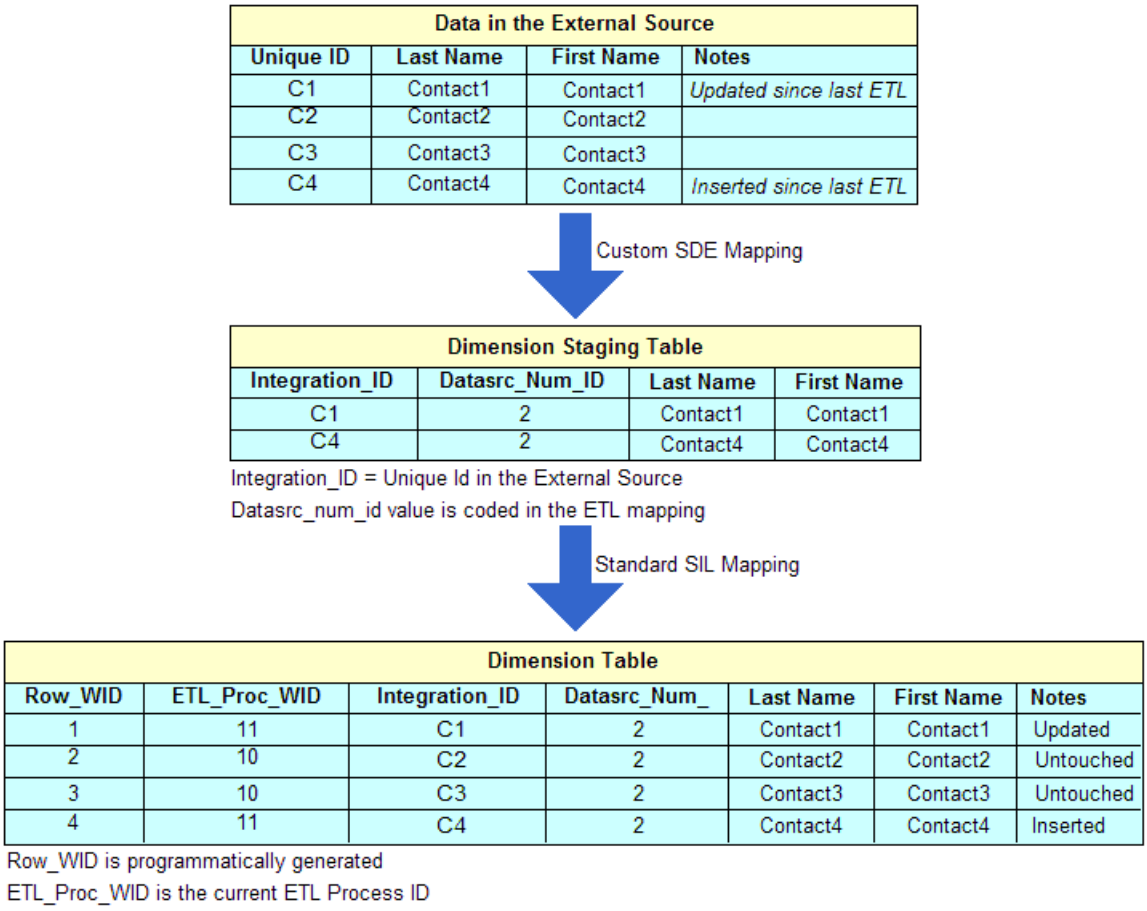
It is very important to decide when the data is going to be loaded. If it is going to be loaded along with the Siebel source data, you must be careful with how you handle failure recovery. The preconfigured workflows truncate the target staging table prior to loading. Upon failure, when the DAC server restarts the task, all the data is truncated and all the data is loaded again.

If the data from the external source gets loaded into the same staging table, be careful with how you handle this situation, since you cannot use the truncate table functionality. The data migrating into the staging table is not incrementally loaded, and, therefore, should be cleaned up prior to attempting to load this table again.

In such a case, it is recommended that you encapsulate the extract part from both the sources inside an Informatica workflow. Upon failure of either of the extracts, the whole workflow gets rerun. Note that the data from both the sources should be run at the same time, all the time.

If it is decided that the data is going to be loaded at different time frequencies, then the new SDE workflows need not depend on the preconfigured SDE workflows and can use the Truncate Table option for failure recovery. In this case, in the DAC Design view, define a new execution plan in the Execution Plans tab, and define the new data source in the Database Connections subtab. Make sure the shared SIL process depends on the SDE processes from both sources.

Figure 16 provides a graphical representation of this process.



Data in the External Source

Unique ID	Last Name	First Name	Notes
C1	Contact1	Contact1	*Updated since last ETL*
C2	Contact2	Contact2	
C3	Contact3	Contact3	
C4	Contact4	Contact4	*Inserted since last ETL*

Custom SDE Mapping

Dimension Staging Table

Integration_ID	Datasrc_Num_ID	Last Name	First Name
C1	2	Contact1	Contact1
C4	2	Contact4	Contact4

Integration_ID = Unique Id in the External Source

Datasrc_num_id value is coded in the ETL mapping

Standard SIL Mapping

Dimension Table

Row_WID	ETL_Proc_WID	Integration_ID	Datasrc_Num_	Last Name	First Name	Notes
1	11	C1	2	Contact1	Contact1	Updated
2	10	C2	2	Contact2	Contact2	Untouched
3	10	C3	2	Contact3	Contact3	Untouched
4	11	C4	2	Contact4	Contact4	Inserted

Row_WID is programmatically generated

ETL_Proc_WID is the current ETL Process ID

Figure 16. Adding New Data as a Whole Row into the Standard Dimension Table

Adding New Data as a Whole Row into the Standard Fact Table in Siebel Data Warehouse

Example: Bringing in activity data from external sources.

- New DDLs: None
- New Mappings: SDE and SIL mappings
- Case: Bringing data into fact tables.
- Reference: SDE and SIL_AssetFact

To add new data as a whole row into a standard fact table

1. Identify and understand the existing structure of staging tables. Refer to *Siebel Data Warehouse Data Model Reference* for the table structure.

   Fact tables contain foreign keys to existing dimension tables. These dimensional key columns in the fact table need to be populated with the correct ROW_WID values from the dimension tables. For dimensions that are populated with Siebel data, identify the correct ROW_WID by looking up the INTEGRATION_ID in the dimension table. For example, to load Order Item details from SAP, the Order Item Fact staging table has the PROD_ID column that needs to be resolved from the main Siebel data source. Looking up the ROW_WID of the S_PROD_INT table later resolves the foreign key to product dimension.

2. Develop custom SIL mappings to handle the specific situation required. The reference SIL mapping can be copied to the Custom folder and customized for specific requirements. If the SIL mappings were copied as a base from the standard folder, it is possible to leverage the existing logic for ROW_WID generation, update strategy, and so on. It may be easier to trim an existing SIL mapping than to create a new one. Apply discretion in creating a custom SIL mapping.

3. While joining with the dimensions in the SIL process, use the DATASOURCE_NUM_ID and INTEGRATION_ID to resolve foreign key values. Note that for dimensions populated entirely from Siebel, DATASOURCE_NUM_ID will be 1.

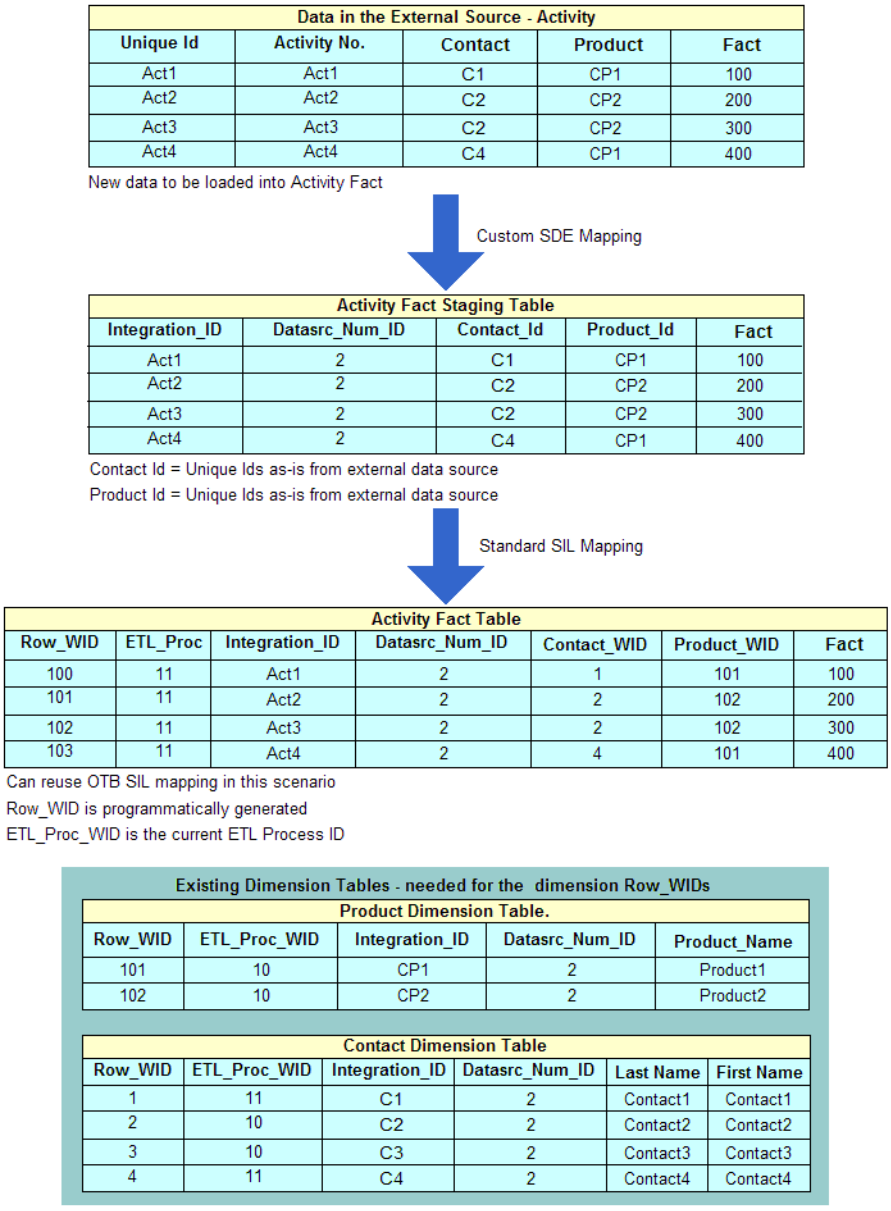
4. For the dimensions populated entirely from other external sources, use the appropriate DATASRC_NUM_ID (not 1) and the corresponding INTEGRATION_ID. This needs to be hard coded in the custom mapping.

5. Use an update strategy that would update the existing records and insert new ones. Inspect SDE and SIL mappings provided as reference to understanding the programmatic logic.

   - **Scenario 1.** All the dimensions get data from the external source as the fact table. In this case it is not necessary to change the existing standard SIL mappings. The standard SIL process assumes that the dimension rows that a fact needs to join are from the same data source as the fact rows. (See Figure 17 for an example.)

- **Scenario 2.** Some dimensions are populated from external sources and some from the Siebel transactional database. In this case, standard SIL mappings need to be modified so that the appropriate DATASRC_NUM_ID is used when joining to a dimension table. This scenario needs to be custom designed and developed. As in any customization, it will require ongoing maintenance. (See Figure 18 for an example.) Copy the mapping over to the Custom folder and do the modifications. Create a workflow that points to the mapping in the Custom folder. In the DAC repository, search for the task that uses this workflow as a command and change the folder to Custom folder. Change the name as necessary.

NOTE: Changing the standard mapping is an exception. It is important to document changes as updates to standard mapping will be lost during the Informatica repository process. Reapply changes to the standard mappings after the upgrade process. Prior to modifying copy the mapping as-is from OTB folder to the Custom folder, and make changes in the Custom folder. In the DAC repository, find the task that is using the workflow name and change the folder to point to the Custom folder. Adding new data as a whole row into the standard Fact table: Scenario 1.



Data in the External Source - Activity

Unique Id	Activity No.	Contact	Product	Fact
Act1	Act1	C1	CP1	100
Act2	Act2	C2	CP2	200
Act3	Act3	C2	CP2	300
Act4	Act4	C4	CP1	400

New data to be loaded into Activity Fact

Custom SDE Mapping

Activity Fact Staging Table

Integration_ID	Datasrc_Num_ID	Contact_Id	Product_Id	Fact
Act1	2	C1	CP1	100
Act2	2	C2	CP2	200
Act3	2	C2	CP2	300
Act4	2	C4	CP1	400

Contact Id = Unique Ids as-is from external data source

Product Id = Unique Ids as-is from external data source

Standard SIL Mapping

Activity Fact Table

Row_WID	ETL_Proc	Integration_ID	Datasrc_Num_ID	Contact_WID	Product_WID	Fact
100	11	Act1	2	1	101	100
101	11	Act2	2	2	102	200
102	11	Act3	2	2	102	300
103	11	Act4	2	4	101	400

Can reuse OTB SIL mapping in this scenario

Row_WID is programmatically generated

ETL_Proc_WID is the current ETL Process ID

Existing Dimension Tables - needed for the dimension Row_WIDs

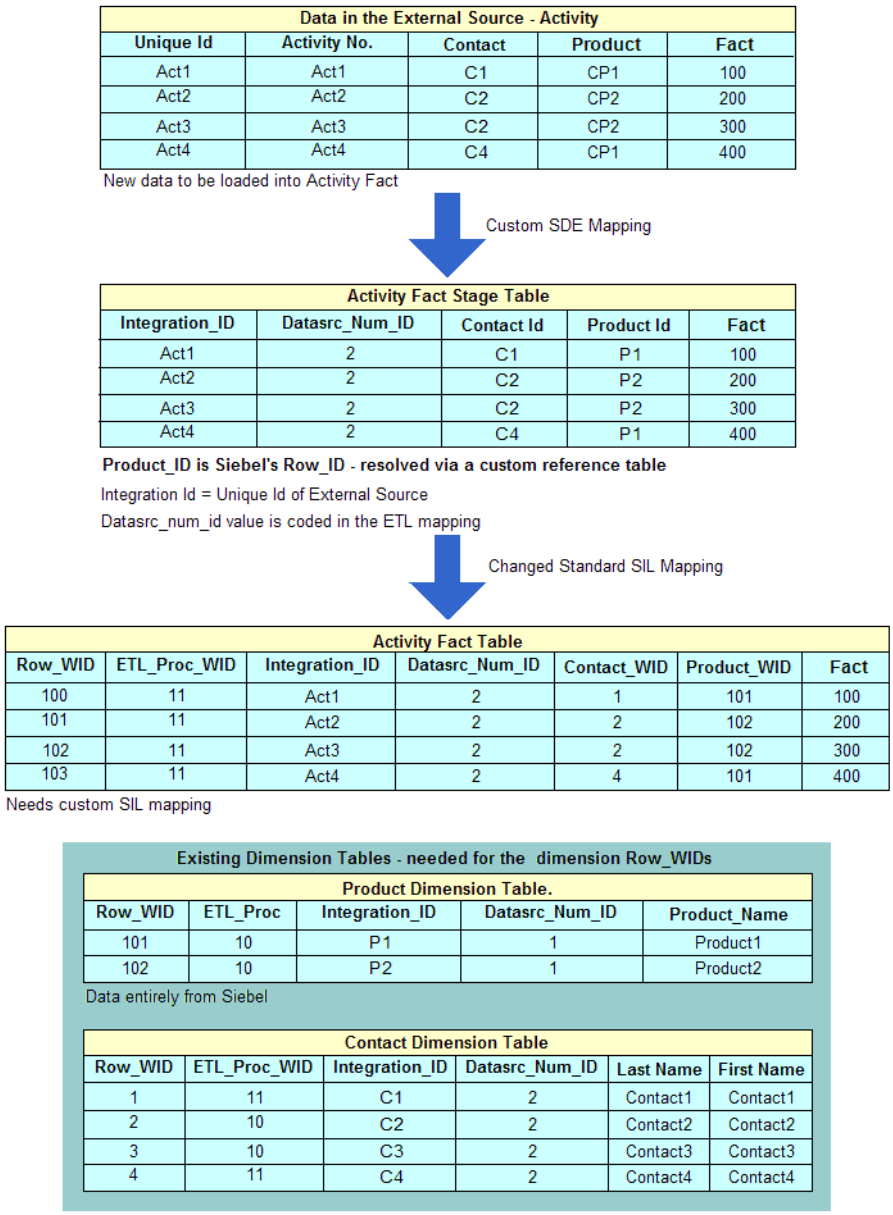
Product Dimension Table.

Row_WID	ETL_Proc_WID	Integration_ID	Datasrc_Num_ID	Product_Name
101	10	CP1	2	Product1
102	10	CP2	2	Product2

Contact Dimension Table

Row_WID	ETL_Proc_WID	Integration_ID	Datasrc_Num_ID	Last Name	First Name
1	11	C1	2	Contact1	Contact1
2	10	C2	2	Contact2	Contact2
3	10	C3	2	Contact3	Contact3
4	11	C4	2	Contact4	Contact4

Figure 17. Adding New Data as a Whole Row into the Standard Fact Table - Scenario 1



Data in the External Source - Activity

Unique Id	Activity No.	Contact	Product	Fact
Act1	Act1	C1	CP1	100
Act2	Act2	C2	CP2	200
Act3	Act3	C2	CP2	300
Act4	Act4	C4	CP1	400

New data to be loaded into Activity Fact

Custom SDE Mapping

Activity Fact Stage Table

Integration_ID	Datasrc_Num_ID	Contact Id	Product Id	Fact
Act1	2	C1	P1	100
Act2	2	C2	P2	200
Act3	2	C2	P2	300
Act4	2	C4	P1	400

Product_ID is Siebel's Row_ID - resolved via a custom reference table

Integration Id = Unique Id of External Source

Datasrc_num_id value is coded in the ETL mapping

Changed Standard SIL Mapping

Activity Fact Table

Row_WID	ETL_Proc_WID	Integration_ID	Datasrc_Num_ID	Contact_WID	Product_WID	Fact
100	11	Act1	2	1	101	100
101	11	Act2	2	2	102	200
102	11	Act3	2	2	102	300
103	11	Act4	2	4	101	400

Needs custom SIL mapping

Existing Dimension Tables - needed for the dimension Row_WIDs

Product Dimension Table.

Row_WID	ETL_Proc	Integration_ID	Datasrc_Num_ID	Product_Name
101	10	P1	1	Product1
102	10	P2	1	Product2

Data entirely from Siebel

Contact Dimension Table

Row_WID	ETL_Proc_WID	Integration_ID	Datasrc_Num_ID	Last Name	First Name
1	11	C1	2	Contact1	Contact1
2	10	C2	2	Contact2	Contact2
3	10	C3	2	Contact3	Contact3
4	11	C4	2	Contact4	Contact4

Figure 18. Adding New Data as a Whole Row into the Standard Fact Table - Scenario 2

In Figure 18, because the contact data and product data are from different data sources, it is necessary to modify the SIL session and the SQL override in the following manner:

```
SELECT

    ACT.INTEGRATION_ID,

    P.ROW_WID,
```

```
    C.ROW_WID

FROM

    W_ACTIVITY_FS ACT,

    W_CONTACT_D C,

    W_PRODUCT_D P

WHERE

    ACT.PRODUCT_ID = P.INTEGRATION_ID

    AND P.DATASRC_NUM_ID = 1 /* Product is from Siebel Source having datasrc_num_id
    = 1

ln dimension */

    AND ACT.CONTACT_ID = C.INTEGRATION_ID

    AND ACT.DATASRC_NUM_ID = C.DATASRC_NUM_ID

/* The rows in Contact dimension that the fact refer to are from the same external
datasrc, 2 */
```

Adding a Column in a Standard Dimension Table in Siebel Data Warehouse

Example: Loading an attribute called SKU for existing products that have already been loaded.

- New DDLs: New extension tables or preconfigured extension tables
- New Mappings: SDE and SIL mappings for extension tables
- Case: Bringing data in for dimension tables
- Reference: CustomTemplateSDE and SIL_WaveDimension in Custom_DW_Folder

To add a column in a standard dimension table

1. Add a column either in the standard or custom extension tables. There should be a 1:1 relationship with the parent table. Do not modify the standard tables, because doing so will affect upgrades.
2. The custom stage table should be populated with incremental data. This is an important performance consideration. It requires a process on the source side that detects new or modified records. This allows custom SDE mappings to process incremental data during each Refresh ETL execution.

3 While populating the extension table, design lookups (or write SQL overrides) for the SQL Qualifier Transformation in the Informatica mapping to identify the primary key (ROW_WID) of the parent row for the additional attribute. The primary key (ROW_WID) of the parent row is then used as the primary key (ROW_WID) of the extension table. Template mappings are provided in the CUSTOM_DW_REP folder (for example, CustomTemplateSDE_WaveDimension').

4 The custom SDE mappings should populate the stage extension tables (standard or custom). The custom extension table for staging should include the INTEGRATION_ID and DATASRC_NUM_ID. This combination enables the unique identification for each row by the data source. The data for this INTEGRATION_ID column should be populated by a process that identifies such an ID from the parent data source. The DATASRC_NUM_ID column should have the same value of the DATASRC_NUM_ID of the parent it is extending.

5 Once stage data is populated, a custom SIL mapping can be developed to move data from the extension stage to the extension table (of either the factor dimension table). The process should contain a look up either based on the INTEGRATION_ID and DATASRC_NUM_ID to find the ROW_WID for the extension table from the parent fact/dimension table. For an example, refer to CustomTemplateSIL_WaveDimension.

6 If there are rows for which there is no matching ROW_WID in the parent tables, they should be ignored. Use the appropriate update strategy for modifying existing data.

7 Make the custom SIL process depend on the custom SDE process and SIL process of the dimension table you are extending, because this mapping will require the borrowing of the foreign key for the link.

Adding a Column for a Standard Fact Table in Siebel Data Warehouse

Example: Loading shipping order information into the data warehouse.

- New DDLs: Custom or standard extension tables.
- New Mappings: SDE and SIL mappings.
- Case: Bringing data in for fact tables.
- Reference: CustomSDE and SIL_AssetFact in Custom_DW_Folder.

To add a column for a standard fact table

1 Follow the steps in "Adding a Column in a Standard Dimension Table in Siebel Data Warehouse" on page 223.

2 Use the CustomTemplateSDE_AssetFact, and CustomTemplateSIL_AssetFact as examples.

If the additional columns include foreign keys to dimension tables, consider the following:

- While joining with the dimensions in the SIL process, use the DATASRC_NUM_ID as 1 for the dimensions that have only data from the main Siebel source and join based on INTEGRATION_ID.

- For the dimensions that have only data from external sources use the appropriate DATASRC_NUM_ID and the corresponding INTEGRATION_ID.
- Use the appropriate update strategy to change and insert rows.

3 Quite likely the SDE process for this will not depend on anything else.

4 Make the Custom SIL process to depend on its custom SDE process and SIL process of the fact table you are extending.

Building a Custom Dimension Table from External Sources in Siebel Data Warehouse

- New DDLs: New staging and new dimension tables.
- New Mappings: Dimension SDE and SIL mappings.
- Case: Loading data for custom dimension tables.
- Reference: SDE and SIL_WaveDimension in Siebel_DW_Folder.

To build a custom dimension table from external sources

1 Follow the steps in "Adding a New Dimension Table for a New Fact Table in the Siebel Data Warehouse" on page 215. Use a different DATASRC_NUM_ID while populating the staging table.

2 Populate the custom table with incremental data. This is an important performance consideration that requires a process on the source side that detects new or modified records.

3 Make sure the SIL process takes care of the update strategy when loading data incrementally. Look for examples of how it is done by inspecting any of the SIL mappings in the standard repository like SIL_WaveDimension.

4 In the DAC, make the SIL process depend on SDE process for this custom table.

5 Register the table.

6 Add it to appropriate table groups.

Linking the Custom Dimension to a Standard Fact Table in Siebel Data Warehouse

- New DDLs: New fact extension tables or standard fact extension tables
- New Mappings: Fact SDE and SIL mappings
- Case: Loading data for fact tables that reference the new custom dimension tables
- Reference: CustomSDE and SIL_AssetFact in Custom_DW_Folder

To link the custom dimension to a standard fact table

1. Extend the fact tables by creating or using fact extension tables.
2. A custom SDE mapping has to be designed to populate the additional foreign key column in the extension stage table. Make sure that the dimensions ID in the fact staging is the same as the integration of the custom dimension. This may require some preprocessing.
3. A custom SIL process loads data from the staging table to the fact extension table.
4. Develop the SQL override for populating the fact extension table to extract foreign keys for the new dimension table. Specifically, write the query to extract the ROW_WID of the custom dimension table using INTEGRATION_ID and DATASRC_NUM_ID of the dimension in combination.
5. Ignore the rows for which there is not a matching ROW_WID value.
6. In the DAC, register the SDE and SIL tasks. Make the SIL mapping depend on its corresponding SDE mapping and the SIL mapping of the base fact table.
7. Add the custom dimension to the fact table group.

Building Custom Fact Tables from External Sources in Siebel Data Warehouse

- New DDLs: New staging and new fact tables
- New Mappings: SDE and SIL mappings
- Case: Loading data for new fact tables
- Reference: SDE and SIL_AssetFact from Siebel_DW_Rep Folder

To build custom fact tables from external sources

1. Follow the steps in "Building a Custom Dimension Table from External Sources in Siebel Data Warehouse" on page 225.
2. In the custom fact SDE process, make sure that fact columns that need to reference standard Siebel dimensions contain ROW_IDS from the Siebel data source.
3. Make sure that the dimension IDs in fact staging table are the same as the INTEGRATION_ID in the corresponding dimension tables. This may require some preprocessing.
4. While creating a custom fact SIL process, make sure the SQL override statement joins with dimension tables. Use DATASRC_NUM_ID as 1 for the dimensions that have data from Siebel and join based on INTEGRATION_ID.
5. For the dimensions that use data entirely from external sources, use the appropriate DATASRC_NUM_ID and the corresponding INTEGRATION_ID. Use the appropriate update strategy to add new or change rows.
6. In the DAC, register the tasks. Make the SIL task depend on its corresponding SDE workflow and all the SIL tasks of its dimensions.

11 Using Siebel Data Warehouse Exception Reports

This chapter covers exception reports, which provide information about the source data used for ETL processes that can lead to erroneous results in the Siebel Data Warehouse or may cause data loss during the ETL process. These reports point out some of the known problematic areas, but they should not be relied upon to find all potential data inconsistencies in the source data.

This chapter includes the following topics:

Understanding Siebel Data Warehouse Exceptions

Exception reports are defined for the following components:

- **List of Values.** Identifies gaps and overlaps for certain LOV types.
- **Cost Lists.** Identifies products for which the cost lists have not been defined. Identifies the cost lists which define costs for a product in a certain currency for overlapping periods of time.
- **Exchange Rates.** Currency Exchange rates that do not change over a period of time. If exchange rates are not defined for more than 30-day intervals, then they are flagged as an exception.
- **Hierarchies**. Entities that have circular references are flagged as exceptions. The Siebel Data Warehouse supports 10 levels of hierarchies. If there are entities that have more than 10 levels of hierarchies defined, they are flagged as exceptions.

List of Values Exceptions

List of Values include High and Low values that can be used as bucket values in categories for effective analysis. If these values are not contiguous (such as gaps or overlaps in defined values), the ETL process cannot accurately categorize the values.

An example of List of Values exceptions is shown in Table 51.

Table 51. List of Values Exceptions (Example)

Type	Name	Low	High
MY_TYPE	< 100	10	100
MY_TYPE	80 – 200	80	200

Table 51. List of Values Exceptions (Example)

Type	Name	Low	High
MY_TYPE	250 – 300	250	300
MY_TYPE	350 – 400	350	400

Notice that, in the example for Type MY_TYPE, there are overlaps and gaps between records. There is an overlap of ranges between the first and second row. There is a gap between second and third row, and between third and fourth rows.

The following LOV types are analyzed for List of Value Exceptions:

- ACCNT_REVENUE_SIZE
- ACCNT_EMP_SIZE
- LEAD_AGE_DAYS
- OPTY_REVENUE_SIZE
- OPTY_UNIT_SIZE
- ACCNT_REVENUE
- QUOTE_AGE_DAYS
- ACCNT_REVN_GROWTH
- APPROVAL_AUTH_SIZE
- SR_CHART_AGE
- ASSET_COST_CATEGORY

Cost List Exceptions

Cost Lists for specified products and currency should not have overlapping time periods. If multiple cost lists are defined for a product and currency during a given time period, then the cost for the product may not be computed correctly in the Siebel Data Warehouse.

An example of Cost List exceptions is shown in Table 52.

Table 52. Cost List Exceptions (Example)

Cost List	Product Name	Currency	Start Date (MM-DD-YYYY)	End Date (MM-DD-YYYY)	Cost
Cost List 1	Product 1	USD	01-01-2000	12-31-2000	10.00
Cost List 2	Product 1	USD	06-01-2000	06-01-2001	12.00
Cost List 3	Product 1	USD	06-01-2001	06-01-2002	13.00

In the example, Cost List 1 and 2 have definitions of cost overlapping over 06-01-2000 to 12-31-2000.

Products Without a Cost List

During the ETL process, the costs of the products are calculated based on the Cost List table. If the cost lists are not defined correctly, the cost of the products cannot be calculated correctly in the Siebel Data Warehouse. This exception mapping queries the product table and looks for a minimum of one cost list to be defined. The products with no cost list definition are flagged as exceptions.

Exchange Rate Exceptions

The Siebel Data Warehouse supports transactions in many different currencies. Siebel Analytics converts all currencies in the Siebel Data Warehouse to a single currency for analysis purposes. The ETL Base Exchange Currency parameter in System Preferences, indicates the currency to which all the financial amounts will be converted. The Exchange rates are derived from the Exchange Rate tables in the Siebel transactional database. If the currency exchange rates do not change for a period of 30 days, then Siebel Analytics flags it as an exception.

If there are time period gaps in the exchange rate data, the ETL process defaults to the most recent recorded exchange rate. If the actual exchange rate is significantly more or less favorable than what is recorded in the database, the outdated exchange rate distorts the true value of currency amounts in the Siebel Data Warehouse.

NOTE: Exchange rates are derived from records that are of type “Daily” in the Siebel transactional database. If any other types have been defined, they are not handled without some customization.

Invalid Hierarchy Exceptions

Accounts, divisions, products, and opportunities can all have hierarchical relationships. These entities are denormalized within the Siebel Data Warehouse database to a fixed number of levels. Siebel Analytics supports up to ten hierarchies levels in the Siebel transactional database. If the depth of hierarchies extends beyond this number, results become inconsistent or incomplete.

Circular Hierarchy Exceptions

Circular Hierarchies arise when the parent-child relationship has circular references. See Table 53 for an example.

Table 53. Circular Hierarchy Example 1

Child	Parent
A1	A2
A2	A1

Siebel Analytics flags exceptions for two levels. Circular references over two hierarchies are not flagged. See Table 54 for an example.

Table 54. Circular Hierarchy Example 2

Child	Parent
A1	A2
A2	A3
A3	A1

These produce infinite levels of hierarchies. The same records will be captured under the Invalid Hierarchy exceptions as their hierarchy depths will increase beyond 10 levels.

Executing Siebel Data Warehouse Exception Reports

Before loading the Siebel Data Warehouse for the first time and for the subsequent refreshes, you should plan to spend time cleansing your Siebel transactional database data using the exception reports. The process is iterative, and requires coordination with other team members who have responsibility for data in the Siebel transactional database, such as the Siebel database administrator. After the initial cleansing, you should generate the exception reports on a scheduled basis to maintain the integrity of your data.

NOTE: Rerunning the exception reports overwrites all data contained in this view.

To cleanse the Siebel transactional database, repeat these actions until the exception report is empty:

- In the DAC, run the Exception Reports execution plan.
- In the Siebel user interface, navigate to Analytics Administration > Exception Reports.

  NOTE: In Siebel Financial Services, this screen is called DataMart Administration.
- For every line in the ETL Exception Reports list, fix the cause of the problem. For information on fixing problems, see "Cleansing Data."

Cleansing Data

Use a combination of the Exception Reports and the Diagnostic views to assess changes that need to be made external to the Siebel transactional database, and changes to the Siebel transactional database directly.

The ETL Exception Reports list contains one record for each exception flagged in the Siebel transactional database. The ETL Exception Explanation form, located below the ETL Exception Reports list, describes the selected exception, its effect on the Siebel Data Warehouse building process, and offers suggestions for repairing the data.

To fix an exception

1. Select an exception record.
2. Read and understand the text in the ETL Exception Explanation form.
3. Click the report link.

   The object's data appears. (For example, if the object is an account, then the Account form appears. If the object is a cost list, then the Cost List list appears.)
4. Repair the problem, using the text in the ETL Exception Explanation form as a guide.
5. Return to the ETL Exception Reports list and place a check mark in the Fixed column to indicate to others that this exception has now been fixed.

Using the List of Values View

Use the List of Values view, shown in Figure 19, to visually compare how the list of values data extracted from the Siebel transactional database coordinates with the values loaded into the Siebel Data Warehouse. The ETL process removes duplicates and overlaps and fills data gaps. Values are extended to span the List of Values (LOV) minimum and maximum values. Duplicates, Range Gaps, and Overlaps are flagged by the exception reports.

ETL History | List of Values | Exchange Rates | Cost Lists

Query | Clear Cache | 1 - 7 of 7+

Type	Language Independent Code	Display Value	Low	High	Order	Language Name
ABS_COST_FCT_AREA	Field Service	Field Service			1	English-American
ABS_COST_FCT_AREA	Professional Services	Professional Services			2	English-American
ABS_COST_FCT_AREA	Workforce Management	Workforce Management			3	English-American
ABS_COST_FCT_TYPE	Normal	Normal			1	English-American
ABS_COST_FCT_TYPE	Emergency	Emergency			2	English-American
ABS_COST_FCT_VARIABLE	Constraint Violation	Constraint Violation			1	English-American
ABS_COST_FCT_VARIABLE	FSE Overtime	FSE Overtime			2	English-American

List Of Values (Data Warehouse)

Query | 1 - 7 of 7+

Type	Language Independent Code	Display Value	Low	High	Order
ABS_COST_FCT_AREA	Field Service	Field Service	0	9,999,999,999	1
ABS_COST_FCT_AREA	Professional Services	Professional Services	0	9,999,999,999	2
ABS_COST_FCT_AREA	Workforce Management	Workforce Management	0	9,999,999,999	3
ABS_COST_FCT_TYPE	Normal	Normal	0	9,999,999,999	1
ABS_COST_FCT_TYPE	Emergency	Emergency	0	9,999,999,999	2
ABS_COST_FCT_VARIABLE	Constraint Violation	Constraint Violation	0	9,999,999,999	1
ABS_COST_FCT_VARIABLE	FSE Overtime	FSE Overtime	0	9,999,999,999	2

Figure 19. List of Values List

The top List of Values list shows values from the Siebel transactional database and the bottom List of Values (Data Warehouse) list shows the data that is to be used in ETL process. You can edit the Siebel transactional database data directly in this view, but the Siebel Data Warehouse list is read-only.

NOTE: The List of Values is extracted into the Siebel Data Warehouse where the language is the same as the ETL Default Language set in the DAC ETL Preferences, or whose translate flag is set to "N," or those that are active.

Using the Exchange Rates View

Use the Exchange Rates view to diagnose currency translation issues in the Siebel Data Warehouse. The ETL process removes duplicates, fills gaps, and removes overlaps. The ETL process computes exchange rates based on commutative and associative properties, such as product and reverse rates.

The top Exchange Rates list shows currencies, the middle Exchange Rates list shows the Siebel Data Warehouse values for active currencies and their exchange rates, and the bottom Exchange Rates (Data Warehouse) list shows the values loaded into the Siebel Data Warehouse for the selected currency in the upper Exchange Rates list to the ETL Base Exchange Currency. The Exchange Rates (Data Warehouse) list is read-only.

- The Active Currencies predefined query restricts the list to the active currencies in the Siebel transactional database.
- The exception reports flag any exchange rates to the ETL Base Exchange Currency that have not been defined within a specified period (30 days) in the DAC ETL Preferences.

Using the Cost List View

Use the Cost List view to display the cost lists from the Siebel transactional database from the point of view of the product, and a read-only view of the values to be loaded into the Siebel Data Warehouse. The ETL process removes duplicates, overlaps, and fills gaps.

The Cost List list (top) shows products, and the Cost List Line Items list (middle) shows the cost lists associated with the selected product. The Cost Lists (Data Warehouse) list (bottom) shows the data as it is transformed for the Siebel Data Warehouse.

- The exception reports flag products that do not appear in the Cost List list or have Cost List time gaps and overlaps.
- The Siebel Data Warehouse contains only one Cost List for a product and a currency at a time.

Using the ETL History View

After all of the exceptions are corrected, the building of the data warehouse can be initiated. This view lists the history of the ETL processes and their statuses. When each ETL batch starts, the name of the process along with the timestamp is set, the status is set to STARTED. When the batch completes, its status is updated to COMPLETED.

Additional Exceptions

The above mentioned exceptions are not an exhaustive list of all possible exceptions. Other exceptions are:

- The Analysis start and end date in the DAC ETL Preferences must span the entire period of time during which the transactions have occurred. For example, you may want to choose an early and late date range to cover the entire time period you are analyzing. These dates in the DAC ETL Preferences are crucial for the building of Day Dimension, flattening of Exchange Rates, Cost Lists, and KPI (Key Performance Indicator fact) calculations.

- The DAC ETL Preferences — ETL Date Format, ETL Analysis Start, ETL Analysis End parameters, and the List of Values — ETL_UNSPEC_DATE must be defined in the same data format. If one is changed, the others must be changed accordingly.
- List of Values must be defined appropriately. If there is no appropriate entry in List of Values, the strings that depend on List of Values in the Siebel transactional database will not be translated.
- There must be exchange rates defined for the currencies your organization deals with. If the appropriate exchange values are not found, the ETL process uses the ETL Unknown Exchange Rate defined in the DAC ETL Preferences.

12 Troubleshooting the Siebel Data Warehouse

This section details some common problems related to the Siebel Data Warehouse and suggests possible solutions. It contains the following topics:

Siebel Data Warehouse Installation and Initial Configuration Issues

Table 55 provides information about problems and solutions related to the installation and initial configuration of the Siebel Data Warehouse.

Table 55. Siebel Data Warehouse Installation and Configuration Issues

Symptom / Error Message	Probable Cause / Solution
Cannot connect to the Siebel Data Warehouse running on DB2.	Make sure that the DB2 configuration matches the requirements listed in "IBM DB2 UDB-Specific Database Guidelines for Siebel Data Warehouse" on page 34.
After installing Informatica's ODBC driver (V3.5) on a Windows computer, you cannot open the Query Analyzer, and the Enterprise Manager shows an error message.	The ODBC driver installation replaces the odbcbcp.dll file with an older version. The Informatica documentation contains the procedure for correcting this problem.
Cannot connect to Siebel transactional database from Informatica.	Make sure that the server running the Informatica Server software has an ODBC connection to the Siebel transactional database using a Siebel ODBC driver and an ODBC connection to the Siebel Data Warehouse using the Merant Closed 3.5 32-bit driver.
Error: ORA-12541: TNS:no listener.	Check the Compatibility tab in Informatica service configuration. It should be Oracle 8. If it is not set to Oracle 8, change it and restart the Informatica server.

Table 55. Siebel Data Warehouse Installation and Configuration Issues

Symptom / Error Message	Probable Cause / Solution
Error 2140 (Informatica service failed to start).	The server may not be started or the Informatica Service may not be started. See Informatica's installation and configuration guide on *Siebel eBusiness Third-Party Bookshelf* for detailed information.
Informatica installation fails with an Unhandled Exception error and displays a message similar to this: "Error Number: 0x80040707. Description: Dll function call crashed: ISRT._DoInstall."	The computer is probably running out of virtual memory. Restart the computer and reinstall Informatica.
After selecting "Warehouse Default Amount (Yen)" as the currency symbol, the dollar sign ($) is still displayed.	The currencies.xml file must be edited. For instructions, see *Siebel Analytics Installation and Configuration Guide*.

Informatica and Loading Issues

Table 56 provides information about problems and solutions related to issues with Informatica and loading. To view the Informatica log file details, double-click the workflow.

Table 56. Informatica and Loading Issues

Symptom / Error Message	Probable Cause / Solution
Double-clicking the workflow yields a Workflow Manager error message: "The system cannot find the file specified."	The session log files are not set up properly. You also may need to change the text editor.
Using Oracle, some mappings hang while running when performance statistics are switched on.	When running some Informatica mappings for loading the Siebel Data Warehouse, turning on the Performance Statistics can cause the mapping to hang. The only workaround is to increase the values of the LMSharedMemory and MaxSessions variables in Informatica. The risk of increasing the LMSharedMemory too much is that it may start to have a serious effect on overall performance of the machine that the Informatica server is running on.

Table 56. Informatica and Loading Issues

Symptom / Error Message	Probable Cause / Solution
When you execute a workflow on the Informatica Workflow Manager, Informatica returns the following error message: "Request to start workflow (workflow name) on server (server name) not successful."	This can happen due to a server time-out property that is usually set to 20 or 40 seconds. When you try to run a large workflow, every session in that workflow is fetched into the server's memory. If this takes longer than the server time-out property, the server returns a message that the workflow was unable to run. However, the workflow is running, but the server just needs time to complete fetching the sessions into memory. Double-click the workflow to view the log file details.

When running Full_Extract_Siebel_DW or Refresh_Extract_Siebel_DW, Informatica returns errors similar to:
 	
`TE_7007 Transformation Evaluation Error; current row skipped...`
 	
`TE_7007 [<<Transformation Error>> [to_date]: Date function error to_date('19010101', 'DD-MON-YYYY')`
	Incorrect date entry in the SME Date Format field in the System Preferences. The format is YYYYMMDD.

When running Full_Load_Siebel_DW, Informatica returns errors similar to:
 	
`CMN_1014 Error creating semaphore...`
 	
`TM_6006 Error initializing DTM for session...`
 	
`TM_6006 [s_CR18a1. Load W_PROG_DM_TMP - Program Records]`
	Insufficient semaphores allocated. Allocate more semaphores on the Data Warehouse Server. The change becomes effective when you reboot.

Table 56. Informatica and Loading Issues

Symptom / Error Message	Probable Cause / Solution

Informatica (RDBMS is DB2) gives the following error message:
 	
`Error occurred unlocking [SDE_ServiceRequestDimension1].`
 	
`An error occurred while accessing the repository[[IBM][CLI Driver][DB2/ 6000] SQL0955C`
 	
`Sort memory cannot be allocated to process the statement. Reason code = "".`
 	
`SQLSTATE=57011]`
 	
`DB2 Fatal Error[FnName: ExecuteDirect -- SQLSTATE=57011 [IBM][CLI Driver][DB2/6000]`
	The DB2 parameter “SHEAPTHRES” is too small.

Informatica produces the error “Unable to connect to the server” when running a full load of the Siebel Data Warehouse (Full_Load_Siebel_DW_Dimensions).	The last Designer session was not validated. Part of the development process of working with Designer is to always validate any changes to Informatica mappings definitions and sessions after the change is saved in repository.
When loading the data warehouse, Informatica reports a lock problem.	Either someone has a session open or there is a dead session. Make sure no one has any open sessions. If no sessions are open, then follow the Informatica documentation on removing locks caused by dead sessions.
After changing an Informatica mapping, you may get an error message when trying to execute “Full_Load_Siebel_DW_Facts.” The error message is “Unable to connect to the server.”	This is due to Informatica mapping objects that have been modified and this does not automatically validate the session objects. You must validate all changes to any existing mappings in the Informatica repository.

Table 56. Informatica and Loading Issues

Symptom / Error Message	Probable Cause / Solution
Session SDE_RecordLoadStart fails due to unique constraint error while executing Full_Load_Siebel_DW_Dimensions or Full_Load_Siebel_DW_Facts.	This could be because the previous full load did not complete successfully. Fix the problem that caused the previous load session to fail. Make sure you start the process from the last entry of Load_RestartNextWorkflow # before the failed session, and restart the workflow process from that point. If you have to re-extract the data from the Siebel transactional database because something had to be fixed in the source database to resolve the load error, then you must restart the ETL process. Truncate S_ETL_INC_STAT in the Siebel transactional database, then enable the Extract and Load workflows and rerun them.
Session SDEINC_RecordExtractStart fails due to unique constraint error while executing a Refresh workflow.	This could be because the previous load or refresh did not complete successfully. Fix the problem that caused the previous refresh session to fail. Make sure you start the process from last entry of %RestartNextWorkflow # before the failed session, and restart the workflow process from that point.
The session fails and you receive the following error code: `Error "TE_7042 Aggregate Error: File Operation Error`	This is due to a disk space limitation. Check the /Informatica/PowerMart/Cache/Check directory for available disk space, also check the limits (ulimit) of the account used to start PowerMart.

Informatica sessions get deadlocked and eventually fail when they try to do a “select” from the repository table OPB_OBJECT_LOCKS. This problem sometimes occurs on MSSQL server databases.	This is possibly caused by a limited number of resources on the MSSQL Database Server. The workaround is to execute the following MSSQL specific SQL command on the Siebel Data Warehouse:
	
	`DROP INDEX OPB_OBJECT_LOCKS.OPB_OBJ_LOCKS_IDX`
	
	`DROP INDEX OPB_OBJECT_LOCKS.OPB_OBJ_LOCKS_IDX2`
	
	`DROP INDEX OPB_OBJECT_LOCKS.OPB_OBJ_LOCKS_IDX3`
	
	Upon completion of executing these commands, continue executing the workflow processes to load the Siebel Data Warehouse.

Table 56. Informatica and Loading Issues

Symptom / Error Message	Probable Cause / Solution
An error may occur when trying to send a post session email notification using MS Outlook 2000. Refer to Informatica release notes for further information.	After installing Informatica Server on Windows, copy the file mapi32.dll from winnt\\system32 to the bin folder where the Informatica Server is installed, overwriting the existing mapi32.dll in that directory. Start the Informatica Server so that the Informatica Server can use the new mapi32.dll. The Extended MAPI Error. MAPILogonEx failed[2147746065] error indicates that the logon is not configured correctly. Check the following: 1 Under Services > Informatica> Logon, make sure the login (domain\username) and password are correct. 2 Under Control Panel > Mail (it may also be called Mail and Fax or Exchange) > Services > Show Profiles, make sure the mail profile is correct. 3 Under Programs> Informatica Server > Informatica Server Setup> Miscellaneous, make sure the MS Exchange profile is correct.
While creating a custom session, bulk load mode does not work properly with SQL Server.	Change the mode to “normal” in Informatica repository for the session. The “normal” mode must be used everywhere for SQL Server in all of your custom sessions.
When running IMR, you may receive an error message box which states “The evaluation period for this Oracle ODBC driver has expired. Please call Merant to obtain a production version of this Oracle ODBC driver.”	This is caused by a down-level ODBC driver license key. Rename or move ivodbc.lic, lvodbc.lic (if it exists), and lvdw.lic (if it exists). Make sure you have only one license file named ivdw.lic in winnt\system32. This eliminates the problem.
Outlook closes when sending out a notification of finishing the ETL process.	Informatica is closing Outlook. This issue is known to Informatica and is scheduled to be resolved in an upcoming release. Until then, create a second profile in Outlook and add that profile name to the Informatica server setup.

Table 56. Informatica and Loading Issues

Symptom / Error Message	Probable Cause / Solution

Oracle 8i sessions running in bulk mode fail and return an error message similar to:
 	
`WRITER_1_1_1> CMN_1022 Database driver error...`
`CMN_1022 [`
`ORA-00600: internal error code, arguments: [kpodpmop_01], [2], [], [], [], [],[], []`
 	
Or
 	
`MAPPING> TE_7022 TShmWriter: Initialized`
`MAPPING> Sat Jan 26 13:54:45 2002`
`MAPPING> TE_7001 Internal error: Failed to allocate a target slot. Sat`
`MAPPING> Jan 26 13:54:45 2002 TE_7017 Failed to Initialize Server`
`MAPPING> Transformation BLK_ALL_DATATYPES1 Sat Jan 26 13:54:45 2002MAPPING> TM_6006 Error initializing DTM for session...MAPPING> TM_6020 Session [s_BULK_LONG] completed at [Sat Jan 26 13:54:45 2002]`
	This problem is an Oracle 8i defect. It has been resolved in Oracle 9i. The workaround is to run the session in Normal mode. To do so, in Workflow Manager navigate to the Targets window, and change the Target Load type to Normal.

Table 56. Informatica and Loading Issues

Symptom / Error Message	Probable Cause / Solution

During an ETL execution, when Informatica and DAC servers use DB2 Connect version 7 to talk to DB2/390 version 7 OLTP and data warehouse databases, you receive an error message similar to the following:
 	
`SEVERE: [IBM][CLI Driver][DB2] SQL0191N Error occurred because of a fragmented MBCS character. SQLSTATE=22504`
 	
`103 SEVERE Tue May 11 21:37:29 CDT 2004 [IBM][CLI Driver][DB2] SQL0191N Error occurred because of a fragmented MBCS character. SQLSTATE=22504`
	This problem is a DB2 Connect version 7 (IBM) defect related to code page conversion. The problem has been resolved in DB2 Connect version 8.
	
	To correct the problem, do the following:
	
	1 Download the file IBM01140.ucs from
	
	ftp://ftp.software.ibm.com/ps/products/db2/fixes/english/siebel/siebel7/Conversion_Files
	
	to the /sqllib/conv directory.
	
	2 Make a copy of the file and rename it to IMB05348.ucs.

When an ETL process is running and tasks fail, Informatica returns an error similar to the following `Error while running Workflow Description: ERROR : TM_6292 : (3040|4868) Session task instance REP_12400 [Repository Error ([REP_51055] Repository agent connection failed. [System Error (errno = 121): The semaphore timeout period has expired. . (Cannot read message. Read 5824 bytes.)])]`	This issue is related to the network. The workaround is to increase the Timeout parameter values in the Informatica Repository Server Administration Console. 1 In the left pane of the Repository Server Administration Console window, right click your repository and select Edit Connection. 2 In the Network tab, enter 9 as the value for the parameters MessageReceiveTimeout and MessageSendTimeout. 3 Stop and start the Informatica Repository Server. 4 Start the Informatica Server.

Unicode Issues with Informatica and the Siebel Data Warehouse

Table 57 provides information about problems and solutions related to Unicode issues with Informatica and the Siebel Data Warehouse.

Table 57. Unicode Issues with Informatica and Siebel Data Warehouse

Symptom / Error Message	Probable Cause / Solution
Task fails and generates the error message: "TRANSF_1_1_1_1> TE_7073 Aggregate Error: Expecting keys to be ascending." Tasks known to fail are SDE_DtlForecastFact and SDE_CostList	In Informatica Mapping Designer, open the corresponding mapping and in the Aggregator transformation, remove the check from the Sortedinput check box.
In the Windows 2000 environment, with a double-byte language, such as Japanese, when you try to input the path for the directory where Informatica is installed (such as c:\Program Files\Informatica\Informatica PowerMart 6.2.1 Siebel XXX(JPN char) OEM - Server) for the value $PMRootDir in the Workflow Manager, the double-byte characters are converted to question marks. This causes the ETL to fail because it cannot find the Informatica PowerMart 6.2.1 Siebel OEM - Server folder and its subdirectories.	Share the folder under a new name with no double-byte characters, and use the network share as the value for $PMRootDir.
When Configuring the Informatica Server or Informatica Repository Server, there are major truncations in some of the tabs. For example, in the Server tab and the Repository tab in the Configure Informatica Service screen (Start > Programs > Informatica Server > Informatica Server Setup > Configure Informatica Service), the field names are truncated.	Informatica 6.x does not support non-English messages and resources. Delete the following files to display all messages in English. PC/PM client folder*411.dll PMServer folder\bin*411.dll PMRepServer folder\bin*411.dll
Unable to start the Informatica repository with the Informatica Repository Server on UNIX. The Siebel_DW_Rep (MS Windows Japanese and superset of Shift-JIS) is not compatible with that of pmrepagent (UTF-8 encoding of Unicode).	Make sure Japanese locales are installed on the UNIX machine. The environment variables on the UNIX machine are not complete. Set the environment variable PM_CODEPAGENAME to MS932. For other languages, set the variable appropriately.

Table 57. Unicode Issues with Informatica and Siebel Data Warehouse

Symptom / Error Message	Probable Cause / Solution
The database connection code page is incorrect for Unicode/Non-Latin code pages.	The database connection code page has to be set manually through the Informatica Workflow Manager. 1 Start the Workflow Manager. 2 Connect to the Siebel_DW_Rep using Administrator as the user. 3 Click Menu Connection > Select Relational. 4 Select the appropriate OLTP connection (DB2_OLTP for DB2, ORA_OLTP for Oracle, MSSQL_OLTP or ODBC_OLTP for MSSQL). 5 For the code page select the UTF-8 encoding of Unicode or the appropriate code page, and then click OK. 6 Select the OLAP connection (DB2_OLAP for DB2, ORA_OLAP for Oracle, MSSQL_OLAP or ODBC_OLTP for MSSQL). 7 For the code page select the UTF-8 encoding of Unicode or the appropriate code page, and then click OK.
Unable to install Informatica PowerMart on non-English UNIX locales (for example, ESN, DEU, FRA, ITA). The following error message appears: "FATAL ERROR during initialization, Invalid language specified, Application terminated."	This error will occur when the locale environment variables are set inconsistently or not at all. These settings are determined by entering the command locale at the UNIX command prompt. Make sure the correct locales are installed on the UNIX machine. Set the environment variable PM_CODEPAGENAME correctly (For example set it to MS932 for Japanese).

UNIX Installation Issues with the Siebel Data Warehouse

Table 58 provides information about issues related to installing the Siebel Data Warehouse on UNIX.

Table 58. UNIX Installation Issues with Siebel Data Warehouse

Symptom / Error Message	Probable Cause / Solution
When starting Informatica in pmserver.log, the following error messages appear: FATAL ERROR: LM_36023 [Tue Mar 11 23:12:41 2003] : (9968|1) The IP address [172.0.129.141] for this machine did not match the IP address [172.20.94] in the repository for server [SIEBEL_DW_SERVER]. FATAL ERROR: SF_34014 [Tue Mar 11 23:13:41 2003] : (9968|1) Server initialization failed. INFO: SF_34014 [Tue Mar 11 23:13:41 2003] : (9968|1) Server shut down.	Launch Workflow Manager, then right-click on SIEBEL_DW_REP and click More. Make sure the Host Name is the machine name where the Informatica Repository Server resides.

When connecting to the Informatica Repository Server from the Repository Administration Console, the following error messages appear:
 	
■ Oracle databases running on Solaris or HP:
 	
Database Connection Error: Database driver event...Error occurred loading library [Bad magic number for shared library: /export/home/oracle/9.0.1.3/lib/libclntsh.sl]Database driver event...Error occurred loading library [libpmora8.sl]
 	
■ DB2 databases running on AIX:
 	
(17764|1) Database Connection Error : Database driver event...Error occurred loading library [No such file or directory]Database driver event...Error occurred loading library [libpmdb2.a]
	Use Oracle 32-bit library.
	
	a Set the environment variable as follows: LD_LIBRARY_PATH=/export/home/oracle/9.0.1.3/lib32
	
	b Restart the Informatica Repository Server service.
	
	■ DB2 databases running on AIX:
	
	(17764|1) Database Connection Error : Database driver event...Error occurred loading library [No such file or directory]Database driver event...Error occurred loading library [libpmdb2.a]
	
	Set the environment variable as follows:
	
	■ For DB2 v8:
	
	`setenv LIBPATH ${LIBPATH}:/usr/opt/db2_08_01/lib`
	
	■ For DB2 v7
	
	`setenv LIBPATH ${LIBPATH}:/usr/lpp/db2_07_01/lib`
	
	Restart the Informatica Repository Server service, after setting the variable.

Table 58. UNIX Installation Issues with Siebel Data Warehouse

Symptom / Error Message	Probable Cause / Solution
When restoring the Informatica Repository, the following error message appears in the dwdb.log: ERROR : OBJM_54543 [Tue Mar 25 17:47:16 2003] : (164|2484) DataBase error: ORA-01461: can bind a LONG value only for insert into a LONG column	The database is Unicode and the environment variable is needed. ■ For Unicode UTF8, set the environment variable to `American_America.UTF8`. ■ For Unicode AL32UTF8, set the environment variable to `American_America.AL32UTF8`.
When starting pmserver on UNIX, the following error message appears: INFO : LM_36039 [Thu Mar 27 11:40:29 2003] : (6772|1) The maximum number of sessions that can run simultaneously is [10]. FATAL ERROR : CMN_1011 [Thu Mar 27 11:40:29 2003] : (6772|1) Error allocating system shared memory of [3000000] bytes for [Load Manager Shared Memory]. Error is [0]: [] FATAL ERROR : SF_34004 [Thu Mar 27 11:40:29 2003] : (6772|1) Server initialization failed. INFO : SF_34014 [Thu Mar 27 11:40:29 2003] : (6772|1) Server shut down.	In the pmserver.cfg file, change MaxSessions to 15 and change LMSharedMem to `3000000`. **NOTE:** For each 10 count increase of MaxSessions, increase the LMSharedMem by 2000000 bytes.

Table 58. UNIX Installation Issues with Siebel Data Warehouse

Symptom / Error Message	Probable Cause / Solution
A session fails (any session), and the following error message appears: TM_6227 Error: codepage incompatible in session [SDE_RecordExtractStart]. The source database DB2_OLTP and server SIEBEL_DW_SERVER do not have compatible code pages. (One way compatibility is required).	For Informatica servers running on UNIX, in the pmserver.cfg file, set ValidateDataCodePages to No. For Informatica servers running on Windows, navigate to Informatica Server setup > Configuration, and remove the check from the Validate Data CodePages check box.

When trying to start the Informatica Server, the following error message appears:
 	
sdcb50a125{qauser7a}/> pmserver
 	
exec(): 0509-036 Cannot load program pmserver because of the following errors:
0509-150 Dependent module libpmcef.a could not be loaded.
0509-022 Cannot load module libpmcef.a.
0509-026 System error: A file or directory in the path name does not exist.
	Set the environment parameters as follows:
	
	`setenv PATH /export/home/informatica/ pm:.:${PATH}`
	`setenv LD_LIBRARY_PATH /export/home/ informatica/ pm:${LD_LIBRARY_PATH}`

Siebel Data Warehouse Application Console (DAC) Issues

Basic Checks

- Make sure that you have installed Java SDK 1.4.2 (and not just the JRE).
- Make sure that in the config.bat file for Windows, and the config.sh file for UNIX, JAVA_HOME is pointing to the JDK (and not the JRE).
- Make sure that the DAC_HOME variable has been properly set.

Table 59 provides information about problems and solutions related to the DAC.

Table 59. Siebel Data Warehouse DAC Issues

Symptom / Error Message	Probable Cause / Solution
Email notification does not work.	To make sure that email notification works with the DAC server, make sure that the following are true: ■ To set up the email notification feature, you must bring up the DAC client on the same machine as the DAC server. ■ The Email server settings (using the DAC client's server setup) point to a valid SMTP server. ■ The SMTP server must be supporting Login Authentication. In the Exchange server, this amounts to enabling the Basic Authentication for SMTP module.
The DAC server does not start.	When trying to bring up the DAC server, make sure that the following are taken care of: ■ The DAC System properties contain the name of the DAC Server Host machine. This specifies the machine on which the DAC server will be run. Make sure that this entry is correct, and points to where you are bringing up the DAC server. Furthermore, there is an entry for the DAC Server OS, which should specify the OS where the DAC server is running. Make sure that this entry is correct. ■ Make sure that the $DAC_HOME/lib directory contains the needed JDBC libraries. ■ You may want to edit the startserver.bat file temporarily, and run the DAC server in such a way that messages show in the command window. (Use JAVA rather than JAVAW for Windows OS). This will allow you to see a lot of run-time information, as the server boots up.

Table 59. Siebel Data Warehouse DAC Issues

Symptom / Error Message	Probable Cause / Solution
The DAC server does not start.	■ In the DAC System properties, there is an entry called Logging Level. You can increase the amount of information logged by decreasing the logging-threshold. In other words, if the value currently is SEVERE, make it INFO. This will show more information. If you still cannot find the cause, you may want to change it to FINE or FINEST. These levels are the standard levels of Java's Logging API. (SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST). ■ Be aware that as you lower the logging threshold, a lot of run-time information will start getting logged. While this may be advantageous for troubleshooting, you do not want to leave it like this in production mode. Excessive logging will generate large log files quickly and use excessive disk space. There is also a modest performance hit with excessive logging. ■ Make sure that there is no DAC server already running on the machine. Only one DAC server instance can exist per machine (and per DAC repository). Some non-visible forms of the DAC server may be running as a background process. ■ Make sure that the DAC repository database instance is actually up and running. The DAC repository database being down or the loss of network connectivity to the database will cause the DAC server not to come up, or to go down if it was running. ■ Make sure that the DAC repository database connection pool size is moderate. In other words, do not try to acquire an excessive number of connections when the database administrator may have put restrictions on how many a single application can request.

Table 59. Siebel Data Warehouse DAC Issues

Symptom / Error Message	Probable Cause / Solution
When starting the DAC client and server for the first time (after configuring the DAC server by using the DAC client), the DAC client's icon for the Server status stays red. In other words, it suggests the server is down, even though it is running.	This is expected behavior. The first time the DAC server comes up against a repository, it creates a Unique ID for itself, and the repository. Thereafter, when any client is started against the repository, it knows where to expect the DAC server to be, and furthermore, sends a valid identification token containing the Unique ID. Therefore, after bringing up the DAC server the first time, you may want to close the existing open clients, and bring them up again. This problem is specific to bringing up the DAC client and server for the first time. Once Unique ID has been created, the behavior of the Server Monitor icon will be as expected.
When exporting the DAC repository against a DB2 database, failures occur. An error message may appear and the export.log file may have an entry similar to the following: "Error while moving to reading next record."	This problem is specific to DB2 databases and occurs only when the following tables are exported: ■ WETL_DBCONN ■ W_ETL_SYSPROP ■ W_ETL_SERVER This problem tends to occur when the database is under significant load. Therefore, it is recommended that you export these tables individually.
Running the DAC server on AIX machines.	Special care must be taken when bringing up the DAC server on AIX machines. For example, if the repository already contains database connection information entries, and Informatica Server entries, these must be modified if you are moving the DAC server to an AIX machine. In particular, you need to delete all the password fields, and re-enter them. For more information, see "Siebel Analytics Applications Installation and Configuration Process on UNIX" on page 106.
The DAC client does not run on machines where PopChart Image Server from CORDA Technologies, Inc., is running.	Run the DAC client on a machine that does not have PopChart Image Server from CORDA Technologies, Inc., installed.

Siebel Data Warehouse Upgrade Issues

Table 60 provides information about problems and solutions related to issues with upgrading the Siebel Data Warehouse.

Table 60. Upgrade Issues

Symptom / Error Message	Probable Cause / Solution
When upgrading from Siebel Data Warehouse version 7.5.3 to version 7.7, duplicate rows are inserted into a mini dimension table from the dimension table.	Before upgrading, all leading and trailing spaces need to be removed from the data. 1 Before running the workflow for the mini dimension that appears in the upgrade folder, modify the data to remove leading and trailing spaces by using the RTRIM and LTRIM functions. For example: set(col)=rtrim(col) 2 In cases where you find leading and trailing spaces, modify the SQL override for the mini dimension mapping in the main folder as shown in the example below: SELECT DISTINCT CASE WHEN G1.VAL IS NULL THEN RTRIM(LTRIM(DS.AREA)) ELSE G1.VAL END AS AREA, CASE WHEN G1.NAME IS NULL THEN RTRIM(LTRIM(DS.AREA)) ELSE G1.NAME END AS AREA_I, CASE WHEN G2.VAL IS NULL THEN RTRIM(LTRIM(DS.SUB_AREA)) ELSE G2.VAL END AS SUB_AREA CASE WHEN G2.NAME IS NULL THEN RTRIM (LTRIM(DS.SUB_AREA)) ELSE G2.NAME END AS SUB_AREA_I CASE WHEN G3.VAL IS NULL THEN RTRIM(LTRIM(DS.SEV_CD)) ELSE G3.VAL END AS SEV_CD CASE WHEN G3.NAME IS NULL THEN RTRIM (LTRIM(DS.SEV_CD)) ELSE G3.NAME END AS SEV_CD_I CASE WHEN G4.VAL IS NULL THEN RTRIM(LTRIM(DS.PRIO_CD)) ELSE G4.VAL END AS PRIO_CD CASE WHEN G4.NAME IS NULL THEN RTRIM (LTRIM(DS.PRIO_CD)) ELSE G4.NAME END AS PRIO_CD_I FROM W_SRVREQ_DS DS, W_LOV_G G1, W_LOV_G G2, W_LOV_G G3, W_LOV_G G4 WHERE W_SRVREQ_DS LEFT OUTER JOIN V_LOV_G G1 ON DS.AREA = G1.NAMEVAL AND G1.TYPE = ‘SR_AREA’ LEFT OUTER JOIN W_LOV_G G2 ON DS.SUB_AREA = G2.NAMEVAL AND G2.TYPE = 'SR_AREA' LEFT OUTER JOIN W_LOV_G G3 ON DS.SEV_CD = G3.NAMEVAL AND G3.TYPE = 'SR_SEVERITY' LEFT OUTER JOIN W_LOV_G G4 ON DS.PRIO_CD = G4.NAMEVAL AND G4.TYPE = 'SR_PRIORITY'}

A Siebel Analytics Bridge Applications

This appendix describes the Siebel Analytics Bridge applications and provides information about setting up the Siebel transactional database for use with the Bridge applications.

This appendix contains the following topics:

Overview of Analytics Bridge Applications

Siebel Analytics Bridge applications allow you to use the newer functionality of Siebel Analytics with previous versions of Siebel Business and Industry operational applications. Bridge applications are installed, based on the license file, when you install Siebel Analytics Applications.

After you install a Bridge application, you then need to set up the Siebel transactional database for use with the Bridge application. For instructions, see "Setting Up the Siebel Transactional Database for Analytics Bridge Applications" on page 257.

Versions of Analytics Bridge Applications and SIF Files

Table 61 shows which Siebel Analytics Bridge application should be used for the different versions of Siebel Business Applications.

Table 61. Siebel Analytics Bridge Applications and SIF Files

Siebel Business Applications Version	Analytics Platform Version	Analytics Applications Version	Analytics Bridge
6.3.x	7.8.2	7.8.2	Siebel Analytics Bridge - 6.3
7.5.x	7.8.2	7.8.2	Siebel Analytics Bridge - 7.5
7.7.x	7.8.2	7.8.2	Siebel Analytics Bridge - 7.7

Analytics Bridge Application Architecture

Figure 20 shows the architecture of Siebel Analytics Bridge applications.

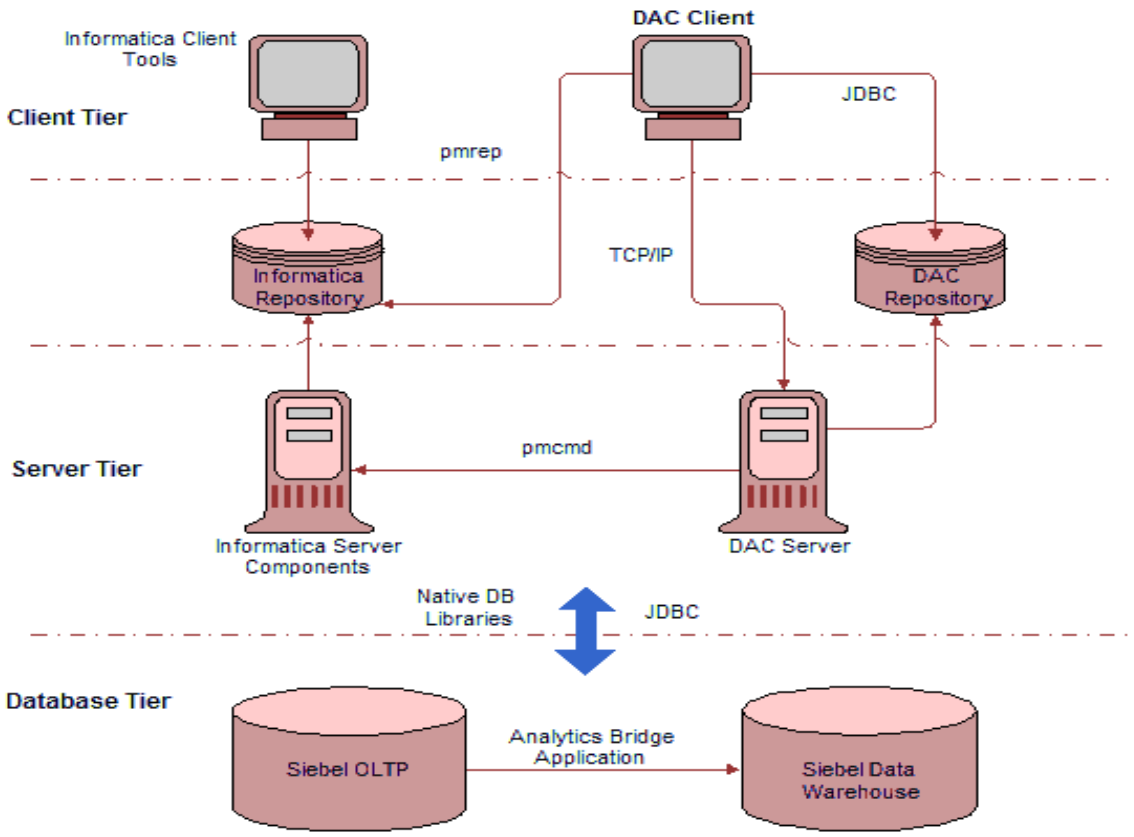


Figure 20. Siebel Analytics Bridge Application Architecture

Siebel Data Warehouse Data Model

The Siebel Analytics Bridge applications use the data warehouse as defined in *Siebel Data Warehouse Data Model Reference, Version 7.8.2.*

CAUTION: Because of inherent differences in the underlying application schemas, the Siebel Analytics Bridge applications do not perform a complete mapping of all Presentation layer columns in the Siebel Analytics repository. This may also affect those dashboard reports built from the columns that are not mapped. See *Technical Note 429*, located on Siebel SupportWeb, for a list of Presentation layer columns and reports known to be affected.

Setting Up the Siebel Transactional Database for Analytics Bridge Applications

Follow this procedure to set up the Siebel transactional database for use with a Siebel Analytics Bridge application.

To set up the Siebel transactional database

1 Import new transactional database schema definitions:

   a Copy the appropriate .sif file for the Analytics Bridge application you installed:

Bridge Application	Copy	From	To
Analytics Bridge - 6.3	SRW_OLTP_63.sif	SiebelAnalytics\dwrep	\tools\objects
Analytics Bridge - 7.5	SRW_OLTP_75.sif	SiebelAnalytics\dwrep	\tools\objects
Analytics Bridge - 7.7	SRW_OLTP_77.sif	SiebelAnalytics\dwrep	\tools\objects

   b Open Siebel Tools and point to the development database.

   c Lock the appropriate project.

      ❑ For Siebel Business Applications, lock the Newtable project

      ❑ For Siebel Industry Applications, lock the Table ETL and Table ETL CCM projects.

   d Navigate to Tools > Import from Archive.

   e Select Open.

      The Import Wizard—Preview screen appears, listing the Analytics Bridge tables.

   f In the Conflict Resolution area, select the option Merge the object definition from the archive file with the definition in the repository, and click Next.

   g A dialog box appears, informing you that the operation will modify your repository and asking if you want to proceed. Click Yes.

   h The Import Wizard — Summary screen appears, showing a summary of the .sif changes to the repository files. Click Finish.

2 Apply the new schema definitions to the transactional database:

   a In Siebel Tools, go to the Table screen and search for the tables S_ETL*.

   b Select all the S_ETL* tables that appear, and click Apply at the top of the screen.

   c In the Apply Schema dialog box, under the entry Tables, select Current Query from the dropdown list.

   d Under the subsequent entries, enter the connection parameters for the development database. (Your connection information will vary depending on your particular database platform and on the configuration of your particular database environment.)

e Click Apply.

3 Create Image tables:

a In the DAC, navigate to Design > Tables.

b On the toolbar, click Query.

c In the Image Suffix field, enter !Null.

d Select a record.

e Right-click and select Change Capture Scripts > Generate Image and Trigger Scripts.

f In the Trigger and Image Tables dialog box, select All Tables in the List.

g Select the appropriate database type, and check the Unicode Database option if the database is Unicode.

h Check the Generate Image Table Scripts option, and click OK.

This process generates a SQL script.

i Save the SQL script and pipe it to the transactional database.

4 For the Analytics Bridge - 6.3, import language-independent Siebel seed data.

NOTE: This step is not required for Analytics Bridge - 7.5 or Analytics Bridge - 7.7.

a Copy the following files:

Copy	From	To
seed_63_bridge.dat	SiebelAnalytics\dwrep	Local machine
seed_63_bidge_<*xxx*>.inp (where *xxx* corresponds to the database platform being used)	SiebelAnalytics\dwrep	Local machine

b Open a command line, and go to the same directory where you copied the .dat and .inp files.

c Replace each of the connection parameters with the values appropriate to your database environment, and then run the appropriate import command from the following table.

Database Platform	Run Import Command
DB2UDB	$SIEBELSERVERROOT\bin\dataimp /u $UserName /p $Password /c "$ODBCDataSource" /d $DatabaseOwner /f seed_63_bridge.dat /w y /q 100 /h Log /x f /i seed_63_bridge_db2udb.inp /l seed_63_bridge_db2udb.log

Database Platform	Run Import Command
MSSQL	$SIEBELSERVERROOT\bin\dataimp /u $UserName /p $Password /c "$ODBCDataSource" /d $DatabaseOwner /f seed_63_bridge.dat /w y /q 100 /h Log /x f /i seed_63_bridge_mssql.inp /l seed_63_bridge_mssql.log
Oracle	$SIEBELSERVERROOT\bin\dataimp /u $UserName /p $Password /c "$ODBCDataSource" /d $DatabaseOwner /f seed_63_bridge.dat /w y /q 100 /h Log /x f /i seed_63_bridge_oracle.inp /l seed_63_bridge_oracle.log

5 Using the DAC, run the Complete ETL execution plan to perform a full load of data into the data warehouse.

For instructions on running an ETL using the DAC, see "Executing an ETL with the DAC" on page 155.

Files for Siebel Analytics Bridge - 6.3

Table 62 describes the application files for Siebel Analytics Bridge - 6.3.

Table 62. Files for Siebel Analytics Bridge - 6.3

File Name	Purpose of File
SRW_OLTP_<*xxx*>.sif where *xxx* is the transactional database version	The Siebel Archive File containing the logical definition of the ETL tables.
seed_<*xxx*>_bridge.dat where *xxx* is the transactional database version	The Siebel Seed Data file containing language-independent seed data records for the following tables: ■ S_LST_OF_VAL ■ S_RESP ■ S_APP_VIEW ■ S_APP_VIEW_RESP
seed_<*xxx*>_bridge_<db>.inp where *xxx* is the transactional database version and <db> is one of: ■ DB2UDB ■ MSSQL ■ Oracle	Import file used to control the import of the seed_<*xxx*>_bridge.dat file from a Siebel Cross-Industry Application to a particular database platform.

Table 62. Files for Siebel Analytics Bridge - 6.3

File Name	Purpose of File
seed_<*xxx*>_bridge_<lan>.dat where *xxx* is the transactional database version	Set of Siebel seed data files containing translations for all the Siebel Analytics metadata objects. Each file contains all the translations for a single language. These files place translation-related seed data into the S_MSG table in the Siebel OLTP database.
seed_<*xxx*>_bridge_<lan>.inp where *xxx* is the transactional database version	Set of import files used to control the import of the language seed files to the database.

Platform Support for Analytics Bridge Applications

For information about hardware and software requirements for Siebel Analytics Bridge applications, see "Operating System, Driver, and Connectivity Requirements for Siebel Data Warehouse" on page 31.

NOTE: The Siebel Analytics Bridge - 7.7 does not support MSSQL Server 7.0.

B Installing the Usage Accelerator

This appendix provides instructions for installing the Usage Accelerator application that is a part of the Siebel eBusiness 7.5.3 to 7.7 Analytics Bridge application. For information about this bridge application, see "Overview of Analytics Bridge Applications" on page 255.

NOTE: The instructions in this appendix are required only if you are planning to use the Usage Accelerator application that is a part of the Siebel eBusiness 7.5.3 to 7.7 Analytics Bridge application. If you licensed the Usage Accelerator application versions 7.7 or 7.7.1, you should follow the installation instructions in *Siebel Analytics Installation and Configuration Guide*. No additional steps are required to install these versions of the Usage Accelerator application.

This appendix contains the following topics:

Installing the Usage Accelerator

The Usage Accelerator installation process consists of installing the Usage Accelerator application and importing seed data into the transactional database.

You install the Usage Accelerator application using the standard Siebel Analytics Installer. The installation options that are available to you depend on the license key you enter.

To install the Usage Accelerator application

1. In the Siebel Analytics Installer, click Next in the Welcome screen.
2. Accept the terms of the license agreement.
3. Browse for the appropriate license key file.
4. Select the Usage Accelerator installation option.

   The Installer installs the application on your machine.
5. Import new transactional database schema definitions required for running the ETL process for the Usage Accelerator.
   a. Copy the following file:

Copy	From	To
SRW_OLTP_75.sif	SiebelAnalytics\dwrep	\tools\objects

   b. Open Siebel Tools and point to the development database.

c Navigate to Repository > Import from Archive, and select the file SRW_OLTP_75.sif.

d Click Open.

The Import Wizard—Preview screen appears, listing the Analytics Bridge tables.

e Click Next.

NOTE: All the objects imported in this process should be new to your repository; therefore, there should be no conflict due to an import. In case of conflict, in the Conflict Resolution area, select the option Merge the object definition from the archive file with the definition in the repository.

f A dialog box appears, informing you that the operation will modify your repository and asking if you want to proceed. Click Yes.

g The Import Wizard — Summary screen appears, showing a summary of the .sif changes to the repository files. Click Finish.

6 Apply the new schema definitions to the transactional database:

a In Siebel Tools, go to the Table screen. Enter the following search string:

```
S_ETL_PARAM or S_ETL_SRC_TABLE or S_NQ_ACCT or S_NQ_JOB or S_ETL_R_IMG_* or
S_ETL_D_IMG_* OR S_ETL_I_IMG_*
```

You should see the following tables:

- S_ETL_D_IMG_1
- S_ETL_D_IMG_10 to S_ETL_D_IMG_13
- S_ETL_D_IMG_133 to S_ETL_D_IMG_139
- S_ETL_D_IMG_14
- S_ETL_D_IMG_140 to S_ETL_D_IMG_147
- S_ETL_D_IMG_15 to S_ETL_D_IMG_19
- S_ETL_D_IMG_2
- S_ETL_D_IMG_20 to S_ETL_D_IMG_29
- S_ETL_D_IMG_3
- S_ETL_D_IMG_30 to S_ETL_D_IMG_39
- S_ETL_D_IMG_4
- S_ETL_D_IMG_40 to S_ETL_D_IMG_49
- S_ETL_D_IMG_5
- S_ETL_D_IMG_50 to S_ETL_D_IMG_59
- S_ETL_D_IMG_6
- S_ETL_D_IMG_60 to S_ETL_D_IMG_69
- S_ETL_D_IMG_7
- S_ETL_D_IMG_70 to S_ETL_D_IMG_79

- S_ETL_D_IMG_8
- S_ETL_D_IMG_80 to S_ETL_D_IMG_83
- S_ETL_D_IMG_9
- S_ETL_I_IMG_1
- S_ETL_I_IMG_10 to S_ETL_I_IMG_13
- S_ETL_I_IMG_133 to S_ETL_I_IMG_139
- S_ETL_I_IMG_14
- S_ETL_I_IMG_140 to S_ETL_I_IMG_147
- S_ETL_I_IMG_15 to S_ETL_I_IMG_19
- S_ETL_I_IMG_2
- S_ETL_I_IMG_20 to S_ETL_I_IMG_29
- S_ETL_I_IMG_3
- S_ETL_I_IMG_30 to S_ETL_I_IMG_39
- S_ETL_I_IMG_4
- S_ETL_I_IMG_40 to S_ETL_I_IMG_49
- S_ETL_I_IMG_5
- S_ETL_I_IMG_50 to S_ETL_I_IMG_59
- S_ETL_I_IMG_6
- S_ETL_I_IMG_60 to S_ETL_I_IMG_69
- S_ETL_I_IMG_7
- S_ETL_I_IMG_70 to S_ETL_I_IMG_79
- S_ETL_I_IMG_8
- S_ETL_I_IMG_80 to S_ETL_I_IMG_83
- S_ETL_I_IMG_9
- S_ETL_R_IMG_1
- S_ETL_R_IMG_10 to S_ETL_R_IMG_13
- S_ETL_R_IMG_133 to S_ETL_R_IMG_139
- S_ETL_R_IMG_14
- S_ETL_R_IMG_140 to S_ETL_R_IMG_147
- S_ETL_R_IMG_15 to S_ETL_R_IMG_19
- S_ETL_R_IMG_2
- S_ETL_R_IMG_20 to S_ETL_R_IMG_29
- S_ETL_R_IMG_3

- S_ETL_R_IMG_30 to S_ETL_R_IMG_39
- S_ETL_R_IMG_4
- S_ETL_R_IMG_40 to S_ETL_R_IMG_49
- S_ETL_R_IMG_5
- S_ETL_R_IMG_50 to S_ETL_R_IMG_59
- S_ETL_R_IMG_6
- S_ETL_R_IMG_60 to S_ETL_R_IMG_69
- S_ETL_R_IMG_7
- S_ETL_R_IMG_70 to S_ETL_R_IMG_79
- S_ETL_R_IMG_8
- S_ETL_R_IMG_80 to S_ETL_R_IMG_83
- S_ETL_R_IMG_9
- S_ETL_PARAM
- S_ETL_SRC_TABLE
- S_NQ_ACCT
- S_NQ_JOB

b. Select all these tables, and click Apply at the top of the screen.

c. In the Apply Schema dialog box, under the entry Tables, select Current Query from the dropdown list.

d. Under the subsequent entries, enter the connection parameters for the development database. (Your connection information will vary depending on your particular database platform and on the configuration of your particular database environment.)

e. Click Apply.

7. Import repository objects to enable embedded Analytics.

NOTE: This step is required only if are planning to use the Usage Accelerator in an integrated environment with a Siebel eBusiness application version 7.5.3.

a. Lock the following projects:

- Analytics
- ERM
- Siebel Field Service
- Siebel Marketing Enterprise
- Siebel Channel
- Siebel Sales Enterprise
- Siebel Service Enterprise

- Siebel Universal Agent

b Copy the following file:

Copy	From	To
SRW_753_77_Plus_UA.sif	SiebelAnalytics\dwrep	\tools\objects

c In Siebel Tools, navigate to Repository > Import from Archive, and select the file SRW_753_77_Plus_UA.sif.

d Click Open.

The Import Wizard—Preview displays a list of tables.

e In the Conflict Resolution area, select the option Merge the object definition from the archive file with the definition in the repository, and click Next.

NOTE: Do not select the Overwrite option, because this will delete existing objects in the repository.

f The Import Wizard—Summary screen displays a summary of the .sif changes to the repository files. Click Finish.

8 Recompile the SRF file.

NOTE: This step is required only if are planning to use the Usage Accelerator in an integrated environment with a Siebel eBusiness application version 7.5.3.

a In Windows Explorer, navigate to \tools\objects\enu, and rename the siebel.srf file.

b In Siebel Tools, navigate to Tools > Compile Projects, and select the Locked Projects radio button.

c Under Siebel Repository, select the new SRF file you created in Step a, and click Compile.

To see the Usage Accelerator screen in your application, the new SRF file must replace the siebel.srf file you were using.

9 Confirm the Analytics Server name to run Embedded Analytics.

NOTE: This step is required only if are planning to use the Usage Accelerator in an integrated environment with a Siebel eBusiness application version 7.5.3.

a Log in to the Siebel application as the administrator.

b Navigate to Site Map > Administration - Integration > WI Symbolic URL List.

c In the Administration - Integration screen, from the visibility filter, select Host Administration.

d In the Host Administration list, locate the record Name=[AnalyticsServerName].

e Change [AnalyticsServerName] to the name of the machine hosting your Analytics Server.

10 Import language-independent Siebel seed data.

a Copy the appropriate seed data files:

Copy	From	To
seed_753_77_plus.dat	SiebelAnalytics\dwrep	Local machine
seed_753_77_plus_<*xxx*>.inp where *xxx* corresponds to the database platform you are using	SiebelAnalytics\dwrep	Local machine

b Open a command line, and go to the same directory where you copied the .dat and .inp files.

c Replace each of the connection parameters with the values appropriate to your database environment, and then run the appropriate import command from the following table.

Database Platform	Run Import Command
DB2UDB	$SIEBELSERVERROOT\bin\dataimp /u $UserName /p $Password /c "$ODBCDataSource" /d $DatabaseOwner /f seed_753_77_plus.dat /w y /q 100 /h Log /x f /i seed_753_77_plus_db2udb.inp /l seed_753_77_plus_db2udb.log
MSSQL	$SIEBELSERVERROOT\bin\dataimp /u $UserName /p $Password /c "$ODBCDataSource" /d $DatabaseOwner /f seed_753_77_plus.dat /w y /q 100 /h Log /x f /i seed_753_77_plus_mssql.inp /l seed_753_77_plus_mssql.log
Oracle	$SIEBELSERVERROOT\bin\dataimp /u $UserName /p $Password /c "$ODBCDataSource" /d $DatabaseOwner /f seed_753_77_plus.dat /w y /q 100 /h Log /x f /i seed_753_77_plus_oracle.inp /l seed_753_77_plus_oracle.log

11 Associate the new Usage Accelerator responsibilities with the users whom you want to see the Usage Accelerator screen.

The Usage Accelerator responsibilities are the following:

- Usage Accelerator Tracking
- Usage Accelerator - Sales Rep
- Usage Accelerator - Sales Manager
- Usage Accelerator - Sales Executive
- Usage Accelerator - Administrator

7.7 Usage Accelerator-Analytics Bridge Files

The contents of each of the files included in the Inca_Horz_Bridge folder are shown in Table 63.

Table 63. 7.7 Usage Accelerator Files

File Name	Purpose of File
SRW_OLTP_75.sif	Archive file containing schema definitions for tables and indices that support ETL processes and Analytics functionality.
SRW_753_77_Plus_UA.sif	Archive file containing the logical definition of the Usage Accelerator integration objects.
seed_753_77_plus.dat	Siebel Seed Data file containing language-independent seed data records for the following tables: ■ S_WI_SYMURL ■ S_WE_SYMURL_ARG ■ S_RESP ■ S_APP_VIEW ■ S_APP_VIEW_RESP
■ seed_753_77_plus_db2udb.inp ■ seed_753_77_plus_mssql.inp ■ seed_753_77_plus_oracle.inp	Import file used to control the import of the seed_inca_bridge.dat file to a particular database platform.

C Siebel Data Warehouse for Life Sciences Data Considerations

This chapter contains the following topics:

Importing Syndicated Data into Siebel Data Warehouse

Pharmaceutical companies purchase weekly and monthly sales and prescription data, known as syndicated data, from third-party vendors such as IMS, NDC, and Cegedim. Syndicated data vendors acquire data from drug wholesalers and retailers on a daily, weekly, and monthly basis and compile a master file of customers (wholesalers, pharmacies, hospitals, and doctors) and sales or prescription transactions for customers. Measures include indirect sales, indirect units, and prescriptions and differ by vendor and periodicity. It is used for sales force analysis reporting and customer targeting.

The data is derived from panels of physicians, pharmacies, and so on, and projected nationally. Since the panels may change on a monthly basis, syndicated data suppliers tend to change their projections of sources on a monthly basis leading to full restatements of historical data. Thus, pharmaceutical companies are required to refresh fully the data in their data warehouses. In addition, weekly data requires incremental loading.

After it is delivered by the vendor, the syndicated data must be fully reloaded into the Siebel Pharma data warehouse in a timely manner and made available to users in order for them to make use of sales force analysis reporting.

This section includes the following topics:

Syndicated Loading Definitions

Table 64 provides terms and definitions related to syndicated loading.

Table 64. Syndicated Loading Definitions

Term	Definition
Syndicated data	Third-party data from vendors that shows sales and prescription results for client defined markets.
Brick	Micro sales geographic area defined by vendor that contains prescription and sales trends for clients' product groupings or markets. Bricks do not exist in the US where the micro sales geographic area is commonly the postal code or zip code.
NRx	Abbreviation of new prescriptions. A new prescription is defined as dispensed prescriptions given a new number by the pharmacy, not necessarily new therapy for the patient.
TRx	Abbreviation of total prescriptions. TRx = NRx + Refills. After the first time a prescription is filled, when it is refilled, the pharmacy refers back to the previous drug ID number and notes this as a refill.
Indirect sales	Total drug wholesalers product sales values to drug retailers (pharmacies) by brick or zip code. Sales values are calculated as units multiplied by client-selected price, where the price is the warehouse acquisition cost unless the client has chosen to have a different price applied to its direct sales or government depot sales.
Indirect units	Total drug wholesalers product sales units to drug retailers (pharmacies) by brick or zip code. Sales unit quantity is determined by multiplying package factor to obtain unit value (for example, one carton x 24 bottles = 24 units).

Data Types Supported in the Siebel Data Warehouse

The Siebel Data Warehouse supports multiple data types defined by data source type and periodicity, as described in Table 65.

Table 65. Data Types Supported in Siebel Data Warehouse

Data Type	Description
Prescription data by contact	Monthly NRx and TRx data for client defined markets that include competitor products data by physician.
Indirect sales brick level	Monthly wholesaler sales and unit values data for client defined markets that include competitor products aggregated by brick.
Indirect sales account level	Monthly wholesaler sales and unit values for company products by pharmacy.

Table 65. Data Types Supported in Siebel Data Warehouse

Data Type	Description
Indirect sales zip level	Monthly wholesaler sales and unit values for client defined markets that include competitor products aggregated by postal code.
Direct sales account level	Monthly direct factory sales data and unit values to wholesalers for company's products by wholesaler.
Direct sales brick level	Monthly direct factory sales data and unit values to wholesalers for company's products by brick.
Direct sales zip level	Monthly direct factory sales data and unit values to wholesalers for company's products by postal code.
Weekly prescription data	Weekly NRx and TRx data for client defined markets that include competitor products by physician.
Plan level prescription data	Monthly prescription data by managed care plan for client defined markets that includes competitor products.
Sales market	**Incentives.** Monthly incentive compensation data sourced from internal incentive compensation systems and loaded as a new data type in the Siebel Data Warehouse.
Sales market	**Modified.** Monthly incentive compensation data sourced from internal incentive compensation systems and loaded as a new data type in the Siebel Data Warehouse.
Plan level physician prescription data	Monthly prescription data for physicians associated with a managed care plan.
Prescription data by zip code	Monthly prescription data for client defined markets that includes competitor products aggregated by postal code.
Prescription data by brick	Monthly prescription data for client defined markets that include competitor products aggregated by brick.

Loading Syndicated Data into the Siebel Data Warehouse

The options for loading syndicated data into the Siebel Data Warehouse include the following:

- Loading the Siebel Data Warehouse from the Siebel transactional database (OLTP) S_SYND_DATA table.

  Customers who are already using the EIM process or SQL Loader to load external syndicated data into the Siebel operational system can use data populated in S_SYND_DATA as the source to the Siebel Data Warehouse. Existing syndicated data from the Siebel transactional database, stored in the S_SYND_DATA table, is extracted, transformed, and loaded by prebuilt routines and populated into the W_SYND_DATA_F and W_SYND_MKT_F fact tables in the Siebel Data Warehouse. W_SYND_DATA_F stores Rx or sales measures by product. W_SYND_MKT_F stores Rx or sales measures by market and allows calculation of market share data.

  The data is first copied from external data tapes (or other provided media) into the interface tables for the Siebel Industry Application (SIA), using a native database data loading utility, such as SQL* Loader, and following the rules in Table 66 on page 273. Using EIM, data is transferred from the Siebel interface tables to predefined destination columns in the base tables of the Siebel Life Sciences database. The application administrator uses SQL*Loader to populate the ID field with a unique sequential value, a process that provides either full or partial table level extraction.

  In order to correctly load the Siebel Data Warehouse, there is a set of population rules that are required to load S_SYND_DATA. This will guarantee minimal configuration required on the Analytics side and correct population of the Siebel Data Warehouse.

  Once the data is loaded into S_SYND_DATA, you can use the LS SYNDICATED FACT GROUP, in the Pharma subject area, to load the syndicated data into the Siebel Data Warehouse.

- Loading from a flat file source using syndicated data staging tables, cross-referencing against extension columns in Siebel base tables.

  Syndicated data source files can be loaded directly into Siebel Data Warehouse staging tables. In the staging tables, Siebel row IDs are matched with external sources keys for joining the external source keys with Siebel Analytics keys.

  You can cross-reference between Siebel row IDs and the external source keys by using existing cross-reference columns in S_CONTACT and S_ORG_EXT for customer data. For example, the IMS ID for a physician may be populated on the ME number column. The configured ETL will perform a lookup against the new column to resolve the IMS ID to Siebel row ID relationship. The process entails first populating the necessary cross-references in the Siebel base tables. The next step is to configure the SDE ETL to extract from external source data files. The prebuilt SIL ETL performs a lookup against the cross-reference to associate the syndicated data record with the correct Siebel row ID and creates the fact record and warehouse ID with the correct relationship.

- Loading product data into the Siebel Data Warehouse by creating a new staging table that looks like the source data and then creating the market aggregation from the product table.

  This method requires the most customization. It involves creating a new staging table with columns for each time period metric and configuring the SDE and SIL mappings to load the data directly into the Data Warehouse. You then need to create a mapping to populate the staging table and a new SIL mapping that will use the staging table as a source and the W_SYND_DATA_F table as a target. The W_SYND_MKT_F table can then be loaded from W_SYND_DATA_F.

NOTE: To make sure the data is correctly loaded in the Siebel Data Warehouse, the Siebel source table S_SYND_DATA needs to be populated correctly, based on Table 66 on page 273.

Table 66. S_SYND_DATA Data Population Rules

Required Codes	Data Types	Required fields (x) and data population rules for correct rollup in Siebel Analytics														
Syndicated Data mandatory codes to be populated in DATA_SRC_CD	Data Type supported with DATA_SRC_CODE	Territory ID TERR_ID	Period ID PERIOD_ID	Position ID POSITION_ID	Postal Code ZIPCODE	Account ID OU_EXT_ID	Contact ID CON_ID	Brick ID AREA_ID	Payor Level PLAN_ID Foreign Key to S_INS_PLAN	Product Level PRDINT_ID (always try to populate data at the lowest level - form strength)	Market Level MARKET_ID	Attrbute 1 (Attrib_01)	Attrbute 1 (Attrib_02)	Attrbute 1 (Attrib_03)	Attrbute 1 (Attrib_04)	Siebel Pharma Analytics Business Model Data Source Codes
RXPrf	Prescription data by Contact		x	x			x		Total	Form Strength	x	NRx	TRx	Market NRx	Market TRx	1
SlsIndBrk	Indirect Sales Brick Level		x	x				x	Total	Form Strength	x	Product Sales $	Product Sales Units	Market Sales $	Market Sales Units	22
SlsIndAct	Indirect Sales Account Level		x	x		x			Total	Form Strength	x	Product Sales $	Product Sales Units	Market Sales $	Market Sales Units	21

Table 66. S_SYND_DATA Data Population Rules

Required Codes	Data Types	Required fields (x) and data population rules for correct rollup in Siebel Analytics														
Syndicated Data mandatory codes to be populated in DATA_SRC_CD	Data Type supported with DATA_SRC_CODE	Territory ID TERR_ID	Period ID PERIOD_ID	Position ID POSITION_ID	Postal Code ZIPCODE	Account ID OU_EXT_ID	Contact ID CON_ID	Brick ID AREA_ID	Payor Level PLAN_ID Foreign Key to S_INS_PLAN	Product Level PRDINT_ID (always try to populate data at the lowest level - form strength)	Market Level MARKET_ID	Attrbute 1 (Attrib_01)	Attrbute 1 (Attrib_02)	Attrbute 1 (Attrib_03)	Attrbute 1 (Attrib_04)	Siebel Pharma Analytics Business Model Data Source Codes
SlsIndZip	Indirect Sales Zip Level		x	x	x				Total	Form Strength	x	Product Sales $	Product Sales Units	Market Sales $	Market Sales Units	23
SlsDirAct	Direct Sales Account Level		x	x		x			Total	Form Strength	x	Product Sales $	Product Sales Units			11
SlsDirBrk	Direct Sales Brick Level		x	x				x	Total	Form Strength	x	Product Sales $	Product Sales Units			12
SlsDirZip	Direct Sales Zip Level		x	x	x				Total	Form Strength	x	Product Sales $	Product Sales Units			13

Table 66. S_SYND_DATA Data Population Rules

Required Codes	Data Types	Required fields (x) and data population rules for correct rollup in Siebel Analytics														
Syndicated Data mandatory codes to be populated in DATA_SRC_CD	Data Type supported with DATA_SRC_CODE	Territory ID TERR_ID	Period ID PERIOD_ID	Position ID POSITION_ID	Postal Code ZIPCODE	Account ID OU_EXT_ID	Contact ID CON_ID	Brick ID AREA_ID	Payor Level PLAN_ID Foreign Key to S_INS_PLAN	Product Level PRDINT_ID (always try to populate data at the lowest level - form strength)	Market Level MARKET_ID	Attrbute 1 (Attrib_01)	Attrbute 1 (Attrib_02)	Attrbute 1 (Attrib_03)	Attrbute 1 (Attrib_04)	Siebel Pharma Analytics Business Model Data Source Codes
RXEVM	Weekly RX Data by Contact		x	x			x		Total	Form Strength	x	Product NRx	Product TRx	Market NRx	Market TRx	0
RXPT	Plan Level Rx Data		x	x					Plan	Form Strength	x	Product NRx	Product TRx	Market NRx	Market TRx	6
RXSMI	Sales Market - Incentives		x	x	x				Total	Form Strength	x	Product NRx	Product TRx	Market NRx	Market TRx	3
RXSMM	Sales Market - Modified		x	x	x				Total	Form Strength	x	Product NRx	Product TRx	Market NRx	Market TRx	4
RXXPT	Plan Level Physician Rx Data		x	x			x		Plan	Form Strength	x	Product NRx	Product TRx	Market NRx	Market TRx	5

Table 66. S_SYND_DATA Data Population Rules

Required Codes	Data Types	Required fields (x) and data population rules for correct rollup in Siebel Analytics														
Syndicated Data mandatory codes to be populated in DATA_SRC_CD	Data Type supported with DATA_SRC_CODE	Territory ID TERR_ID	Period ID PERIOD_ID	Position ID POSITION_ID	Postal Code ZIPCODE	Account ID OU_EXT_ID	Contact ID CON_ID	Brick ID AREA_ID	Payor Level PLAN_ID Foreign Key to S_INS_PLAN	Product Level PRDINT_ID (always try to populate data at the lowest level - form strength)	Market Level MARKET_ID	Attrbute 1 (Attrib_01)	Attrbute 1 (Attrib_02)	Attrbute 1 (Attrib_03)	Attrbute 1 (Attrib_04)	Siebel Pharma Analytics Business Model Data Source Codes
RXZip	Prescription data by Zipcode		x	x	x				Total	Form Strength	x	Product NRx	Product TRx	Market NRx	Market TRx	7
RXBrk	Prescription data by Brick		x	x				x	Total	Form Strength	x	Product NRx	Product TRx	Market NRx	Market TRx	8

W_SYND_DATA_F Data Population Rules for Populating TRx Data

This section provides information about various columns in the W_SYND_DATA_F table related to populating TRx data.

- INTEGRATION_ID
  - Data type is varchar(30).
  - Value can be any unique number or combination of values coming in from the external file source. If the data is not already unique, the EXPTRANS transformation will manipulate the value to make it unique.
  - Typically, contains the row ID of the source table record in the Siebel transactional database.
- DATASOURCE_NUM_ID
  - ETL run specific (a numeric value that indicates the source for the data).
  - Used in standard Siebel mappings to ensure each record carries a value indicating its source.
  - Used in source qualifiers to ensure that lookups on dimension tables when populating fact tables are referencing data that was loaded from a uniform data source.
  - Data source number 1 is reserved for the Siebel transactional database. The external source should use a number other than 1.
  - Combination of INTEGRATION_ID and DATASOURCE_NUM. Must be a unique value (mandated by a unique index on both columns).

    This means that the system will not accept the same record from the same source with identical unique IDs.
- DATA_SRC_CD

  Standard Siebel mapping transforms an incoming character data source code value into a numeric range between 1 and 8, using the following formula:

  ```
  DECODE(DATA_SRC_CD

      RXPrf',1,
      RXEVM',0,
      RXSMI',3,
      RXSMM',4,
      RXXPT',5,
      RXPT',6,
      RXZip',7,
      RXBrk',8
  ```

  The resulting numeric value is used to populate the DATA_TYPE column in the fact table, which is used in the Analytics business models to logically partition data when running different types of reports against the W_SYND_DATA table.
- PREFIX

  The following transformation is performed on DATA_SRC_CD to populate the PREFIX column:

  ```
  DECODE(DATA_SRC_CD,
  ```

```
RXPrf','Prf',
RXEVM','EVM',
RXSMI','SMI',
RXSMM','SMM',
RXXPT','XPT',
RXPT','PT',
RXZip','Zip',
RXBrk','Brk')
```

The resulting three-character code is used to prefix the ROW_WID of the W_SYND_DATA table. Instead of 1, the first record is populated with Prf1 when loading prescriptions by professional. This is done in order to facilitate parallel load streams in Informatica so that ROW_WIDs of records for different data types do not conflict with each other.

- PERIOD_TYPE_WID
  - Stores a foreign key in the W_LOV_D table for the record corresponding to the period for the data.
  - PERIOD_CD comes from the S_SYND_DATA as Month or Week, and so on.
  - The standard mapping uses the code to do a lookup against the W_LOV_D table to secure the ROW_WID of that period type and inserts it into the W_SYND_DATA_F table PERIOD_TYPE_WID column.
- OWNER_POSTN_WID
  - W_SYND_DATA maps each record to a Siebel position using OWNER_POSTN_ID. Typically, when the source of the ETL is S_SYND_DATA in the transactional database, the field is already populated, but this is not the case when the source is the syndicated data file.
  - The W_ALIGNMT_DH table in the data warehouse contains information that maps Siebel positions to zip codes, bricks, and accounts. ALIGN_ITEM_TYPE contains literal type values for this data, such as zip code, brick or account.
  - Since the alignment process could assign a territory to multiple zip codes, the W_ALIGNMT_DH table could contain multiple records for the same zip code to position and territory combinations. In such cases, the ALIGN_TYPE_FLG column will designate whether the relationship between the territory and the zip code is primary (P), mirror (M), share (S), and so on.
  - The Source qualifier in the custom mapping that populates W_SYND_DATA_F then retrieves the correct data warehouse position ID based on the following join criteria:

```
SELECT
SYND.IMS_NUM,
SYND.CLIENT_NUM,
SYND.START_DT,
SYND.IMS_SPEC_CD,
SYND.PROD_GRP,
SYND.ME_NUM,
SYND.PR_CITY,
SYND.PR_COUNTRY,
SYND.ZIPCODE,
ALIGN.LVL1ANC_POSTN_ID as OWNER_POSTN_ID,
PER.ROW_WID as MAP_CON_ROW_WID,
```

```
PROD_MKT.ROW_WID as MAP_MKT_ROW_WID,
PROD.ROW_WID as PROD_ROW_WID,
PROD.RX_AVG_PRICE,
POSTN.ROW_WID as POSTN_ROW_WID,
POSTN.INTEGRATION_ID as OWNER_POSTN_ID,
SYND.TRX01,
SYND.TRX02,
SYND.TRX03,
SYND.TRX04,
SYND.TRX05,
SYND.TRX06,
SYND.TRX07,
SYND.TRX08,
SYND.TRX09,
SYND.TRX10,
SYND.TRX11,
SYND.TRX12,
SYND.TRX13,
ALIGN.ALIGN_WID,
ALIGN.ALIGN_TYPE_FLG,
PER.INTEGRATION_ID,
PROD_MKT.PROD_NAME,
PROD.PROD_NAME
from WC_SYND_DATA_FS as SYND,

W_ALIGNMT_DH as ALIGN,
W_PERSON_D as PER,
W_PRODUCT_D as PROD_MKT,
W_PRODUCT_D as PROD,
W_POSITION_D as POST

where
(SYND.ZIPCODE = ALIGN.ALIGN_ITEM and ALIGN.ALIGN_ITEM_TYPE = 'Zipcode' and
ALIGN.ALIGN_TYPE_FLG = 'P') and
SYND.ME_NUM = PER.CSN and
SYND.PHYNTYPE = PROD_MKT.PROD_NAME and
SYND.PROD_GRP = PROD.PROD_NAME and
ALIGN.LVL1ANC_POSTN_ID = POSTN.INTEGRATION_ID
```

- GEO_WID
  - In the Siebel Data Warehouse, the GEO_WID is a foreign key to the W_GEO_D dimension, which contains a geographical hierarchy based on zip code as the lowest level of detail. It is preconfigured to allow users to roll up a TRx measure to levels such as city, state or province, county, country, and continent.
  - In the standard Siebel mappings that populate W_SYND_DATA_F, the following three attributes are brought directly from the S_SYND_DATA table: city, country, and zip code.
  - A complex lookup is then performed utilizing two possible logic paths:

- MPLT_GEO_WID_ZIPCODE (returns IN_ZIPCODE_GEO_WID): Takes in the zip code value, and a literal NO_DUP_FLG = 'N' ports do a lookup against W_GEO_D for any record where the ZIPCODE column matches the incoming zip code value and the DUP_ZIPCODE column = 'N'. (Sample data in the W_GEO_D table contains multiple city records per zip code, only one of which is marked as DUP_ZIPCODE = 'N'). The mapping also ensures that even with multiple matches only the first record is retrieved, and with no matches, the data comes back with the appropriate Unspecified code.
- EXP_CITY_COUNTRY_ZIP_CHKNULL & MPLT_GEO_WID1 (returns IN_GEO_WID): Uses the combination of city, country, and zip code to retrieve the first available record from W_GEO_D even if duplicates exists and regardless of the NO_DUP_FLG.

■ Once the two values are brought into the EXPTRANS transformation, the following formula is used:

```
IIF(DATA_SRC_CD = 'RXZip' OR DATA_SRC_CD = 'RXBrk' OR DATA_SRC_CD = 'SlsDirBrk'
OR DATA_SRC_CD = 'SlsDirZip'   OR DATA_SRC_CD = 'SlsIndBrk' OR DATA_SRC_CD =
'SlsIndZip',IN_ZIPCODE_GEO_WID,IN_GEO_WID)
```

Life Sciences Data Loading Issues with Siebel Data Warehouse

This issue is specific to Analytics for Life Sciences and does not affect other products.

Siebel Data Warehouse for Life Sciences updates the Siebel Data Warehouse with a full refresh or using incremental updates. In Informatica, the Full_Load_Siebel_DW workflow does a complete refresh and is usually only used for the initial build. Refresh_Siebel_DW incrementally updates the Siebel Data Warehouse.

Running Full_Load_Siebel_DW deletes all existing information stored in the fact and dimension tables. The following types of incremental updates rules are supported on the fact tables.

- **Activity Product Fact.** Add new records (call activity) to the fact table.
- **Rx Sales (syndicated data) Fact.** ETL does not support incremental updates. When running Full_Load_Siebel_DW, all records in the fact and dimension tables are deleted. To maintain a history in the dimension tables (such as multiple alignments), use Refresh_Siebel_DW. If you need to incrementally update the RxSales fact table for incremental syndicated data loading, use one of the following strategies:
  - **For incremental insert.** Load the data directly into staging and modify the session not to truncate the fact table. Then use the existing ETL to load it to the fact table.
  - **For incremental update.** Create a new mapping that does a lookup against the fact table and run the update.
- **Objective.** Add new records as per the new ePharma Objective. Update existing records with a changed ePharma Objective to the fact table.
- **Profile Rank.** Add new records to the fact table and update existing records in the fact table.

- **Med Ed.** Add new MedEd events to the fact table. A MedEd event is assumed to be locked in the Siebel ePharma MedEd. If the information is changed in an existing MedEd event, the ETL creates a new record in the fact table. If an existing MedEd event is changed, the ETL process does not load the data. The ETL always skips already loaded MedEd events.

Known Issues with the RxSales Fact Table

The following are known issues with creation of aggregate measure columns in the RxSales fact table.

- With large volumes of syndicated data, the creation of aggregate measures in the Siebel Data Warehouse can take four times the amount of time needed to load the fact table. This may be unacceptable under circumstances when there are large amounts of syndicated data.
- All aggregate measures have been recreated at the metadata level except for MAT. MAT can be entered as a formula for ad-hoc analysis in Siebel ePharma Answers using the function msum(measure, window). For more information, see *Siebel Analytics User Guide*.
- The LS syndicated fact table contains following aggregate measure columns to speed enterprise wide reporting: TTD, TTDLY, TTDPT, MAT, YTD. To update these aggregate columns ETL process requires following steps:

Upgrading aggregate columns

1. Join the syndicated data to the day dimension to denormalize by time.
2. Create a second set of time denormalized syndicated data.
3. Join these two data sets to obtain appropriate measure for TTD, TTDLY, TTDPT, MAT, and YTD aggregation.
4. Aggregate the measures.
5. Update the syndicated data aggregate columns.

Incremental Updates in the Siebel Data Warehouse LS Dimension Tables

This issue is specific to Analytics for Life Sciences and does not affect other products. In the LS dimension tables, the following incremental updates are supported. Some dimensions have more than one set of attributes.

W_ALIGNMT_DH

Slowly Changing Dimension Type. 2

Attribute. Link between ZIP, Brick, Account, and Contact Hierarchy

Description. Tracks historical alignments.

- If ETL Alignment Version is set to N, ETL makes changes to existing alignment if there is a change on an assignment rule.

- If ETL Alignment Version is set to Y, it creates a new alignment version.

  The dimension has these characteristics:

  - The first alignment after Full load is 1.
  - A new version is created when a new assignment rule (new position or position relationship) or an assignment criteria is modified (change in postal code, brick, contact, or account) if ETL Alignment Version is set to Y.
  - Activation date changes have to be entered manually. The ETL looks for the change and creates a new version, even though the rules have not been applied to the Siebel transactional database.
  - Assignment criteria:
    - Contact ZIP Code or Account ZIP Code cannot use ZIP Code ranges. Each ZIP Code assigned to a territory needs to be on a separate row, so the same value needs to be entered for both ZIP Code Low and ZIP Code High.
      For example, if ZIP Code 09654 is assigned to a territory, the value for both ZIP Code High and ZIP Code Low should be 09654. Also, the same ZIP Code should not be assigned twice to the same territory and a ZIP Code should be unique to a territory when using assignment criteria for Contact ZIP Code or Account ZIP Code.
    - Contact Brick or Account Brick require unique bricks assigned to a territory.
  - Every new version increases the counter by one and is a sequential number. When a dimension table gets too large, your administrator can create an SQL query to delete all attributes for version numbers.
  - Effective date of the alignment is assignment activation date.

W_ORG_D

Slowly Changing Dimension Type. 3

Attributes. ACCNT_TYPE_CD, ACCNT_TYPE_CD1, ACCNT_TYPE_CD2, ACCNT_TYPE_CD3

Description. Tracks three past account types. If the account type changes a fourth time, the first change is deleted and only the past three changes are maintained. Effective date is ETL run date to each group of attributes.

Attributes. NAME, NAME1, NAME2, NAME3

Description. Tracks three past account names. If the account name changes a fourth time, the first change is deleted and only the past three changes are maintained. Effective date is ETL run date.

Attributes. NUMB_OF_BEDS, NUMB_OF_BEDS1, NUMB_OF_BEDS2, NUMB_OF_BEDS3

Description. For an account type of hospital and clinic, tracks three past quantities. If the number of beds changes a fourth time, the first change is deleted and only the past three changes are maintained. Effective date is ETL run date.

Attributes. PAR_INTEGRATION_ID, PAR_ITGR_ID1, PAR_ITGR_ID2, PAR_ITGR_ID3

Description. Tracks three past parent accounts. If the parent account changes a fourth time, the first change is deleted and only the past three changes are maintained. Effective date is ETL run date.

W_POSTN_CON_D

Slowly Changing Dimension Type. 3

Attributes. STATUS, STATUS1, STATUS2, STATUS3

Description. Tracks three past contact statuses. If the contact status changes a fourth time, the first change is deleted and only the past three changes are maintained. Effective date is ETL run date.

W_POSITION_D

Slowly Changing Dimension Type. 3

Attributes. EMP_FST_NAME, EMP_MID_NAME, EMP_LAST_NAME, EMP_FST_NAME_H1, EMP_MID_NAME_H1, EMP_LAST_NAME_H1, EMP_FST_NAME_H2, EMP_MID_NAME_H2, EMP_LAST_NAME_H2, EMP_FST_NAME_H3, EMP_MID_NAME_H3, EMP_LAST_NAME_H3

Description. Tracks three past employee names assigned to a position. If the employee name changes a fourth time, the first change is deleted and only the past three changes are maintained. Effective date is ETL run date.

W_PROD_RANK_D

Slowly Changing Dimension Type. 3

Attributes. RANKING, RANKING1, RANKING2, RANKING3

Description. Keep three previous rankings to allow restatement. Effective date is ETL run date.

Attributes. RATING, RATING1, RATING2, RATING3

Description. Keep three previous ratings to allow restatement. Effective date is ETL run date.

W_PRODUCT_D

Slowly Changing Dimension Type. 3

Attributes. VENDOR_LOC, VENDOR_LOC1, VENDOR_LOC2, VENDOR_LOC3

Description. Tracks three past vendor locations. If the vendor location changes a fourth time, the first change is deleted and only the past three changes are maintained. Effective date is ETL run date.

Index

J

L

M

O

P

Q

R

S

T

U

W