Oracle® Migration Workbench

Reference Guide for IBM DB2/400 V4R5 Migrations

Release 9.2.0 for Microsoft Windows 98/2000 and Microsoft Windows NT

March 2002

Part Number: A97252-01

This guide describes how to migrate from IBM DB2/400 V4R5 to Oracle9*i*.

ORACLE®

Oracle Migration Workbench Reference Guide for IBM DB2/400 V4R5 Migrations, Release 9.2.0 for Microsoft Windows 98/2000 and Microsoft Windows NT

Part Number: A97252-01

Contents

Send Us Your Comments

Oracle Migration Workbench Reference Guide for IBM DB2/400 V4R5 Migrations, Release 9.2.0 Microsoft Windows 98/2000 and Microsoft Windows NT

Part Number: A97252-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: gpe_techpubs_ie@ORACLE.COM
- Tel: +353-1-8031000, Fax: +353-1- 8033321
- Attn: Oracle Migration Workbench
- Postal service:
  Oracle Migration Workbench Documentation
  Oracle Corporation
  East Point Business Park
  Clontarf, Dublin 3
  Ireland

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

This guide provides detailed information about migrating a database from IBM DB2/400 V4R5 to Oracle8, Oracle8*i*, or Oracle8*i* Appliance. It describes several differences between IBM DB2/400 V4R5 and Oracle and describes how those differences are handled by the Migration Workbench during the conversion process.

It also provides guidelines on how to modify IBM DB2/400 V4R5 applications to work with the new Oracle database. If you have an investment in IBM DB2/400 V4R5 applications, you can retain this investment while adding the advanced features of Oracle to the application architecture.

This preface contains the following sections:

- Audience
- What You Should Already Know
- How This Reference Guide is Organized
- How to Use This Reference Guide
- Related Documentation
- Conventions

Audience

This guide is intended for anyone using the Migration Workbench to convert an IBM DB2/400 V4R5 database to Oracle.

What You Should Already Know

You should be familiar with relational database concepts and with the operating system environments under which you are running Oracle and IBM DB2/400 V4R5.

How This Reference Guide is Organized

This reference guide is organized as follows:

- Chapter 1, "Overview"

  Introduces the Migration Workbench and describes the features of this tool.

- Chapter 2, "Oracle and IBM DB2/400 V4R5 Compared"

  Contains detailed information about the differences between data types, data storage concepts, and schema objects in IBM DB2/400 V4R5 and Oracle.

- Chapter 3, "Migration Process"

  Describes the architecture of IBM DB2/400 V4R5 and Oracle. It explains how to prepare the IBM DB2/400 V4R5 database for migration. This chapter also describes how to migrate from IBM DB2/400 V4R5 to Oracle using the Migration Workbench.

- Chapter 4, "Troubleshooting"

  Describes how to solve problems you might encounter when using the Migration Workbench.

How to Use This Reference Guide

You must read Chapter 1, "Overview", which provides an introduction to the concepts and terminology of the Migration Workbench.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at

```
http://www.oracle.com/accessibility/
```

Accessibility of Code Examples in Documentation

JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Related Documentation

For more information, see these Oracle Migration Workbench documents:

- *Oracle Migration Workbench Frequently Asked Questions (FAQ)*
- *Oracle Migration Workbench Release Notes*

To download release notes, installation documentation, white papers, or other collateral, visit the Oracle Technology Network (OTN). You must register online before using OTN. You can register at:

```
http://otn.oracle.com/membership/index.htm
```

If you already have a user name and password for OTN, then you can go directly to the Migration Workbench documentation section of the OTN Web site at:

```
http://otn.oracle.com/tech/migration/workbench
```

Conventions

This section describes the various conventions used in the text and code examples in this documentation. It describes:

- Conventions in Text
- Conventions in Code Examples

Conventions in Text

This guide uses various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use:

Convention	Meaning	Example
Bold	Bold type indicates GUI options. It also indicates terms that are defined in the text, terms that appear in the glossary, or both.	The C datatypes such as **ub4**, **sword**, or **OCINumber** are valid. When you specify this clause, you create an **index-organized table**.
Italics	Italic type indicates book titles, emphasis, syntax clauses, or placeholders.	*Reference Guide* Run `U`*`old_release`*`.SQL` where *old_release* refers to the release you installed before upgrading.
`UPPERCASE` (monspace)	Uppercase monospace type indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, user names, and roles.	You can specify this clause only for a `NUMBER` column. You can back up the database using the `BACKUP` command.
`lowercase` (monspace)	Lowercase monospace type indicates executables and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, user names and roles, program units, and parameter values.	Enter `sqlplus` to start SQL*Plus. The `department_id`, `department_name`, and `location_id` columns are in the `hr.departments` table.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace font and are separated from normal text as shown in this example:

```
SELECT * FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use:

Convention	Meaning	Example
Square Brackets [ ]	Indicates that the enclosed arguments are optional. Do not enter the brackets.	`DECIMAL (digits [ , precision ])`
Curly Braces { }	Indicates that one of the enclosed arguments is required. Do not enter the braces.	`{ENABLE | DISABLE}`

Vertical Line |	Separates alternative items that may be optional or required. Do not type the vertical bar.	`{ENABLE | DISABLE}`
		`[COMPRESS | NOCOMPRESS]`

`Ellipses ...`	Indicates that the preceding item can be repeated. You can enter an arbitrary number of similar items. In code fragments, an ellipsis means that code not relevant to the discussion has been omitted. Do not type the ellipsis.	`CREATE TABLE ... AS subquery;`
		
		`SELECT col1, col2, ... , col`*`n`*` FROM employees;`

`Italics`	Indicates variables that you must supply particular values.	`CONNECT SYSTEM/`*`system_password`*

`UPPERCASE`	Indicates case-insensitive filenames or directory names, commands, command keywords, initialization parameters, data types, table names, or object names. Enter text exactly as spelled; it need not be in uppercase.	`SELECT last_name, employee_id FROM employees;`
		`SELECT * FROM USER_TABLES;`
		`DROP TABLE hr.employees;`

`lowercase`	Indicates words supplied only for the context of the example. For example, lowercase words may indicate the name of a table, column, or file.	`SELECT last_name, employee_id FROM employees;`
		`sqlplus hr/hr`

1 Overview

This chapter introduces the Oracle Migration Workbench (Migration Workbench). It includes information on the following:

- Introduction to the Migration Workbench
- Introduction to IBM DB2/400 V4R5

Introduction to the Migration Workbench

The Migration Workbench is a tool that simplifies the migration of data and applications from an IBM DB2/400 V4R5 environment to an Oracle8, Oracle8*i*, or Oracle Appliance destination database. The Oracle RDBMS is a modern, scalable, high performance database server that can run on a wide range of computers from PCs to mainframes. Oracle operates in a networked, client/server environment. It can support tens, hundreds, or thousands of users, depending on the server.

The Migration Workbench allows you to migrate an entire application system and database schema in an integrated, visual environment.

The Migration Workbench provides an intuitive and informative user interface and a series of wizards to simplify the migration process. To ensure portability, all components of the Migration Workbench are written in Java.

The Migration Workbench uses a repository to store migration information. The repository allows you to query the initial state of the application before migration. By initially loading the components of the application system that you want to migrate into a repository, you can work independently of the production application.

Furthermore, the Migration Workbench uses a repository to store migration information about the components you are converting. You can use this information to understand the effect of modifying a given table.

Using the Migration Workbench you can:

- Make a complete data migration from IBM DB2/400 V4R5 to Oracle8 or Oracle8*i*
- Migrate users, tables, table aliases, primary keys, foreign keys, unique constraints, indexes, views, and table-level privileges to Oracle8 or Oracle8*i*
- Generate the Oracle SQL*Loader and IBM DB2/400 V4R5 flat files for offline data loading
- Display a representation of the source database and its Oracle equivalent
- Generate a summary report of the migration
- Customize users, tables, indexes, and tablespaces
- Customize the default data type mapping rules
- Create ANSI-compliant names
- Automatically resolve conflicts, such as Oracle reserved words
- Remove and rename objects in the Oracle Model

Introduction to IBM DB2/400 V4R5

IBM DB2/400 V4R5 is a relational database management system (RDBMS). IBM offers DB2 products on a number of platforms, including OS/390, AS/400, and UNIX-based systems. The Migration Workbench DB2/400 plug-in is a migration tool for Version 4 Release 3 of DB2 on the AS/400 system, (IBM DB2/400 V4R5), which is no longer supported by IBM.

The IBM DB2/400 V4R5 database is integrated with the AS/400 operating system, OS/400. It includes the following features:

- Referential Integrity
- Triggers
- Stored Procedures
- Two-Phase Commitment Control
- Distributed Relational Database Access (DRDA) Level 2

The database uses SQL/400 as its native interface. SQL/400 includes non-SQL standard facilities such as Command Language (CL) and Data Description

Specifications (DDS). These facilities originated with the IBM System/38 and existed before SQL became standard.

2

Oracle and IBM DB2/400 V4R5 Compared

This chapter describes Oracle and IBM DB2/400 V4R5 database concepts, including similarities and differences between the two database. It includes information on the following:

- Data Storage Concepts
- Schema Migration
- Data Types
- Database Security

Data Storage Concepts

This section provides an overview of the data storage concepts and methods used by IBM DB2/400 V4R5, and the similarities or differences between these methods and concepts and those used by Oracle.

Oracle Data Storage Concepts

An Oracle database consists of one or more tablespaces. Tablespaces provide logical storage space that links a database to the physical disks which hold the data. A tablespace is created from one or more datafiles. Datafiles are files in the file system or an area of disk space specified by a device. A tablespace can be enlarged by adding more datafiles.

A basic Oracle database consists of a SYSTEM tablespace, where the Oracle data dictionary tables are stored. It can also consist of user-defined tablespaces. A tablespace is the logical storage location for database objects. For example, you can specify where a particular table or index is created in the tablespace. The size of a tablespace is determined by the amount of diskspace allocated to it. Each tablespace is made up of one or more datafiles.

IBM DB2/400 V4R5 Data Storage Concepts

IBM DB2/400 V4R5 is an integrated part of OS/400, the AS/400 operating system. Everything in the OS/400 operating system, including the database, is organized as objects. OS/400 contains over 80 types of objects, including programs, database files, and user profiles. OS/400 stores program instructions, application data, and other system components on disk and loads them into main memory as required. However, OS/400 does not permit direct access to the memory on disk or to memory directly. Instead, you must always use specific commands or system interfaces that are valid for each type of object.

The IBM DB2/400 V4R5 Library

OS/400 controls how you use an object by storing some descriptive information with the content of the object. All objects have a header, which consists of two parts, a standard part and a type-specific part. The type-specific part specifies the object. For example: `*PGM` equals program and `*FILE` equals file type. The standard part of the header contains the following information:

- The library that contains the object
- The object's name
- The object's type and subtype
- The user profile that owns the object

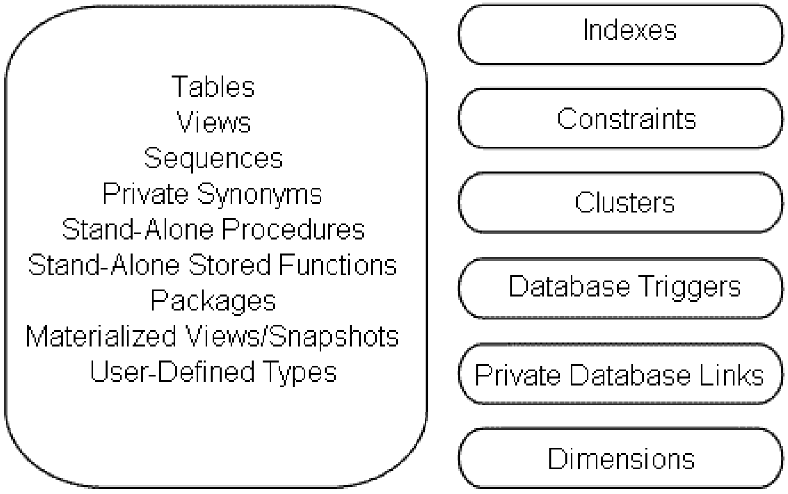
An IBM DB2/400 V4R5 library is an object that contains other objects, but a library cannot contain another library object. As the containment properties are similar, the Migration Workbench maps library names to Oracle tablespaces.

Object Name IBM DB2/400 V4R5 object names can be up to 10 alphanumeric characters in length, beginning with a letter or a national character. Libraries also have names, for example the name, `custdata` might be used for a library that contains data relating to a company's customers. An object's qualified name is the combination of the name of the library that contains it, and the object's unqualified name, separated by a forward slash, for example `custdata/abcproductions`.

Object Type and Subtype Each object type in AS/400 is designated by a special value, for example, `*PGM` for program and `*FILE` for file. Some object types also have subtypes. The `*FILE` object includes physical files, logical files, printer files, display files, and communication files. An OS/400 object is uniquely identified by the combination of its qualified name and its object type, which means OS/400 allows only one object on an AS/400 database that has a given combination of library, name and object type. You cannot have two files with the same name in the same library, however, you can have two different object types of the same name in the same library.

This concept of unique names for objects is similar to the Oracle criteria that no two objects in a namespace can have the same name. Figure 2–1 shows the namespace for schema objects in Oracle. The tables and views are in the same namespace within a schema and therefore cannot have the same name, whereas tables and indexes which are not in the same namespace can have the same name within a schema.

Figure 2–1 Namespace for Schema Objects in Oracle



During the capture phase of the migration, if the Migration Workbench finds object names that do not conform to the Oracle schema-naming convention, it substitutes the original schema names with names allowed by Oracle by introducing an underscore to the original name. For example, as Oracle does not allow whitespaces in schema names, the Workbench would convert COLUMN NAME to COLUMN_ NAME. You can see this in the Migration Workbench after the Capture phase, by comparing schema names in the Source Model with the schema names in the Oracle Model.

User Profiles Each OS/400 object is owned by a user profile. A user profile is another type of OS/400 object. Each user profile stores information about a system user, including the user name, password, and authority to access data or use system functions. Whenever you log in, you supply a user profile name and password, allowing AS/400 to control your use of the system, including access to the database. This security system is similar to the Oracle security system, where each user has a password that allows access to the database. Each user also has a set of privileges for objects in the database. In Oracle, the creator or owner of an object is automatically given full access rights to that object. The owner of an object can, in turn, grant privileges such as select and update for this object to other users. In Oracle, the database catalogs store the privilege information for each user.

IBM DB2/400 V4R5 File Types

There are two types of IBM DB2/400 V4R5 files, physical files and logical files.

Physical Files

Application data is stored in physical files. Each record, or entry in a database file, occupies a unique location in a physical file. The records are not necessarily in any order based on their content. A record's location is identified by its relative record number (RRN), which starts at 1 for the first record in the member and increases by 1 for each location. When you delete a record IBM DB2/400 V4R5 sets an internal deleted record flag in the record's location. When you insert a new record, IBM DB2/400 V4R5 either puts it in the first available location with a deleted record flag, or after the last record in the file.

A physical file always has just one record format, and all records in the same physical file have the same record layout.

Logical Files

Logical files provide an alternative way to access data in one or more physical files. You can use a logical file to:

- Select a subset of the records in a physical file
- Merge the records from multiple physical file members
- Select a subset of the fields in a physical files record format
- Combine related records in two or more physical files

- Provide an index so records can be retrieved in particular order
- Perform a Backup and Recovery operation

IBM DB2/400 V4R5 Backup Facility

The IBM DB2/400 V4R5 system has backup facilities for everything on the system. The following table shows the save and restore commands for database files and associated objects such as journals:

IBM DB2/400 V4R5 Command	Description
`SAVLIB`	Saves one or more libraries and all objects in them
`SAVOBJ`	Saves one or more specific objects in one or more libraries
`SAVCHGOBJ`	Saves one or more objects that have been changed since the last time they were saved by a `SAVLIB` command or since a specified date and time
`RSTLIB`	Restores database library
`RSTOBJ`	Restores database objects

In Oracle, each database has a set of two or more redo log files. All changes made to a database are recorded in these files. You can use redo log files to restore corrupt Oracle databases. Oracle allows mirrored redo log files, allowing you to maintain two or more copies of these files. By using mirrored redo log files you can protect them from loss, for example, from a hardware failure.

Schema Migration

The schema contains the definitions of the tables, views, indexes, and other database-specific objects.

This section includes information on the following:

- Schema Object Similarities
- Schema Object Names
- Table Design Considerations
- Schema Migration Limitations for IBM DB2/400 V4R5

Schema Object Similarities

There are many similarities between schema objects in Oracle and IBM DB2/400 V4R5. However, some schema objects differ between these databases. For specific information about schema objects within Oracle8*i*, refer to the SQL Statements topic in the *Oracle8i SQL Reference*. Table 2–1 shows the similarities and differences between the schema objects in the two databases.

Table 2–1 Schema Objects in Oracle and IBM DB2/400 V4R5

Oracle	**IBM DB2/400 V4R5**
Database	Database
Tablespace	Collection
User	User Profile
Role	Group Profile/Authorization list (not migrated)
Table	Table
Temporary Table	Not applicable
Index	Index
Check constraint	Check constraint (not migrated)
Column default	Column default
Unique key	Unique key
Primary key	Primary key
Foreign key	Foreign key
PL/SQL procedure	External procedure/CL procedure (not migrated)
PL/SQL function	SQL/400 function (not migrated)
Packages	Not applicable
Triggers	Triggers (not migrated)
Private synonyms	Table alias
Sequences	Not applicable
Snapshot	Not applicable
View	View (migrated but not parsed)

Schema Object Names

Reserved words differ between Oracle and IBM DB2/400 V4R5. Some Oracle reserved words are valid object or column names in IBM DB2/400 V4R5. Use of reserved words as schema object names makes it impossible to use the same names across databases. The Migration Workbench appends an underscore (_) to the end of the name of an IBM DB2/400 V4R5 object that is an Oracle reserved word.

Object names are not case sensitive in Oracle or IBM DB2/400 V4R5. For a list of Oracle reserved words, refer to the *Oracle8i SQL Reference*.

Table Design Considerations

This section describes table design issues that you must consider when converting IBM DB2/400 V4R5 databases to Oracle. It contains information on the following:

- Referential Integrity Constraints
- Collection Mappings
- Alias Mappings

Referential Integrity Constraints

A referential integrity constraint is the rule that governs the relationship between columns in different tables. Integrity constraints are very similar in the Oracle and IBM DB2/400 V4R5 databases, but there are some differences. Both Oracle and IBM DB2/400 V4R5 use the `ON DELETE` clause in referential integrity constraints, but, in addition, IBM also uses the `ON UPDATE` clause.

The `ON UPDATE` clause determines what action is taken if an insertion in the designating table creates an unmatched, non-null foreign key value. There is no direct mapping of the IBM DB2/400 V4R5 `ON UPDATE` clause to Oracle, so this clause is not migrated to Oracle.

The `ON DELETE` clause determines what action is taken if a primary key value in the target table is deleted. The deletion of a primary key value could potentially leave orphan rows in the designating table that no longer reference an existing row in the target table. There are a number of possible `On Delete` clauses in IBM DB2/400 V4R5. However, only the `ON DELETE CASCADE` clause is migrated to from IBM DB2/400 V4R5 to Oracle

Note: The `CASCADE` action causes IBM DB2/400 V4R5 to propagate the `On Delete` operation to all dependent rows. The `CASCADE` action is available in both the Oracle and IBM DB2/400 V4R5 databases.

Collection Mappings

A library is an OS/400 object that contains other objects, such as files and programs. In SQL/400 terminology, this container is called a collection. The Oracle equivalent of an IBM DB2/400 V4R5 collection is a tablespace. An Oracle tablespace is used to group related logical structures together, so collection names in IBM DB2/400 V4R5 are mapped to tablespaces in Oracle.

Alias Mappings

An alias in IBM DB2/400 V4R5 is an alternative name for a table or view. An alias can be created or dropped. No authority is required to use an alias. However, access to the tables and views referred to by an alias still require the appropriate authorization. An alias can be up to 30 characters long.

The Oracle equivalent of an IBM DB2/400 V4R5 alias is a synonym. A synonym is an alternative name for any table, view, snapshot, sequence, procedure, function, or package. As it is only an alternative name, it does not require storage, other than the storage of its definition in the data dictionary.

You can create both public and private synonyms in Oracle. A public synonym is owned by the special user group named `PUBLIC` and every user in a database can access it. A private synonym is owned by a specific user who has control over its availability to others. Table aliases in IBM DB2/400 V4R5 are mapped to private synonyms in Oracle.

Schema Migration Limitations for IBM DB2/400 V4R5

This section describes IBM DB2/400 V4R5 schema migration limitations. It contains information on the following:

- Parsable Objects
- Privileges

Parsable Objects

Except for view, the migration of parsable objects such as stored procedures, triggers and check constraints is not supported by the Migration Workbench. However, in IBM DB2/400 V4R5, views are defined using SQL/400, which does not map directly to PL/SQL. When the SQL text for IBM DB2/400 V4R5 views is migrated to Oracle, this SQL text is not converted to PL/SQL.

Note: In many cases, the SQL/400 text should conform to the SQL standard and should compile in the PL/SQL environment, however this is not guaranteed.

Privileges

The Migration Workbench migrates all user names, including their table-level privileges. The access privilege system in IBM DB2/400 V4R5 is integrated into the OS/400 operating system, and is therefore not available in IBM DB2/400 V4R5 catalogs. You can use special add-in programs to determine the users of the system and their table-level privileges. These add-in programs are installed in the `$ORACLE_HOME/Omwb/addins/as400/v4r3` directory. The Migration Workbench automatically transfers these programs to the source AS/400 system during the migration using File Transfer Protocol (FTP). They are then executed on the AS/400 system. The Migration Workbench does not support the privileges defined in IBM DB2/400 V4R5 Groups or Authorization lists.

Data Types

This section describes the differences between the data types used in IBM DB2/400 V4R5 and Oracle databases. Specifically, this section contains information on the following:

- Data Types in Oracle and IBM DB2/400 V4R5
- Character Strings
- Graphic Strings
- Numeric Types
- Date Data Types

Data Types in Oracle and IBM DB2/400 V4R5

Table 2–2 shows the data types available in IBM DB2/400 V4R5 and their Oracle equivalents.

Table 2–2 Data Types in Oracle and IBM DB2/400 V4R3

DB2400 V4R3 Data type	Description	Oracle Data type
CHAR*(length)*	Fixed-length character string with a length between 1 and 32766 characters (if the length is omitted, it defaults to 1)	CHAR
CHARACTER*(length)*	Fixed-length character string with a length between 1 and 32766 characters (if the length is omitted, it defaults to 1)	CHAR
CHAR FOR BIT DATA*(length)*	Fixed-length character string with a length between 1 and 32766 characters (if the length is omitted, it defaults to 1) To be treated as binary data.	CHAR
CHARACTER FOR BIT DATA*(length)*	Fixed-length character string with a length between 1 and 32766 characters (if the length is omitted, it defaults to 1) To be treated as binary data.	CHAR
VARCHAR*(length)*	Variable-length character string with a maximum length of 4000 bytes	VARCHAR2
CHAR VARYING*(length)*	Variable-length character string with a maximum length of 4000 bytes	VARCHAR2
CHARACTER VARYING*(length)*	Variable-length character string with a maximum length of 4000 bytes	VARCHAR2
VARCHAR FOR BIT DATA*(length)*	Variable-length character string with a maximum length of 4000 bytes. To be treated as binary data.	VARCHAR2
CHAR VARYING FOR BIT DATA*(length)*	Variable-length character string with a maximum length of 4000 bytes. To be treated as binary data.	VARCHAR2

Table 2–2 Data Types in Oracle and IBM DB2/400 V4R3(Cont.)

DB2400 V4R3 Data type	Description	Oracle Data type
CHARACTER VARYING FOR BIT DATA*(length)*	Variable-length character string with a maximum length of 4000 bytes. To be treated as binary data.	VARCHAR2
LONG VARCHAR*(length)*	Variable-length character string with a maximum length of 32700 bytes	VARCHAR2
GRAPHIC*(length)*	Fixed-length graphic string with a length between 1 and 127 double-byte characters	VARCHAR2
VARGRAPHIC*(length)*	Variable-length graphic string with a maximum length of 2,000 double-byte characters	VARCHAR2
LONG VARGRAPHIC*(length)*	Variable-length graphic string with a maximum length of 16,350 double-byte characters.	CLOB
BLOB	Variable-length binary large object string that can be up to 2GB long.	BLOB
CLOB	Variable-length character large object string that can be up to 2GB long. A CLOB can store single-byte character strings or multibyte, character-based data.	CLOB
	Variable-length, double-byte character string that can store up to 1,073,741,823 characters.	BLOB
DBCLOB	Variable-length, double-byte character string that can store up to 1,073,741,823 characters.	BLOB

IBM DB2/400 V4R5 User-Defined Data Types

IBM DB2/400 V4R5 provides a method for a user to declare specialized usages of datatypes and the rules which apply to them. You can use a user-defined or distinct data type as the data type for any column in the database. Defaults and rules (check constraints) are also bound to these user-defined data types, and are applied automatically to the individual columns of these user-defined data types.

When migrating to Oracle using the Oracle Migration Workbench, the base data type for each user-defined type is determined, and the PL/SQL equivalent of that base type is applied to the column definitions of the target database's tables, in place of the user-defined data types.

Note: Data definition language code and procedural SQL code are less portable across different database servers when user-defined data types are used.

Character Strings

There are several types of character strings. Each string can be categorized further into one of the types shown in the following table:

Type	Description
Bit data	Data not associated with a coded character set and is never converted.
Single byte character set (SBCS) data	Data in which every character is represented by a single character set. Each SBCS data character string has an associated coded character set identifier (CCSID).
Mixed data	Data that can contain a mixture of characters for the SBCS and the double byte character set (DBCS).

The Database Manager in IBM DB2/400 V4R5 recognizes DBCS strings by enclosing them in two EBCIDIC codes:

- `X'0E'` marks the beginning of a sequence of double-byte codes
- `X'0F'` marks the end of a sequence of double byte-codes

The length of a mixed-data character string is its total number of bytes, counting two bytes for each double-byte character and one byte for each single-byte character.

A character string is a sequence of bytes. The length of the string is the number of bytes in the sequence. If the length of the string is zero, then this is called the empty string. This string should not be confused with `NULL`.

Fixed-Length Character Strings

All values of a fixed-length character string column have the same length. This is determined by the length attribute of the column. The length attribute must be between 1 and 32766.

Varying-Length Character Strings

All values of a varying-length character string column have the same maximum length, which is determined by the length attribute of the column. The length attribute must be from 1 and 32740.

Table 2–3 provides definitions of each character string.

Table 2–3 Character String Definitions

String	Definition
CHAR or CHARACTER	A single character.
CHAR FOR BIT DATA or CHARACTER FOR BIT DATA	A single character that is to be treated as binary data.
CHAR(length) or CHARACTER(length)	A fixed-length sequence of characters ranging in length between 1 and 254.
CHAR(length) FOR BIT DATA or CHARACTER(length) FOR BIT DATA	This is a fixed-length sequence of characters that must be treated as binary data ranging in length between 1 and 254.
VARCHAR*(length)* or CHARACTER VARYING*(length)* or CHAR VARYING*(length)*	This is a varying-length sequence of characters ranging in length between 1 and 32,672.
VARCHAR*(length)* FOR BIT DATA or CHARACTER VARYING*(length)* FOR BIT DATA or CHAR VARYING*(length)* FOR BIT DATA	This is a varying length sequence of characters that is to be treated as binary data ranging in length between 1 and 32,672.

Large Object Data Types

Large object data types store data ranging in size from zero bytes to 15 megabytes. There are three large object data types:

Character Large Objects (CLOBs) A character string comprised of single-byte characters with an associated code page. This data type holds text-oriented information, where the amount of information could grow beyond the limits of a regular VARCHAR data type (upper limit of 32K bytes). The Migration Workbench supports code page conversion of the information and compatibility with the other character types.

Double-Byte Character Large Objects (DBCLOBs) A character string comprised of double-byte characters with an associated code page. This data type holds text-oriented information where double-byte character sets are used. This data type has a maximum length of 7864320.

Binary Large Objects (BLOBs) A binary string comprised of bytes with no associated code page.

Graphic Strings

A graphic string is a sequence of two-byte characters. The length of the string is the number of its characters. Like character strings, graphic strings can be empty. All values of a fixed-length graphic string column have the same length, which is determined by the length attribute of the column.

The length attribute must be between 1 and 16383. The values of a column with a varying-length graphic string, such as a VARGRAPHIC or DBCLOB, can have different lengths. The length attribute of the column determines the maximum length that a value can have.

For a VARGRAPHIC column, the length attribute must be between 1 and 16370. For a `DBCLOB` column, the length attribute must be between 1 and 7864320.

Each graphic data type can be further categorized into either DBCS data or UCS-2 data.

DBCS Data A data type where every character is represented by a character from the double-byte character set. Every DBCS graphic has a Coded Character Set Identifier (CCSID) that identifies a double-byte coded character set.

UCS-2 Data A data type where every character is represented by a character from the Universal Coded Character Set (UCS-2).

Table 2–4 provides definitions of each graphic string.

Table 2–4 Graphic String Definitions

String	Definition
GRAPHIC	This is a single graphic character.
GRAPHIC*(length)*	This is a fixed length graphic string of Length graphic characters. Length can range between 1 and 127.
VARGRAPHIC*(length)*	This is a varying length graphic string of up to Length graphic characters. Length can range between 1 and 16,336.

Table 2–4 Graphic String Definitions(Cont.)

String	Definition
LONG VARGRAPHIC	This is a varying length graphic string of up to 16,350 characters.

Numeric Types

All numbers have a sign and a precision. The precision is the total number of binary or decimal digits, excluding the sign. The sign is positive if the value is zero.

Table 2–5 describes the numeric types available in IBM DB2/400 V4R5 and their Oracle equivalents.

Table 2–5 Numeric Types in Oracle and IBM DB2/400 V4R5

DB2400 V4R3 Data type	Description	Oracle Data type
SMALLINT	Two-byte binary integer with a precision of 5 bits. The range of this data type is -32, 768 to +32, 767	NUMBER (6, 0)
INTEGER	Four-byte binary integer. With a precision of 10 digits. The range of this data type is -2, 147, 483, 648 to +2, 147, 483, 647	NUMBER (11, 0)
BIGINT	An eight-byte large binary integer with a precision of 19 digits. The range of this data type is -9, 223, 372, 036, 854, 775, 808 to +9, 223, 372, 036, 854, 775, 807	NUMBER (19, 0)

Table 2–5 Numeric Types in Oracle and IBM DB2/400 V4R5 (Cont.)

DB2400 V4R3 Data type	Description	Oracle Data type
FLOAT*(precision)*	A floating-point number. The precision is the number of digits, which can range between 1 and 53. A precision value between 1 and 24 indicates a single-precision, floating-point number. A precision value between 25 and 53 indicates a double-precision, floating point number. If you do not specify a precision value then this is a double-precision, floating-point number. A single-precision, floating-point number is a 32-bit approximation of a real number with a range of 1.17549436 x 10^{-38} to 3.40282356 x 10^{+38}. A double-precision, floating-point number is a 64-bit approximation of a real number with a range of 2.2250738585072014 x 10^{-308} to 1.7976931348623158 x 10^{308}.	FLOAT
DOUBLE	A double-precision floating-point number, which is a 64-bit approximation of a real number. The range of this data type is 2.2250738585072014 x 10^{-308} to 1.7976931348623158 x 10^{+308}.	FLOAT(53)

Table 2–5 Numeric Types in Oracle and IBM DB2/400 V4R5 (Cont.)

DB2400 V4R3 Data type	Description	Oracle Data type
DECIMAL*(precision, scale)*	A packed-decimal number. The precision is the number of digits and can range between 1 and 31. The scale is the number of digits to the right of the decimal point and can range from 0 to the value specified for precision. You can use DECIMAL<*precision*> for DECIMAL(*precision*, 0). You can also use DECIMAL by itself for DECIMAL(5, 0). The position of the decimal point is determined by the precision and scale (number of digits in the fractional part of the number) of the number. All values of a decimal column have the same precision and scale. The range of a decimal variable or the numbers in a decimal column is -*n* to +*n*, where the absolute value of *n* is the largest number that can be represented with the applicable precision and scale. The maximum range is $-10^{31 + 1}$ to $10^{31 - 1}$.	FLOAT (24)
REAL	A single-precision floating-point number.	FLOAT (24)
NUMERIC*(precision, scale)*	Zoned-decimal number. The precision is the number of digits and can range between 1 and 31. The scale is the number of digits to the right of the decimal point and can range from 0 to the value specified for precision. You can use DECIMAL<*precision*> for DECIMAL(*precision*, 0). You can also use DECIMAL by itself for DECIMAL(5, 0).	NUMBER

Date Data Types

Although date data types are used in certain arithmetic and string operations and they are compatible with certain strings, they are neither strings nor numbers.

Table 2–6 Date Data Types in Oracle and IBM DB2/400 V4R5

DB2400 V4R3 Data type	Description	Oracle Data type
DATE	This type consists of three parts: year, month, and day, and represents a calendar date. Year can range between 0001 and 9999, month can range between 1 and 12, and day can range between 1 and *n*, which can be 28, 29, 30, or 31, depending on the value of month and year.	DATE
TIME	This type is made up of three parts - hour, minute, and second represents a 24-hour clock time value. Hour can range between 0 and 24, minute can range between 0 and 59, and second can range between 0 and 59. Minute and second must be set to 0 if hour is set to 24.	CHAR(8)
TIMESTAMP	This type consists of seven parts: year, month, day, hour, minute, second, and millisecond represents a calendar date and 24-hour clock time value. The ranges of year, month, and day are the same as the date data type, and the ranges for the hour, minute, and second are the same as the time data type. Milliseconds can range between 0 and 999. The TIMESTAMP data type maps to the DATE data type in Oracle by default.	DATE

Database Security

This section includes information on IBM DB2/400 V4R5 and Oracle database security.

IBM DB2/400 V4R5 Privilege System

In OS/400, everything is an object. OS/400 objects include the following types that are essential to IBM DB2/400 V4R5 operations:

- User profiles (*USRPRF)
- Libraries (*LIB)
- Database files (*FILE)
- Programs (*PGM)
- Authorization lists (*AUTL)

These types are described in the following User Profiles and Authorities sections.

User Profiles

A user profile object represents a user of an AS/400 system. A user profile has a name, password, and a set of values that control various aspects of security. The user profile has a unique password that the user must enter to sign-on to an AS/400 system. The security officer (QSECOFR), or individual in charge of an organization's applications, grants the user profile the appropriate authority to access various OS/400 objects so that the user can work with application programs and data. After a user has signed on with a particular user profile, that user profile governs all access to other objects, including libraries, programs, and files.

OS/400 security is conceptually simple. When a user profile attempts an operation, OS/400 checks to ensure that the user profile has adequate authority to perform the operation on the target object.

Authorities for Libraries, Programs, and Database Files

To access any object, a user profile must have *EXECUTE authority for the library containing the object. Without this authority, the user profile cannot perform any actions with objects in the library. To enable a user to access one or more of the objects in a library, the user profile *USE authority (which includes *EXECUTE authority) must normally be granted to the library.

To access data through an open physical file member, a user profile must have *OBJOPR authority for the field, and one or more of the *READ, *ADD, *UPD, or *DLT data authorities for the physical file. The data authorities control access to records by the read, add, update, and delete operations.

To access data in a physical file member by opening a logical file member, a user profile must have *OBJOPR authority and one or more of the *READ, *ADD, *UPD, or *DLT data authorities for the logical file. In addition, the user profile must have the required data authorities for the physical file. For example, to read and update records through a logical file, the user must have *OBJOPR, *READ, and *UPD authorities for the logical file and *READ and *UPD authorities to the physical file.

Public Authority

Public authority lets you grant authority to more than one user profile at a time. Every object has public authority, which controls access by user profiles that are not otherwise authorized to the object.

In Oracle, there is a special user group named PUBLIC. Every user's security domain includes the privileges and roles granted to the PUBLIC user group.

Group Profiles and Authorization Lists

Groups provide a way to identify an individual user profile as a member of one or more groups. The migration of groups is not supported by this release of the Migration Workbench for IBM DB2/400 V4R5.

Authorization lists provide a somewhat comparable feature for groups, allowing you to organize sets of objects that have been granted identical authorities. The migration of authorization lists is not supported by this release of the Migration Workbench for IBM DB2/400 V4R5.

Object Ownership

Every object is owned by a user profile. The owner of an object normally has all authorities for an object.

For every table object in the source database, the Migration Workbench determines the owner of the table object, and the users granted privileges for that table object. The Migration Workbench then determines if the user has SELECT, INSERT, UPDATE, DELETE, and EXECUTE privileges for the table object. It also ensures that these privileges are reinstated during migration, to ensure the user has the same table-level privileges in the destination Oracle database as in the source IBM DB2/400 V4R5 database. Therefore, a large portion of the original privilege system is maintained during the migration process.

Migration Workbench Add-In Programs

As stated in Chapter 1, IBM DB2/400 V4R5 is integrated with to OS/400, the operating system for AS/400. The database catalogs for IBM DB2/400 V4R5 do not contain the user information (names, IDs, and privileges) because of this integration with the operating system. To migrate this information, the Migration Workbench requires add-in programs, which are copied to, and run on, the source AS/400 system. The Migration Workbench uses FTP to send a `SaveFile` containing programs that collect this information. During the loading phase of the migration, these programs are copied to the `OMWB` library on the AS/400 system.

Note: These programs are installed as a SaveFile in the `$ORACLE_HOME/addins/AS400/V4R3` directory during installation of the Migration Workbench. The source code for both programs is also installed in that directory. Both programs are also executed automatically during the load phase of the Workbench installation.

The SaveFile contains a progam called `GETAUTHOBJ` that contains the object authorities and permissions, and a program that obtains the users of the system. These programs run under the authority of the owner of these programs, that is under the authority of the OS/400 user profile, which ran the `DORST` program on the `SaveFile`. After the Migration Workbench has successfully migrated data, the library containing these two programs can be deleted.

3

Migration Process

This chapter introduces the migration process by outlining the architecture of both IBM DB2/400 V4R5 and Oracle. It includes information on the following:

- IBM DB2/400 V4R5 Architecture
- Oracle Architecture
- Preparing for Migration
- Extending the Application
- Using Offline Data Loading

IBM DB2/400 V4R5 Architecture

IBM DB2/400 V4R5 is an integrated database management system for OS/400, the AS/400 operating system. It is a database system used for storing and manipulating large volumes of data and it forms the basis for most of the business applications that run on the AS/400.

Everything in the OS/400 operating system, including the database, is organized as objects. Each object has a qualified name. Examples of OS/400 objects are program files, database files, and user profiles. There are about 80 types of objects and some object types contain sub-types, for example the file object type `*FILE` includes physical files, logical files, printer files, display files, and communications files. An OS/400 object is uniquely identified by its qualified name and object type.

The Distributed Data Management (DDM) architecture provides a basis for distributed file access. Only native data access is allowed for DDM files. IBM created the Distributed Relational Database Architecture (DRDA) layer on top of DDM, and this provides the protocol that a SQL application can use to access distributed tables and data. Oracle supports the DRDA standard and the Oracle Transparent Gateway for IBM DB2/400 V4R5 provides the capability to transparently access and update data stored in distributed locations in IBM DB2/400 V4R5 databases.

High-Level Languages (HLLs) supported by IBM DB2/400 V4R5 include RPG, COBOL, C++, and CL (Command Language).

IBM DB2/400 V4R5 databases can be accessed from any application program using the Microsoft Open Database Connectivity (ODBC) interface, the Java Database Connectivity (JDBC) interface, or a Common Object Request Broker Architecture (CORBA) interface broker.

IBM DB2/400 V4R5 is no longer supported by IBM. The Migration Workbench provides customers with the ability to migrate from IBM DB2/400 V4R5 to Oracle, on any Oracle-supported platform.

Oracle Architecture

Oracle9*i* is a powerful, flexible, and scalable relational database management system (RDBMS) server, that run on a range of computer systems, from personal computers to the largest mainframes.

The architectural features described in this chapter are only a few of the features provided by Oracle. The features relate only to an IBM DB2/400 V4R5 migration. Refer to the following Oracle Server manuals for a complete description of the Oracle architecture. These manuals can also be found in online format on CD-ROM:

- *Getting to Know Oracle8i*
- *Oracle8i Concepts, Release*
- *Oracle8i Administrator’s Guide*
- *PL/SQL User's Guide and Reference*
- *Oracle8i Error Messages*

PL/SQL Programming Language

PL/SQL is a modern, full-featured programming language with exception handling. You can use PL/SQL to write stored programs and triggers in Oracle. It is also the programming language used in many of the client-side tools available from Oracle, such as Forms from the Oracle Developer suite of products.

Triggers and Stored Procedures

Oracle allows you to write and store code in the DBMS along with data. You can associate trigger code with an UPDATE, INSERT, or DELETE event for each row or for a table as a whole. You can also set a trigger to run before or after the event. For example, you can set a trigger to run after any row is updated.

A stored procedure is a general routine, either function or subroutine, that is stored in precompiled form on the server. A trigger can call stored procedures, but triggers are activated only by specific database activity, such as the insertion of a row in a table.

Sequences

A sequence is a unique number generator that is implemented in shared memory on a server. It is designed to provide a set of unique values for use as primary keys in high-performance applications.

Transactions

Oracle supports an implicit transaction model. Each SQL statement is part of a logical transaction. A logical transaction begins with the first SQL statement and ends with a `COMMIT` or `ROLLBACK` statement. Immediately after either of these statements, a new transaction takes effect with the next SQL statement.

Other Oracle Features

A database administrator has great flexibility when configuring Oracle. The administrator can write data on multiple disks for increased performance, tune rollback and recovery options, and allocate computer resources to optimize the configuration for each server. Oracle also supports distributed processing, so data can be distributed across multiple systems. Oracle offers a version of the server called Trusted Oracle Server for applications that require a higher level of user and use authentication.

Preparing for Migration

You must back up the IBM DB2/400 V4R5 database files before using the Migration Workbench Capture wizard to migrate to Oracle.

Extending the Application

After you move the data management portion of the IBM DB2/400 V4R5 application to Oracle, you can rely on Oracle to protect the data and maintain all referential integrity and business rules that you have encoded in PL/SQL. With this foundation, you can use a wide range of tools such as Oracle JDeveloper and Oracle Objects for OLE to extend the application.

In addition, if the application grows, you can move the Oracle server to larger computers without changing the application.

Using Offline Data Loading

You can use the `extract_nn` add-in program, shipped with the Migration Workbench, to extract the data of an IBM DB2/400 V4R5 database into delimited flat files. The Migration Workbench uses the `extract_nn` add-in with SQL*Loader to provide an offline data loading capability for large tables. The following topics explain the process of offline data loading:

- extract_nn Add-in
- Script Directory Structure
- Generating Extract Scripts
- Using the Extract Scripts
- Installing the extract_nn program to the source AS/400 system

extract_nn Add-in

The `extract_nn` add-in uses SQL to describe the fields of a file and retrieve the data from the file, given a library name and a file name. The field data is written to a specified file in the Integrated File System, as a parameter to the `extract_nn` add-in itself. The data is separated as follows:

- Fields are separated by field-terminator strings
- Rows are separated by record-terminator strings
- Last field in a record is also terminated by the record-terminator.

NULL entries in the table are not explicitly returned. A NULL value can be any of the following:

NULL entry location	Terminator
NULL neither at the start or end of a row	Two successive field-terminators
NULL in the last column	A field-terminator followed by a record-terminator
NULL in first column in any row other than the first row	A record-terminator followed by a field-terminator
NULL in first column of the first row of the table	A field-terminator at the very start of the file

The output file cannot contain any binary data. Decimal and zoned decimal fields are presented as numeric strings with the appropriate sign and decimal-point, as required. Floating-point columns are formatted using the `%f` format. The column types that are valid for the `extract_nn` program are:

- CHAR
- CHAR(n)
- VARCHAR(n)
- LONG VARCHAR(n)
- NULL-terminated CHAR
- INTEGER
- SMALLINT
- FLOAT
- DOUBLE
- DECIMAL *p, s*
- NUMERIC *p, s*
- DATE : **ISO date: yyyy-mm-dd*
- TIME : **ISO time: hh.mm.ss*
- TIMESTAMP : **ISO: yyyy-mm-dd-hh.mm.ss.nnnnnn*

For more information on data types refer to "Data Types in Oracle and IBM DB2/400 V4R5" on page 2-10.

GRAPHIC types, Datalink columns, and BLOB columns are not valid types for the `extract_nn` program. All CHAR and VARCHAR columns must be SBCS and not have a CCSID of 65535.

Script Directory Structure

The `%ORACLE_HOME%\Omwb\sqlloader_scripts` directory contains all data extraction scripts. There is a subdirectory in this directory named `db2400v4r3` that contains the SQL*Loader script output for IBM DB2/400 V4R5. The Migration Workbench creates a subdirectory in the `db2400v4r3` directory using the date and time the SQL*Loader scripts were generated. For example, a subdirectory named `1-06-01_17-56-16` contains scripts generated at 17:56 P.M. on June 1st 2001.

The `extract.omwb` file is created by the Generate SQL*Loader Script command and is located in this subdirectory. The file contains a series of command lines that are required for the `extract_nn` add-in to extract data from the specified tables. You must edit some of the fields in this file, refer to "Using the Extract Scripts" for more information. Use ftp to copy this file and the `extract_nn` add-in program file, to the `omwb_lib` library on the source AS/400 system.

When you are generating SQL*Loader Scripts in the Migration Workbench, a subdirectory called `oracle` is created in the *`timestamp`* directory. The `oracle` directory contains SQL*Loader control files and a SQL*Loader script called `sql_load_script.bat`. The SQL*Loader control files and the data files that you create must be located in this directory. Therefore, after running the `extract_nn` add-in on the source AS/400 system, use ftp to copy the resulting data files back to the `sqlloader_scripts/db2400v4r3`/*timestamp*/oracle directory on the target system before executing the `sql_load_script.bat` file.

Generating Extract Scripts

To create the `extract_nn` data extraction scripts and the SQL*Loader control files for all tables:

1. From the Oracle Model, select the **Tables** folder.
2. Choose **Object** -> **Generate SQL*Loader Scripts**.

Note: You can also generate the scripts for a specific table by selecting that table from the Oracle Model, then choosing **Object** -> **Generate SQL*Loader Scripts**.

3. When you are sure you want to generate the SQL*Loader scripts for the tables specified, click **Yes**.
4. After noting the location of the SQL*Loader scripts, click **OK**.

Using the Extract Scripts

After generating the SQL*Loader scripts, you can use them to load the data into the Oracle database. You must copy the `extract_nn` file using FTP to the host AS/400 system.

Each command line for each table in the `extract.omwb` file is as follows:

```
EXTRACT( "SCHEMA_NAME" , "TABLE_NAME" , "<OUTPUT INTEGRATED FILE SYSTEM
NAME>/TABLE_NAME.DAT", "<ec>" , "<er>" , "" , "<LOGFILE INTEGRATED FILE SYSTEM
NAME> /TABLE_NAME_LOG.DAT" )
```

The parameters of the above example are described in the following table:

Parameter	Description
"SCHEMA_NAME"	The schema containing the table (file) that you want to extract. The Migration Workbench automatically completes this parameter.
"TABLE_NAME"	The table name from which you want to extract the data. The Migration Workbench automatically completes this parameter.
"<OUTPUT INTEGRATED FILE SYSTEM NAME>"	The file in the Integrated File System (IFS) containing the extract from the table. You must fill in this parameter in the `extract.omwb` before running the batch file.
"<ec>"	The character string used to delimit each field (terminated by a `0x00` character). Each CHAR, VARCHAR, and null-CHAR string is checked for this character if the length of the string is 1. If the character appears in the database string it is doubled, creating the field-terminator string, which is set to `"<ec>"` by default.
"<er>"	The character string used to delimit each record (terminated by a 0x00 character). This is the record-terminator string and is set to `"<er>"` by default.

Parameter	Description
`"NULL"`	The character string used as the value for any NULL values. Since zero-length data and NULL entries are both output to the file in the same format, `"<ec><ec>"`, there is nothing else between the column terminators, which could indicate a string of zero length or a NULL. You may want to specify what the value of NULL should be set to.
`"<LOGFILE INTEGARATED FILE SYSTEM NAME>"`	The file in the Integrated File System (IFS) containing the logfile generated from the extract process. It contains descriptions of any errors encountered. You must fill in this parameter in the `extract.omwb` file before running the batch file.

Installing the `extract_nn` program to the source AS/400 system

The `extract_nn` is included with the other add-in programs in the `savefile` program that is copied to the `%oracle_home%\omwb\addins\DB2400V4R3` directory during the installation process of the Migration Workbench.

To run the `extract_nn` add-in program, you must FTP the `savefile` to the source AS/400 system. The add-in program is installed automatically to the omwb_lib directory on the source AS/400 system during the database capture process of the Migration Workbench. If you have used the Capture Wizard for the IBM DB2/400 V4R3 plug-in, the `extract_nn` program can be run from the omwb_lib library on the source AS/400 system using the call commands outlined in "Calling the extract_nn program".

If, however, you want to use the `extract_nn` program without going through the capture process, you have to carry out the following steps:

1. Login to the source AS/400 system using a user id that has *SECOFR or *SECADM authority - QSECOFR is a suitable user id.
2. Create a library on the source AS/400 system. A good name would be omwb_lib -- but you can use any library name you want. We will use omwb_lib for that name in what follows.
3. Create a `savefile` (object type `*SAVF`) in the library created in step 2. Call that file `savefile`. For example:

   ```
   CRTSAVF OMWB_lib/SAVEFILE
   ```

4. Open an FTP connection to the target AS/400 system.

5. Set the transfer mode to Binary.
6. Use a `cd` subcommand to make sure the data goes to the correct library on the AS/400. For example, `cd omwb_lib.`
7. FTP the `savefile` in binary mode, from the `%oracle_home%\omwb\addins\DB2400V4R3` directory on the Migration Workbench installation system to the a library on the source AS/400 system using the `put` command. This file is approximately 1 MB in size. For example:

   ```
   PUT SAVEFILE
   ```

8. The `savefile` on the target AS/400 system contains nine `*MODULE` objects and a `*FILE` object.

   Restore the objects from the `savefile` using the `RSTOBJ` command. The command should be entered on one line:

   ```
   RSTOBJ OBJ(*ALL) SAVLIB(QTEMP) DEV(*SAVF) SAVF(OMWB_lib/SAVEFILE)
   RSTLIB(OMWB_lib)
   ```

9. Create the `DORST` program by entering the following command:

   ```
   CRTPGM PGM(OMWB_lib/DORST) MODULE(OMWB_lib/DORST)
   ```

10. Run the `DORST` program to complete the installation. A single parameter is required. The parameter is the name of the library in which the add-in's program is placed. In the following example, the program is placed in the same library as the `savefile` and the objects extracted from that `savefile:`

    ```
    CALL PGM(OMWB_lib/DORST) PARM(OMWB_lib)
    ```

11. You also need to create extract scripts for input to the `extract_nn` program. The format of these scripts is defined in "Using the Extract Scripts".
12. You can now run the `extract_nn` program.

Calling the extract_nn program

You can run the `extract_nn` program when the following steps have been completed:

- The add-in programs' `savefile` is installed correctly on the source AS/400 system, either manually or through the Migration Workbench capture phase.

- The `extract.omwb` file is generated and copied by FTP to the source system.

The call to the `extract_nn` program accepts up to two parameters.

1. The first parameter, which is not optional, is a string containing the location of the `extract.omwb` file. An example of a call to the `extract_nn` program using this single parameter is as follows:

   ```
   CALL OMWB_lib/EXTRACT_NN PARM( '/home/usr/EXTRACT.OMWB' )
   ```

2. A second optional parameter can be added to the call to the `extract_nn` program. This parameter is used to:

   - Set the length of the returned string for a `TIMESTAMP` value.
   - Request an estimate of file sizes to be returned by the `extract_nn` program.

   An example of a call to the `extract_nn` program to retrieve a TIMESTAMP value no longer than 19 characters is as follows:

   ```
   CALL OMWB_lib/EXTRACT_NN PARM('home/usr/EXTRACT.OMWB', 'TIMESTAMP_LEN=19')
   ```

   The above call tells the `extract_nn` program to truncate any `TIMESTAMP` values to 19 characters.

   An example of a call to the `extract_nn` program to retrieve an estimate of file sizes the `extract_nn` program will generate is as follows:

   ```
   CALL OMWB_lib/EXTRACT_NN PARM('home/usr/EXTRACT.OMWB', 'DISKSPACE')
   ```

   The above call does not extract any data to files. It places comments about how large the output file might be in the log file.

   Note: The file sizes estimate is approximate and depends on the real data.

To use the EXTRACT script to execute a manual data extraction:

1. Capture the schema model of the source IBM DB2/400 V4R5 database using the Migration Workbench, and generate the extraction scripts as described in "Generating Extract Scripts" on page 3-6.
2. Migrate the IBM DB2/400 V4R5 table schema to your Oracle database. Select **No** when asked "Do you want to migrate table data to Oracle?" in Step 3 of the Migration Wizard.

3. Follow the steps listed in "Generating Extract Scripts" on page 3-6.
4. Edit the resulting extract.omwb in the `%ORACLE_HOME%\Omwb\sqlloader_scripts\DB2400V4R3` directory and enter the values for the required fields (refer to "Using the Extract Scripts" on page 3-7 for more information). Copy the edited `extract.omwb` in the `%ORACLE_HOME%\Omwb\sqlloader_scripts\DB2400V4R3` directory. If required, install the `extract_nn` program to the source AS/400 system. Refer to "Installing the extract_nn program to the source AS/400 system" on page 3-8.
5. Call the `extract_nn` add-in using the following command:

   ```
   CALL LIBRARY_NAME/EXTRACT_NN PARM ( "<EXTRACT.OMWB IFS >/EXTRACT.OMWB")
   ```

   In the above example, `LIBRARY_NAME` is the library where the `extract_nn` program was copied and `<EXTRACT.OMWB IFS>` is the Integrated File System name for the location where the `extract.omwb` file was copied. Wait until each command line in the batch file has completed. For examples of how to call the `extract_nn` program and the options available see "Calling the extract_nn program".
6. Copy the resulting output files from the AS/400 system to the `%ORACLE_HOME\Omwb\sqlloader_scripts\`*`timestamp`*`\Oracle` directory on the Migration Workbench client system.
7. Run the `sql_load_script.bat` file in the `%ORACLE_HOME%\Omwb\sqlloader_scripts\`*`timestamp`*`\Oracle` directory to input the data from the flat files into the Oracle database.

4 Troubleshooting

This chapter provides troubleshooting solutions, information on offline data loading, and information on avoiding connection issues from the Migration Workbench to the IBM DB2/400 V4R5 database (AS/400 server).

Connecting to the IBM DB2/400 V4R5 Database

Using the Migration Workbench capture Wizard you should connect to the IBM DB2/400 V4R5 database as a user with DBA privileges. It is recommended to log on as QSECOFR, as this user has total control over all aspects of the database. The Migration Workbench will attempt to FTP add-in programs to the AS/400 source system, in order to capture data relevant to the database which cannot be sourced in the database catalog. Therefore, the user you log on as will need write and read privileges for the AS/400 machine.

Support for DBCS Data

This version of the Migration Workbench does not support the migration of databases containing DBCS data. The Migration Workbench appears to freeze or quit prematurely if it encounters DBCS data.

Offline Data Loading

A fast method of migrating data is to use the offline data loading facility. Using the Migration Workbench, the user can automatically create extraction scripts to export the IBM DB2/400 V4R5 database data into flat text files as well as SQL*Loader scripts. These are used to pump the IBM DB2/400 V4R5 flat file data into an Oracle database. This method of migration is considerably faster than via the Migration

Workbench UI. In order to complete a successful migration using this method, the user should take note of the following:

Index