Oracle® Territory Manager

Implementation Guide

Release 12.2

Part No. E48989-04

September 2020

ORACLE

Oracle Territory Manager Implementation Guide, Release 12.2

Part No. E48989-04

Primary Author: Ashita Mathur

Contributor: Stacey Blosch, Udit Verma

Contents

Send Us Your Comments

Preface

1 Introduction

Overview of Oracle Territory Manager 1-1
Sales Territories 1-2
Non-Sales Territories 1-3
Oracle Territory Manager Features 1-4

2 Dependencies and Integration Points

Oracle Territory Manager Mandatory Dependencies 2-1
Oracle Territory Manager Optional Integrations 2-1

3 Implementation Overview

Process Description 3-1

4 Territory Planning

User Security 4-2
Planning Your Territories 4-2
Sales Territory Planning Example 4-5
Step 1: What business objects are we assigning? 4-6
Step 2: What matching attributes should we use? 4-6
Step 3: Territory hierarchy - Define date effectivity and number of winners 4-7
Step 4: Territory hierarchy – Placeholder territories 4-8
Step 5: Implement Self Service 4-8

Step 6: Territory hierarchy – Define Catch Alls 4-9
Step 7: How to implement named account territories 4-9
Step 8: How to implement geographic territories 4-10
Step 9: How to support overlays 4-11
Step 10: What is an appropriate territory hierarchy for overlays? 4-12
Step 11: What rank should each territory have? 4-16
Step 12: Have you met all your business requirements? 4-16
Leverage Territory Hierarchies and Inheritance 4-16
Leverage Territory Ranking and Number of Winners 4-17
Choosing Appropriate Matching Attributes 4-17
Matching Rules 4-19
Migrating Territories 4-19

5 Implementation Tasks

Setting Up User Security 5-1
Setting Up Multiple Organization Access Control 5-2
If You Have Value Added Tax (VAT) 5-3
Creating Custom Matching Attributes 5-3
Enabling Matching Attributes 5-6
Sales Matching Attributes 5-6
Running Concurrent Programs 5-8
Setting Up Territory Assignment Program 5-10
Setting Up to Use Dun & Bradstreet Data 5-10
Setting Up Territory Alignment 5-10
Proxy User 5-11
Setting Up Export 5-12
Loading Postal Codes 5-12
APIs 5-12
CREATE_GEO API 5-13
DELETE_GEO API 5-15
UPDATE_GEO API 5-15

6 Verify and Troubleshoot

Verification Tasks 6-1
Profile Options 6-1
Prerequisites for Using Web ADI 6-2
Troubleshooting Export to Excel 6-3
Party, Party Site, and Account Merge 6-3
Tips for Fine-tuning Territory Assignment Performance 6-5
If the Synchronize Territory Assignment Rules Fails 6-5

Frequently Asked Questions (FAQs) about Transaction Matching Attributes 6-5

A Creating Custom Matching Attributes

CMA for Account: Party Type A-1
CMA for Lead: Lead Status A-16
CMA for Quote: Quote Source A-37
CMA for Proposal: Proposal Status A-52
CMA for Opportunity: Budget Status A-61
CMA for Oracle Collections, Customer: Customer Party Type A-83
Script 1 ..... A-96

Index

Send Us Your Comments

Oracle Territory Manager Implementation Guide, Release 12.2

Part No. E48989-04

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document. Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

Note: Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the new Oracle E-Business Suite Release Online Documentation CD available on My Oracle Support and www.oracle.com. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: appsdoc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at www.oracle.com.

Preface

Intended Audience

Welcome to Release 12.2 of the *Oracle Territory Manager Implementation Guide.*

This document is intended for implementers and territory planners.

This guide assumes that you have a working knowledge of the following:

- The principles and customary practices of your business area
- The Oracle Territory Manager application
- Oracle Application Framework Applications
- The Oracle Applications graphical user interface

  To learn more about the Oracle Applications graphical user interface, read the *Oracle E-Business Suite User's Guide.*

See Related Information Sources on page x for more Oracle E-Business Suite product information.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?

ctx=acc&id=trs if you are hearing impaired.

Structure

1 Introduction
2 Dependencies and Integration Points
3 Implementation Overview
4 Territory Planning
5 Implementation Tasks
6 Verify and Troubleshoot
A Creating Custom Matching Attributes

Related Information Sources

You can choose from many sources of information, including online documentation, training, and support services, to increase your knowledge and understanding of Oracle Territory Manager.

Integration Repository

The Oracle Integration Repository is a compilation of information about the service endpoints exposed by the Oracle E-Business Suite of applications. It provides a complete catalog of Oracle E-Business Suite's business service interfaces. The tool lets users easily discover and deploy the appropriate business service interface for integration with any system, application, or business partner.

The Oracle Integration Repository is shipped as part of the Oracle E-Business Suite. As your instance is patched, the repository is automatically updated with content appropriate for the precise revisions of interfaces in your environment.

Online Documentation

All Oracle E-Business Suite documentation is available online (HTML or PDF).

- **PDF** - See the Oracle E-Business Suite Documentation Library for current PDF documentation for your product with each release. The Oracle E-Business Suite Documentation Library is also available on My Oracle Support and is updated frequently

- **Online Help** - Online help patches (HTML) are available on My Oracle Support.

- **Release Notes** - For information about changes in this release, including new features, known issues, and other details, see the release notes for the relevant product, available on My Oracle Support.

- **Oracle Electronic Technical Reference Manual -** The Oracle Electronic Technical Reference Manual (eTRM) contains database diagrams and a detailed description of

database tables, forms, reports, and programs for each Oracle E-Business Suite product. This information helps you convert data from your existing applications and integrate Oracle E-Business Suite data with non-Oracle applications, and write custom reports for Oracle E-Business Suite products. The Oracle eTRM is available on My Oracle Support.

Guides Related to All Products

Oracle E-Business Suite User's Guide

This guide explains how to navigate, enter and query data, and run concurrent requests using the user interface (UI) of Oracle E-Business Suite. It includes information on setting preferences and customizing the UI. In addition, this guide describes accessibility features and keyboard shortcuts for Oracle E-Business Suite.

Guides Related to This Product

Oracle Advanced Pricing User's Guide

Oracle Advanced Pricing calculates prices including promotional prices for Oracle Order Management and other Oracle Applications based on pricing rules, pricing relationships, item hierarchies, usage brackets, and deals and promotions.

Oracle Sales Contracts Implementation and Administration Guide

Oracle Sales Contracts enhances the ability of sales organizations to manage their contracts by adding sophisticated contract management and compliance features to quoting, ordering, and negotiating long-term agreements. You can use the Contract Expert to define rules for creating contracts and reporting policy deviations. This guide describes how to establish contract standards, author and negotiate contracts, and approve, sign, and manage them.

Oracle Sales for Handhelds Implementation Guide

This guide describes how to set up Data Quality Management to manage customers, set appointment preferences of timezone and categories, and map appointment, task, and contact on your handheld device with Oracle E-Business Suite. You can implement clients to synchronize your handheld with Oracle Sales and Microsoft Desktop Outlook and subscribe to Short Message Service alerts.

Oracle Sales for Handhelds User Guide

Oracle Sales for Handhelds enables traveling sales professionals to access enterprise information from their pocket PC, Blackberry, palm-based devices, and Nokia using an HTML browser. You can use Outlook for your appointments, view emails received in outlook from contacts as Oracle Sales interaction history, and receive Short Message

Service alerts for service contract expiry, escalated service requests, and invoice overdue. You can manage customers, contacts, and customer visits using your handheld.

Oracle Sales Offline Implementation Guide

Oracle Sales Offline enables you to remotely manage your sales efforts without logging in every time to Oracle Sales. You can use the template provided by Oracle Sales Offline to download and upload sales information without the need to install additional software. To implement this, you must have first installed Oracle Sales, Oracle Quoting, and Oracle Web Applications Desktop Integrator. You must also configure timezones and timezone conversions.

Oracle Sales Offline User Guide

Oracle Sales Offline is a mobile sales application that uses templates to enable sales representatives remotely manage their day-to-day sales activities. You do not have to log into Oracles Sales to download and upload the template and template data. Oracle Sales Offline works with Oracle Sales and with Oracle TeleSales to create a virtual sales team that enables the sharing of opportunities, contacts, notes, and other customer information between sales team members. You can use Oracle Sales Offline to manage leads and opportunities, build forecasts from opportunities, manage customers, maintain and develop customer quotes, and create and manage assigned tasks.

Oracle Sales Implementation Guide

This guide enables you to set up users, user groups, and roles, define forecast categories that group products and services to be included in a forecast, set up and enable currency conversion, and set up the sales dashboard linking the sales funnel to sales stages of a sales methodology. You can also set up Oracle Sales and Oracle Telesales interoperability and set up Oracle Sales for integration with Oracle Territory Manager, Oracle Marketing, Oracle Quoting, Oracle Proposals, Oracle Channel Revenue Management, Oracle Partner Management, and Oracle Incentive Compensation.

Oracle Sales User Guide

Oracle Sales enables sales professionals plan and manage the sales process from leads to opportunities to quotes including the tracking of competitors for products within opportunities. It is integrated in the E-Business Suite and optimized for use with wireless. You can use the sales dashboard to view open opportunities, proposals, quotes, top customers, leads by age and by campaign, the latest sales forecast, and your calendar tasks.

Oracle TeleSales Implementation Guide

This guide describes how you can set up Oracle TeleSales so telesales agents can convert a sales inquiry or a customer call into an order. You must set up agent and customer interaction tracking, enable web directory assistance for agents, enable web

collaboration, set up opportunity forecasting so agents enter forecast amounts for a product line and receive sales credits, and set up marketing source codes to track the marketing activity responsible for a sale or a sales activity. Oracle TeleSales interacts with Oracle Scripting, Oracle Email Center, Oracle Marketing, Oracle Territory Manager, Oracle Product Hub, Oracle One-to-One Fulfillment, Oracle Universal Work Queue, Oracle Sales, and Oracle Quoting.

Oracle TeleSales User Guide

Oracle TeleSales enables telesales agents manage the sales cycle, from prospects to booked orders. It offers a multi-channel selling solution that leverages all sales channels: whether selling over the phone, through the web or through mobile devices. Its E-Business Center offers a cross-application desktop for all Oracle call center applications, and provides elements of Service and Collections for a comprehensive customer view. You can use Oracle TeleSales for comprehensive customer management, list generation, lead, opportunity, and pipeline management, quote and order generation, event registration and collateral fulfillment.

Oracle Territory Manager User Guide

Oracle Territory Manager enables you to distribute sales and after sales tasks by geographical location, account, task priority, and resource skills. Oracle Sales, Oracle Field Service, Oracle Service Contracts, Oracle Collections, Oracle Partner Manager, and Oracle Channel Revenue Management all use Oracle Territory Manager to define ownership of transactions.

Oracle Channel Revenue Management Implementation and Administration Guide

Channel Revenue Management enables users to efficiently plan, promote, execute, and manage the order to cash process for improved sales and return on investment (ROI), and reduced loss in revenue. Use this guide to learn about the different products in the Oracle Channel Revenue Management Suite and the other Oracle E-Business Suite products with which this product family integrates. You can learn how to set up users, customers, and suppliers, and perform the basic configurations that will be used by all the products in this suite.

Oracle Channel Rebate and Point-of-Sale Management User Guide

Oracle Channel Rebate and Point of Sales Management enables suppliers to manage their product inventory and prices, create budgets for customer and partner rebates, offers, and incentives, and enlist the help of channel partners such as distributors and retailers to manage execution of these offers at the points of sale in the channel. This guide describes how to use the Account Manager Dashboard to manage products and price lists, create and manage budgets, quotas, and offers, and plan and manage customer accounts.

Oracle Workflow User's Guide

This guide describes how users can view and respond to workflow notifications and monitor the progress of their workflow processes.

Installation and System Administration

Oracle Alert User's Guide

This guide explains how to define periodic and event alerts to monitor the status of your Oracle E-Business Suite data.

Oracle E-Business Suite Concepts

This book is intended for all those planning to deploy Oracle E-Business Suite Release 12.2, or contemplating significant changes to a configuration. After describing the Oracle E-Business Suite architecture and technology stack, it focuses on strategic topics, giving a broad outline of the actions needed to achieve a particular goal, plus the installation and configuration choices that may be available.

Oracle E-Business Suite CRM System Administrator's Guide

This manual describes how to implement the CRM Technology Foundation (JTT) and use its System Administrator Console.

Oracle E-Business Suite Developer's Guide

This guide contains the coding standards followed by the Oracle E-Business Suite development staff. It describes the Oracle Application Object Library components needed to implement the Oracle E-Business Suite user interface described in the *Oracle E-Business Suite User Interface Standards for Forms-Based Products*. It also provides information to help you build your custom Oracle Forms Developer forms so that they integrate with Oracle E-Business Suite. In addition, this guide has information for customizations in features such as concurrent programs, flexfields, messages, and logging.

Oracle E-Business Suite Installation Guide: Using Rapid Install

This book is intended for use by anyone who is responsible for installing or upgrading Oracle E-Business Suite. It provides instructions for running Rapid Install either to carry out a fresh installation of Oracle E-Business Suite Release 12.2, or as part of an upgrade to Release 12.2.

Oracle E-Business Suite Maintenance Guide

This guide contains information about the strategies, tasks, and troubleshooting activities that can be used to help ensure an Oracle E-Business Suite system keeps

running smoothly, together with a comprehensive description of the relevant tools and utilities. It also describes how to patch a system, with recommendations for optimizing typical patching operations and reducing downtime.

Oracle E-Business Suite Security Guide

This guide contains information on a comprehensive range of security-related topics, including access control, user management, function security, data security, and auditing. It also describes how Oracle E-Business Suite can be integrated into a single sign-on environment.

Oracle E-Business Suite Setup Guide

This guide contains information on system configuration tasks that are carried out either after installation or whenever there is a significant change to the system. The activities described include defining concurrent programs and managers, enabling Oracle Applications Manager features, and setting up printers and online help.

Oracle E-Business Suite User Interface Standards for Forms-Based Products

This guide contains the user interface (UI) standards followed by the Oracle E-Business Suite development staff. It describes the UI for the Oracle E-Business Suite products and tells you how to apply this UI to the design of an application built by using Oracle Forms.

Other Implementation Documentation

Oracle Approvals Management Implementation Guide

This guide describes transaction attributes, conditions, actions, and approver groups that you can use to define approval rules for your business. These rules govern the process for approving transactions in an integrated Oracle application. You can define approvals by job, supervisor hierarchy, positions, or by lists of individuals created either at the time you set up the approval rule or generated dynamically when the rule is invoked. You can learn how to link different approval methods together and how to run approval processes in parallel to shorten transaction approval process time.

Oracle Diagnostics Framework User's Guide

This guide contains information on implementing, administering, and developing diagnostics tests for Oracle E-Business Suite using the Oracle Diagnostics Framework.

Oracle E-Business Suite Flexfields Guide

This guide provides flexfields planning, setup and reference information for the Oracle E-Business Suite implementation team, as well as for users responsible for the ongoing maintenance of Oracle E-Business Suite product data. This guide also provides

information on creating custom reports on flexfields data.

Oracle E-Business Suite Integrated SOA Gateway Implementation Guide

This guide explains the details of how integration repository administrators can manage and administer the entire service enablement process based on the service-oriented architecture (SOA) for both native packaged public integration interfaces and composite services - BPEL type. It also describes how to invoke Web services from Oracle E-Business Suite by working with Oracle Workflow Business Event System, manage Web service security, and monitor SOAP messages.

Oracle E-Business Suite Integrated SOA Gateway User's Guide

This guide describes how users can browse and view the integration interface definitions and services that reside in Oracle Integration Repository.

Oracle E-Business Suite Multiple Organizations Implementation Guide

This guide describes how to set up multiple organizations and the relationships among them in a single installation of an Oracle E-Business Suite product such that transactions flow smoothly through and among organizations that can be ledgers, business groups, legal entities, operating units, or inventory organizations. You can use this guide to assign operating units to a security profile and assign this profile to responsibilities such that a user can access data for multiple operating units from a single responsibility. In addition, this guide describes how to set up reporting to generate reports at different levels and for different contexts. Reporting levels can be ledger or operating unit while reporting context is a named entity in the selected reporting level.

Oracle e-Commerce Gateway Implementation Guide

This guide describes implementation details, highlighting additional setup steps needed for trading partners, code conversion, and Oracle E-Business Suite. It also provides architecture guidelines for transaction interface files, troubleshooting information, and a description of how to customize EDI transactions.

Oracle e-Commerce Gateway User's Guide

This guide describes the functionality of Oracle e-Commerce Gateway and the necessary setup steps in order for Oracle E-Business Suite to conduct business with trading partners through Electronic Data Interchange (EDI). It also describes how to run extract programs for outbound transactions, import programs for inbound transactions, and the relevant reports.

Oracle iSetup User's Guide

This guide describes how to use Oracle iSetup to migrate data between different instances of the Oracle E-Business Suite and generate reports. It also includes configuration information, instance mapping, and seeded templates used for data

migration.

Oracle Product Hub Implementation Guide

This guide explains how to set up hierarchies of items using catalogs and catalog categories and then to create user-defined attributes to capture all of the detailed information (such as cost information) about an object (such as an item or change order). It also explains how to set up optional features used in specific business cases; choose which features meet your business' needs. Finally, the guide explains the set up steps required to link to third party and legacy applications, then synchronize and enrich the data in a master product information repository.

Oracle Product Hub User's Guide

This guide explains how to centrally manage item information across an enterprise, focusing on product data consolidation and quality. The item information managed includes item attributes, categorization, organizations, suppliers, multilevel structures/bills of material, packaging, changes, attachments, and reporting.

Oracle Web Applications Desktop Integrator Implementation and Administration Guide

Oracle Web Applications Desktop Integrator brings Oracle E-Business Suite functionality to a spreadsheet, where familiar data entry and modeling techniques can be used to complete Oracle E-Business Suite tasks. You can create formatted spreadsheets on your desktop that allow you to download, view, edit, and create Oracle E-Business Suite data, which you can then upload. This guide describes how to implement Oracle Web Applications Desktop Integrator and how to define mappings, layouts, style sheets, and other setup options.

Oracle Workflow Administrator's Guide

This guide explains how to complete the setup steps necessary for any Oracle E-Business Suite product that includes workflow-enabled processes. It also describes how to manage workflow processes and business events using Oracle Applications Manager, how to monitor the progress of runtime workflow processes, and how to administer notifications sent to workflow users.

Oracle Workflow Developer's Guide

This guide explains how to define new workflow business processes and customize existing workflow processes embedded in Oracle E-Business Suite. It also describes how to define and customize business events and event subscriptions.

Oracle Workflow User's Guide

This guide describes how Oracle E-Business Suite users can view and respond to workflow notifications and monitor the progress of their workflow processes.

Oracle XML Gateway User's Guide

This guide describes Oracle XML Gateway functionality and each component of the Oracle XML Gateway architecture, including Message Designer, Oracle XML Gateway Setup, Execution Engine, Message Queues, and Oracle Transport Agent. It also explains how to use Collaboration History that records all business transactions and messages exchanged with trading partners.

The integrations with Oracle Workflow Business Event System, and the Business-to-Business transactions are also addressed in this guide.

Oracle XML Publisher Administration and Developer's Guide

Oracle XML Publisher is a template-based reporting solution that merges XML data with templates in RTF or PDF format to produce outputs to meet a variety of business needs. Outputs include: PDF, HTML, Excel, RTF, and eText (for EDI and EFT transactions). Oracle XML Publisher can be used to generate reports based on existing Oracle E-Business Suite report data, or you can use Oracle XML Publisher's data extraction engine to build your own queries. Oracle XML Publisher also provides a robust set of APIs to manage delivery of your reports via e-mail, fax, secure FTP, printer, WebDav, and more. This guide describes how to set up and administer Oracle XML Publisher as well as how to use the Application Programming Interface to build custom solutions. This guide is available through the Oracle E-Business Suite online help.

Oracle XML Publisher Report Designer's Guide

Oracle XML Publisher is a template-based reporting solution that merges XML data with templates in RTF or PDF format to produce a variety of outputs to meet a variety of business needs. Using Microsoft Word or Adobe Acrobat as the design tool, you can create pixel-perfect reports from the Oracle E-Business Suite. Use this guide to design your report layouts. This guide is available through the Oracle E-Business Suite online help.

Training and Support

Training

Oracle offers a complete set of training courses to help you master your product and reach full productivity quickly. These courses are organized into functional learning paths, so you take only those courses appropriate to your job or area of responsibility.

You have a choice of educational environments. You can attend courses offered by Oracle University at any of our many Education Centers, you can arrange for our trainers to teach at your facility, or you can use Oracle Learning Network (OLN), Oracle University's online education utility. In addition, Oracle training professionals can tailor standard courses or develop custom courses to meet your needs. For example, you may

want to use your organization structure, terminology, and data as examples in a customized training session delivered at your own facility.

Support

From on-site support to central support, our team of experienced professionals provides the help and information you need to keep your product working for you. This team includes your Technical Representative, Account Manager, and Oracle's large staff of consultants and support specialists with expertise in your business area, managing an Oracle server, and your hardware and software environment.

Do Not Use Database Tools to Modify Oracle E-Business Suite Data

Oracle STRONGLY RECOMMENDS that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle E-Business Suite data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Oracle E-Business Suite data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle E-Business Suite tables are interrelated, any change you make using an Oracle E-Business Suite form can update many tables at once. But when you modify Oracle E-Business Suite data using anything other than Oracle E-Business Suite, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle E-Business Suite.

When you use Oracle E-Business Suite to modify your data, Oracle E-Business Suite automatically checks that your changes are valid. Oracle E-Business Suite also keeps track of who changes information. If you enter information into database tables using database tools, you may store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

1

Introduction

This chapter covers the following topics:

- Overview of Oracle Territory Manager
- Oracle Territory Manager Features

Overview of Oracle Territory Manager

Oracle Territory Manager assigns business objects (customers and leads, for example) to resources based on configurable business rules. It defines who owns what.

When concurrent programs are run, the territory assignment engine assigns business objects such as the following to resources:

- customers
- leads
- opportunities
- service requests
- tasks
- contract renewals
- trade management claims and offers

Additional information on concurrent programs for sales is in the "Setting Up and Using Territory Assignment Program (TAP)" section of either the *Oracle Sales Implementation Guide* or the *Oracle TeleSales Implementation Guide.*

Components

Territory implementations use all or a subset of the following components to build

territories:

- Territory Types: Required. Territory Types are the building blocks for all territories. A type is blueprint for territories, defined by a specific set of transaction types and corresponding matching attributes.

- Self Service Named Accounts: Optional. Sales administrators use the seeded named account territory type to create named account territories. Sales managers then assign their named accounts to individual sales representatives. Sales managers use the alignment tool to create different territory assignment models and compare them.

- Self Service Geographic Territories: Optional. Sales administrators create a self-service geographic type to create geographic territories and assign them to sales managers. Sales managers can then allocate their geographic region to their directs.

- Concurrent programs enable territory definitions.

- A stand-alone sales team search is available to any user.

Sales Territories

Sales organizations use territories to automatically assign prospects to salespeople.

An example of a sales territory is: High-tech companies within a specific geographic area. This territory is defined using the following matching attributes:

- Account Classification = High Tech

- State = California

The resource assigned to the territory is Joe who is in Sam's sales group. When the assignment engine is run Joe is assigned to all high-tech companies that have an address in the state of California. Because Joe is in Sam's sales group, his manager and the other resources within his group can be granted access to these same companies.

Named Accounts

Most customers fall into sales territories segmented along geographic or industry boundaries. Named accounts represent individual customers elevated from geographic territories and deemed by a sales organization as critical enough to have their own salesperson or account manager.

By their very nature, named account territories are difficult and complex to maintain and revolve around a decentralized business process.

Upper levels of sales management identify a set of named accounts and associate them to a sales division. The sales vice presidents responsible for the sales division distribute named accounts to their directs in a top down fashion through the sales hierarchy until

all named accounts are owned by salespersons.

Self-Service Named Account Alignment for Sales

Proper territory alignment is frequently overlooked and can have a high impact on sales force productivity. Without any additional salespeople, studies have shown 2-7% increases in sales revenue due to proper territory alignments.

Sales managers start the year with compensation plans, named accounts, and quotas. They can then build and save what-if territories in an iterative manner until their alignment goals are met.

Sales managers export a territory alignment to spreadsheet, change named account assignments, and upload the changes for the alignment. Through the HTML interface they use analytic data to compare two alignments using graphs and a calculated balance index. When managers have finalized an alignment they can activate it, making it the one of record.

At the end of the alignment process, you can manually enter named account quotas into the Oracle Incentive Compensation Quota Planning module.

Self-Service Geographic Territories for Sales

Geographies are centrally identified to a sales organization and distributed top down to individual salespeople. Ownership changes are reflected quickly to all levels of sales management and for incoming leads and opportunities.

At each level of the distribution process, the ownership of geographies is clearly and accurately communicated. Sales management interfaces are simplified for common administrative tasks such as the transfer of geographic territories between salespeople.

Non-Sales Territories

The following Oracle modules can use Oracle Territory Manager to assign business objects to resources:

- Oracle Collections
- Oracle Partner Manager
- Oracle Field Service
- Oracle Service Contracts
- Oracle Trade Management

Example

In Oracle Field Service, all high urgency service requests for internal IT support should go to the senior IT support person, John Gray. A territory is defined, with John Gray as the resource, using the following matching attributes for service requests:

- Request Type = Internal: IT
- Request Severity = High1
- Request Status = Open
- Request Urgency = Immediate

Oracle Territory Manager Features

Oracle Territory Manager includes the following features:

- Over 100 matching attributes through which to define territory rules
- Territory Types for ease of territory creation for any usage (such as Oracle Sales or Oracle Service)
- Assignment to individual resources, groups, or teams
- Named account support
- Self-service distribution of named accounts
- Named account alignments with visual comparison of what-if territories
- Export to spreadsheet for ease of territory maintenance
- Self-service distribution of geographic territories
- Configurable territory exception handling through Oracle Workflow
- Ability to create custom matching attributes through a public API

2

Dependencies and Integration Points

This chapter covers the following topics:

- Oracle Territory Manager Mandatory Dependencies
- Oracle Territory Manager Optional Integrations

Oracle Territory Manager Mandatory Dependencies

The following are the modules that Oracle Territory Manager depends upon:

- **Oracle Application Object Library (AOL)**: Territory Manager uses AOL to manage responsibilities that are used in various modules.

- **Resource Manager:** Territory Manager uses resources defined in the Resource Manager to assign resources to a territory.

Oracle Territory Manager Optional Integrations

The following modules are required for specific features to work:

- **Oracle Web Applications Desktop Integrator:** The application uses Web ADI to export and import named account and geography assignments to and from Microsoft Excel. (Required for Geographic Territories and Named Account Territory Alignment.) Also, Oracle Web ADI requires that you use Microsoft Internet Explorer for your browser.

- **Oracle Field Sales or Oracle TeleSales:** Named account alignment metrics depend upon past opportunity data. (Required for Named Account Territory Alignment.)

- **Oracle Trading Community Architecture (TCA):** TCA provides the DUNS number, DNB annual revenue, and DNB number of employees. (Required for Named Accounts if you want to use this information.)

3

Implementation Overview

This chapter covers the following topics:

- Process Description

Process Description

Before using the Oracle Territory Manager, your functional implementation team must analyze your business and organization needs. This step is key before implementing the application.

Based on planning decisions, the implementation team enables seeded matching attributes to be used in defining your territories.

The territory administrator then begins the territory creation process, according to the implementation.

If you implement named accounts for sales, then the administrator creates territories, selecting named accounts, and assigning them to the top level of sales management. Sales managers in turn assign the named accounts to the salespeople or sales managers who report to them. The next level of sales managers in turn assign named accounts to their directs.

After territories are manually created, you can search and view territory hierarchies. The territory administrator must run the *Synchronize Territory Assignment Rules (STAR)* concurrent program to generate territories before modules can assign resources defined in your territories.

Oracle Territory Manager is implemented in the following four phases:

Phase I: Territory Planning

In Phase I, your implementation team analyzes business and organization needs and plans territories accordingly.

Phase II: Setting Up Territories

In Phase II, the territory administrator starts the territory creation process based on

territory planning. If you plan to implement sales territories, then you also need to set up the Territory Assignment Program per the steps provided in either the *Oracle Sales Implementation Guide* and the *Oracle TeleSales Implementation Guide.*

Phase III: Creating Named Account Territories

Phase III is optional. If you have sales territories and your planning includes the use of named accounts, then the territory administrator and sales managers use the self-service application to create named accounts and territories for sales.

Phase IV: Creating Geographic Territories

Phase IV is optional. If you have geographic sales territories, then the territory administrator and sales managers use the self-service application to create geographic territories for sales.

Phase V: Managing Territories

In Phase V, the territory administrator manages territory changes, such as copying or moving an entire territory in the hierarchy. In addition, the administrator can run territory reports to verify territory change information. See the *Oracle Territory Manager User Guide* for more information.

After updating existing territories, the *Synchronize Territory Assignment Rules (STAR)* concurrent program still needs to be run to generate territories for all usages.

4

Territory Planning

This chapter covers the following topics:

- User Security
- Planning Your Territories
- Sales Territory Planning Example
- Step 1: What business objects are we assigning?
- Step 2: What matching attributes should we use?
- Step 3: Territory hierarchy - Define date effectivity and number of winners
- Step 4: Territory hierarchy – Placeholder territories
- Step 5: Implement Self Service
- Step 6: Territory hierarchy – Define Catch Alls
- Step 7: How to implement named account territories
- Step 8: How to implement geographic territories
- Step 9: How to support overlays
- Step 10: What is an appropriate territory hierarchy for overlays?
- Step 11: What rank should each territory have?
- Step 12: Have you met all your business requirements?
- Leverage Territory Hierarchies and Inheritance
- Leverage Territory Ranking and Number of Winners
- Choosing Appropriate Matching Attributes
- Matching Rules
- Migrating Territories

User Security

Oracle Territory Manager supports role-based access control of data security. Each user is assigned one or more territory roles. When users log in to Oracle Applications, all menu items associated to their assigned roles appear on their personal home pages. The following roles are available for assignment:

- Sales Team Search User: Provides access to the stand-alone sales team search
- Territory Reports User
- Sales Territory Reports User (includes Territory Reports User role)
- Sales Territory User (includes Sales Team Search User)
- Territory Manager Application Administrator
- Sales Territory Administrator (includes Sales Team Search User and Sales Territory Reports User)
- Collections Territory Administrator (includes Territory Reports User)
- Partner Management Territory Administrator (includes Territory Reports User)
- Service Contracts Territory Administrator (includes Territory Reports User)
- Service Territory Administrator (includes Territory Reports User)
- Trade Management Territory Administrator (includes Territory Reports User)

Having a separate role for each usage makes it possible to restrict a user to see and work with territories, matching attributes, and so on within their usage only.

Example

For example, a user with only the Service Territory Administrator role can only manage service territories and will not have access to territories for another usage, such as sales or collections.

Related Topics

Oracle E-Business Suite Security Guide

Planning Your Territories

The planning phase is the most important step in territory setup. Before using Oracle Territory Manager, a territory planning team should be established to analyze the territory setup in your organization. This territory planning process needs enterprise-

wide cooperation and feedback.

> **Note:** Multiple territory revisions in the first months of operation should be expected as your enterprise discovers omitted information or territories that do not work on a day-to-day basis

Based on your business needs, your team needs to determine the following:

- Usage: What applications need territories?
- Transaction Type (which is based on usage): for example, leads, opportunities, service requests, and delinquencies.
- How many territories are needed?
- What transaction matching attributes should be enabled?
- What resources should be attached to the territories?
- What should the territory hierarchy structure be?
- How many winners are allowed? A winner is the territory that receives the transaction or customer.
- How are winning territories determined?
- Will Sales implement named account territories?
- Do you want to implement self service territory deployment?

This list is not all-inclusive and planning factors depend on your business needs.

Perform the following steps to plan your territories. This procedure is usually done in a group with pen and paper.

Steps:

1. Review your existing territories.

   You need the following types of information:

   - What is your usage? In other words, what business applications are you building territories for? For example, Oracle Sales, Oracle TeleSales, Oracle Collections, or Oracle Service.
   - What transactional objects within your chosen usage are you assigning resources to? This is your transaction type. For example, for Sales, it is account, opportunity, or lead.

- How are your territories currently assigned (by state, by industry, by zip code, by account, and so on)?
- Is Sales currently using named account territories, even if being manually maintained?
- What are the names and current territory assignments for your sales or service personnel?
- What are the names of employees in other organizations who receive account, lead, and opportunity information and how is that information accessed and used?
- What are your products and how are they differentiated?

2. Decide what matching attributes you want to use to assign objects to territories.
3. Decide on the hierarchy of territories.
4. Identify any overlapping territories and decide the order in which the application chooses them.

   Rank any overlapping territories from 1 to N to determine the order. A territory with a lower number wins over a territory with a higher number rank within the same level of the hierarchy. In case of a tie, the assignment is made randomly. The Territory Assignment Program for Sales objects assigns the tied object to the territory that was created first.
5. Decide how many territories will win. For example, do you need one territory to win, which can be appropriate for service, or multiple winners, which can be appropriate for sales?
6. Decide if named account territories are needed.
7. Decide if you will use self service deployment for named account or geographic territories.
8. Test the strategy before implementing territories throughout the company and consider any future territory maintenance efforts.
9. Consider future territory maintenance efforts.

   Note: Remember that the first territory setup is not necessarily the one that works best. You achieve optimum territory definition only gradually after much fine-tuning to accommodate user reactions and various interests in your organization.

Sales Territory Planning Example

This territory planning example utilizes best practices for a fictitious company with a typical sales model involving named accounts and geographic territories with overlay sales organizations.

Business World is a large manufacturer of computer equipment selling in the US and Canada. They organize their products into three families: servers, desktops/laptops, and storage. In their direct sales model, Business World has a named account sales force consisting of an account manager working specific accounts, and a telephony sales force working the remaining general pool of customers. A product overlay sales force works with account managers and telesales reps based on what products a customer is interested in. Each account manager or telesales representative works with three overlay specialists, one for each product family. In planning for FY2008, Business World is expected to have the following:

- US and Canada have separate sales forces
- The direct sales forces manage their top 200 key accounts
- The general business telesales forces manage their remaining customer pool
- The overlay sales forces service all opportunities for a product family and associated customers. There are 3 types of product specialists: Server, Desktop/Laptop, and Storage product specialists.

The following table shows the sales model:

Sales territory model

Role	US	Canada	Comments
Account Managers	15 account managers manage 150 large, key customers	5 account managers manage 50 large, key customers	Account manager territories encompass all leads and opportunities associated to a key customer.
Telesales Representatives	6 Telesales representatives in 6 geographic territories	3 Telesales representatives in 3 geographic territories	Telesales representative territories encompass all leads and opportunities associated to a customer.

Role	US	Canada	Comments
Overlay Product Specialists	3 product families, 9 product specialists for each product family and region	3 product specialists. Product specialists cover their mutually exclusive product families for the entire country.	Product specialists service all opportunities for a product family and geographic location and their related customers (they are only allowed to view customer information). Each account manager and telesales representative has a Server, Desktop/Laptop, and Storage product specialist.

Step 1: What business objects are we assigning?

Which business objects (transaction types) are being assigned to each type of resource? Are we defining territories to access customers, leads, or opportunities for account managers, telesales reps, and product specialists? Do you need to provide read access only or update privileges as well? To what territory usage are the business objects associated?

Business World is going to assign customer, lead, and opportunity transaction types for all territories assigned to account managers and telesales reps.

Territories assigned to overlay product specialists will have a transaction type of opportunity because they need update privileges for opportunities. Oracle Sales products (Sales and TeleSales) provide, by default, read only access to customers if a resource is assigned to the sales team of any of the customer's opportunities.

Investigate how to set up Oracle Territory Manager business objects in conjunction with the Oracle Sales data security model. If you need update access to customers, leads, or opportunities in Oracle Sales or Oracle TeleSales, then you will need to assign them in Oracle Territory Manager.

Step 2: What matching attributes should we use?

We enable the following matching attributes:

- CUSTOMER NAME RANGE, to identify the 200 named accounts
- COUNTRY, because this qualifier is used as a first criterion in identifying territory and offers the ability to support "Catch All" territories
- POSTAL CODE, to create geographic territories composed of postal codes. One can use less granular geographic matching attributes such as state or province as well.
- STATE, to create geographic "Catch All" territories for overlay product specialists
- OPPORTUNITY PRODUCT CATEGORY, to create overlay territories for product specialists.

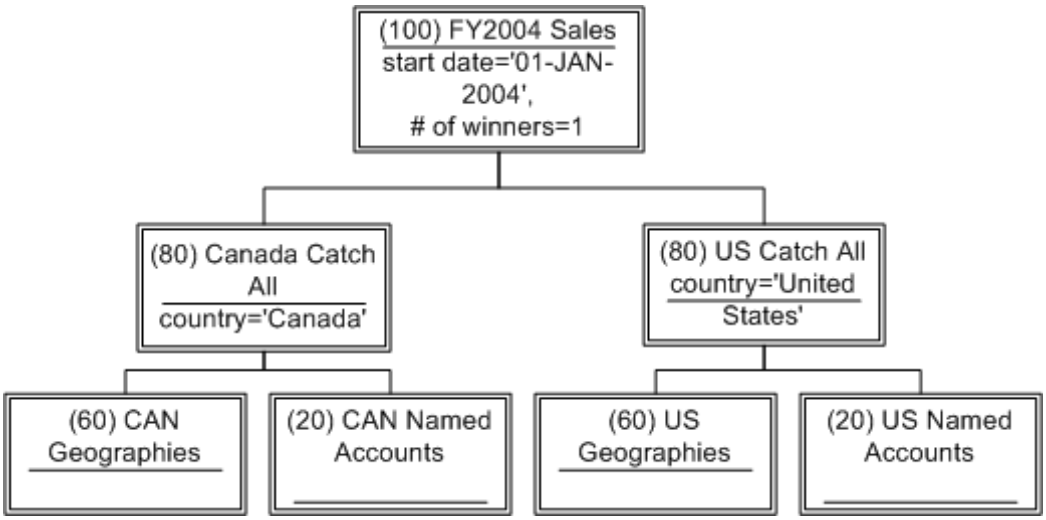
Review Choosing Appropriate Matching Attributes, page 4-17 to analyze your particular situation in greater detail.

Step 3: Territory hierarchy - Define date effectivity and number of winners

The hierarchy is largely designed for ease of maintenance and for the creation of "Catch All" territories. Hierarchies allow you to inherit matching rules and territory properties such as date effectivity and number of winners. Catch alls will be discussed later in Step 4.

For the Sales and TeleSales usage, different territory structures are supported for a business unit by selecting an independent number of winners within the top five levels of the territory hierarchy.

Under the Sales and TeleSales usage, we need to create a top-level territory representing Business World's FY2008 Sales territories. This FY2008 Sales territory will be effective from January 1, 2008 and does not have any transactional matching rules or resources. It is the top of the FY2008 Sales territories and is used to maintain the date effectivity of all territories underneath it and is used to set the number of winning territories (number of winners = 1).



Step 4: Territory hierarchy – Placeholder territories

Sometimes we create territories purely for organization and ease of maintenance. The territories under the US Catch All territory are an example of this.

Within the US and Canada, there are named account sales forces and geographic sales forces. We create two territories under the US Catch All territory:

- US Named Accounts, with no matching attributes or resources
- US Geographies, with no matching attributes or resources

Similarly, Canada has 2 territories called CAN Named Accounts and CAN Geographies underneath the Canada territory.

Step 5: Implement Self Service

Oracle Territory Manager supports the self-service deployment of named accounts and geographic territories. Following are some things to consider before deciding whether or not to utilize this feature:

1. How often do your named account assignments change? If they change frequently, then a self service deployment is recommended.

2. How large is your territory administration staff? What is the required timeliness of activating territory reassignments? How important is it to expose territory ownership to the sales force?

   If representatives have very few named accounts and there is little change, then the self service deployment may not be necessary. However, we believe visibility to clear ownership is a best practice.

3. How many named accounts do you have? At what granularity do you need to define named accounts, for example, by customer name range and country or by customer name range and postal code?

   If you defined your named accounts by customer name range and country and have a manageable number of customers, use the Forms Navigator to create your territories. If you have many named accounts and need to define territories by customer name range and postal code, then use self-service named account territory management.

Self service allows your total cost of ownership to decrease, reduces your time to implementation, and at the same time empowers your sales management with the ability to manage their own territory distribution and midyear adjustments. Steps 6 (Define Catch Alls) to 11 (Rank) are still relevant regardless of your implementation, but can be managed and generated through the self service flows for geographic and named account territories.

Step 6: Territory hierarchy – Define Catch Alls

Business World has separate sales forces for US and Canada so we will create two child territories underneath FY2008 Sales; one called US with a transaction matching rule COUNTRY = 'UNITED STATES' and another called Canada with a transaction matching rule COUNTRY = 'CANADA'. The COUNTRY matching rules will be inherited by all child territories, easing maintenance. These two territories will also serve as "Catch Alls" for exceptions of the assignment process. These are important because customers, leads or opportunities that do not find matching leaf node territories will fall into these "Catch All" territories based on their country matching attribute and can be assigned to a designated owner (typically the territory administrator) for resolution. The territory administrator should have customer, lead, and opportunity access.

Catch Alls are used to "catch" business transactions that fall through the cracks (leaf node territories) and usually are assigned to territory administrators. To be more exact, they "catch" all business transactions that fall into the catchall territory itself or any territory below it in the hierarchy that does not have a resource.

It is recommended that you use the term "Catch All" in the naming of these territories to help visually distinguish this type of territory.

Step 7: How to implement named account territories

From a business perspective, there are various types of territories. Organizations will pull particular customers from the general pool that they deem as critical and assign a specific resource to it. These are termed "named accounts". In an attempt to organize the remaining pool of customers, general business customers are segmented by a simple criteria such as SIC code, state, or area code.

Named account territories may have rules utilizing the CUSTOMER NAME RANGE

matching attribute in combination with a geographic matching attribute such as state or country. For example, if IBM is a named account you define a territory with CUSTOMER NAME RANGE like 'IBM%' and CUSTOMER NAME RANGE like 'International Business Machines%'. This assigns IBM to the account manager regardless of how many customers exist that begin with IBM or International Business Machines.

The US direct sales force consists of 15 account managers, each responsible for a territory of large key accounts. There will be 15 US named account territories as children of the US Named Accounts territory, one for each account manager. There will also be a catch all territory for the US named accounts. Similarly there will be 5 Canadian named account territories as children of the CAN Named Accounts territory plus a catch all territory.

Lowest level territories or leaf node territories should always have resources assigned to them and therefore also require access types to be defined. All account managers are associated to territories with customer, lead, and opportunity access.

See Choosing Appropriate Matching Attributes, page 4-17 for a discussion on the selection of matching attributes for named accounts.

Step 8: How to implement geographic territories

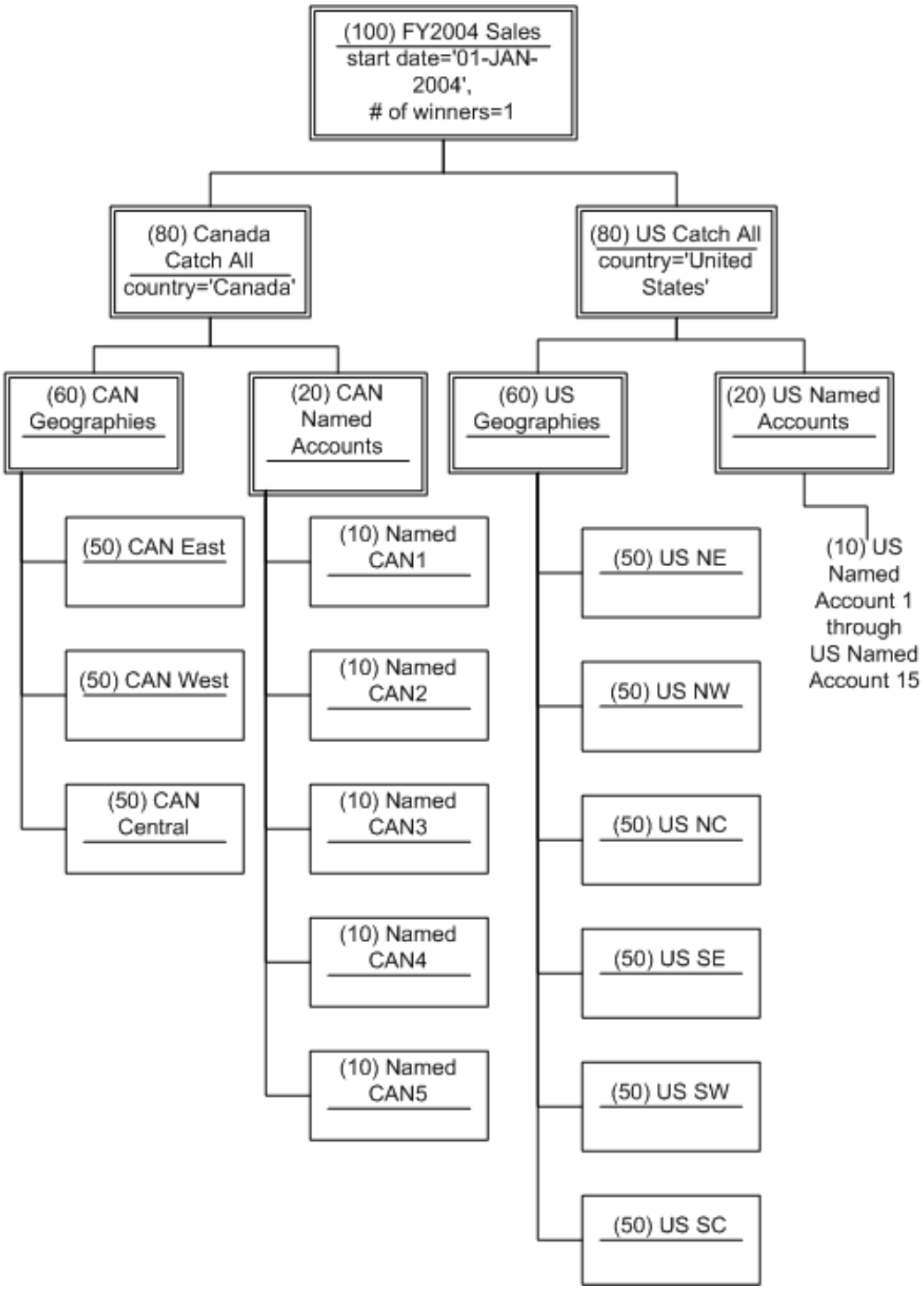
The decision to utilize geographic territories should be based on your business requirements. Following are questions to ask before implementing this type of territory:

1. What granularity is used to distribute your geographies?

2. Do you distribute territories based on geographical requirements such as states, provinces, or postal codes? This will help determine what geographic matching attribute you use.

In our test case the US and Canada each have separate telesales forces, which are responsible for all non-named accounts in a particular geography. The US telesales force has 6 geographic territories as children of the US Geographies territory. The Canada telesales force has 3 geographic territories as children of the CAN Geographies territory. Each geographic territory will have transactional matching rules containing the postal codes that the respective telesales representatives will be responsible for. All telesales representatives will be assigned customer, lead, and opportunity access.

The following chart displays this example.



Step 9: How to support overlays

We recommend that you implement Overlays in a separate territory hierarchy.

The desired business behavior is to find:

1. Either a named account OR a geographic general business territory
2. AND one overlay territory.

Increasing the number of winners in the FY2008 Sales hierarchy and putting the overlay territories underneath it would not accomplish this. However, with the FY2008 Sales hierarchy and number of winners set to one, Territory Manager selects either a named account territory or a general business territory. With a separate FY2008 Overlay hierarchy and number of winners set to one, Territory Manager selects a single overlay territory.

Under the sales usage, you will need to create another top-level territory representing Business World's FY2008 Overlay territories. This FY2008 Overlay territory will be effective from January 1, 2008 and does not have any resources. It does have transaction matching rules to distinguish it as an overlay hierarchy, for example:

OPPORTUNITY PRODUCT CATEGORY= 'SERVER',

OPPORTUNITY PRODUCT CATEGORY= 'DESKTOP',

OPPORTUNITY PRODUCT CATEGORY= 'LAPTOP',

OPPORTUNITY PRODUCT CATEGORY= 'STORAGE'.

It is the top of the FY2008 Overlay territories and is also used to maintain the date effectivity of all territories underneath it and is used to set the number of winning territories. Note the OPPORTUNITY PRODUCT CATEGORY matching attributes are necessary to distinguish these territories from Business World's geographic TeleSales territories.

Product specialists work opportunities, so we are going to assign Opportunity transaction types for all overlay territories.

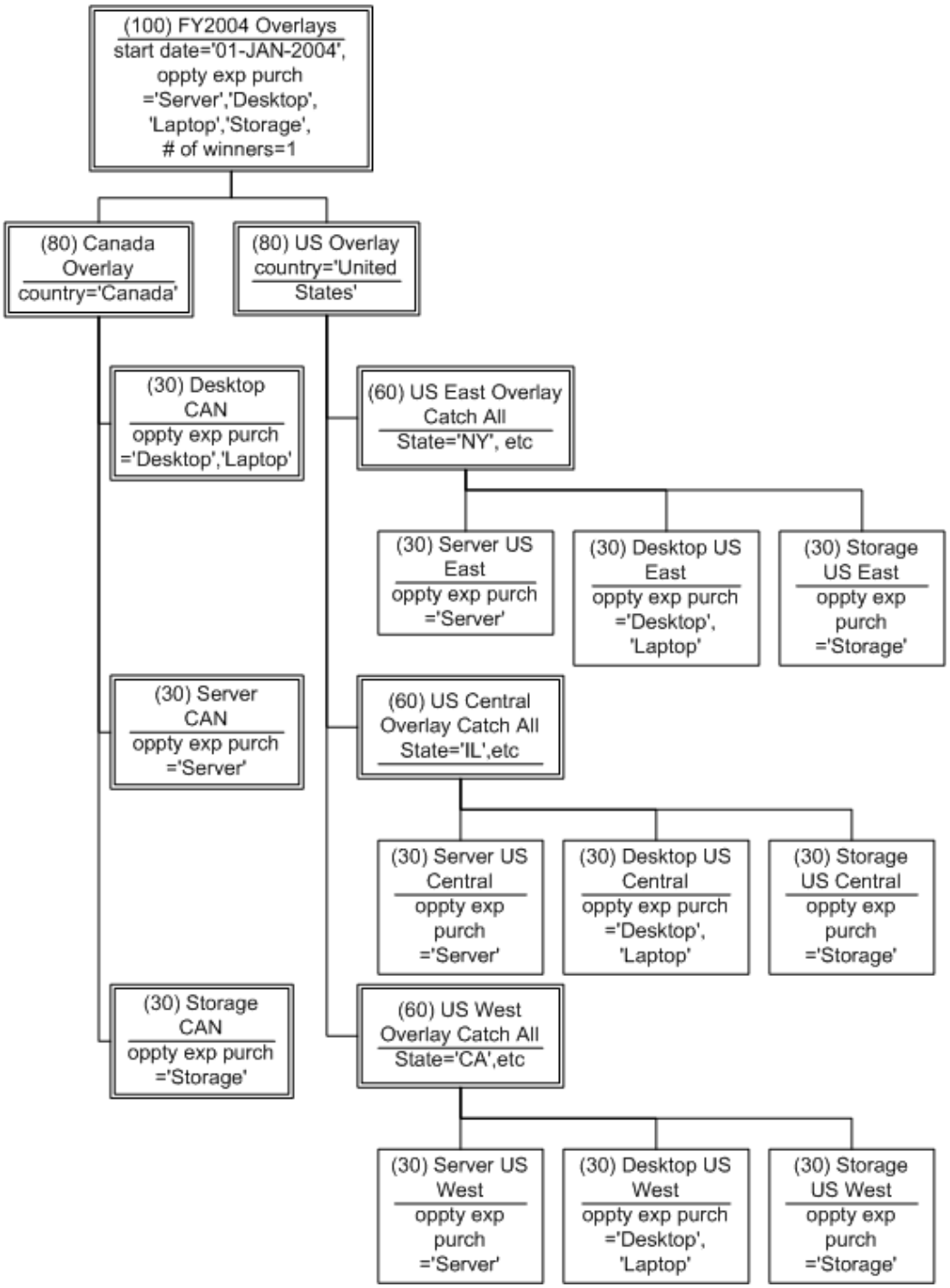
Step 10: What is an appropriate territory hierarchy for overlays?

We have separate sales forces for the US and Canada so we will create two child territories underneath FY2008 Overlay: one called USA Overlay with a transaction matching rule COUNTRY = 'UNITED STATES' and another called Canada Overlay with transaction matching rule COUNTRY = 'CANADA'.

Do you require Catch All territories and how are they organized? If Catch All territories are required, these need to be reflected in the hierarchy.

Is your sales management organized by product family or by geographic area? The territory hierarchy should closely mimic your sales management hierarchy for ease of understanding and maintenance.

Overlay territory hierarchy BY GEOGRAPHY



If Business World's overlay sales force is organized by geography and wants Catch Alls by geography, then we create three territories underneath the US Overlay territory:

- "US East Overlay Catch All", with transactional matching rules STATE = 'NY', STATE = 'NJ', STATE = 'MA', STATE = 'VT', and so on, and resource = Eastern Overlay territory administrator

- "US Central Overlay Catch All", with transactional matching rules STATE = 'IL', STATE = 'OH', STATE = 'AK', STATE = 'WI', and so on, and resource = Central Overlay territory administrator

- "US West Overlay Catch All", with transactional matching rules STATE = 'CA', STATE = 'NV', STATE = 'OR', STATE = 'WA', and so on, and resource = Western Overlay territory administrator

Similarly underneath the Canada Overlay territory, we create three territories:

- "Server CAN", with the transactional matching rule OPPORTUNITY PRODUCT CATEGORY = 'SERVER' and resource = Canadian server specialist

- "Desktop CAN", with the transactional matching rule OPPORTUNITY PRODUCT CATEGORY = 'DESKTOP' and OPPORTUNITY PRODUCT CATEGORY = 'LAPTOP' and resource = Canadian desktop/laptop specialist

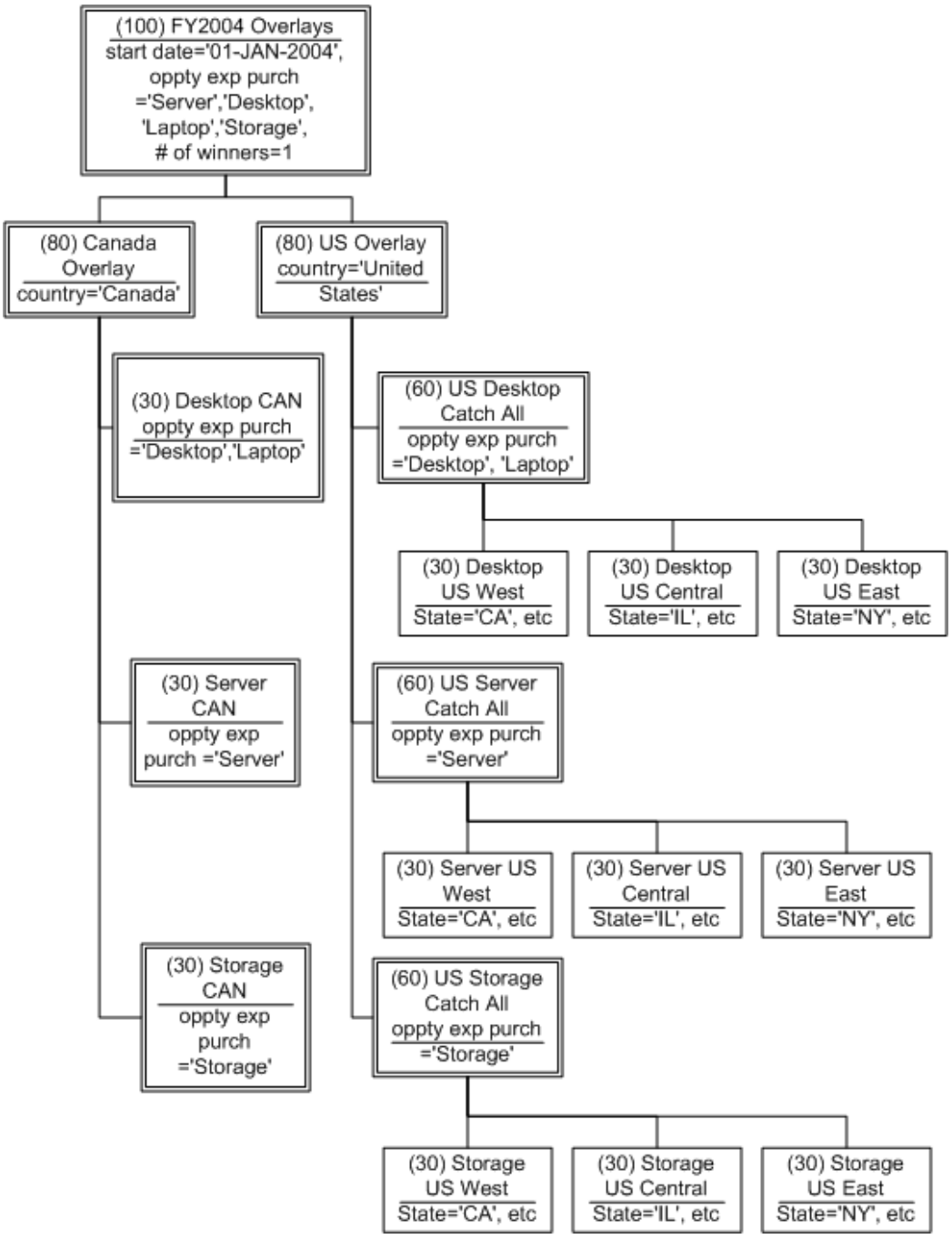
- "Storage CAN", with the transactional matching rule OPPORTUNITY PRODUCT CATEGORY = 'STORAGE' and resource = Canadian storage specialist

As Canada has a smaller number of customers, three product specialists cover the entire country, one for each product family.

The US overlay sales force consists of nine product specialists, each responsible for a territory of product family and geography. As there is no fixed teaming of account managers or telesales representatives to product specialists, we have chosen to implement the overlays separately as children of the country overlay territory. There will be nine US overlay territories as children of the US Overlay territory, one for each product specialist. Each specialist will cover one of three geographies: East, West, or Central.

All product specialists will be associated to territories with customer and opportunity access.

Alternative overlay territory hierarchy BY PRODUCT FAMILY



If Business World's US overlay sales force requires Catch Alls by product family or is organized by product family, we would create three territories underneath the US Overlay territory:

- "US Server Catch All", with the transactional matching rule OPPORTUNITY PRODUCT CATEGORY = 'SERVER' and resource = Server territory administrator

- "US Desktop Catch All", with the transactional matching rule OPPORTUNITY PRODUCT CATEGORY = 'DESKTOP' and OPPORTUNITY PRODUCT CATEGORY = 'LAPTOP' and resource = Desktop territory administrator

- "US Storage Catch All", with the transactional matching rule OPPORTUNITY PRODUCT CATEGORY = 'STORAGE' and resource = Storage territory administrator

The same 9 US overlay territories created in the first example are re-shuffled underneath the appropriate US product family Catch Alls.

Step 11: What rank should each territory have?

To the left of all the territory names in Figures 1, 2 and 3, is the rank of each territory. Named account territories are always ranked higher than geographic territories because a named account in California should fall in a named account territory and not the geographic territory that includes California. Catch Alls are always ranked lower than their associated territories. The rank applies to the territories on the same level in the hierarchy. Territories lower on a hierarchy always win over territories higher in the hierarchy. See Leverage Territory Ranking and Number of Winners, page 4-17 for a detailed discussion of territory rankings.

Step 12: Have you met all your business requirements?

Stand back and review your business requirements for sales territories. Ensure your territory implementation has met all your business requirements. How will you validate your territories are correct? Create and enable your territories on a test environment. Ensure you have an implementation plan in place that involves systematic validation and user acceptance testing. Don't forget about your ongoing maintenance requirements. See Migrating Territories, page 4-19 for a discussion on how to migrate territories from one year to the next.

For any implementation to succeed in the long run, clear and consistent business processes should be implemented to complement your territory setup.

Leverage Territory Hierarchies and Inheritance

Territory hierarchies do more than organize your territories. They also organize your transactional matching rules, critical to improving the ease of maintenance. Child territories always inherit transactional matching rules.

If you are building a set of 100 representative territories exclusive to the US, it is a best practice to introduce a hierarchy layer representing countries which should include the transactional matching rule "Country = United States". In this manner, all the 100 child territories will inherit its transaction matching rules. Instead of maintaining the rule in 100 territories it can be maintained in one parent territory.

These parent territories can also be assigned resources and act as "Catch All" territories in case the customer, lead or opportunity did not match one of the child territories. We have two catch alls in our business world example: one for Canada and the other for the US. Assigning a resource to "Catch All" territories will route customers, leads, and opportunities to the designated resource for resolution. Catch alls are typically assigned to territory administrators.

Leverage Territory Ranking and Number of Winners

The number of winners refers to the number of winning territories allowed. The number of winners can be defined for territories up to five levels down the hierarchy for the Sales and TeleSales usage. It can only be defined at the top level for other usages. Territory rankings work in conjunction with the number of winners = "w" by selecting the top ranked "w" territories within a level of the hierarchy. Territories lower in a hierarchy have an inherent higher ranking compared to territories higher in the hierarchy. When a winning territory is found, all of the resources assigned to it are assigned to the business object.

A good example of this is the difference between named account territories and geographic territories. You don't want to have to maintain the exclusion of named accounts from the geographic territories explicitly. Rather, you would set the number of winners to one and use the customer name range matching attribute in the named account territories and rank these higher than the geographic territories utilizing postal code qualifier. The territory engine finds that both territories match but the number of winners and rankings dictate that the higher ranked named account territory wins.

Choosing Appropriate Matching Attributes

The matching attributes used in defining named account territories are:

- Customer Name Range and Postal Code

  The Customer Name and Postal Code matching attribute identifies organizations through ranges of names or even partial name matches. Unless you have strict data quality management policies in place and there is only one occurrence of each customer, we recommend that you use the Customer Name and Postal Code matching attribute for named accounts.

- DUNS Number

  The DUNS Number qualifier identifies organizations through DUNS number matches.

- Site Number

- Registry ID from TCA

- Party Hierarchy from TCA

These matching attributes identify TCA (Oracle Trading Community Architecture) organizations or customer sites.

Be careful not to confuse SIC code, geographic, and so on territories with named accounts. Many organizations will attempt to implement geographic or SIC code based territories as named accounts because they say they have always done it this way.

Questions to ask:

- How many named accounts does the organization have?

  If sales management claims that named accounts make up more than 20% of all TCA organizations, then they are likely to have incorrectly implemented named accounts.

  For example, a Telecom territory is composed of 50 SIC codes. Implementing a Telecom territory as 20,000 named accounts would require a minimum of 20,000 CUSTOMER NAME RANGE qualifier rules. Clearly, it would be better implemented as a territory with 50 SIC CODE matching rules. There is a diagnostic test within the Oracle Diagnostic Framework for this.

- How many named accounts does a typical resource have?

  If sales management claims their reps have any more than a hundred named accounts, they are likely to have incorrectly implemented a simple territory as named accounts. Investigate how the business derived the set of named accounts. Typically, reps do not have the bandwidth to manage more than 100 named accounts and give them the proper attention associated to critical customers.

- Does your business matching attribute fluctuate in the context of the business object you are assigning? Does it segment customers periodically based on a dynamic business matching attribute?

  It is important to examine the fluctuation of the dynamic matching attribute in the context of the business object you are assigning. For instance in the credit card industry, customers are segmented by their total A/R balance every quarter into three territories (e.g., <$250k, $250k-$500k, >$500k) and customers maintain their segmentation even though their balances change.

  In these cases, we recommend that you designate account classifications based on the dynamic matching attribute periodically and then use the account classification matching attribute in Oracle Territory Manager.

Using named accounts in lieu of geographic territories is an ineffective way to distribute a representative's territory. It is ineffective in terms of application performance, scalability, and ease of maintenance.

Example

20,000 customers are segmented by the number of employees quarterly into three

territories:

- Less than 100 employees
- 100 to 1000 employees
- Greater than 1000 employees

In this case, the territory assignment is quickly performed because there are only three rules.

Suppose that instead of maintaining three matching rules based on the number of employees matching attribute, you are maintaining a minimum of 20,000 customer name range matching rules. Territory assignment performance is directly correlated to the number of matching rules. Territory assignment will be slower with 20,000 matching rules than with three.

Matching Rules

Best practices around matching rules are the simplest to follow because they are very concrete. Matching rules are converted to SQL and are subject to the same performance constraints. The use of the % wildcard as the first character of the matching attribute value prevents the use of indexes.

Migrating Territories

For how long are your territories active? Your sales organization may migrate to new territories yearly or quarterly and you want to use the existing territory hierarchy as a starting baseline. By following best practices, it will be easier to migrate territories to the next period. The trick is to define a territory start date at the top level territory, since all those below it will be gated by the start date property. Copy the top-level territory, which will automatically copy all the child territories as well. Rename the top-level territory and give it a new start date. Don't forget to go back to the old territory hierarchy and end date the top-level territory. You can create territories with a future start date in order to have your territories created and ready to become active.

5

Implementation Tasks

This chapter covers the following topics:

- Setting Up User Security
- Setting Up Multiple Organization Access Control
- If You Have Value Added Tax (VAT)
- Creating Custom Matching Attributes
- Enabling Matching Attributes
- Sales Matching Attributes
- Running Concurrent Programs
- Setting Up Territory Assignment Program
- Setting Up to Use Dun & Bradstreet Data
- Setting Up Territory Alignment
- Proxy User
- Setting Up Export
- Loading Postal Codes

Setting Up User Security

In order for users to access Oracle Territory Manager, you need to assign roles to each user in addition to assigned the Territory Management responsibility. When users log in to Oracle Applications, all menu items associated to their assigned roles appear on their personal home pages. The following roles are available for assignment:

- Sales Team Search User: Provides access to the stand-alone sales team search
- Territory Reports User

- Sales Territory Reports User (includes Territory Reports User)
- Sales Territory User (includes Sales Team Search User)
- Territory Manager Application Administrator
- Sales Territory Administrator (includes Sales Team Search User and Sales Territory Reports User)
- Collections Territory Administrator (includes Territory Reports User)
- Partner Management Territory Administrator (includes Territory Reports User)
- Service Contracts Territory Administrator (includes Territory Reports User)
- Service Territory Administrator (includes Territory Reports User)
- Field Service Territory Administrator (includes Territory Reports User)
- Trade Management Territory Administrator (includes Territory Reports User)

Related Topics

See the *Oracle E-Business Suite Security Guide* for information on how to assign roles to users.

Setting Up Multiple Organization Access Control

When properly set up, users in Oracle Territory Manager can make territory changes in different operating units.

1. Create a Global Security Profile in HRMS using the HRMS Management Responsibility > HRMS Manager > Security > Global Security Profile. Enter a Security Type: Secure organizations by organization hierarchy and/or organization list.
2. Set the *MO: Security Profile* profile option value at the Territory Management responsibility level to the security profile that you just created.
3. Run the *Security List Maintenance* concurrent program. You can run the program to process just your security profile by entering the following parameters:
   - Set Generate Lists for to One Named Security Profile
   - Set Security Profile to the profile you created in step 1.
4. Log into the Oracle Territory Manager application and verify that the multiple

operating units appear in the Operating Unit dropdown list in either the Territory tab, Territory Type tab, or the Enable Matching Attributes page. If they do not appear, try bouncing Apache.

If You Have Value Added Tax (VAT)

If you have VAT and want to use geographic matching attributes for the Sales usage, then you need to set up the geographies so that they appear in the lists of values (LOVs) for territory qualifiers in the Forms Territory Details window.

As the Sales Administrator, navigate to Sales Setup > Lookup Codes and create the lookup types and values for any of the following lookup types you may need:

- TAP_CITY: to add cities
- TAP_STATE: to add states
- TAP_COUNTY: to add counties
- TAP_PROVINCE: to add provinces

Creating Custom Matching Attributes

For the Sales or Collections usages, the system administrator can create custom matching attributes based on:

- Any APPS schema table column attribute
- Third party system table column attribute

To create a custom matching attribute:

1. Obtain script number 1.
2. Modify the variable `p_source_id` if needed. The number in the script is -1001, which is for the Sales usage. If you need to change the usage, query JTF_SOURCES_ALL for the correct number.
3. Modify the variable `p_trans_type_id` to the value for the transaction type your new matching attribute relates to. The value in the script is -1002. You can query JTF_QUAL_TYPES_ALL to determine the correct ID.
4. Run script number 1, which retrieves the transaction type SQL.
5. Save the output of script number 1 to a text file, saved as name.sql. This becomes script number 2.

6. Modify each query from script number 2 to include the attribute you want to use. The queries are:

   - Real Time SQL
   - Batch Total SQL
   - Batch Incremental
   - Batch Date of Effectivity
   - Incremental Reassign

   Make a note of the attribute and its alias. You will need this for script number 3.

7. Near the end of the script, modify values for `p_source_id` and `p_trans_type_id` to the same values as for script number 1.

8. Modify the value for `p_program_name`. You can query JTY_TRANS_USG_PGM_DETAILS for the value.

9. Modify the values for `p_version_name` and `p_enabled_flag`.

10. Run script number 2, which adds the attribute to the transaction type SQL.

11. Obtain script number 3, which creates the custom matching attribute (CMA).

12. Verify that there is not an existing CMA record for the unique matching attribute ID that you will use for the new CMA.

13. Follow the instructions in the script. Make sure you give a unique qualifier ID for each attribute you create. The ID numbers usually start from 9000. The ID in the script is -9010. Give your new attribute a name and description using the variables `p_name` and `p_description`.

14. Modify values for `p_source_id` and `p_trans_type_id` to the same values as for scripts 1 and 2.

15. Specify which columns to use in the values table to store the values for your custom matching attribute.

16. Enter the alias from step 6 for `p_qual_coll`.

17. Run script number 3.

18. Enable your new matching attribute.

19. Associate the matching attribute with a territory type.

Restrictions

If you create a custom matching attribute for the Account transaction, then you must also add the same custom matching attribute to the other Sales transactions.

The values table stores the values for your custom matching attributes. You can query JTF_TERR_VALUES_ALL to see existing the display type and corresponding columns for existing matching attributes. The following table lists the available columns and display types for storing your custom matching attributes.

DISPLAY_TYPE	CONVERT_TO_ID_FLAG	COLUMNS
CHAR	Y	low_value_char_id
CHAR	N	low_value_char, high_value_char
CHAR_2IDS		value1_id, value2_id
COMPETENCE		not used
CURRENCY		low_value_number, high_value_number, currency_code
DEP_2FIELDS		value1_id, value2_id
DEP_2FIELDS_CHAR_2IDS		value1_id, value2_id, value3_id
DEP_3FIELDS_CHAR_3IDS		value1_id, value2_id, value3_id, value4_id
INTEREST_TYPE		interest_type_id, primary_interest_code_id, secondary_interest_code_id
NUMERIC		low_value_number, high_value_number

Related Topics

Enabling Matching Attributes, page 5-6

Example of a Custom Matching Attribute for Account: Party Type, page A-1

Example of Custom Matching Attribute for Lead: Lead Status, page A-16

Example of Custom Matching Attribute for Quote: Quote Source, page A-37

Example of Custom Matching Attribute for Proposal: Proposal Status, page A-52

Example of Custom Matching Attribute for Opportunity: Budget Status, page A-61

Example of Custom Matching Attribute for Oracle Collections, Customer: Customer Party Type, page A-83

Script 1, page A-96

Enabling Matching Attributes

Matching attributes are seeded criteria used to identify territories. For example, country is a geographic matching attribute. Oracle Territory Manager has seeded matching attributes for the following usages and transaction types:

- Collections: customer
- Partner Management: partner
- Sales and TeleSales: account, lead, opportunity, proposal, quote
- Service: service request, service request and task, task
- Service Contracts: contract renewal
- Trade Management: claim, offer

In addition, you can create your own custom matching attribute through a public API. Before you can assign a matching attribute to a territory type, you need to enable it.

Role: Territory Manager Application Administrator

Restrictions

The *Service Request and Task* transaction type means tasks are created through a service request. If it is a stand-alone task, use the *Task* transaction type instead.

You cannot disable a matching attribute that is part of a territory type definition.

Sales Matching Attributes

The following table lists the transaction matching attributes used by Oracle Sales products and their respective uses. The Account transaction type is also used by Oracle Incentive Compensation.

Sales Matching Attributes

Transaction Type	Territory Matching Attribute	Sales/TeleSales Attribute
Account	Account Classification	Interest of "party site"
Account	Account Hierarchy	"Subsidiary Of" a particular organization
Account	Area Code	Area Code
Account	City	City
Account	City	City
Account	Country	Country
Account	Company Annual Revenue	Annual Revenue
Account	Customer Category	Customer Category
Account	Customer Name	Customer Name
Account	Customer Name Range	Customer Name
Account	DUNS Number	D-U-N-S Number
Account	Number of Employees	Total Employees
Account	Site Number	Party Site Num
Account	Postal Code	Postal Code
Account	Product Hierarchy	Product Category
Account	Province	Province
Account	Registry ID	Party Number or Registry ID
Account	SIC Code	SIC Code

Transaction Type	Territory Matching Attribute	Sales/TeleSales Attribute
Account	Sales Partner Of (This attribute is available only for Oracle Incentive Compensation)	Partner Of
Account	State	State
Lead	Budget Amount	Budget
Lead	Lead Product Category	Product Category
Lead	Lead Inventory Item	Inventory Item
Lead	Lead Source	Source Name
Lead	Purchase Amount	Amount
Lead	Sales Channel	
Opportunity	Opportunity Channel	Sales Channel
Opportunity	Opportunity Classification	Classification
Opportunity	Opportunity Product Category	Product Category
Opportunity	Opportunity Inventory Item	Inventory Item
Opportunity	Opportunity Status	Status
Opportunity	Total Amount	Total
Quoting	Product Category	

Running Concurrent Programs

It is important that the territory administrator runs the concurrent programs regularly in order to generate the territories and reflect all changes made.

- **Synchronize Territory Assignment Rules (STAR):** It builds the territory

assignment rules, builds the API that returns the winning territories and resources attached to the winning territories, which are defined in territory setup, generates all self-service territories.

- **Calculate Territory Alignment Metrics:** Calculates information such as DNB annual revenue for named accounts for the time period specified in the system profile options *Territory Alignment Metric Calculation From Date* and *Territory Alignment Metric Calculation To Date.*

- **Named Account Territory Post Processing:**
  - Kicks off named account catchall workflow processing.
  - Refreshes territory administration portals for Sales.
  - Refreshes the named account and geography distribution information for the Sales User responsibility.

Batch Size

The profile Batch Size for Territory Concurrent Programs commits the Generate Territory Concurrent (GTP) program transaction at regular intervals instead of committing the program after all the transactions are done. GTP populates all the territory details in denormalized form in the territory tables. Instead of committing after forming the denormalized territory details for all the territories, GTP commits after reaching the number of transactions mentioned in profile Batch Size for Territory Concurrent Programs. Doing so improves the performance of GTP. If this profile is not set, the default value for this profile is 50000.

Parameters

STAR uses the following parameters:

- Usage: Select the correct usage from the LOV, such as *Oracle Sales and TeleSales* or *Oracle Service.*
- Run Mode: Set to **Total Refresh** the first time you run the concurrent program. Subsequently, set to **Incremental Refresh** to process only the changes in named account and geographic territories, which saves processing time. Set to **Date Effective** to assign resources for a specified date range.
- Start Date
- End Date
- Debug Flag: Yes to write debug messages.

Setting Up Territory Assignment Program

If you are implementing sales territories, then you need to set up the Territory Assignment Program per the steps provided in either the *Oracle Sales Implementation Guide* and the *Oracle TeleSales Implementation Guide.*

Setting Up to Use Dun & Bradstreet Data

If you want to use Dun & Bradstreet data for your named accounts, or use the DUNS number qualifier, then you need to purchase the data. See the *Oracle Trading Community Architecture User Guide* for more information.

Setting Up Territory Alignment

The metrics DNB Number of Employees and DNB Annual Revenue, used for comparing alignments, depend upon information provided by Trading Community Architecture.

Login

Log in to the Personal Home Page.

Responsibility

Trading Community Manager

Navigation

Trading Community : Administration : Enrichment : Third Party Data Rules

Prerequisites

- ❒ Use the System Administrator responsibility to assign a user the Trading Community Manager responsibility.
- ❒ Purchase Dun & Bradstreet information. See the *Oracle Trading Community Architecture User Guide* for more information. Do not import the data into TCA until after performing this procedure.

Steps:

1. Click **Party Profile Entities**.
2. In the Organization Profile Entity region, click **View Attributes.**

   The View Attributes for Organization Profile Entity page displays a list of attributes.

3. Select the attribute Employees at Primary Address.

4. Click **Update Individually.**

   The Select and Rank Data Sources for Attributes: Employees at Primary Address page appears.

5. Move the data source Dun & Bradstreet from the Select Data Sources box to the Rank Data Sources box and move it to the top of the box ahead of User Entered.

6. Click **Finish.**

7. Using the System Administrator responsibility, set the following two profile options to set the date range for metric calculations. Profile category is Territory Administration.

   - Territory Alignment Metric Calculation From Date (mm/dd/yyyy)
   - Territory Alignment Metric Calculation To Date (mm/dd/yyyy)

8. Using the CRM Administrator responsibility, run the concurrent program *Calculate Territory Alignment Metrics.*

   This concurrent program should be run whenever there are changes in the Dun & Bradstreet information, such as number of employees or annual revenue; changes in the desired date range; and when new named accounts are created.

Metric information is now calculated and available to use for territory alignments.

Proxy User

A user who is a member of a sales group and who is assigned the proxy user role can assign the named accounts or geographic territories owned by the manager of that sales group to any member of the sales group hierarchy that reports up to that manager. This role is typically assigned to a sales administrator or sales operations who can then assign accounts in the sales manager's organization.

Using Resource Manager, create a special role for this function of the type Sales and select the Administrator or Member flag.

Then assign the newly created role to a member of the sales group to be the proxy user.

Next, enter the role in the system profile option *Self Service Named Account Proxy User Role.* The category for this profile is Territory Administration.

When the proxy user logs in, he will see the same list of named accounts and geographic territories as the manager of the sales group that he is the proxy user for.

Setting Up Export

Territory creation requires the ability to export to and upload from spreadsheet. To use the export to Microsoft Excel and upload from spreadsheet features you need to do the following:

1. Install the Oracle Web Applications Desktop Integrator.
2. Enable *Initialize and script Active X controls not marked as safe* in your browser.
3. In your Microsoft Excel, set the macro security to low: Tools > Macro > Security.

Loading Postal Codes

You need to load geographic data into the table that is used by self-service geography. This section covers the public APIs you use to do so, and provides some sample SQL code. You must purchase your geography data, for example from the United States Postal Service. It needs to contain city, state or province, country, and postal code.

Table

The name of the table that contains the geographic data is JTF_TTY_GEOGRAPHIES.

Supported Geography Types

The types of geography supported in Oracle Territory Manager are:

- Country (COUNTRY)
- State (STATE)
- City (CITY)
- Postal Code (POSTAL_CODE)

APIs

The PL/SQL package JTF_TTY_GEOSOURCE_PUB contains three APIs:

- CREATE_GEO
- DELETE_GEO
- UPDATE_GEO

CREATE_GEO API

This API creates a geography. This is the API you will use to initially populate geographic data into the JTF_TTY_GEOGRAPHIES table.

Procedure Specification:

```
Create_geo(
p_geo_type IN VARCHAR2,
p_geo_name IN VARCHAR2,
p_geo_code IN VARCHAR2,
p_country_code IN VARCHAR2,
p_state_code IN VARCHAR2 default null,
p_province_code IN VARCHAR2 default null,
p_county_code IN VARCHAR2 default null,
p_city_code IN VARCHAR2 default null,
p_postal_code IN VARCHAR2 default null,
x_return_status IN OUT NOCOPY VARCHAR2,
x_error_msg IN OUT NOCOPY VARCHAR2)
```

Example

```
DECLARE

 err_status VARCHAR2(80);
 err_msg VARCHAR2(255);

BEGIN

-- to populate country with 'US'

JTF_TTY_GEOSOURCE_PUB.create_geo(p_geo_type => 'COUNTRY',
                     p_geo_name => 'United States',
                     p_geo_code => 'US',
                     p_country_code => 'US',
                     p_state_code => null,
                     p_province_code => null,
                     p_county_code => null,
                     p_city_code => null,
                     p_postal_code => null,
                     x_return_status => err_status,
                     x_error_msg => err_msg);

-- to populate state with 'California'

JTF_TTY_GEOSOURCE_PUB.create_geo(p_geo_type => 'STATE',
                     p_geo_name => 'California',
                     p_geo_code => 'CA',
                     p_country_code => 'US',
                     p_state_code => 'CA',
                     p_province_code => null,
                     p_county_code => null,
                     p_city_code => null,
                     p_postal_code => null,
                     x_return_status => err_status,
                     x_error_msg => err_msg);

-- to populate city with 'San Francisco'

JTF_TTY_GEOSOURCE_PUB.create_geo(p_geo_type => 'CITY',
                     p_geo_name => 'San Francisco',
                     p_geo_code => 'SAN_FRANCISCO',
                     p_country_code => 'US',
                     p_state_code => 'CA',
                     p_province_code => null,
                     p_county_code => null,
                     p_city_code => 'SAN_FRANCISCO',
                     p_postal_code => null,
                     x_return_status => err_status,
                     x_error_msg => err_msg);

-- to populate postal code with '94065'

JTF_TTY_GEOSOURCE_PUB.create_geo(p_geo_type => 'POSTAL_CODE',
                     p_geo_name => '94065',
                     p_geo_code => '94065',
                     p_country_code => 'US',
                     p_state_code => 'CA",
                     p_province_code => null,
                     p_county_code => null,
                     p_city_code => 'SAN_FRANCISCO',
                     p_postal_code => '94065',
                     x_return_status => err_status,
                     x_error_msg => err_msg);

COMMIT;
```

```
END;
/
```

DELETE_GEO API

This API deletes a geography. This is the API you will use to delete geographic data into the JTF_TTY_GEOGRAPHIES table.

Procedure Specification:

```
delete_geo(
p_geo_type IN VARCHAR2,
p_geo_code IN VARCHAR2,
p_country_code IN VARCHAR2,
p_state_code IN VARCHAR2 default null,
p_province_code IN VARCHAR2 default null,
p_county_code IN VARCHAR2 default null,
p_city_code IN VARCHAR2 default null,
p_postal_code IN VARCHAR2 default null,
p_delete_cascade_flag IN VARCHAR2 default 'N',
x_return_status IN OUT NOCOPY VARCHAR2,
x_error_msg IN OUT NOCOPY VARCHAR2)
```

Example to delete state of 'Texas'

```
JTF_TTY_GEOSOURCE_PUB.delete_geo(p_geo_type => 'STATE',
                    p_geo_code => 'TX',
                    p_country_code => 'US',
                    p_state_code => 'TX',
                    p_province_code => null,
                    p_county_code => null,
                    p_city_code => null,
                    p_postal_code=> null,
                    p_delete_cascade_flag => 'N',
                    x_return_status => err_status,
                    x_error_msg => err_msg);
```

UPDATE_GEO API

This API updates a geography. This is the API you will use to edit geographic data into the JTF_TTY_GEOGRAPHIES table.

Procedure Specification:

```
update_geo(
    p_geo_id IN VARCHAR2,
    p_geo_name IN VARCHAR2,
    x_return_status IN OUT NOCOPY VARCHAR2,
    x_error_msg IN OUT NOCOPY VARCHAR2)
```

Example to update country name

```
 JTF_TTY_GEOSOURCE_PUB.update_geo(p_geo_id => 249203,
                    p_geo_name => 'CANADA',
                    x_return_status => err_status,
                    x_error_msg => err_msg);
```

6

Verify and Troubleshoot

This chapter covers the following topics:

- Verification Tasks
- Profile Options
- Prerequisites for Using Web ADI
- Troubleshooting Export to Excel
- Party, Party Site, and Account Merge
- Tips for Fine-tuning Territory Assignment Performance
- If the Synchronize Territory Assignment Rules Fails
- Frequently Asked Questions (FAQs) about Transaction Matching Attributes

Verification Tasks

Verify your implementation by performing one or more of the following tasks:

1. Create a territory for your usage and run the concurrent program.
2. Create a transaction for your usage.
3. Verify that the transaction is assigned to the correct resources.

Profile Options

The following profile options are in the profile category Territory Administration. These system profile options are discussed in the preceding chapter.

- Territory Alignment Metric Calculation From Date (mm/dd/yyyy)
- Territory Alignment Metric Calculation To Date (mm/dd/yyyy)

The From and To dates specify the time period for information to be calculated by concurrent programs for metric calculation used by territory alignment.

- Self Service Named Account Proxy User Role

  When the role specified in this profile option is assigned to a user, that user becomes a proxy user and is able to view and manage self-service geographic and named account territories controlled by the manager of the sales group for whom the user is a proxy user.

Prerequisites for Using Web ADI

The following are the prerequisites for Web ADI:

- Client PC with Windows ME, Windows NT 4.0 (with Service Pack 3 or later), Windows 2000, Windows XP, or Windows 98 installed on it
- Internet Explorer 5.0 or greater installed on the client PC
- Microsoft Excel 97, 2000, 2003, or XP installed on the client PC. For Web ADI to work with Microsoft Excel XP/2003, you must change the macro settings for Excel XP/2003. To do so:

1. In Excel, go to Tools > Macro > Security > Trusted Sources.
2. Select the "Trust access to Visual Basic Project" option.

- To allow spreadsheets to be created on your desktop, change your intranet browser security settings as follows:

1. Navigate to Tools > Internet Options > Security > Custom Level.
2. Set the "Initialize and script ActiveX controls not marked as safe" option to "Prompt".

   Note: Web ADI is not certified on Mac operating system with Safari browser.

- Refer to the Web ADI documentation for more information.
- To export using Web ADI
  1. Navigate to Tools > Internet Options > Security > Trusted Sites > Sites
  2. Click the Add button to add the server name to the list of trusted sites

3. Deselect the Require Server verification (https) for all the sites in this zone check box.

Troubleshooting Export to Excel

Please refer to the Oracle Web Applications Desktop Integrator User Guide for more information when you troubleshoot your export to Excel. For the very latest updates to troubleshooting information for Web ADI, refer to related links in the release notes for Oracle Territory Manager. Following are some troubleshooting notes:

- If images are not displaying properly in the Web ADI page flow, then check the following profile options :
  - BNE UIX Base Path: Default value is "/OA_HTML/cabo"
  - BNE UIX Physical Directory: This should be set to the physical directory that is mapped to the value being used for BNE UIX Base Path. The default is derived by looking up the Java system variable OA_HTML in jserv.properties. If OA_HTML cannot be found, then a search is done for MEDIA_DIRECTORY. Once the path of either of these is found, it is appended with "/cabo/".
- If you are getting an "Unexpected error" during download, then make sure that the log file has the write permission. The log file name and directory are specified by the following profile options:
  - BNE Server Log Filename: default value bne.log
  - BNE ServerLog Path: default is $FND_TOP/log
- Oracle Web ADI provides a diagnostic tool named BNETEST to confirm the configuration of Web ADI. To run this tool, download the patch for the diagnostic tool that matches the version of Web ADI you have installed and enter the following URL: `http://:/servlets/BNETEST`

Party, Party Site, and Account Merge

When you merge parties in Oracle Trading Community Architecture, it affects territories in the following ways:

- Named party merged into named party

  The merge-from named account and the associated sales team are deleted if they belong to the same territory. If the merge-from named account belongs to a separate territory, then the merge-from named account is deleted but the associated sales team becomes part of the merge-to named account, and the merge-to named account now belongs to all of the separate territories.

- Non-named party merged into named party

  The merge-from party is deleted with no impact on territories.

- Named party merged into non-named party

  The non-named party replaces the merge-from named party in the existing territory, and the territory and the sales team of the merge-from named party is retained.

Party Sites

If the party or party site records are identified as duplicates, then the territory definition values that are based on parties are transferred.

If both the merge-from party site and the merge-to party site are named accounts in Oracle Territory Manager, then:

- If both the named accounts belong to the same parent territory, then the merge-from named account and the associated sales team are deleted.

- If the named accounts belong to different parent territories, then the merge-to named account is added to the parent territory of the merge-from named account with the sales team of the merge-from named account. The merge-from named account is then deleted.

If the merge-from party site was set up as a named account in Oracle Territory Manager and the merge-to party site was not a named account in Oracle Territory Manager, then:

- If the merge-from named account belongs to a territory whose matching rule is either DUNS or REGISTRY ID, then:

  - If the party of the merge-to party site already exists as a named account in the territory, then the merge-from named account and the associated sales team are deleted.

  - If the party of the merge-to party site does not exist as a named account in the territory, then the merge-to party site will replace the merge-from party site as a named account.

- If the merge-from named account belongs to a territory whose matching rule is neither DUNS nor REGISTRY ID, then the merge-to party site replaces the merge-from party site as a named account.

Tips for Fine-tuning Territory Assignment Performance

Comparison Operators

Avoid using "<>" operator.

Additional Tips

The following tips can be useful.

- Set up territories in hierarchical fashion for easy maintenance.
- Make the territories as generic as possible.
- If you create a territory and do not assign resources to it, then the Territory Manager does not return this territory as a winning territory.
- You can create your own module specific "Catch All."

If the Synchronize Territory Assignment Rules Fails

If you run the Synchronize Territory Assignment Rules concurrent program and it errors out, run it again with the Debug and Trace flags set to Yes. Then review the log file that results. The log file gives clear information on exactly what step the error occurred at in the program and what the error is.

Frequently Asked Questions (FAQs) about Transaction Matching Attributes

The following are frequently asked questions about transaction matching attributes. Answers to these questions may help you in troubleshooting problems with Territory Manager.

How to Find Possible Transaction Matching Attribute Values?

Answer: In the Define Matching Attributes step, you must enter at least three letters in order to bring up the LOV in the Value From and Value To fields. For example, enter "Bus" to launch the LOV starting with "Bus". If wildcard "%" is used, then use it with letters, such as "B%%", or "%%%" to query the list of values.

What is Customer Name Range Transaction Matching Attribute?

Answer: Customer Name Range is used to group similar customer names. It allows you

to define access based on pre-defined customer names.

What Is the Difference Between the Following Service Transaction Types?

Service Request: This transaction type allows you to define access for account and service request related transaction qualifiers. It limits the transaction access to a service request only.

For example, a territory is defined with appropriate rank information and has Service Request transaction type identified. That territory can be a winning territory among other competitive territories only if it is for a service request transaction, but not a task or a task associated with a service request transaction.

Task: This type allows you to define access for account and task related transaction qualifiers. It limits the transaction access to a task only, not a task created within a service request.

Service Request and Task: This type allows you to define access for account, task and service request related transaction qualifiers. However, it limits the transaction access to a task associated with a service request only, not a task or a service request.

For example, a territory is created with "Service Request" and "Task" transaction types. When a transaction is lunched through a task within a service request, that territory will not be selected as a winning territory even the qualifier and its value is matched to the transaction (a task associated with a service request).

A

Creating Custom Matching Attributes

This appendix covers the following topics:

- CMA for Account: Party Type
- CMA for Lead: Lead Status
- CMA for Quote: Quote Source
- CMA for Proposal: Proposal Status
- CMA for Opportunity: Budget Status
- CMA for Oracle Collections, Customer: Customer Party Type
- Script 1

CMA for Account: Party Type

This is an example of creating a custom matching attribute for Sales using the account transaction and the party type attribute.

Modify Script 1

Modify Script 1, page A-96 to retrieve the relevant account transaction type SQL meta-data.

1. Set the source ID corresponding to the usage (Sales or Collections). The source ID -1001 corresponds to the Sales usage. To find the desired usage ID query JTF_SOURES_ALL.

   ```
   p_source_id      NUMBER := -1001;
   ```

2. Set the transaction type ID. The ID -1002 corresponds to the Account transaction type ID. To find the desired transaction type ID, query JTF_QUAL_TYPES_ALL

   ```
   p_trans_type_id  NUMBER := -1002;
   ```

3. Run the script and save the output to a file. Running this command will save the

Account transaction type SQL meta-data to the "R12AccountTT.sql" file.

```
SQL>spool <output filename - e.g., "d:\R12AccountTT.sql">
SQL> set serveroutput on size 999999
SQL> @Script1.sql /
```

Modify Transaction SQL Meta-Data

Modify the transaction type SQL meta-data (R12AccountTT.sql) to get the CMA's corresponding transaction attribute value.

1. Add the party_type attribute to each of the following SQLs:
    - l_real_time_sql :=

```
 SELECT
   party.party_id    trans_object_id,
   null     trans_detail_object_id,
   TO_CHAR(NULL)                                        city,
   TO_CHAR(NULL)
postal_code,
   TO_CHAR(NULL)                                         state,
   TO_CHAR(NULL)                                         province,

   TO_CHAR(NULL)                                         county,
   TO_CHAR(NULL)                                         country,
   party.party_id
party_id,
   TO_NUMBER(NULL)                                    party_site_id,

   upper(party.primary_phone_area_code)       area_code,
   upper(party.party_name)
comp_name_range,
   party.party_id                                        partner_id,

   party.employees_total
num_of_employees,
   upper(party.category_code)
category_code,
   party.party_id
party_relationship_id,
   upper(party.sic_code_type||  :   ||party.sic_code)  sic_code,

   orgp.CURR_FY_POTENTIAL_REVENUE                   squal_num06,

   upper(orgp.PREF_FUNCTIONAL_CURRENCY)        car_currency_code,

   upper(party.duns_number_c)
squal_char11,
   l_txn_date                                            txn_date

-- Party Type CMA
  , UPPER(party.party_type)
Q9010_PARTY_TYPE
--

 FROM
   HZ_PARTIES party,
   HZ_ORGANIZATION_PROFILES ORGP
 WHERE party.party_type IN (''PERSON'',''ORGANIZATION'')
 AND   party.status=''A''
 AND   party.party_id = orgp.party_id(+)
AND   nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate
AND   party.party_id = l_trans_object_id1 UNION ALL
SELECT
  party.party_id
trans_object_id,
  addr.party_site_id
trans_detail_object_id,
  upper(LOC.city)                                       city,
  upper(LOC.postal_code)                                postal_code,

  upper(LOC.state)                                     state,
  upper(LOC.province)                                   province,
  upper(LOC.county)                                     county,
  upper(LOC.country)                                    country,
  party.party_id                                        party_id,
  addr.party_site_id
party_site_id,
  upper(phone.phone_area_code)                         area_code,
```

```
upper(party.party_name)
comp_name_range,
  party.party_id                                        partner_id,

  party.employees_total
num_of_employees,
  upper(party.category_code)
category_code,
  party.party_id
party_relationship_id,
  upper(party.sic_code_type||'': ''||party.sic_code) sic_code,
  orgp.CURR_FY_POTENTIAL_REVENUE                      squal_num06,
  upper(orgp.PREF_FUNCTIONAL_CURRENCY)        car_currency_code,

  upper(party.duns_number_c)
squal_char11,
  l_txn_date                                            txn_date

-- Party Type CMA
 , UPPER(party.party_type)
Q9010_PARTY_TYPE
--

FROM
  HZ_CONTACT_POINTS phone,
  HZ_PARTIES party,
  HZ_PARTY_SITES addr,
  HZ_LOCATIONS LOC,
  HZ_ORGANIZATION_PROFILES ORGP
WHERE  phone.owner_table_name(+) = ''HZ_PARTY_SITES''
AND    phone.primary_flag(+) = ''Y''
AND    phone.status(+) = ''A''
AND    phone.contact_point_type (+) = ''PHONE''
AND    addr.party_site_id = phone.owner_table_id(+)
AND    addr.party_id=party.party_id
AND    party.party_id = orgp.party_id(+)
AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate
AND    party.party_type IN (''PERSON'',''ORGANIZATION'')
AND    party.status=''A''
AND    LOC.location_id = addr.location_id
AND    addr.status=''A''
AND    party.party_id = l_trans_object_id1
AND    addr.party_site_id = nvl(l_trans_object_id2, addr.
party_site_id) ;
```

- l_batch_total_sql :=

```
select /*+ parallel(ORGPRO) parallel(CNTPNT) parallel(X) use_hash
(ORGPRO CNTPNT X) */
  -1001      source_id,
  -1002     trans_object_type_id,
  X.party_id    trans_object_id,
  X.party_site_id                    trans_detail_object_id,
  x.party_type                              party_type,
  UPPER(X.party_name)   squal_char01,
  UPPER(X.city)    squal_char02,
  UPPER(X.county)    squal_char03,
  UPPER(X.state)    squal_char04,
  UPPER(X.province)                   squal_char05,
  UPPER(X.postal_code)   squal_char06,
  UPPER(X.country)                   squal_char07,
  UPPER(CNTPNT.phone_area_code)    squal_char08,
  UPPER(X.category_code)              squal_char09,
  UPPER(X.sic_code_type||'': ''||X.sic_code)   squal_char10,
  UPPER(X.duns_number_c)               squal_char11,
  UPPER(ORGPRO.pref_functional_currency)    squal_curc01,
  UPPER(X.party_name_substring)  squal_fc01,
  X.party_id    squal_num01,
  X.party_site_id    squal_num02,
  X.party_id    squal_num03,
  X.party_id    squal_num04,
  X.employees_total                   squal_num05,
  ORGPRO.curr_fy_potential_revenue         squal_num06

-- Party Type CMA
, UPPER(X.party_type)
Q9010_PARTY_TYPE
--

from HZ_ORGANIZATION_PROFILES   ORGPRO ,
     HZ_CONTACT_POINTS CNTPNT,
     (select /*+  parallel(PARTY) parallel(SITE) parallel(LOC)
use_hash(SITE) use_hash(PARTY) use_hash(LOC) */
        PARTY.party_type party_type,
        SITE.party_site_id party_site_id,
        Site.party_id party_id,
        LOC.city city,
        LOC.country country,
        LOC.county county,
        LOC.state state,
        LOC.province province,
        LOC.postal_code postal_code,
        PARTY.employees_total employees_total,
        PARTY.sic_code_type sic_code_type,
        PARTY.sic_code sic_code,
        upper(substr(PARTY.party_name,1,1)) party_name_substring,

        upper(PARTY.party_name) party_name,
        PARTY.category_code category_code,
        ''HZ_PARTY_SITES'' owner_table_name,
        SITE.party_site_id owner_table_id,
        PARTY.duns_number_c

-- Party Type CMA
       , party.party_type party_type
--

      from  HZ_PARTY_SITES   SITE,
            HZ_LOCATIONS   LOC,
            HZ_PARTIES   PARTY
      where  SITE.status = ''A''
      and SITE.party_id = PARTY.party_id
```

```
and PARTY.party_type in (''PERSON'', ''ORGANIZATION'')
      and PARTY.status = ''A''
      and LOC.location_id = SITE.location_id
      union all
      select /*+ parallel(PARTY)  */
        PARTY.party_type party_type,
        to_number(NULL) party_site_id ,
        PARTY.party_id party_id ,
        to_char(NULL) city ,
        to_char(NULL) country ,
        to_char(NULL) county ,
        to_char(NULL) state ,
        to_char(NULL) province ,
        to_char(NULL) postal_code ,
        PARTY.employees_total employees_total,
        PARTY.sic_code_type sic_code_type,
        PARTY.sic_code sic_code,
        upper(substr(PARTY.party_name,1,1)) party_name_substring,

        upper(PARTY.party_name) party_name,
        PARTY.category_code category_code,
        ''HZ_PARTIES'' owner_table_name,
        PARTY.party_id owner_table_id,
        PARTY.duns_number_c


-- Party Type CMA
       , party.party_type party_type
--

      from  HZ_PARTIES PARTY'
      where PARTY.party_type in (''PERSON'', ''ORGANIZATION'')
      and PARTY.status = ''A''
      ) X
where CNTPNT.owner_table_name(+) = X.owner_table_name
and CNTPNT.owner_table_id(+) = X.owner_table_id
and CNTPNT.status(+)=''A''
and CNTPNT.primary_flag(+)=''Y''
and CNTPNT.contact_point_type(+)=''PHONE''
and ORGPRO.party_id(+) = X.party_id
and nvl(ORGPRO.effective_end_date(+),sysdate+1) > sysdate ;
```

- l_batch_incr_sql :=

```
 select /*+ use_nl(ORGPRO CNTPNT) */
  -1001 source_id,
  -1002 trans_object_type_id,
  X.party_id trans_object_id,
  X.party_site_id trans_detail_object_id,
  x.party_type party_type,
  X.party_id squal_num01,
  X.party_site_id squal_num02,
  X.party_id squal_num03,
  X.party_id squal_num04,
  X.employees_total squal_num05,
  ORGPRO.curr_fy_potential_revenue squal_num06,
  UPPER(ORGPRO.pref_functional_currency) squal_curc01,
  UPPER(X.party_name_substring) squal_fc01,
  UPPER(X.party_name) squal_char01,
  UPPER(X.city) squal_char02,
  UPPER(X.county) squal_char03,
  UPPER(X.state) squal_char04,
  UPPER(X.province) squal_char05,
  UPPER(X.postal_code) squal_char06,
  UPPER(X.country) squal_char07,
  UPPER(CNTPNT.phone_area_code) squal_char08,
  UPPER(X.category_code) squal_char09,
  UPPER(X.sic_code_type||'': ''||X.sic_code) squal_char10,
  UPPER(X.duns_number_c) squal_char11

-- Party Type CMA
 , UPPER(X.party_type)
Q9010_PARTY_TYPE
--

from HZ_ORGANIZATION_PROFILES   ORGPRO ,
     HZ_CONTACT_POINTS CNTPNT,
     (select   /*+ leading(y) */
        Z.party_site_id,        Z.party_id,          z.
party_type, Z.city,
        Z.country,              Z.county,            Z.state,
        Z.province,             Z.postal_code,       Z.
employees_total,
        Z.sic_code_type,        Z.sic_code,          Z.
party_name_substring,
        Z.party_name,           Z.category_code,     Z.
owner_table_name,
        Z.owner_table_id,       Z.duns_number_c
       from (select  /*+ no_merge */ distinct customer_id
             from AS_CHANGED_ACCOUNTS_ALL CHGACC
             where lead_id is null and sales_lead_id is null
       and CHGACC.request_id = l_REQUEST_ID) Y,
            (select SITE.party_site_id party_site_id,
                    SITE.party_id party_id,
                    PARTY.PARTY_TYPE party_type,
                    LOC.city city,
                    LOC.country country,
                    LOC.county county,
                    LOC.state state,
                    LOC.province province,
                    LOC.postal_code postal_code,
                    PARTY.employees_total employees_total,
                    PARTY.sic_code_type sic_code_type,
                    PARTY.sic_code sic_code,
                    upper(substr(PARTY.party_name,1,1))
party_name_substring,
                    upper(PARTY.party_name) party_name,
                    PARTY.category_code category_code,
                    ''HZ_PARTY_SITES'' owner_table_name,
```

```
SITE.party_site_id owner_table_id,
                   PARTY.duns_number_c

-- Party Type CMA
       , party.party_type'
--


            from  HZ_PARTY_SITES  SITE,
                  HZ_LOCATIONS  LOC,
                  HZ_PARTIES  PARTY
            where SITE.status = ''A''
            and   SITE.party_id = PARTY.party_id
            and   PARTY.party_type in (''PERSON'',
''ORGANIZATION'')
            and   PARTY.status = ''A''
            and   LOC.location_id = SITE.location_id
union all
            select to_number(NULL) party_site_id,
                   PARTY.party_id party_id,
                   PARTY.PARTY_TYPE party_type,
                   to_char(NULL) city,
                   to_char(NULL) country,
                   to_char(NULL) county,
                   to_char(NULL) state,
                   to_char(NULL) province,
                   to_char(NULL) postal_code ,
                   PARTY.employees_total employees_total,
                   PARTY.sic_code_type sic_code_type,
                   PARTY.sic_code sic_code,
                   upper(substr(PARTY.party_name,1,1))
party_name_substring,
                   upper(PARTY.party_name) party_name,
                   PARTY.category_code category_code,
                   ''HZ_PARTIES'' owner_table_name,
                   PARTY.party_id owner_table_id,
                   PARTY.duns_number_c


-- Party Type CMA
       , party.party_type
--
            from  HZ_PARTIES PARTY
             where PARTY.party_type in (''PERSON'',
''ORGANIZATION'')
             and PARTY.status = ''A'') Z
 where y.customer_id = z.party_id) X
 where CNTPNT.owner_table_name(+) = X.owner_table_name
 and   CNTPNT.owner_table_id(+) = X.owner_table_id
 and   CNTPNT.status(+)=''A''
 and   CNTPNT.primary_flag(+)=''Y''
 and   CNTPNT.contact_point_type(+)=''PHONE''
 and   ORGPRO.party_id(+) = X.party_id
 and   nvl(ORGPRO.effective_end_date(+),sysdate+1) > sysdate ;
```

- l_batch_dea_sql := NO
- l_incr_reassign_sql :=

```
select /*+ use_nl(ORGPRO CNTPNT) */
  -1001 source_id,
  -1002 trans_object_type_id,
  X.party_id trans_object_id,
  X.party_site_id trans_detail_object_id,
  x.party_type party_type,
  X.party_id squal_num01,
  X.party_site_id squal_num02,
  X.party_id squal_num03,
  X.party_id squal_num04,
  X.employees_total squal_num05,
  ORGPRO.curr_fy_potential_revenue squal_num06,
  UPPER(ORGPRO.pref_functional_currency) squal_curc01,
  UPPER(X.party_name_substring) squal_fc01,
  UPPER(X.party_name) squal_char01,
  UPPER(X.city) squal_char02,
  UPPER(X.county) squal_char03,
  UPPER(X.state) squal_char04,
  UPPER(X.province) squal_char05,
  UPPER(X.postal_code) squal_char06,
  UPPER(X.country) squal_char07,
  UPPER(CNTPNT.phone_area_code) squal_char08,
  UPPER(X.category_code) squal_char09,
  UPPER(X.sic_code_type||'': ''||X.sic_code) squal_char10,
  UPPER(X.duns_number_c) squal_char11

-- Party Type CMA
 , UPPER(X.party_type)
Q9010_PARTY_TYPE
--

from HZ_ORGANIZATION_PROFILES   ORGPRO ,
     HZ_CONTACT_POINTS CNTPNT,
     (select   /*+ leading(y) */
        Z.party_site_id,        Z.party_id,            z.
party_type, Z.city,
        Z.country,              Z.county,              Z.state,
        Z.province,             Z.postal_code,        Z.
employees_total,
        Z.sic_code_type,        Z.sic_code,            Z.
party_name_substring,
        Z.party_name,           Z.category_code,      Z.
owner_table_name,
        Z.owner_table_id,       Z.duns_number_c
       from ( select distinct ACC.customer_id customer_id
              from ( select distinct terr_id terr_id
                     from jty_changed_terrs
                     where tap_request_id = l_request_id )
CHG_TERR,
                   AS_ACCESSES_ALL ACC,
                   AS_TERRITORY_ACCESSES TERR_ACC
             where CHG_TERR.terr_id = TERR_ACC.territory_id
       and TERR_ACC.access_id = acc.access_id
       and acc.lead_id is null
       and acc.sales_lead_id is null ) Y,
           ( select SITE.party_site_id party_site_id,
                    SITE.party_id party_id,
   PARTY.PARTY_TYPE party_type,
                    LOC.city city,
                    LOC.country country,
                    LOC.county county,
                    LOC.state state,
                    LOC.province province,
                    LOC.postal_code postal_code,
                    PARTY.employees_total employees_total,
```

```
PARTY.sic_code_type sic_code_type,
                    PARTY.sic_code sic_code,
                    upper(substr(PARTY.party_name,1,1))
party_name_substring,
                    upper(PARTY.party_name) party_name,
                    PARTY.category_code category_code,
                    ''HZ_PARTY_SITES'' owner_table_name,
                    SITE.party_site_id owner_table_id,
                    PARTY.duns_number_c
-- Party Type CMA
       , party.party_type
--

             from  HZ_PARTY_SITES  SITE,
                   HZ_LOCATIONS  LOC,
                   HZ_PARTIES  PARTY
             where SITE.status = ''A''
             and SITE.party_id = PARTY.party_id
             and PARTY.party_type in (''PERSON'',
''ORGANIZATION'')
             and PARTY.status = ''A''
             and LOC.location_id = SITE.location_id union all
             select to_number(NULL) party_site_id,
                   PARTY.party_id party_id,
 PARTY.PARTY_TYPE party_type,
 to_char(NULL) city,
                    to_char(NULL) country,
                    to_char(NULL) county,
                    to_char(NULL) state,
                    to_char(NULL) province,
                    to_char(NULL) postal_code ,
                    PARTY.employees_total employees_total,
                    PARTY.sic_code_type sic_code_type,
                    PARTY.sic_code sic_code,
                    upper(substr(PARTY.party_name,1,1))
party_name_substring,
                    upper(PARTY.party_name) party_name,
                    PARTY.category_code category_code,
                    ''HZ_PARTIES'' owner_table_name,
                    PARTY.party_id owner_table_id,
                    PARTY.duns_number_c
-- Party Type CMA
       , party.party_type
--
             from  HZ_PARTIES PARTY
             where PARTY.party_type in (''PERSON'',
''ORGANIZATION'')
             and PARTY.status = ''A'') Z
      where y.customer_id = z.party_id) X
where CNTPNT.owner_table_name(+) = X.owner_table_name
and CNTPNT.owner_table_id(+) = X.owner_table_id
and CNTPNT.status(+)=''A''
and CNTPNT.primary_flag(+)=''Y''
and CNTPNT.contact_point_type(+)=''PHONE''
and ORGPRO.party_id(+) = X.party_id
and nvl(ORGPRO.effective_end_date(+),sysdate+1) > sysdate;



  JTY_TRANS_USG_PGM_SQL_PKG.Insert_Row(
   p_source_id => -1001
  ,p_trans_type_id => -1002
  ,p_program_name => 'SALES/ACCOUNT PROGRAM'
  ,p_version_name => '18-NOV-05: CMA PARTY TYPE ADDED'
```

```
,p_real_time_sql => l_real_time_sql
  ,p_batch_total_sql => l_batch_total_sql
  ,p_batch_incr_sql => l_batch_incr_sql
  ,p_batch_dea_sql => l_batch_dea_sql
  ,p_incr_reassign_sql => l_incr_reassign_sql
  ,p_use_total_for_dea_flag => null
  ,p_enabled_flag => 'Y'
  ,retcode => retcode
  ,errbuf => errbuf);


dbms_output.put_line('SALES/ACCOUNT PROGRAM retcode : ' ||
retcode);
dbms_output.put_line('SALES/ACCOUNT PROGRAM errbuf : ' ||
errbuf);

COMMIT;

END;
```

2. Modify the R12AccountTT.sql file with the appropriate package parameters to make the SQL meta-data change.

```
JTY_TRANS_USG_PGM_SQL_PKG.Insert_Row(
   p_source_id => -1001 /* From JTF_SOURCES_ALL*/
  ,p_trans_type_id => -1002 /*From JTF_QUAL_TYPES_ALL*/
  ,p_program_name => 'SALES/ACCOUNT PROGRAM' /*FROM
JTF_TRANS_USG_PGM_DETAILS*/
  ,p_version_name => '18-NOV-05: CMA PARTY TYPE ADDED'
  ,p_real_time_sql => l_real_time_sql
  ,p_batch_total_sql => l_batch_total_sql
  ,p_batch_incr_sql => l_batch_incr_sql
  ,p_batch_dea_sql => l_batch_dea_sql
  ,p_incr_reassign_sql => l_incr_reassign_sql
  ,p_use_total_for_dea_flag => null
  ,p_enabled_flag => 'Y'
  ,retcode => retcode
  ,errbuf => errbuf);
```

3. Run the R12AccountTT.sql file.

```
SQL> @R12AccountTT.sql /
```

Create the Custom Matching Attribute

Create the CMA. The basis of this script is to populate all the necessary information in the following tables:

JTF_SEEDED_QUAL_ALL_B

JTF_SEEDED_QUAL_ALL_TL

JTF_QUAL_USGS_ALL

1. Ensure that there is not an existing CMA record for the unique matching attribute ID that you will use for the new CMA.

```
set serveroutput ON SIZE 999999
DECLARE
    l_real_time_sql VARCHAR2(32000) := NULL;
    l_batch_total_sql VARCHAR2(32000) := NULL;
    l_batch_incr_sql VARCHAR2(32000) := NULL;
    l_batch_dea_sql VARCHAR2(32000) := NULL;
    l_incr_reassign_sql VARCHAR2(32000);
         retcode VARCHAR2(250);
           errbuf VARCHAR2(1000);

      TYPE namesarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE descarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE langarray IS VARRAY(5) OF VARCHAR2(50);
      names namesarray;
      descript descarray;
      lang langarray;
      total INTEGER;
 BEGIN
    BEGIN
        DELETE FROM jtf_seeded_qual_all_b
        WHERE seeded_qual_id = -9010;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_seeded_qual_all_tl
        WHERE seeded_qual_id = -9010;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_qual_usgs_all
        WHERE qual_usg_id = -9010;
            EXCEPTION
                  WHEN no_data_found THEN
                    NULL;
            END;
      BEGIN
```

2. Create the meta-data for the CMA. Follow the comments provided in the script to enter the relevant CMA information.

```
/* Provide CMA details in at least one language, such as 'US'*/
/* To create a CMA in multi-language, provide those details in
'names', 'descript' and 'lang' as an array*/

            names := namesarray('Party Type (CMA)');
            descript := descarray('Party Type (CMA)');
            lang := langarray('US');
            total := names.count;

    FOR i in 1 .. total LOOP
/* The following meta-data will create the custom matching attribute
for the Party Type.*/
     JTY_CUST_QUAL_PKG.Create_qual(
    p_seeded_qual_id     => -9010,
/* This number should correspond to the number used in the alias for
the attribute*/
      p_name            => names(i),
      p_description     => descript(i),
      p_language        => lang(i),

    ,/* Sales Usage: FK to
       JTF_SOURCES_ALL.SOURCE_ID*/
     p_source_id         => -1001

    ,/* Sales Account Transaction Type: FK to
        JTF_QUAL_TYPE_USGS_ALL.QUAL_TYPE_USG_ID*/
     p_trans_type_id      => -1002
    ,p_enabled_flag       => 'N'

    ,/* QUAL_RELATION_FACTOR
     This needs to be the next prime number greater than
     the value from the following SQL:

     SELECT MAX(qual_relation_factor)
     FROM jtf_qual_usgs_all

     The first 1000 primes can be found at:
     http://primes.utm.edu/lists/small/1000.txt

     Custom matching attributes should start at the 303rd prime
     which is 1999. This is to keep 1st to 302nd primes
     for product development to seed matching attributes.
     */
     p_qual_relation_factor       => 1999


    /* The following set of meta-data setups determine
    ** the behavior of the matching attribute in
    ** HTML and Excel UIs: how it is displayed; what
    ** is the LOV SQL; and, what comparison operators
    ** are supported.
    */

    ,/* CONVERT_TO_ID_FLAG: displayed qualifier value is a CHAR
but stored qualifier value is stored as an internal id. For example,
customer name
is displayed as a char but stored by party_id value. */
     p_convert_to_id_flag         => 'N'

    ,/* DISPLAY_TYPE: display type on UI (NUMERIC/CHAR) */
     p_display_type               => 'CHAR'

    ,/* LOV SQL */
    p_html_lov_sql1              =>
```

```
' SELECT a.meaning col1_value, a.lookup_code col2_value ' || CHR(10)
||
     ' FROM ar_lookups a ' || CHR(10) ||
     ' WHERE a.lookup_type = ''PARTY_TYPE'' ' || CHR(10) ||
     ' AND a.lookup_code IN (''ORGANIZATION'', ''PERSON'') ' || CHR
(10) ||
     ' ORDER BY col1_value '

/*Use the following parameters for dependent LOV SQLs */
    ,p_html_lov_sql2             => null
    ,p_html_lov_sql3             => null
    ,p_display_sql1              => null
    ,p_display_sql2              => null
    ,p_display_sql3              => null

/* The p_hierarchy_type value should always be set to null */
    ,p_hierarchy_type            => null

    ,/* Is the "=" operator supported? */
    p_equal_flag                 => 'Y'
    ,/* Is the "LIKE" operator supported? */
    p_like_flag                  => 'N'
    ,/* Is the "BETWEEN" operator supported? */
    p_between_flag               => 'N'

    /* Values table ? this table will store the matching attribute
vales of a territory*/
    /* Specify which columns to use by giving it a name for the
corresnding custom matching attribute values. In this example, the
party type value (person, organization), will be stored in the
'q9010_low_value_CHAR' column. In order to avoid confusion, it is
recommended that you start the column name with the custom matching
attribute unique identifier (ie. "q9010"). Use the following table
to determine which column to use based on the display type of the
custom matching attribute. You can query JTF_TERR_VALUES_ALL to see
existing matching attributes' display type and the corresponding
column(s). */
```

```
    ,p_comparison_operator        => 'q9010_cp'
    ,p_low_value_char             => 'q9010_low_value_CHAR'
    ,p_high_value_char            => null
    ,p_low_value_char_id          => null
    ,p_low_value_number           => null
    ,p_high_value_number          => null
    ,p_interest_type_id           => null
    ,p_primary_interest_code_id   => null
    ,p_sec_interest_code_id       => null
    ,p_value1_id                  => null
    ,p_value2_id                  => null
    ,p_value3_id                  => null
    ,p_value4_id                  => null
    ,p_first_char                 => null
    ,p_currency_code              => null


    ,/* Transaction Type Alias/_TRANS table column mapping */
     p_qual_col1                  => 'Q9010_PARTY_TYPE' /*This is
the alias used in script #2*/

    ,/* TCA Classification for derivation. This value should remain
null*/
     p_alias_rule1                => null

    ,/* Rule SQL.
In this example we are matching the Q9010_PARTY_TYPE value from the
transaction type SQL with the q9010_low_value_CHAR value stored in
JTF_TERR_ALL_VALUES.
*/
     p_op_eql                     =>
      ' (A.Q9010_PARTY_TYPE = B.q9010_low_value_CHAR AND B.q9010_cp
= ''='')'
    ,p_op_like                    => null
    ,p_op_between                 => NULL
    ,p_op_common_where            => null

    ,p_real_time_select           =>
    'SELECT DISTINCT A.trans_object_id, A.trans_detail_object_id, A.
txn_date, B.terr_id, B.absolute_rank, B.top_level_terr_id, B.
num_winners'
    ,p_real_time_from             => null
    ,p_real_time_where            =>
    'WHERE '  || CHR(10) ||
    '  B.q9010_low_value_CHAR = A.Q9010_PARTY_TYPE ' || CHR(10) ||
    '  AND B.q9010_cp = ''='' ' || CHR(10) ||
    '  AND B.source_id = -1001 ' || CHR(10) ||
    '  AND A.txn_date between B.start_date and B.end_date'


    ,retcode                      => retcode
    ,errbuf                       => errbuf);

  dbms_output.put_line('Retcode : ' || retcode);
  dbms_output.put_line('Errbuf : ' || errbuf);

  ad_morg.replicate_seed_data(NULL, 'JTF', NULL);

EXIT WHEN retcode=2;
END LOOP;
END;
COMMIT;

exception
  when others then
```

```
dbms_output.put_line('Retcode : ' || retcode);
    dbms_output.put_line('Errbuf : ' || errbuf);
    raise;
end;
/
```

3. Run this script.

CMA for Lead: Lead Status

This is an example of creating a custom matching attribute for Sales using the lead transaction and the lead status attribute.

Modify Script 1

Modify Script 1, page A-96 to retrieve the relevant lead transaction type SQL meta-data.

1. Set the source ID corresponding to the usage (Sales or Collections). The source ID -1001 corresponds to the Sales usage. To find the desired usage ID query JTF_SOURES_ALL.

   ```
   p_source_id      NUMBER := -1001;
   ```

2. Set the transaction type ID. The ID -1003 corresponds to the Lead transaction type ID. To find the desired transaction type ID, query JTF_QUAL_TYPES_ALL

   ```
   p_trans_type_id  NUMBER := -1003;
   ```

3. Run the script and save the output to a file. Running this command will save the Lead transaction type SQL meta-data to the "R12LeadTT.sql" file.

   ```
   SQL>spool <output filename - e.g., "d:\R12LeadTT.sql">
   SQL> set serveroutput on size 999999
   SQL> @Script1.sql /
   ```

Modify Transaction SQL Meta-Data

Modify the transaction type SQL meta-data (R12ALeadTT.sql) to get the CMA's corresponding transaction attribute value.

1. Add the status_code attribute to each of the following SQLs:

   - l_real_time_sql :=

```
 'SELECT  ' || CHR(10) ||

'   LEAD.sales_lead_id trans_object_id  ' || CHR(10) ||

'  ,to_number(NULL) trans_detail_object_id  ' || CHR(10) ||

'  ,LEAD.sales_lead_id sales_lead_id  ' || CHR(10) ||

'  ,to_number(NULL) sales_lead_line_id  ' || CHR(10) ||

'  ,TO_CHAR(NULL) city  ' || CHR(10) ||

'  ,TO_CHAR(NULL) postal_code  ' || CHR(10) ||

'  ,TO_CHAR(NULL) state  ' || CHR(10) ||

'  ,TO_CHAR(NULL) province  ' || CHR(10) ||

'  ,TO_CHAR(NULL) county  ' || CHR(10) ||

'  ,TO_CHAR(NULL) country  ' || CHR(10) ||

'  ,PARTY.PARTY_ID party_id  ' || CHR(10) ||

'  ,to_number(null) party_site_id  ' || CHR(10) ||

'  ,upper(PARTY.PRIMARY_PHONE_AREA_CODE) area_code  ' || CHR(10)
||
'  ,upper(PARTY.PARTY_NAME) comp_name_range  ' || CHR(10) ||

'  ,PARTY.PARTY_ID partner_id  ' || CHR(10) ||

'  ,PARTY.EMPLOYEES_TOTAL num_of_employees  ' || CHR(10) ||

'  ,upper(PARTY.CATEGORY_CODE) category_code  ' || CHR(10) ||

'  ,PARTY.PARTY_ID party_relationship_id  ' || CHR(10) ||

'  ,upper(party.sic_code_type||'': ''||party.sic_code) sic_code
' || CHR(10) ||
'  ,LEAD.budget_amount budget_amount  ' || CHR(10) ||

'  ,upper(lead.currency_code) currency_code  ' || CHR(10) ||

'  ,LEAD.source_promotion_id source_promotion_id  ' || CHR(10) ||

'  ,ORGP.CURR_FY_POTENTIAL_REVENUE squal_num06  ' || CHR(10) ||

'  ,upper(ORGP.PREF_FUNCTIONAL_CURRENCY) car_currency_code  ' ||
CHR(10) ||
'  ,PARTY.PARTY_ID squal_num01  ' || CHR(10) ||

'  ,UPPER(PARTY.DUNS_NUMBER_C) SQUAL_CHAR11  ' || CHR(10) ||

'  ,UPPER(LEAD.channel_code) SQUAL_CHAR30  ' || CHR(10) ||

'  ,l_txn_date txn_date  ' || CHR(10) ||
' , UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(LEAD.status_code) Q9100_LEAD_STATUS  ' || CHR(10) ||

'FROM  ' || CHR(10) ||

'  AS_SALES_LEADS LEAD,  ' || CHR(10) ||
```

```
'  HZ_PARTIES PARTY,  ' || CHR(10) ||

'  HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'  AS_STATUSES_B STATUS  ' || CHR(10) ||

'WHERE  LEAD.ADDRESS_ID IS NULL  ' || CHR(10) ||

'AND    LEAD.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    LEAD.status_code = STATUS.status_code  ' || CHR(10) ||

'AND    STATUS.lead_flag = ''Y''  ' || CHR(10) ||

'AND    LEAD.sales_lead_id = l_trans_object_id1  ' || CHR(10) ||

'UNION ALL  ' || CHR(10) ||

'SELECT  ' || CHR(10) ||

'   LEAD.sales_lead_id trans_object_id  ' || CHR(10) ||

'  ,SITE.PARTY_SITE_ID trans_detail_object_id  ' || CHR(10) ||

'  ,LEAD.sales_lead_id sales_lead_id  ' || CHR(10) ||

'  ,SITE.PARTY_SITE_ID sales_lead_line_id  ' || CHR(10) ||

'  ,upper(LOC.CITY) city  ' || CHR(10) ||

'  ,upper(LOC.POSTAL_CODE) postal_code  ' || CHR(10) ||

'  ,upper(LOC.STATE) state  ' || CHR(10) ||

'  ,upper(LOC.PROVINCE) province  ' || CHR(10) ||

'  ,upper(LOC.COUNTY) county  ' || CHR(10) ||

'  ,upper(LOC.COUNTRY) country  ' || CHR(10) ||

'  ,PARTY.PARTY_ID party_id  ' || CHR(10) ||

'  ,SITE.PARTY_SITE_ID party_site_id  ' || CHR(10) ||

'  ,upper(CNTPNT.PHONE_AREA_CODE) area_code  ' || CHR(10) ||

'  ,upper(PARTY.PARTY_NAME) comp_name_range  ' || CHR(10) ||

'  ,PARTY.PARTY_ID partner_id  ' || CHR(10) ||

'  ,PARTY.EMPLOYEES_TOTAL num_of_employees  ' || CHR(10) ||

'  ,upper(PARTY.CATEGORY_CODE) category_code  ' || CHR(10) ||

'  ,PARTY.PARTY_ID party_relationship_id  ' || CHR(10) ||

'  ,upper(party.sic_code_type||'': ''||party.sic_code) sic_code
' || CHR(10) ||
'  ,LEAD.budget_amount budget_amount  ' || CHR(10) ||
```

```
'  ,upper(lead.currency_code) currency_code  ' || CHR(10) ||

'  ,LEAD.source_promotion_id source_promotion_id  ' || CHR(10) ||

'  ,ORGP.CURR_FY_POTENTIAL_REVENUE squal_num06  ' || CHR(10) ||

'  ,upper(ORGP.PREF_FUNCTIONAL_CURRENCY) car_currency_code  ' ||
CHR(10) ||
'  ,PARTY.PARTY_ID squal_num01  ' || CHR(10) ||

'  ,UPPER(PARTY.DUNS_NUMBER_C) SQUAL_CHAR11  ' || CHR(10) ||

'  ,UPPER(LEAD.channel_code) SQUAL_CHAR30  ' || CHR(10) ||

'  ,l_txn_date txn_date  ' || CHR(10) ||

' , UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(LEAD.status_code) Q9100_LEAD_STATUS  ' || CHR(10) ||

'FROM  ' || CHR(10) ||

'  AS_SALES_LEADS LEAD,  ' || CHR(10) ||

'  HZ_CONTACT_POINTS CNTPNT,  ' || CHR(10) ||

'  HZ_PARTY_SITES SITE,  ' || CHR(10) ||

'  HZ_LOCATIONS LOC,  ' || CHR(10) ||

'  HZ_PARTIES PARTY,  ' || CHR(10) ||

'  HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'  AS_STATUSES_B STATUS  ' || CHR(10) ||

'WHERE  CNTPNT.OWNER_TABLE_NAME(+) = ''HZ_PARTY_SITES''  ' || CHR
(10) ||
'AND    SITE.PARTY_SITE_ID = CNTPNT.OWNER_TABLE_ID(+)  ' || CHR
(10) ||
'AND    CNTPNT.PRIMARY_FLAG(+) = ''Y''  ' || CHR(10) ||

'AND    CNTPNT.STATUS(+) <> ''I''  ' || CHR(10) ||

'AND    CNTPNT.contact_point_type(+)=''PHONE''  ' || CHR(10) ||

'AND    LEAD.ADDRESS_ID = SITE.PARTY_SITE_ID  ' || CHR(10) ||

'AND    SITE.LOCATION_ID = LOC.LOCATION_ID  ' || CHR(10) ||

'AND    SITE.STATUS = ''A''  ' || CHR(10) ||

'AND    LEAD.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    LEAD.status_code = STATUS.status_code  ' || CHR(10) ||

'AND    STATUS.lead_flag = ''Y''  ' || CHR(10) ||

'AND    LEAD.sales_lead_id = l_trans_object_id1  ';
```

- l_batch_total_sql :=

```
l_batch_total_sql :=

'SELECT  ' || CHR(10) ||

'  -1001                              source_id,  ' || CHR(10) ||

'  -1003                                 trans_object_type_id,  '
|| CHR(10) ||
'  LEAD.sales_lead_id                 trans_object_id,  ' || CHR
(10) ||
'  to_number(NULL)
trans_detail_object_id,  ' || CHR(10)
||

'  STATUS.OPP_OPEN_STATUS_FLAG                       OPEN_FLAG,
' || CHR(10) ||
'  PARTY.PARTY_ID                       squal_num01,  ' || CHR
(10) ||
'  to_number(null)                     squal_num02,  ' || CHR
(10) ||
'  PARTY.PARTY_ID                   squal_num03,  ' || CHR(10) ||

'  lead.sales_lead_id                 squal_num04,  ' || CHR(10)
||
'  PARTY.EMPLOYEES_TOTAL                 squal_num05,  ' || CHR
(10) ||
'  ORGP.CURR_FY_POTENTIAL_REVENUE         squal_num06,  ' || CHR
(10) ||
'  LEAD.budget_amount                  squal_num30,  ' || CHR(10)
||
'  LEAD.source_promotion_id               squal_num31,  ' || CHR
(10) ||
'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)        squal_curc01,  '
|| CHR(10) ||
'  upper(lead.currency_code)            squal_curc07,  ' || CHR
(10) ||
'  upper(substr(PARTY.party_name,1,1))       squal_fc01,  ' ||
CHR(10) ||
'  upper(PARTY.PARTY_NAME)             squal_char01,  ' || CHR
(10) ||
'  TO_CHAR(NULL)                      squal_char02,  ' || CHR(10)
||
'  TO_CHAR(NULL)                      squal_char03,  ' || CHR(10)
||
'  TO_CHAR(NULL)                      squal_char04,  ' || CHR(10)
||
'  TO_CHAR(NULL)                      squal_char05,  ' || CHR(10)
||
'  TO_CHAR(NULL)
squal_char06,  ' || CHR(10)
||

'  TO_CHAR(NULL)                      squal_char07,  ' || CHR(10)
||
'  upper(PARTY.PRIMARY_PHONE_AREA_CODE)       squal_char08,  ' ||
CHR(10) ||
'  upper(PARTY.CATEGORY_CODE)
squal_char09,  ' || CHR(10)
||

'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)           squal_char11,  ' || CHR
```

```
(10) ||
'  UPPER(LEAD.channel_code)                squal_char30  ' || CHR
(10) ||
'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(LEAD.status_code) Q9100_LEAD_STATUS  ' || CHR(10) ||

'FROM  ' || CHR(10) ||

'  AS_SALES_LEADS LEAD,  ' || CHR(10) ||

'  HZ_PARTIES PARTY,  ' || CHR(10) ||

'  HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'  AS_STATUSES_B STATUS  ' || CHR(10) ||

'WHERE  LEAD.ADDRESS_ID IS NULL  ' || CHR(10) ||

'AND    LEAD.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    LEAD.status_code = STATUS.status_code  ' || CHR(10) ||

'AND    STATUS.lead_flag = ''Y'' UNION ALL  ' || CHR(10) ||

'SELECT  ' || CHR(10) ||

'  -1001                          source_id,  ' || CHR(10) ||

'  -1003                          trans_object_type_id,  ' ||
CHR(10) ||
'  LEAD.sales_lead_id
trans_object_id,  ' ||
CHR(10) ||

'  SITE.PARTY_SITE_ID                trans_detail_object_id,  '
|| CHR(10) ||
'  STATUS.OPP_OPEN_STATUS_FLAG                       OPEN_FLAG,
' || CHR(10) ||
'  PARTY.PARTY_ID                 squal_num01,  ' || CHR(10) ||

'  SITE.PARTY_SITE_ID                                squal_num02,
' || CHR(10)
||

'  PARTY.PARTY_ID                 squal_num03,  ' || CHR(10) ||

'  LEAD.SALES_LEAD_ID                squal_num04,  ' || CHR(10)
||
'  PARTY.EMPLOYEES_TOTAL                             squal_num05,
' || CHR(10)
||

'  ORGP.CURR_FY_POTENTIAL_REVENUE        squal_num06,  ' || CHR
(10) ||
'  LEAD.budget_amount                 squal_num30,  ' || CHR(10)
||
'  LEAD.source_promotion_id              squal_num31,  ' || CHR
(10) ||
```

```
'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)       squal_curc01,  ' ||
CHR(10) ||
'  upper(lead.currency_code)             squal_curc07,  ' || CHR
(10) ||
'  upper(substr(PARTY.party_name,1,1))       squal_fc01,  ' ||
CHR(10) ||
'  upper(PARTY.PARTY_NAME)             squal_char01,  ' || CHR
(10) ||
'  upper(LOC.CITY)                  squal_char02,  ' || CHR(10)
||
'  upper(LOC.COUNTY)                     squal_char03,  ' || CHR
(10) ||
'  upper(LOC.STATE)                     squal_char04,  ' || CHR
(10) ||
'  upper(LOC.PROVINCE)                  squal_char05,  ' || CHR
(10) ||
'  upper(LOC.POSTAL_CODE)              squal_char06,  ' || CHR(10)
||
'  upper(LOC.COUNTRY)                  squal_char07,  ' || CHR(10)
||
'  upper(CNTPNT.PHONE_AREA_CODE)            squal_char08,  ' ||
CHR(10) ||
'  upper(PARTY.CATEGORY_CODE)                     squal_char09,
' || CHR(10) ||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)            squal_char11,  ' || CHR
(10) ||
'  UPPER(LEAD.channel_code)               squal_char30  ' || CHR
(10) ||
'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(LEAD.status_code) Q9100_LEAD_STATUS  ' || CHR(10) ||

'FROM  ' || CHR(10) ||

'  AS_SALES_LEADS LEAD,  ' || CHR(10) ||

'  HZ_CONTACT_POINTS CNTPNT,  ' || CHR(10) ||

'  HZ_PARTY_SITES SITE,  ' || CHR(10) ||

'  HZ_LOCATIONS LOC,  ' || CHR(10) ||

'  HZ_PARTIES PARTY,  ' || CHR(10) ||

'  HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'  AS_STATUSES_B STATUS  ' || CHR(10) ||

'WHERE  CNTPNT.OWNER_TABLE_NAME(+) = ''HZ_PARTY_SITES''  ' || CHR
(10) ||
'AND    SITE.PARTY_SITE_ID = CNTPNT.OWNER_TABLE_ID(+)  ' || CHR
(10) ||
'AND    CNTPNT.PRIMARY_FLAG(+) = ''Y''  ' || CHR(10) ||

'AND    CNTPNT.STATUS(+) <> ''I''  ' || CHR(10) ||

'AND    CNTPNT.contact_point_type(+)=''PHONE''  ' || CHR(10) ||

'AND    LEAD.ADDRESS_ID = SITE.PARTY_SITE_ID  ' || CHR(10) ||

'AND    SITE.LOCATION_ID = LOC.LOCATION_ID  ' || CHR(10) ||
```

```
'AND    SITE.STATUS = ''A''  ' || CHR(10) ||

'AND    LEAD.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    LEAD.status_code = STATUS.status_code  ' || CHR(10) ||

'AND    STATUS.lead_flag = ''Y''  ';
```

- l_batch_incr_sql :=

```
l_batch_incr_sql :=

'SELECT  ' || CHR(10) ||

'  -1001                  source_id,  ' || CHR(10) ||

'  -1003                       trans_object_type_id,  ' || CHR
(10) ||
'  LEAD.sales_lead_id     trans_object_id,  ' || CHR(10) ||

'  to_number(NULL)             trans_detail_object_id,  ' || CHR
(10) ||
'  STATUS.OPP_OPEN_STATUS_FLAG           OPEN_FLAG,  ' || CHR(10)
||
'  PARTY.PARTY_ID               squal_num01,  ' || CHR(10) ||

'  to_number(null)          squal_num02,  ' || CHR(10) ||

'  PARTY.PARTY_ID       squal_num03,  ' || CHR(10) ||

'  lead.sales_lead_id        squal_num04,  ' || CHR(10) ||

'  PARTY.EMPLOYEES_TOTAL     squal_num05,  ' || CHR(10) ||

'  ORGP.CURR_FY_POTENTIAL_REVENUE   squal_num06,  ' || CHR(10) ||

'  LEAD.budget_amount         squal_num30,  ' || CHR(10) ||

'  LEAD.source_promotion_id    squal_num31,  ' || CHR(10) ||

'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)  squal_curc01,  ' || CHR
(10) ||
'  upper(lead.currency_code)       squal_curc07,  ' || CHR(10) ||

'  upper(substr(PARTY.party_name,1,1)) squal_fc01,  ' || CHR(10)
||
'  upper(PARTY.PARTY_NAME)    squal_char01,  ' || CHR(10) ||

'  TO_CHAR(NULL)          squal_char02,  ' || CHR(10) ||

'  TO_CHAR(NULL)          squal_char03,  ' || CHR(10) ||

'  TO_CHAR(NULL)          squal_char04,  ' || CHR(10) ||

'  TO_CHAR(NULL)          squal_char05,  ' || CHR(10) ||

'  TO_CHAR(NULL)                         squal_char06,  ' || CHR
(10) ||
'  TO_CHAR(NULL)          squal_char07,  ' || CHR(10) ||

'  upper(PARTY.PRIMARY_PHONE_AREA_CODE) squal_char08,  ' || CHR
(10) ||
'  upper(PARTY.CATEGORY_CODE)            squal_char09,  ' || CHR
(10) ||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)   squal_char11,  ' || CHR(10) ||

'  UPPER(LEAD.channel_code)    squal_char30  ' || CHR(10) ||

'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(LEAD.status_code) Q9100_LEAD_STATUS  ' || CHR(10) ||
```

```
'FROM AS_SALES_LEADS LEAD,  ' || CHR(10) ||

'     HZ_PARTIES PARTY,  ' || CHR(10) ||

'     HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'  AS_STATUSES_B STATUS,  ' || CHR(10) ||

'  AS_CHANGED_ACCOUNTS_ALL ACHNG  ' || CHR(10) ||

'WHERE  lead.address_id is null  ' || CHR(10) ||

'AND    lead.customer_id = party.party_id  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    lead.status_code = status.status_code  ' || CHR(10) ||

'AND    status.lead_flag = ''Y''  ' || CHR(10) ||

'AND    achng.sales_lead_id = lead.sales_lead_id  ' || CHR(10) ||

'AND    achng.request_id = l_REQUEST_ID UNION ALL  ' || CHR(10)
||
'SELECT  ' || CHR(10) ||

'  -1001               source_id,  ' || CHR(10) ||

'  -1003               trans_object_type_id,  ' || CHR(10) ||

'  LEAD.sales_lead_id                    trans_object_id,  ' ||
CHR(10) ||
'  SITE.PARTY_SITE_ID     trans_detail_object_id,  ' || CHR(10)
||
'  STATUS.OPP_OPEN_STATUS_FLAG           OPEN_FLAG,  ' || CHR(10)
||
'  PARTY.PARTY_ID      squal_num01,  ' || CHR(10) ||

'  SITE.PARTY_SITE_ID                    squal_num02,  ' || CHR
(10) ||
'  PARTY.PARTY_ID      squal_num03,  ' || CHR(10) ||

'  LEAD.SALES_LEAD_ID     squal_num04,  ' || CHR(10) ||

'  PARTY.EMPLOYEES_TOTAL                 squal_num05,  ' || CHR
(10) ||
'  ORGP.CURR_FY_POTENTIAL_REVENUE  squal_num06,  ' || CHR(10) ||

'  LEAD.budget_amount         squal_num30,  ' || CHR(10) ||

'  LEAD.source_promotion_id    squal_num31,  ' || CHR(10) ||

'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY) squal_curc01,  ' || CHR
(10) ||
'  upper(lead.currency_code)    squal_curc07,  ' || CHR(10) ||

'  upper(substr(PARTY.party_name,1,1)) squal_fc01,  ' || CHR(10)
||
'  upper(PARTY.PARTY_NAME)    squal_char01,  ' || CHR(10) ||

'  upper(LOC.CITY)      squal_char02,  ' || CHR(10) ||
```

```
'  upper(LOC.COUNTY)         squal_char03,  ' || CHR(10) ||

'  upper(LOC.STATE)         squal_char04,  ' || CHR(10) ||

'  upper(LOC.PROVINCE)     squal_char05,  ' || CHR(10) ||

'  upper(LOC.POSTAL_CODE)    squal_char06,  ' || CHR(10) ||

'  upper(LOC.COUNTRY)     squal_char07,  ' || CHR(10) ||

'  upper(CNTPNT.PHONE_AREA_CODE)   squal_char08,  ' || CHR(10) ||

'  upper(PARTY.CATEGORY_CODE)         squal_char09,  ' || CHR(10)
||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)   squal_char11,  ' || CHR(10) ||

'  UPPER(LEAD.channel_code)    squal_char30  ' || CHR(10) ||

'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(LEAD.status_code) Q9100_LEAD_STATUS  ' || CHR(10) ||

'FROM  AS_SALES_LEADS LEAD,  ' || CHR(10) ||

'      HZ_CONTACT_POINTS CNTPNT,  ' || CHR(10) ||

'      HZ_PARTY_SITES SITE,  ' || CHR(10) ||

'      HZ_LOCATIONS LOC,  ' || CHR(10) ||

'      HZ_PARTIES PARTY,  ' || CHR(10) ||

'      HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'   AS_STATUSES_B STATUS,  ' || CHR(10) ||

'      AS_CHANGED_ACCOUNTS_ALL ACHNG  ' || CHR(10) ||

'WHERE  CNTPNT.OWNER_TABLE_NAME(+) = ''HZ_PARTY_SITES''  ' || CHR
(10) ||
'AND    SITE.PARTY_SITE_ID = CNTPNT.OWNER_TABLE_ID(+)  ' || CHR
(10) ||
'AND    CNTPNT.PRIMARY_FLAG(+) = ''Y''  ' || CHR(10) ||

'AND    CNTPNT.STATUS(+) <> ''I''  ' || CHR(10) ||

'AND    CNTPNT.contact_point_type(+)=''PHONE''  ' || CHR(10) ||

'AND    LEAD.ADDRESS_ID = SITE.PARTY_SITE_ID  ' || CHR(10) ||

'AND    SITE.LOCATION_ID = LOC.LOCATION_ID  ' || CHR(10) ||

'AND    SITE.STATUS = ''A''  ' || CHR(10) ||

'AND    LEAD.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
```

```
|| CHR(10) ||
'AND    LEAD.status_code = STATUS.status_code  ' || CHR(10) ||

'AND    STATUS.lead_flag = ''Y''  ' || CHR(10) ||

'AND    achng.sales_lead_id = lead.sales_lead_id  ' || CHR(10) ||

'AND    achng.request_id = l_REQUEST_ID  ';
```

- l_batch_dea_sql :=

```
l_batch_dea_sql :=

NULL;

l_incr_reassign_sql :=

'SELECT  ' || CHR(10) ||

'  -1001                      source_id,  ' || CHR(10) ||

'  -1003                         trans_object_type_id,  ' || CHR
(10) ||
'  LEAD.sales_lead_id         trans_object_id,  ' || CHR(10) ||

'  STATUS.OPP_OPEN_STATUS_FLAG               OPEN_FLAG,  ' || CHR
(10) ||
'  to_number(NULL)                trans_detail_object_id,  ' ||
CHR(10) ||
'  PARTY.PARTY_ID                squal_num01,  ' || CHR(10) ||

'  to_number(null)              squal_num02,  ' || CHR(10) ||

'  PARTY.PARTY_ID           squal_num03,  ' || CHR(10) ||

'  lead.sales_lead_id         squal_num04,  ' || CHR(10) ||

'  PARTY.EMPLOYEES_TOTAL         squal_num05,  ' || CHR(10) ||

'  ORGP.CURR_FY_POTENTIAL_REVENUE    squal_num06,  ' || CHR(10)
||
'  LEAD.budget_amount          squal_num30,  ' || CHR(10) ||

'  LEAD.source_promotion_id        squal_num31,  ' || CHR(10) ||

'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)   squal_curc01,  ' || CHR
(10) ||
'  upper(lead.currency_code)     squal_curc07,  ' || CHR(10) ||

'  upper(substr(PARTY.party_name,1,1))  squal_fc01,  ' || CHR(10)
||
'  upper(PARTY.PARTY_NAME)     squal_char01,  ' || CHR(10) ||

'  TO_CHAR(NULL)              squal_char02,  ' || CHR(10) ||

'  TO_CHAR(NULL)              squal_char03,  ' || CHR(10) ||

'  TO_CHAR(NULL)              squal_char04,  ' || CHR(10) ||

'  TO_CHAR(NULL)              squal_char05,  ' || CHR(10) ||

'  TO_CHAR(NULL)                              squal_char06,  ' ||
CHR(10) ||
'  TO_CHAR(NULL)              squal_char07,  ' || CHR(10) ||

'  upper(PARTY.PRIMARY_PHONE_AREA_CODE)  squal_char08,  ' || CHR
(10) ||
'  upper(PARTY.CATEGORY_CODE)                squal_char09,  ' ||
CHR(10) ||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)    squal_char11,  ' || CHR(10) ||

'  UPPER(LEAD.channel_code)        squal_char30  ' || CHR(10) ||
```

```
'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(LEAD.status_code) Q9100_LEAD_STATUS  ' || CHR(10) ||

'FROM   AS_SALES_LEADS LEAD,  ' || CHR(10) ||

'       HZ_PARTIES PARTY,  ' || CHR(10) ||

'       HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'    AS_STATUSES_B STATUS,  ' || CHR(10) ||

'    ( select distinct ACC.sales_lead_id sales_lead_id  ' || CHR
(10) ||
'         from (select distinct terr_id terr_id  ' || CHR(10) ||

'               from JTY_CHANGED_TERRS  ' || CHR(10) ||

'               where tap_request_id = l_request_id) CHG_TERR,  '
|| CHR(10) ||
'              AS_ACCESSES_ALL ACC,  ' || CHR(10) ||

'              AS_TERRITORY_ACCESSES TERR_ACC  ' || CHR(10) ||

'         where CHG_TERR.terr_id = TERR_ACC.territory_id  ' ||
CHR(10) ||
'         and TERR_ACC.access_id = acc.access_id  ' || CHR(10) ||

'         and acc.sales_lead_id is not null  ' || CHR(10) ||

'         and acc.lead_id is null ) ACHNG  ' || CHR(10) ||

'WHERE  lead.address_id is null  ' || CHR(10) ||

'AND  lead.customer_id = party.party_id  ' || CHR(10) ||

'AND  party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND  party.STATUS = ''A''  ' || CHR(10) ||

'AND  nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  ' ||
CHR(10) ||
'AND  lead.status_code = status.status_code  ' || CHR(10) ||

'AND status.lead_flag = ''Y''  ' || CHR(10) ||

'AND achng.sales_lead_id = lead.sales_lead_id UNION ALL  ' || CHR
(10) ||
'SELECT  ' || CHR(10) ||

'  -1001                  source_id,  ' || CHR(10) ||

'  -1003                  trans_object_type_id,  ' || CHR(10) ||

'  LEAD.sales_lead_id                        trans_object_id,  '
|| CHR(10) ||
'  STATUS.OPP_OPEN_STATUS_FLAG               OPEN_FLAG,  ' || CHR
(10) ||
'  SITE.PARTY_SITE_ID         trans_detail_object_id,  ' || CHR
(10) ||
'  PARTY.PARTY_ID          squal_num01,  ' || CHR(10) ||

'  SITE.PARTY_SITE_ID                        squal_num02,  ' ||
CHR(10) ||
'  PARTY.PARTY_ID          squal_num03,  ' || CHR(10) ||
```

```
'  LEAD.SALES_LEAD_ID         squal_num04,  ' || CHR(10) ||

'  PARTY.EMPLOYEES_TOTAL                     squal_num05,  ' ||
CHR(10) ||
'  ORGP.CURR_FY_POTENTIAL_REVENUE   squal_num06,  ' || CHR(10) ||

'  LEAD.budget_amount          squal_num30,  ' || CHR(10) ||

'  LEAD.source_promotion_id        squal_num31,  ' || CHR(10) ||

'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)  squal_curc01,  ' || CHR
(10) ||
'  upper(lead.currency_code)     squal_curc07,  ' || CHR(10) ||

'  upper(substr(PARTY.party_name,1,1))  squal_fc01,  ' || CHR(10)
||
'  upper(PARTY.PARTY_NAME)     squal_char01,  ' || CHR(10) ||

'  upper(LOC.CITY)          squal_char02,  ' || CHR(10) ||

'  upper(LOC.COUNTY)             squal_char03,  ' || CHR(10) ||

'  upper(LOC.STATE)             squal_char04,  ' || CHR(10) ||

'  upper(LOC.PROVINCE)         squal_char05,  ' || CHR(10) ||

'  upper(LOC.POSTAL_CODE)     squal_char06,  ' || CHR(10) ||

'  upper(LOC.COUNTRY)         squal_char07,  ' || CHR(10) ||

'  upper(CNTPNT.PHONE_AREA_CODE)    squal_char08,  ' || CHR(10)
||
'  upper(PARTY.CATEGORY_CODE)             squal_char09,  ' || CHR
(10) ||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)    squal_char11,  ' || CHR(10) ||

'  UPPER(LEAD.channel_code)        squal_char30  ' || CHR(10) ||

'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(LEAD.status_code) Q9100_LEAD_STATUS  ' || CHR(10) ||

'FROM  AS_SALES_LEADS LEAD,  ' || CHR(10) ||

'      HZ_CONTACT_POINTS CNTPNT,  ' || CHR(10) ||

'      HZ_PARTY_SITES SITE,  ' || CHR(10) ||

'      HZ_LOCATIONS LOC,  ' || CHR(10) ||

'      HZ_PARTIES PARTY,  ' || CHR(10) ||

'      HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'   AS_STATUSES_B STATUS,  ' || CHR(10) ||

'      ( select distinct  ACC.sales_lead_id sales_lead_id  ' ||
CHR(10) ||
'        from (select distinct terr_id terr_id  ' || CHR(10) ||

'              from JTY_CHANGED_TERRS  ' || CHR(10) ||
```

```
'              where tap_request_id = l_request_id) CHG_TERR,  '
|| CHR(10) ||
'             AS_ACCESSES_ALL ACC,  ' || CHR(10) ||

'             AS_TERRITORY_ACCESSES TERR_ACC  ' || CHR(10) ||

'        where CHG_TERR.terr_id = TERR_ACC.territory_id  ' || CHR
(10) ||
'        and TERR_ACC.access_id = acc.access_id  ' || CHR(10) ||

'        and acc.sales_lead_id is not null  ' || CHR(10) ||

'        and acc.lead_id is null ) ACHNG  ' || CHR(10) ||

'WHERE  CNTPNT.OWNER_TABLE_NAME(+) = ''HZ_PARTY_SITES''  ' || CHR
(10) ||
'AND  SITE.PARTY_SITE_ID = CNTPNT.OWNER_TABLE_ID(+)  ' || CHR(10)
||
'AND  CNTPNT.PRIMARY_FLAG(+) = ''Y''  ' || CHR(10) ||

'AND  CNTPNT.STATUS(+) <> ''I''  ' || CHR(10) ||

'AND  CNTPNT.contact_point_type(+)=''PHONE''  ' || CHR(10) ||

'AND  LEAD.ADDRESS_ID = SITE.PARTY_SITE_ID  ' || CHR(10) ||

'AND  SITE.LOCATION_ID = LOC.LOCATION_ID  ' || CHR(10) ||

'AND  SITE.STATUS = ''A''  ' || CHR(10) ||

'AND  LEAD.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND  party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND  party.STATUS = ''A''  ' || CHR(10) ||

'AND  nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  ' ||
CHR(10) ||
'AND LEAD.status_code = STATUS.status_code  ' || CHR(10) ||

'AND STATUS.lead_flag = ''Y''  ' || CHR(10) ||

'AND achng.sales_lead_id = lead.sales_lead_id  ';
```

- l_incr_reassign_sql :=

2. Modify the R12LeadTT.sql file with the appropriate package parameters to make the SQL meta-data change.

```
JTY_TRANS_USG_PGM_SQL_PKG.Insert_Row(
p_source_id => -1001
,p_trans_type_id => -1003
,p_program_name => 'SALES/LEAD PROGRAM'
,p_version_name => '12-JAN-06: CMA LEAD STATUS ADDED'
,p_real_time_sql => l_real_time_sql
,p_batch_total_sql => l_batch_total_sql
,p_batch_incr_sql => l_batch_incr_sql
,p_batch_dea_sql => l_batch_dea_sql
,p_incr_reassign_sql => l_incr_reassign_sql
,p_use_total_for_dea_flag => null
,p_enabled_flag => 'Y'
,retcode => retcode
,errbuf => errbuf);
dbms_output.put_line('SALES/LEAD PROGRAM retcode : ' || retcode);
dbms_output.put_line('SALES/LEAD PROGRAM errbuf : ' || errbuf);
```

3. Run the R12LeadTT.sql file.

```
SQL> @R12LeadTT.sql /
```

Create the Custom Matching Attribute

Create the CMA). The basis of this script is to populate all the necessary information in the following tables:

JTF_SEEDED_QUAL_ALL_B

JTF_SEEDED_QUAL_ALL_TL

JTF_QUAL_USGS_ALL

1. Ensure that there is not an existing CMA record for the unique matching attribute ID that you will use for the new CMA.

```
set serveroutput ON SIZE 999999
DECLARE
    l_real_time_sql VARCHAR2(32000) := NULL;
    l_batch_total_sql VARCHAR2(32000) := NULL;
    l_batch_incr_sql VARCHAR2(32000) := NULL;
    l_batch_dea_sql VARCHAR2(32000) := NULL;
    l_incr_reassign_sql VARCHAR2(32000);
         retcode VARCHAR2(250);
           errbuf VARCHAR2(1000);

      TYPE namesarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE descarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE langarray IS VARRAY(5) OF VARCHAR2(50);
      names namesarray;
      descript descarray;
      lang langarray;
      total INTEGER;
 BEGIN
    BEGIN
        DELETE FROM jtf_seeded_qual_all_b
        WHERE seeded_qual_id = -9100;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_seeded_qual_all_tl
        WHERE seeded_qual_id = -9100;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_qual_usgs_all
        WHERE qual_usg_id = -9100;
            EXCEPTION
                  WHEN no_data_found THEN
                    NULL;
            END;
      BEGIN
```

2. Create the meta-data for the CMA. Follow the comments provided in the script to enter the relevant CMA information.

```
/* Provide CMA details in at least one language, such as 'US'*/
/* To create a CMA in multi-language, provide those details in
'names', 'descript' and 'lang' as an array*/

            names := namesarray('Lead Status (CMA)');
            descript := descarray('Lead Status (CMA)');
            lang := langarray('US');
            total := names.count;

     FOR i in 1 .. total LOOP
/* The following meta-data will create the custom matching attribute
for the Lead Status.*/
       JTY_CUST_QUAL_PKG.Create_qual(
     p_seeded_qual_id  => -9100,
/* This number should correspond to the number used in the alias for
the attribute*/
     p_name            => names(i),
     p_description     => descript(i),
     p_language        => lang(i),

    ,/* Sales Usage: FK to
       JTF_SOURCES_ALL.SOURCE_ID*/
     p_source_id       => -1001

    ,/* Sales Lead Transaction Type: FK to
        JTF_QUAL_TYPE_USGS_ALL.QUAL_TYPE_USG_ID*/
     p_trans_type_id   => -1003

    ,p_enabled_flag    => 'N'

    ,/* QUAL_RELATION_FACTOR
     This needs to be the next prime number greater than
     the value from the following SQL:

     SELECT MAX(qual_relation_factor)
     FROM jtf_qual_usgs_all

     The first 1000 primes can be found at:
     http://primes.utm.edu/lists/small/1000.txt

     Custom qualifiers should start at the 303rd prime
     which is 1999. This is to keep 1st to 302nd primes
     for product development to seed qualifiers.
     */
     p_qual_relation_factor       => 2003


    /* The following set of meta-data setups determine
    ** the behaviour of the matching attribute in
    ** HTML and Excel UIs: how it is displayed; what
    ** is the LOV SQL; and, what comparison operators
    ** are supported.
    */

    ,/* CONVERT_TO_ID_FLAG: displayed qualifier value is a CHAR
        but stored qualifier value is stored as an internal id  */
     p_convert_to_id_flag         => 'N'

    ,/* DISPLAY_TYPE: display type on UI (NUMERIC/CHAR) */
     p_display_type               => 'CHAR'
    ,/* LOV SQL */
    p_html_lov_sql1              =>
     ' SELECT a.meaning col1_value, a.status_code col2_value ' ||
CHR(10) ||
```

```
' FROM as_statuses_vl a ' || CHR(10) ||
     ' WHERE a.enabled_flag = ''Y'' ' || CHR(10) ||
     ' AND a.lead_flag = ''Y'' ' || CHR(10) ||
     ' ORDER BY col1_value '

    ,p_html_lov_sql2              => null
    ,p_html_lov_sql3              => null
    ,p_display_sql1               => ' SELECT meaning FROM
as_statuses_vl
 WHERE enabled_flag = ''Y'' AND lead_flag = ''Y'' AND STATUS_CODE=
'
    ,p_display_sql2               => null
    ,p_display_sql3               => null
,p_hierarchy_type             => null

    ,/* Is the "=" operator supported? */
    p_equal_flag                  => 'Y'
    ,/* Is the "LIKE" operator supported? */
    p_like_flag                   => 'N'
    ,/* Is the "BETWEEN" operator supported? */
    p_between_flag                => 'N'

    /* Values table */
    /* matching attribute comparison operator */
    ,p_comparison_operator        => 'q9100_cp'
    ,p_low_value_char             => 'q9100_low_value_CHAR'
    ,p_high_value_char            => null
    ,p_low_value_char_id          => null
    ,p_low_value_number           => null
    ,p_high_value_number          => null
    ,p_interest_type_id           => null
    ,p_primary_interest_code_id   => null
    ,p_sec_interest_code_id       => null
    ,p_value1_id                  => null
    ,p_value2_id                  => null
    ,p_value3_id                  => null
    ,p_value4_id                  => null
    ,p_first_char                 => null
    ,p_currency_code              => null


    ,/* Transaction Type Alias/_TRANS table column mapping */
     p_qual_col1                  => 'Q9100_LEAD_STATUS'

    ,/* TCA Classification for derivation */
     p_alias_rule1                => null

    ,/* OP_EQL FOR BATCH-MODE ASSIGNMENT:
     ** Since the AS_SALES_LEADS.STATUS_CODE value is not passed to
the
     ** assignment API, we must the PK (SALES_LEAD_ID) to get the
     ** STATUS_CODE value from AS_SALES_LEADS and compare to the
     ** territory qualifier value
     */
     p_op_eql                     =>
      ' (A.Q9100_LEAD_STATUS = B.q9100_low_value_CHAR AND B.q9100_cp
= ''='')'
    ,p_op_like                    => null
    ,p_op_between                 => NULL
    ,p_op_common_where            => null

    ,p_real_time_select           =>
    'SELECT DISTINCT A.trans_object_id, A.trans_detail_object_id, A.
txn_date, B.terr_id, B.absolute_rank, B.top_level_terr_id, B.
num_winners'
```

```
,p_real_time_from             => null
    ,p_real_time_where            =>
    'WHERE '  || CHR(10) ||
    '  B.q9010_low_value_CHAR = A.Q9100_LEAD_STATUS ' || CHR(10) ||
    '  AND B.q9010_cp = ''='' ' || CHR(10) ||
    '  AND B.source_id = -1001 ' || CHR(10) ||
    '  AND A.txn_date between B.start_date and B.end_date'


    ,retcode                      => retcode
    ,errbuf                       => errbuf);

  dbms_output.put_line('Retcode : ' || retcode);
  dbms_output.put_line('Errbuf : ' || errbuf);

  ad_morg.replicate_seed_data(NULL, 'JTF', NULL);

EXIT WHEN retcode=2;
END LOOP;
END;
COMMIT;

exception
  when others then
    dbms_output.put_line('Retcode : ' || retcode);
    dbms_output.put_line('Errbuf : ' || errbuf);
    raise;
end;
/
```

3. Run this script.

CMA for Quote: Quote Source

This is an example of creating a custom matching attribute for Sales using the quote transaction and the quote source attribute.

Modify Script 1

Modify Script 1, page A-96 to retrieve the relevant quote transaction type SQL meta-data.

1. Set the source ID corresponding to the usage (Sales or Collections). The source ID -1001 corresponds to the Sales usage. To find the desired usage ID query JTF_SOURES_ALL.

   ```
   p_source_id      NUMBER := -1001;
   ```

2. Set the transaction type ID. The ID -1105 corresponds to the Quotetransaction type ID. To find the desired transaction type ID, query JTF_QUAL_TYPES_ALL

   ```
   p_trans_type_id  NUMBER := -1105;
   ```

3. Run the script and save the output to a file. Running this command will save the Quote transaction type SQL meta-data to the "R12QuoteTT.sql" file.

```
SQL>spool <output filename - e.g., "d:\R12QuoteTT.sql">
SQL> set serveroutput on size 999999
SQL> @Script1.sql /
```

Modify Transaction SQL Meta-Data

Modify the transaction type SQL meta-data (R12QuoteTT.sql,) to get the CMA's corresponding transaction attribute value.

1. Add the quote_source_code attribute to each of the following SQLs:
    - l_real_time_sql :=

```
'select  ' || CHR(10) ||

'   header.quote_header_id  trans_object_id  ' || CHR(10) ||

'  ,header.quote_header_id  trans_detail_object_id  ' || CHR(10)
||
'  ,upper(location.city) city  ' || CHR(10) ||

'  ,upper(location.postal_code) postal_code  ' || CHR(10) ||

'  ,upper(location.state) state  ' || CHR(10) ||

'  ,upper(location.province) province  ' || CHR(10) ||

'  ,upper(location.county) county  ' || CHR(10) ||

'  ,upper(location.country) country  ' || CHR(10) ||

'  ,header.cust_party_id party_id  ' || CHR(10) ||

'  ,header.sold_to_party_site_id party_site_id  ' || CHR(10) ||

'  ,upper(cust_contact.phone_area_code) area_code  ' || CHR(10)
||
'  ,upper(party.party_name) comp_name_range  ' || CHR(10) ||

'  ,header.cust_party_id partner_id  ' || CHR(10) ||

'  ,party.employees_total num_of_employees  ' || CHR(10) ||

'  ,upper(party.category_code) category_code  ' || CHR(10) ||

'  ,header.cust_party_id party_relationship_id  ' || CHR(10) ||

'  ,upper(party.sic_code_type||'': ''||party.sic_code) sic_code
' || CHR(10) ||
'  ,header.quote_header_id  squal_num50  ' || CHR(10) ||

'  ,UPPER(party.duns_number_c)    squal_char11  ' || CHR(10) ||

'  ,orgp.CURR_FY_POTENTIAL_REVENUE squal_num06  ' || CHR(10) ||

'  ,upper(orgp.PREF_FUNCTIONAL_CURRENCY) car_currency_code  ' ||
CHR(10) ||
'  ,l_txn_date txn_date  ' || CHR(10) ||
'  ,UPPER(party.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(header.quote_source_code)  Q9101_QUOTE_SOURCE ' ||

'from  aso_quote_headers_all  header,  ' || CHR(10) ||

'      hz_parties             party,  ' || CHR(10) ||

'      hz_party_sites         party_site,  ' || CHR(10) ||

'      HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'      hz_locations           location,  ' || CHR(10) ||

'      hz_contact_points      cust_contact  ' || CHR(10) ||

'where header.cust_party_id             = party.party_id  ' ||
CHR(10) ||
'and   header.sold_to_party_site_id     = party_site.
party_site_id  ' || CHR(10)
```

```
||

'and   party_site.location_id           = location.location_id  '
|| CHR(10) ||
'AND   party.party_id                   = orgp.party_id(+)  ' ||
CHR(10) ||
'AND   nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'and   cust_contact.owner_table_id(+)   = header.party_id  ' ||
CHR(10) ||
'and   cust_contact.owner_table_name(+) = ''HZ_PARTIES''  ' ||
CHR(10) ||
'and   cust_contact.contact_point_type(+) = ''PHONE''  ' || CHR
(10) ||
'and   cust_contact.status(+)           = ''A''  ' || CHR(10) ||

'and   cust_contact.primary_flag(+)     = ''Y''  ' || CHR(10) ||

'and   header.quote_header_id           = l_trans_object_id1  ';
```

- l_batch_total_sql :=

```
l_batch_total_sql :=

'select  ' || CHR(10) ||

'  -1001                                                 source_id,
' || CHR(10) ||
'  -1105
trans_object_type_id,  ' ||
CHR(10) ||

'  header.QUOTE_NUMBER
trans_object_id,  ' ||
CHR(10) ||

'  header.sold_to_party_site_id
trans_detail_object_id,  '
|| CHR(10) ||

'  header.order_id                                       order_id,  '
|| CHR(10) ||
'  header.quote_expiration_date
quote_expiration_date,  '
|| CHR(10) ||

'  header.party_id                                squal_num01,  '
|| CHR(10) ||
'  header.sold_to_party_site_id                          squal_num02,
' || CHR(10)
||

'  header.party_id                                       squal_num03,
' || CHR(10)
||

'  header.party_id                                       squal_num04,
' || CHR(10)
||

'  party.employees_total                                 squal_num05,
' || CHR(10)
||

'  orgp.CURR_FY_POTENTIAL_REVENUE                        squal_num06,
' || CHR(10)
||

'  header.quote_header_id                                squal_num50,
' || CHR(10)
||

'  upper(party.party_name)
squal_char01,  ' || CHR(10)
||

'  upper(location.city)
squal_char02,  ' || CHR(10)
||

'  upper(location.county)
squal_char03,  ' || CHR(10)
||

'  upper(location.state)
squal_char04,  ' || CHR(10)
||
```

```
'  upper(location.province)
squal_char05,  ' || CHR(10)
||

'  upper(location.postal_code)
squal_char06,  ' || CHR(10)
||

'  upper(location.country)
squal_char07,  ' || CHR(10)
||

'  upper(cust_contact.phone_area_code)
squal_char08,  ' || CHR(10)
||

'  upper(party.category_code)
squal_char09,  ' || CHR(10)
||

'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(party.duns_number_c)                   squal_char11,  '
|| CHR(10) ||
'  upper(orgp.PREF_FUNCTIONAL_CURRENCY)
squal_curc01,  ' || CHR(10)
||

'  upper(substr(party.party_name,1,1))              squal_fc01
' || CHR(10) ||
'  ,UPPER(party.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(header.quote_source_code)  Q9101_QUOTE_SOURCE ' ||
'from  aso_quote_headers_all  header,  ' || CHR(10) ||

'      hz_parties         party,  ' || CHR(10) ||

'      hz_party_sites     party_site,  ' || CHR(10) ||

'      HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'      hz_locations       location,  ' || CHR(10) ||

'      hz_contact_points  cust_contact  ' || CHR(10) ||

'where header.cust_party_id            = party.party_id  ' ||
CHR(10) ||
'and   header.sold_to_party_site_id    = party_site.
party_site_id  ' || CHR(10)
||

'and   party_site.location_id          = location.location_id  '
|| CHR(10) ||
'AND   party.party_id                  = orgp.party_id(+)  ' ||
CHR(10) ||
'AND   nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'and   cust_contact.owner_table_id(+)   = header.party_id  ' ||
CHR(10) ||
'and   cust_contact.owner_table_name(+) = ''HZ_PARTIES''  ' ||
CHR(10) ||
'and   contact_point_type(+)           = ''PHONE''  ' || CHR(10)
||
```

```
'and   cust_contact.status(+)          = ''A''  ' || CHR(10) ||
'and   primary_flag(+)                 = ''Y''  ' || CHR(10) ||
'and   header.max_version_flag         = ''Y''  ';
l_batch_dea_sql :=
NULL;
```

- l_batch_incr_sql :=

```
l_batch_incr_sql :=

'select  ' || CHR(10) ||

'  -1001                        source_id,  ' || CHR(10) ||

'  -1105                                    trans_object_type_id,
' || CHR(10)
||

'  header.QUOTE_NUMBER             trans_object_id,  ' || CHR(10)
||
'  header.sold_to_party_site_id
trans_detail_object_id,  ' ||
CHR(10) ||

'  header.order_id                 order_id,  ' || CHR(10) ||

'  header.quote_expiration_date        quote_expiration_date,  '
|| CHR(10) ||
'  header.party_id                       squal_num01,  ' || CHR
(10) ||
'  header.sold_to_party_site_id              squal_num02,  ' ||
CHR(10) ||
'  header.party_id                       squal_num03,  ' || CHR
(10) ||
'  header.party_id                       squal_num04,  ' || CHR
(10) ||
'  party.employees_total                     squal_num05,  ' ||
CHR(10) ||
'  orgp.CURR_FY_POTENTIAL_REVENUE                squal_num06,  '
|| CHR(10) ||
'  header.quote_header_id            squal_num50,  ' || CHR(10)
||
'  upper(party.party_name)             squal_char01,  ' || CHR
(10) ||
'  upper(location.city)                      squal_char02,  ' ||
CHR(10) ||
'  upper(location.county)                    squal_char03,  ' ||
CHR(10) ||
'  upper(location.state)                     squal_char04,  ' ||
CHR(10) ||
'  upper(location.province)                  squal_char05,  ' ||
CHR(10) ||
'  upper(location.postal_code)               squal_char06,  ' ||
CHR(10) ||
'  upper(location.country)                   squal_char07,  ' ||
CHR(10) ||
'  upper(cust_contact.phone_area_code)       squal_char08,  ' ||
CHR(10) ||
'  upper(party.category_code)                squal_char09,  ' ||
CHR(10) ||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(party.duns_number_c)            squal_char11,  ' || CHR
(10) ||
'  upper(orgp.PREF_FUNCTIONAL_CURRENCY)          squal_curc01,  '
|| CHR(10) ||
'  upper(substr(party.party_name,1,1))   squal_fc01  ' || CHR(10)
||
'  ,UPPER(party.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(header.quote_source_code)  Q9101_QUOTE_SOURCE ' ||
```

```
'from  aso_quote_headers_all  header,  ' || CHR(10) ||

'      hz_parties         party,  ' || CHR(10) ||

'      hz_party_sites     party_site,  ' || CHR(10) ||

'      HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'      hz_locations       location,  ' || CHR(10) ||

'      hz_contact_points  cust_contact,  ' || CHR(10) ||

'      aso_changed_quotes chng  ' || CHR(10) ||

'where header.party_id                  = party.party_id  ' ||
CHR(10) ||
'and   header.sold_to_party_site_id     = party_site.
party_site_id  ' || CHR(10)
||

'and   party_site.location_id           = location.location_id  '
|| CHR(10) ||
'AND   party.party_id                   = orgp.party_id(+)  ' ||
CHR(10) ||
'AND   nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'and   cust_contact.owner_table_id(+)   = header.party_id  ' ||
CHR(10) ||
'and   cust_contact.owner_table_name(+) = ''HZ_PARTIES''  ' ||
CHR(10) ||
'and   contact_point_type(+)            = ''PHONE''  ' || CHR(10)
||
'and   cust_contact.status(+)           = ''A''  ' || CHR(10) ||

'and   primary_flag(+)                  = ''Y''  ' || CHR(10) ||

'and   header.max_version_flag          = ''Y''  ' || CHR(10) ||

'and   chng.quote_number                = header.quote_number  '
|| CHR(10) ||
'and   chng.conc_request_id             = l_REQUEST_ID  ';
```

- l_batch_dea_sql := NO
- l_incr_reassign_sql :=

```
l_incr_reassign_sql :=

'select  ' || CHR(10) ||

'  -1001       source_id,  ' || CHR(10) ||

'  -1105                                     trans_object_type_id,
' || CHR(10)
||

'  header.QUOTE_NUMBER     trans_object_id,  ' || CHR(10) ||

'  header.order_id     order_id,  ' || CHR(10) ||

'  header.quote_expiration_date    quote_expiration_date,  ' ||
CHR(10) ||
'  header.sold_to_party_site_id
trans_detail_object_id,  ' ||
CHR(10) ||

'  header.party_id                        squal_num01,  ' || CHR
(10) ||
'  header.sold_to_party_site_id             squal_num02,  ' ||
CHR(10) ||
'  header.party_id                        squal_num03,  ' || CHR
(10) ||
'  header.party_id                        squal_num04,  ' || CHR
(10) ||
'  party.employees_total                     squal_num05,  ' ||
CHR(10) ||
'  orgp.CURR_FY_POTENTIAL_REVENUE                squal_num06,  '
|| CHR(10) ||
'  header.quote_header_id            squal_num50,  ' || CHR(10)
||
'  upper(party.party_name)                squal_char01,  ' || CHR
(10) ||
'  upper(location.city)                      squal_char02,  ' ||
CHR(10) ||
'  upper(location.county)                    squal_char03,  ' ||
CHR(10) ||
'  upper(location.state)                     squal_char04,  ' ||
CHR(10) ||
'  upper(location.province)                  squal_char05,  ' ||
CHR(10) ||
'  upper(location.postal_code)               squal_char06,  ' ||
CHR(10) ||
'  upper(location.country)                   squal_char07,  ' ||
CHR(10) ||
'  upper(cust_contact.phone_area_code)       squal_char08,  ' ||
CHR(10) ||
'  upper(party.category_code)                squal_char09,  ' ||
CHR(10) ||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(party.duns_number_c)             squal_char11,  ' || CHR
(10) ||
'  upper(orgp.PREF_FUNCTIONAL_CURRENCY)          squal_curc01,  '
|| CHR(10) ||
'  upper(substr(party.party_name,1,1))       squal_fc01  ' || CHR
(10) ||
'  ,UPPER(party.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(header.quote_source_code)  Q9101_QUOTE_SOURCE ' ||
```

```
'from  aso_quote_headers  header,  ' || CHR(10) ||

'      hz_parties         party,  ' || CHR(10) ||

'      hz_party_sites     party_site,  ' || CHR(10) ||

'      HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'      hz_locations       location,  ' || CHR(10) ||

'      hz_contact_points  cust_contact,  ' || CHR(10) ||

'      ( select distinct acc.quote_number quote_number  ' || CHR
(10) ||
'        from (select distinct terr_id terr_id  ' || CHR(10) ||

'              from JTY_CHANGED_TERRS  ' || CHR(10) ||

'              where tap_request_id = l_request_id) CHG_TERR,  '
|| CHR(10) ||
'             ASO_QUOTE_ACCESSES ACC,  ' || CHR(10) ||

'             ASO_TERRITORY_ACCESSES  TERR_ACC  ' || CHR(10) ||

'        where CHG_TERR.terr_id = TERR_ACC.territory_id  ' || CHR
(10) ||
'        and TERR_ACC.access_id = acc.access_id) chng  ' || CHR
(10) ||
'where header.party_id                  = party.party_id  ' ||
CHR(10) ||
'and   header.sold_to_party_site_id     = party_site.
party_site_id  ' || CHR(10)
||

'and   party_site.location_id           = location.location_id  '
|| CHR(10) ||
'AND   party.party_id                   = orgp.party_id(+)  ' ||
CHR(10) ||
'AND   nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'and   cust_contact.owner_table_id(+)   = header.party_id  ' ||
CHR(10) ||
'and   cust_contact.owner_table_name(+) = ''HZ_PARTIES''  ' ||
CHR(10) ||
'and   contact_point_type(+)            = ''PHONE''  ' || CHR(10)
||
'and   cust_contact.status(+)           = ''A''  ' || CHR(10) ||

'and   primary_flag(+)                  = ''Y''  ' || CHR(10) ||

'and   header.max_version_flag          = ''Y''  ' || CHR(10) ||

'and   chng.quote_number                = header.quote_number  ';
```

2. Modify the R12QuoteTT.sql file with the appropriate package parameters to make the SQL meta-data change.

```
JTY_TRANS_USG_PGM_SQL_PKG.Insert_Row(
p_source_id => -1001
,p_trans_type_id => -1105
,p_program_name => 'SALES/QUOTE PROGRAM'
,p_version_name => '12-JAN-06: CMA QUOTE SOURCE ADDED'
,p_real_time_sql => l_real_time_sql
,p_batch_total_sql => l_batch_total_sql
,p_batch_incr_sql => l_batch_incr_sql
,p_batch_dea_sql => l_batch_dea_sql
,p_incr_reassign_sql => l_incr_reassign_sql
,p_use_total_for_dea_flag => null
,p_enabled_flag => 'Y'
,retcode => retcode
,errbuf => errbuf);
dbms_output.put_line('SALES/QUOTE PROGRAM retcode : ' || retcode);
dbms_output.put_line('SALES/QUOTE PROGRAM errbuf : ' || errbuf);
```

3. Run the R12QuoteTT.sql file.

```
SQL> @R12QuoteTT.sql /
```

Create the Custom Matching Attribute

Create the CMA. The basis of this script is to populate all the necessary information in the following tables:

JTF_SEEDED_QUAL_ALL_B

JTF_SEEDED_QUAL_ALL_TL

JTF_QUAL_USGS_ALL

1. Ensure that there is not an existing CMA record for the unique matching attribute ID that you will use for the new CMA.

```
set serveroutput ON SIZE 999999
DECLARE
    l_real_time_sql VARCHAR2(32000) := NULL;
    l_batch_total_sql VARCHAR2(32000) := NULL;
    l_batch_incr_sql VARCHAR2(32000) := NULL;
    l_batch_dea_sql VARCHAR2(32000) := NULL;
    l_incr_reassign_sql VARCHAR2(32000);
         retcode VARCHAR2(250);
           errbuf VARCHAR2(1000);

      TYPE namesarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE descarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE langarray IS VARRAY(5) OF VARCHAR2(50);
      names namesarray;
      descript descarray;
      lang langarray;
      total INTEGER;
 BEGIN
    BEGIN
        DELETE FROM jtf_seeded_qual_all_b
        WHERE seeded_qual_id = -9101;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_seeded_qual_all_tl
        WHERE seeded_qual_id = -9101;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_qual_usgs_all
        WHERE qual_usg_id = -9101;
            EXCEPTION
                  WHEN no_data_found THEN
                    NULL;
            END;
      BEGIN
```

2. Create the meta-data for the CMA. Follow the comments provided in the script to enter the relevant CMA information.

```
/* Provide CMA details in at least one language, such as 'US'*/
/* To create a CMA in multi-language, provide those details in
'names', 'descript' and 'lang' as an array*/

            names := namesarray('Quote Source (CMA)');
            descript := descarray('Custom Matching Attribute for
Source Marketing Campaign of a Quote');
            lang := langarray('US');
            total := names.count;

     FOR i in 1 .. total LOOP
/* The following meta-data will create the custom matching attribute
for the Quote Source.*/
     JTY_CUST_QUAL_PKG.Create_qual(
     p_seeded_qual_id     => -9101,
/* This number should correspond to the number used in the alias for
the attribute*/
     p_name               => names(i),
     p_description        => descript(i),
     p_language           => lang(i),

,/* Sales Usage: FK to
       JTF_SOURCES_ALL.SOURCE_ID*/
     p_source_id          => -1001
,/* Sales Lead Transaction Type: FK to
        JTF_QUAL_TYPE_USGS_ALL.QUAL_TYPE_USG_ID*/
     p_trans_type_id      => -1105

    ,p_enabled_flag       => 'N'

    ,/* QUAL_RELATION_FACTOR
     This needs to be the next prime number greater than
     the value from the following SQL:

     SELECT MAX(qual_relation_factor)
     FROM jtf_qual_usgs_all

     The first 1000 primes can be found at:
     http://primes.utm.edu/lists/small/1000.txt

     Custom qualifiers should start at the 303rd prime
     which is 1999. This is to keep 1st to 302nd primes
     for product development to seed qualifiers.
     */
     p_qual_relation_factor       => 2017


    /* The following set of meta-data setups determine
    ** the behaviour of the matching attribute in
    ** HTML and Excel UIs: how it is displayed; what
    ** is the LOV SQL; and, what comparison operators
    ** are supported.
    */

    ,/* CONVERT_TO_ID_FLAG: displayed qualifier value is a CHAR
        but stored qualifier value is stored as an internal id  */
     p_convert_to_id_flag         => 'N'

    ,/* DISPLAY_TYPE: display type on UI (NUMERIC/CHAR) */
     p_display_type               => 'CHAR'
    ,/* LOV SQL */
    p_html_lov_sql1              =>
     ' SELECT c.campaign_name || '': '' || b.status_code || '': ''
|| a.source_code || '': '' || c.description col1_value, '  || CHR
```

```
(10) ||
     ' TO_CHAR(a.source_code_id) col2_value ' || CHR(10) ||
     ' FROM ams_source_codes a, ams_campaigns_all_b b,
ams_campaigns_all_tl c  ' || CHR(10) ||
     ' WHERE b.source_code = a.source_code ' || CHR(10) ||
     ' AND a.arc_source_code_for = ''CAMP'' ' || CHR(10) ||
     ' AND b.campaign_id = c.campaign_id ' || CHR(10) ||
     ' AND c.LANGUAGE = userenv(''LANG'') ' || CHR(10) ||
     ' ORDER BY b.status_code, c.campaign_name '

    ,p_html_lov_sql2             => null
    ,p_html_lov_sql3             => null
    ,p_display_sql1              => null
    ,p_display_sql2              => null
    ,p_display_sql3              => null
    ,p_hierarchy_type            => null

    ,/* Is the "=" operator supported? */
    p_equal_flag                 => 'Y'
    ,/* Is the "LIKE" operator supported? */
    p_like_flag                  => 'N'
    ,/* Is the "BETWEEN" operator supported? */
    p_between_flag               => 'N'

    /* Values table */
    /* matching attribute comparison operator */
    ,p_comparison_operator       => 'q9101_cp'
    ,p_low_value_char            => 'q9101_low_value_char'
    ,p_high_value_char           => null
    ,p_low_value_char_id         => null
    ,p_low_value_number          => null
    ,p_high_value_number         => null
    ,p_interest_type_id          => null
    ,p_primary_interest_code_id  => null
    ,p_sec_interest_code_id      => null
    ,p_value1_id                 => null
    ,p_value2_id                 => null
    ,p_value3_id                 => null
    ,p_value4_id                 => null
    ,p_first_char                => null
    ,p_currency_code             => null


    ,/* Transaction Type Alias/_TRANS table column mapping */
     p_qual_col1                 => 'Q9101_QUOTE_SOURCE'

    ,/* TCA Classification for derivation */
     p_alias_rule1               => null

    ,/* OP_EQL FOR BATCH-MODE ASSIGNMENT:
     ** Since the AS_SALES_LEADS.STATUS_CODE value is not passed to
the
     ** assignment API, we must the PK (SALES_LEAD_ID) to get the
     ** STATUS_CODE value from AS_SALES_LEADS and compare to the
     ** territory qualifier value
     */
     p_op_eql                    =>
      ' (A.Q9101_QUOTE_SOURCE = B.q9101_low_value_CHAR AND B.
q9101_cp = ''='')'
    ,p_op_like                   => null
    ,p_op_between                => NULL
    ,p_op_common_where           => null

    ,p_real_time_select          =>
    'SELECT DISTINCT A.trans_object_id, A.trans_detail_object_id, A.
```

```
txn_date, B.terr_id, B.absolute_rank, B.top_level_terr_id, B.
num_winners'
    ,p_real_time_from            => null
    ,p_real_time_where           =>
    'WHERE '  || CHR(10) ||
    '  B.q9101_low_value_CHAR = A.Q9101_QUOTE_SOURCE ' || CHR(10) ||
    '  AND B.q9101_cp = ''='' ' || CHR(10) ||
    '  AND B.source_id = -1001 ' || CHR(10) ||
    '  AND A.txn_date between B.start_date and B.end_date'


    ,retcode                     => retcode
    ,errbuf                      => errbuf);

  dbms_output.put_line('Retcode : ' || retcode);
  dbms_output.put_line('Errbuf : ' || errbuf);

  ad_morg.replicate_seed_data(NULL, 'JTF', NULL);

EXIT WHEN retcode=2;
END LOOP;
END;
COMMIT;

exception
  when others then
    dbms_output.put_line('Retcode : ' || retcode);
    dbms_output.put_line('Errbuf : ' || errbuf);
    raise;
end;
/
```

3. Run this script.

CMA for Proposal: Proposal Status

This is an example of creating a custom matching attribute for Sales using the proposal transaction and the proposal status attribute.

Modify Script 1

Modify Script 1, page A-96 to retrieve the relevant proposal transaction type SQL meta-data.

1. Set the source ID corresponding to the usage (Sales or Collections). The source ID -1001 corresponds to the Sales usage. To find the desired usage ID query JTF_SOURES_ALL.

   ```
   p_source_id      NUMBER := -1001;
   ```

2. Set the transaction type ID. The ID -1106 corresponds to the Proposal transaction type ID. To find the desired transaction type ID, query JTF_QUAL_TYPES_ALL

   ```
   p_trans_type_id  NUMBER := -1106;
   ```

3. Run the script and save the output to a file. Running this command will save the Proposal transaction type SQL meta-data to the "R12ProposalTT.sql" file.

```
SQL>spool <output filename - e.g., "d:\R12ProposalTT.sql">
SQL> set serveroutput on size 999999
SQL> @Script1.sql /
```

Modify Transaction SQL Meta-Data

Modify the transaction type SQL meta-data (R12AccountTT.sql) to get the CMA's corresponding transaction attribute value.

1. Add the proposal_status attribute to each of the following SQLs:
   - l_real_time_sql :=

```
'Select  ' || CHR(10) ||

'   prop.proposal_id TRANS_OBJECT_ID  ' || CHR(10) ||

'  ,SITE.party_site_id trans_detail_object_id  ' || CHR(10) ||

'  ,UPPER(LOC.CITY) city  ' || CHR(10) ||

'  ,UPPER(LOC.POSTAL_CODE) postal_code  ' || CHR(10) ||

'  ,UPPER(LOC.STATE) state  ' || CHR(10) ||

'  ,UPPER(LOC.PROVINCE) province  ' || CHR(10) ||

'  ,UPPER(LOC.COUNTY) county  ' || CHR(10) ||

'  ,UPPER(LOC.COUNTRY) country  ' || CHR(10) ||

'  ,hzp.party_id party_id  ' || CHR(10) ||

'  ,SITE.party_site_id party_site_id  ' || CHR(10) ||

'  ,UPPER(CNTPNT.phone_area_code) area_code  ' || CHR(10) ||

'  ,UPPER(hzp.party_name) comp_name_range  ' || CHR(10) ||

'  ,hzp.party_id partner_id  ' || CHR(10) ||

'  ,hzp.employees_total num_of_employees  ' || CHR(10) ||

'  ,UPPER(hzp.category_code) category_code  ' || CHR(10) ||

'  ,hzp.party_id party_relationship_id  ' || CHR(10) ||

'  ,UPPER(hzp.sic_code_type||'': ''||hzp.sic_code) sic_code  ' ||
CHR(10) ||
'  ,ORGPRO.curr_fy_potential_revenue squal_num06  ' || CHR(10) ||

'  ,UPPER(ORGPRO.pref_functional_currency) car_currency_code  '
|| CHR(10) ||
'  ,hzp.party_id squal_num01  ' || CHR(10) ||

'  ,UPPER(hzp.duns_number_c) SQUAL_CHAR11  ' || CHR(10) ||

'  ,l_txn_date txn_date  ' || CHR(10) ||

'  ,UPPER(hzp.party_type)
Q9010_PARTY_TYPE  ' ||   CHR(10) ||

'  ,UPPER(PROP.proposal_status)  Q9102_PROPOSAL_STATUS'   ||
CHR(10) ||
'FROM HZ_PARTY_SITES   SITE,  ' || CHR(10) ||

'    HZ_LOCATIONS   LOC,  ' || CHR(10) ||

'    HZ_PARTIES   HZP,  ' || CHR(10) ||

'    HZ_ORGANIZATION_PROFILES  ORGPRO ,  ' || CHR(10) ||

'    HZ_CONTACT_POINTS CNTPNT,  ' || CHR(10) ||

'    prp_proposals   PROP  ' || CHR(10) ||

'WHERE CNTPNT.owner_table_id(+) = HZP.party_id  ' || CHR(10) ||
```

```
'and CNTPNT.status(+)=''A''  ' || CHR(10) ||

'and CNTPNT.primary_flag(+)=''Y''  ' || CHR(10) ||

'and CNTPNT.contact_point_type(+)=''PHONE''  ' || CHR(10) ||

'and ORGPRO.party_id(+) = HZP.party_id  ' || CHR(10) ||

'and nvl(ORGPRO.effective_end_date(+),sysdate+1) > sysdate  ' ||
CHR(10) ||
'AND PROP.PARTY_ID = HZP.PARTY_ID  ' || CHR(10) ||

'AND HZP.party_type in (''PERSON'', ''ORGANIZATION'')  ' || CHR
(10) ||
'AND HZP.status = ''A''  ' || CHR(10) ||

'AND  SITE.status = ''A''  ' || CHR(10) ||

'and SITE.party_id = hzp.party_id  ' || CHR(10) ||

'and LOC.location_id = SITE.location_id  ' || CHR(10) ||

'AND PROP.Proposal_id = l_trans_object_id1  ' || CHR(10) ||

'UNION ALL  ' || CHR(10) ||

'select  ' || CHR(10) ||

'   prop.proposal_id TRANS_OBJECT_ID  ' || CHR(10) ||

'  ,to_number(NULL) trans_detail_object_id  ' || CHR(10) ||

'  ,To_char(null) city  ' || CHR(10) ||

'  ,To_char(null) postal_code  ' || CHR(10) ||

'  ,To_char(null) state  ' || CHR(10) ||

'  ,To_char(null) province  ' || CHR(10) ||

'  ,To_char(null) county  ' || CHR(10) ||

'  ,To_char(null) country  ' || CHR(10) ||

'  ,hzp.party_id party_id  ' || CHR(10) ||

'  ,to_number(NULL) party_site_id  ' || CHR(10) ||

'  ,UPPER(CNTPNT.phone_area_code) area_code  ' || CHR(10) ||

'  ,UPPER(hzp.party_name) comp_name_range  ' || CHR(10) ||

'  ,hzp.party_id partner_id  ' || CHR(10) ||

'  ,hzp.employees_total num_of_employees  ' || CHR(10) ||

'  ,UPPER(hzp.category_code) category_code  ' || CHR(10) ||

'  ,hzp.party_id party_relationship_id  ' || CHR(10) ||

'  ,UPPER(hzp.sic_code_type||'': ''||hzp.sic_code) sic_code  ' ||
CHR(10) ||
'  ,ORGPRO.curr_fy_potential_revenue squal_num06  ' || CHR(10) ||

'  ,UPPER(ORGPRO.pref_functional_currency) car_currency_code  '
```

```
|| CHR(10) ||
'  ,hzp.party_id squal_num01  ' || CHR(10) ||

'  ,UPPER(hzp.duns_number_c) SQUAL_CHAR11  ' || CHR(10) ||

'  ,l_txn_date txn_date  ' || CHR(10) ||

'  ,UPPER(hzp.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(PROP.proposal_status)  Q9102_PROPOSAL_STATUS'   ||
CHR(10) ||
CHR(10) ||

'from HZ_ORGANIZATION_PROFILES  ORGPRO ,  ' || CHR(10) ||

'   HZ_CONTACT_POINTS CNTPNT,  ' || CHR(10) ||

'   prp_proposals   PROP,  ' || CHR(10) ||

'   hz_parties hzp  ' || CHR(10) ||

'where CNTPNT.owner_table_id(+) = HZP.party_id  ' || CHR(10) ||

'and CNTPNT.status(+)=''A''  ' || CHR(10) ||

'and CNTPNT.primary_flag(+)=''Y''  ' || CHR(10) ||

'and CNTPNT.contact_point_type(+)=''PHONE''  ' || CHR(10) ||

'and ORGPRO.party_id(+) = HZP.party_id  ' || CHR(10) ||

'and nvl(ORGPRO.effective_end_date(+),sysdate+1) > sysdate  ' ||
CHR(10) ||
'AND PROP.PARTY_ID = HZP.PARTY_ID  ' || CHR(10) ||

'AND HZP.party_type in (''PERSON'', ''ORGANIZATION'')  ' || CHR
(10) ||
'and HZP.status = ''A''  ' || CHR(10) ||

'AND PROP.Proposal_id = l_trans_object_id1  ';
```

- l_batch_total_sql :=

```
l_batch_total_sql :=

NULL;

l_batch_incr_sql :=

NULL;

l_batch_dea_sql :=

NULL;

l_incr_reassign_sql :=

NULL;
```

- l_batch_incr_sql :=

- l_batch_dea_sql := NO
- l_incr_reassign_sql :=

2. Modify the R12ProposalTT.sql file with the appropriate package parameters to make the SQL meta-data change.

```
JTY_TRANS_USG_PGM_SQL_PKG.Insert_Row(

p_source_id => -1001

,p_trans_type_id => -1106

,p_program_name => 'SALES/PROPOSAL PROGRAM'

,p_version_name => '12-JAN-06: CMA PROPOSAL STATUS ADDED'

,p_real_time_sql => l_real_time_sql

,p_batch_total_sql => l_batch_total_sql

,p_batch_incr_sql => l_batch_incr_sql

,p_batch_dea_sql => l_batch_dea_sql

,p_incr_reassign_sql => l_incr_reassign_sql

,p_use_total_for_dea_flag => null

,p_enabled_flag => 'Y'

,retcode => retcode

,errbuf => errbuf);

dbms_output.put_line('SALES/PROPOSAL PROGRAM retcode : ' ||
retcode);
dbms_output.put_line('SALES/PROPOSAL PROGRAM errbuf : ' || errbuf);
```

3. Run the R12ProposalTT.sql file.

```
SQL> @R12ProposalTT.sql /
```

Create the Custom Matching Attribute

Create the CMA. The basis of this script is to populate all the necessary information in the following tables:

JTF_SEEDED_QUAL_ALL_B

JTF_SEEDED_QUAL_ALL_TL

JTF_QUAL_USGS_ALL

1. Ensure that there is not an existing CMA record for the unique matching attribute ID that you will use for the new CMA.

```
set serveroutput ON SIZE 999999
DECLARE
    l_real_time_sql VARCHAR2(32000) := NULL;
    l_batch_total_sql VARCHAR2(32000) := NULL;
    l_batch_incr_sql VARCHAR2(32000) := NULL;
    l_batch_dea_sql VARCHAR2(32000) := NULL;
    l_incr_reassign_sql VARCHAR2(32000);
         retcode VARCHAR2(250);
           errbuf VARCHAR2(1000);

      TYPE namesarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE descarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE langarray IS VARRAY(5) OF VARCHAR2(50);
      names namesarray;
      descript descarray;
      lang langarray;
      total INTEGER;
 BEGIN
    BEGIN
        DELETE FROM jtf_seeded_qual_all_b
        WHERE seeded_qual_id = -9102;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_seeded_qual_all_tl
        WHERE seeded_qual_id = -9102;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_qual_usgs_all
        WHERE qual_usg_id = -9102;
            EXCEPTION
                  WHEN no_data_found THEN
                    NULL;
            END;
      BEGIN
```

2. Create the meta-data for the CMA. Follow the comments provided in the script to enter the relevant CMA information.

```
/* Provide CMA details in at least one language, such as 'US'*/
/* To create a CMA in multi-language, provide those details in
'names', 'descript' and 'lang' as an array*/

            names := namesarray('Proposal Status (CMA)');
            descript := descarray('Custom Matching Attribute for
Proposal Status');
            lang := langarray('US');
            total := names.count;

     FOR i in 1 .. total LOOP
 /* The following meta-data will create the custom matching
attribute for the Proposal Status.*/
        JTY_CUST_QUAL_PKG.Create_qual(
     p_seeded_qual_id     => -9102,
/* This number should correspond to the number used in the alias for
the attribute*/
     p_name               => names(i),
     p_description        => descript(i),
     p_language           => lang(i),

    ,/* Sales Usage: FK to
       JTF_SOURCES_ALL.SOURCE_ID*/
     p_source_id          => -1001

    ,/* Sales Lead Transaction Type: FK to
        JTF_QUAL_TYPE_USGS_ALL.QUAL_TYPE_USG_ID*/
     p_trans_type_id      => -1106

    ,p_enabled_flag       => 'N'

    ,/* QUAL_RELATION_FACTOR
     This needs to be the next prime number greater than
     the value from the following SQL:

     SELECT MAX(qual_relation_factor)
     FROM jtf_qual_usgs_all

     The first 1000 primes can be found at:
     http://primes.utm.edu/lists/small/1000.txt

     Custom qualifiers should start at the 303rd prime
     which is 1999. This is to keep 1st to 302nd primes
     for product development to seed qualifiers.
     */
     p_qual_relation_factor       => 2029


    /* The following set of meta-data setups determine
    ** the behaviour of the matching attribute in
    ** HTML and Excel UIs: how it is displayed; what
    ** is the LOV SQL; and, what comparison operators
    ** are supported.
    */

    ,/* CONVERT_TO_ID_FLAG: displayed qualifier value is a CHAR
        but stored qualifier value is stored as an internal id  */
     p_convert_to_id_flag         => 'N'

    ,/* DISPLAY_TYPE: display type on UI (NUMERIC/CHAR) */
     p_display_type               => 'CHAR'
    ,/* LOV SQL */
    p_html_lov_sql1              => NULL
    ,p_html_lov_sql2              => null
```

```
,p_html_lov_sql3              => null
    ,p_display_sql1               => null
    ,p_display_sql2               => null
    ,p_display_sql3               => null
    ,p_hierarchy_type             => null

    ,/* Is the "=" operator supported? */
    p_equal_flag                  => 'Y'
    ,/* Is the "LIKE" operator supported? */
    p_like_flag                   => 'N'
    ,/* Is the "BETWEEN" operator supported? */
    p_between_flag                => 'N'

    /* Values table */
    /* matching attribute comparison operator */
    ,p_comparison_operator        => 'q9102_cp'
    ,p_low_value_char             => 'q9102_low_value_CHAR'
    ,p_high_value_char            => null
    ,p_low_value_char_id          => null
    ,p_low_value_number           => null
    ,p_high_value_number          => null
    ,p_interest_type_id           => null
    ,p_primary_interest_code_id   => null
    ,p_sec_interest_code_id       => null
    ,p_value1_id                  => null
    ,p_value2_id                  => null
    ,p_value3_id                  => null
    ,p_value4_id                  => null
    ,p_first_char                 => null
    ,p_currency_code              => null


    ,/* Transaction Type Alias/_TRANS table column mapping */
     p_qual_col1                  => 'Q9102_PROPOSAL_STATUS'

    ,/* TCA Classification for derivation */
     p_alias_rule1                => null

    ,/* OP_EQL FOR BATCH-MODE ASSIGNMENT:
     ** Since the AS_SALES_LEADS.STATUS_CODE value is not passed to
the
     ** assignment API, we must the PK (SALES_LEAD_ID) to get the
     ** STATUS_CODE value from AS_SALES_LEADS and compare to the
     ** territory qualifier value
     */
     p_op_eql                     =>
      ' (A.Q9102_PROPOSAL_STATUS = B.q9102_low_value_CHAR AND B.
q9102_cp = ''='')'
    ,p_op_like                    => null
    ,p_op_between                 => NULL
    ,p_op_common_where            => null

    ,p_real_time_select           =>
    'SELECT DISTINCT A.trans_object_id, A.trans_detail_object_id, A.
txn_date, B.terr_id, B.absolute_rank, B.top_level_terr_id, B.
num_winners'
    ,p_real_time_from             => null
    ,p_real_time_where            =>
    'WHERE '  || CHR(10) ||
    '  B.q9102_low_value_CHAR = A.Q9102_PROPOSAL_STATUS ' || CHR(10)
||
    '  AND B.q9102_cp = ''='' ' || CHR(10) ||
    '  AND B.source_id = -1001 ' || CHR(10) ||
    '  AND A.txn_date between B.start_date and B.end_date'
```

```
    ,retcode                      => retcode
    ,errbuf                       => errbuf);

  dbms_output.put_line('Retcode : ' || retcode);
  dbms_output.put_line('Errbuf : ' || errbuf);

  ad_morg.replicate_seed_data(NULL, 'JTF', NULL);

EXIT WHEN retcode=2;
END LOOP;
END;
COMMIT;

exception
  when others then
    dbms_output.put_line('Retcode : ' || retcode);
    dbms_output.put_line('Errbuf : ' || errbuf);
    raise;
end;
/
```

3. Run this script.

CMA for Opportunity: Budget Status

This is an example of creating a custom matching attribute for Sales using the opportunity transaction and the budget status attribute.

Modify Script 1

Modify Script 1, page A-96 to retrieve the relevant Opportunity transaction type SQL meta-data.

1. Set the source ID corresponding to the usage (Sales or Collections). The source ID -1001 corresponds to the Sales usage. To find the desired usage ID query JTF_SOURES_ALL.

   ```
   p_source_id      NUMBER := -1001;
   ```

2. Set the transaction type ID. The ID -1004 corresponds to the Opportunity transaction type ID. To find the desired transaction type ID, query JTF_QUAL_TYPES_ALL

   ```
   p_trans_type_id  NUMBER := -1004;
   ```

3. Run the script and save the output to a file. Running this command will save the Opportunity transaction type SQL meta-data to the "R12OpportunityTT.sql" file.

   ```
   SQL>spool <output filename - e.g., "d:\R12OpportunityTT.sql">
   SQL> set serveroutput on size 999999
   SQL> @Script1.sql /
   ```

Modify Transaction SQL Meta-Data

Modify the transaction type SQL meta-data (R12OpportunityTT.sql) to get the CMA's

corresponding transaction attribute value.

1. Add the budget_status_code attribute to each of the following SQLs:
   - l_real_time_sql :=

```
'SELECT  ' || CHR(10) ||

'  OPP.LEAD_ID
trans_object_id,  ' ||
CHR(10) ||

'  -1
trans_detail_object_id,  '
|| CHR(10) ||

'  OPP.LEAD_ID                                          lead_id,  '
|| CHR(10) ||
'  -1                                                   lead_line_id,
' || CHR(10)
||

'  TO_CHAR(NULL)                                        city,  ' ||
CHR(10) ||
'  TO_CHAR(NULL)                                        postal_code,
' || CHR(10)
||

'  TO_CHAR(NULL)                                        state,  ' ||
CHR(10) ||
'  TO_CHAR(NULL)                                        province,  '
|| CHR(10) ||
'  TO_CHAR(NULL)                                        county,  ' ||
CHR(10) ||
'  TO_CHAR(NULL)                                        country,  '
|| CHR(10) ||
'  PARTY.PARTY_ID                                       party_id,  '
|| CHR(10) ||
'  TO_NUMBER(NULL)
party_site_id,  ' || CHR(10)
||

'  upper(PARTY.PRIMARY_PHONE_AREA_CODE)                 area_code,  '
|| CHR(10) ||
'  upper(PARTY.PARTY_NAME)
comp_name_range,  ' ||
CHR(10) ||

'  PARTY.PARTY_ID                                       partner_id,
' || CHR(10) ||
'  PARTY.EMPLOYEES_TOTAL
num_of_employees,  ' ||
CHR(10) ||

'  upper(PARTY.CATEGORY_CODE)
category_code,  ' || CHR(10)
||

'  PARTY.PARTY_ID
party_relationship_id,  ' ||
CHR(10) ||

'  upper(party.sic_code_type||'': ''||party.sic_code) sic_code,
' || CHR(10) ||
'  OPP.TOTAL_AMOUNT                                     total_amount,
' || CHR(10)
||

'  upper(OPP.CURRENCY_CODE)
currency_code,  ' || CHR(10)
||
```

```
'  NVL(OPP.DECISION_DATE, OPP.LAST_UPDATE_DATE)     pricing_date,
' || CHR(10)
||

'  upper(OPP.CHANNEL_CODE)                          channel_code,
' || CHR(10)
||

'  OPP.SOURCE_PROMOTION_ID                          squal_num40,
' || CHR(10)
||

'  upper(OPP.STATUS)                                squal_char41,
' || CHR(10)
||

'  ORGP.CURR_FY_POTENTIAL_REVENUE                   squal_num06,
' || CHR(10)
||

'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)
car_currency_code,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)                       squal_char11,
' || CHR(10)
||

'  l_txn_date                                       txn_date  '
|| CHR(10) ||
' ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
' ,UPPER(OPP.budget_status_code)
Q9103_BUDGET_STATUS ' ||
'FROM  ' || CHR(10) ||

'  AS_LEADS OPP,  ' || CHR(10) ||

'  HZ_PARTIES PARTY,  ' || CHR(10) ||

'  HZ_ORGANIZATION_PROFILES ORGP  ' || CHR(10) ||

'WHERE  OPP.ADDRESS_ID IS NULL  ' || CHR(10) ||

'AND    OPP.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    OPP.lead_id = l_trans_object_id1 UNION ALL SELECT  ' ||
CHR(10) ||
'  OPP.LEAD_ID
trans_object_id,  ' ||
CHR(10) ||

'  -1
trans_detail_object_id,  '
|| CHR(10) ||

'  OPP.LEAD_ID                                      lead_id,  '
|| CHR(10) ||
'  -1                                               lead_line_id,
```

```
' || CHR(10)
||

'  upper(LOC.CITY)                                    city,  ' ||
CHR(10) ||
'  upper(LOC.POSTAL_CODE)                             postal_code,
' || CHR(10)
||

'  upper(LOC.STATE)                                   state,  ' ||
CHR(10) ||
'  upper(LOC.PROVINCE)                                province,  '
|| CHR(10) ||
'  upper(LOC.COUNTY)                                  county,  ' ||
CHR(10) ||
'  upper(LOC.COUNTRY)                                 country,  '
|| CHR(10) ||
'  PARTY.PARTY_ID                                     party_id,  '
|| CHR(10) ||
'  SITE.PARTY_SITE_ID
party_site_id,  ' || CHR(10)
||

'  upper(PHONE.AREA_CODE)                             area_code,  '
|| CHR(10) ||
'  upper(PARTY.PARTY_NAME)
comp_name_range,  ' ||
CHR(10) ||

'  PARTY.PARTY_ID                                     partner_id,
' || CHR(10) ||
'  PARTY.EMPLOYEES_TOTAL
num_of_employees,  ' ||
CHR(10) ||

'  upper(PARTY.CATEGORY_CODE)
category_code,  ' || CHR(10)
||

'  PARTY.PARTY_ID
party_relationship_id,  ' ||
CHR(10) ||

'  upper(party.sic_code_type||'': ''||party.sic_code) sic_code,
' || CHR(10) ||
'  OPP.TOTAL_AMOUNT                                   total_amount,
' || CHR(10)
||

'  upper(OPP.CURRENCY_CODE)
currency_code,  ' || CHR(10)
||

'  NVL(OPP.DECISION_DATE, OPP.LAST_UPDATE_DATE)     pricing_date,
' || CHR(10)
||

'  upper(OPP.CHANNEL_CODE)                            channel_code,
' || CHR(10)
||

'  OPP.SOURCE_PROMOTION_ID                            squal_num40,
' || CHR(10)
||
```

```
'  upper(OPP.STATUS)                                    squal_char41,
' || CHR(10)
||

'  ORGP.CURR_FY_POTENTIAL_REVENUE                       squal_num06,
' || CHR(10)
||

'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)
car_currency_code,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)                           squal_char11,
' || CHR(10)
||

'  l_txn_date                                           txn_date  '
|| CHR(10) ||
'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(OPP.budget_status_code)
Q9103_BUDGET_STATUS ' ||
'FROM  ' || CHR(10) ||

'  AS_LEADS_ALL OPP,  ' || CHR(10) ||

'  AS_PARTY_PHONES_V PHONE,  ' || CHR(10) ||

'  HZ_PARTY_SITES SITE,  ' || CHR(10) ||

'  HZ_LOCATIONS LOC,  ' || CHR(10) ||

'  HZ_PARTIES PARTY,  ' || CHR(10) ||

'  HZ_ORGANIZATION_PROFILES ORGP  ' || CHR(10) ||

'WHERE  PHONE.OWNER_TABLE_NAME(+) = ''HZ_PARTY_SITES''  ' || CHR
(10) ||
'AND    PHONE.PRIMARY_FLAG(+) = ''Y''  ' || CHR(10) ||

'AND    PHONE.STATUS_CODE(+) <> ''I''  ' || CHR(10) ||

'AND    SITE.PARTY_SITE_ID = PHONE.OWNER_TABLE_ID(+)  ' || CHR
(10) ||
'AND    OPP.ADDRESS_ID = SITE.PARTY_SITE_ID  ' || CHR(10) ||

'AND    SITE.LOCATION_ID = LOC.LOCATION_ID  ' || CHR(10) ||

'AND    SITE.STATUS = ''A''  ' || CHR(10) ||

'AND    OPP.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    OPP.lead_id = l_trans_object_id1  ';
```

- l_batch_total_sql :=

```
l_batch_total_sql :=

'SELECT  ' || CHR(10) ||

'  -1001                               source_id,  ' || CHR(10) ||

'  -1004                                     trans_object_type_id,
' || CHR(10)
||

'  OPP.lead_id                          trans_object_id,  ' || CHR
(10) ||
'  STATUS.OPP_OPEN_STATUS_FLAG                      OPEN_FLAG,
' || CHR(10) ||
'  to_number(null)
trans_detail_object_id,  ' ||
CHR(10) ||

'  PARTY.PARTY_ID                           squal_num01,  ' ||
CHR(10) ||
'  to_number(null)                         squal_num02,  ' || CHR
(10) ||
'  PARTY.PARTY_ID                       squal_num03,  ' || CHR(10)
||
'  opp.lead_id                           squal_num04,  ' || CHR
(10) ||
'  PARTY.EMPLOYEES_TOTAL                    squal_num05,  ' ||
CHR(10) ||
'  ORGP.CURR_FY_POTENTIAL_REVENUE            squal_num06,  ' ||
CHR(10) ||
'  OPP.SOURCE_PROMOTION_ID                    squal_num40,  ' ||
CHR(10) ||
'  OPP.TOTAL_AMOUNT                         squal_num41,  ' || CHR
(10) ||
'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)           squal_curc01,
' || CHR(10) ||
'  upper(OPP.CURRENCY_CODE)                    squal_curc07,  '
|| CHR(10) ||
'  upper(substr(PARTY.party_name,1,1))          squal_fc01,  ' ||
CHR(10) ||
'  upper(PARTY.PARTY_NAME)                squal_char01,  ' || CHR
(10) ||
'  TO_CHAR(NULL)                         squal_char02,  ' || CHR
(10) ||
'  TO_CHAR(NULL)                         squal_char03,  ' || CHR
(10) ||
'  TO_CHAR(NULL)                         squal_char04,  ' || CHR
(10) ||
'  TO_CHAR(NULL)                         squal_char05,  ' || CHR
(10) ||
'  TO_CHAR(NULL)
squal_char06,  ' || CHR(10)
||

'  TO_CHAR(NULL)                         squal_char07,  ' || CHR
(10) ||
'  upper(PARTY.PRIMARY_PHONE_AREA_CODE)          squal_char08,  '
|| CHR(10) ||
'  upper(PARTY.CATEGORY_CODE)
squal_char09,  ' || CHR(10)
||

'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||
```

```
'  UPPER(PARTY.DUNS_NUMBER_C)               squal_char11,  ' ||
CHR(10) ||
'  upper(OPP.CHANNEL_CODE)                 squal_char40,  ' || CHR
(10) ||
'  upper(OPP.STATUS)                                 squal_char41
' || CHR(10) ||
'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(OPP.budget_status_code)
Q9103_BUDGET_STATUS ' ||
'FROM   AS_LEADS OPP,  ' || CHR(10) ||

'       HZ_PARTIES PARTY,  ' || CHR(10) ||

'       HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'    AS_STATUSES_B STATUS  ' || CHR(10) ||

'WHERE  OPP.ADDRESS_ID IS NULL  ' || CHR(10) ||

'AND    OPP.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    OPP.status = STATUS.status_code  ' || CHR(10) ||

'AND    STATUS.opp_flag = ''Y'' UNION ALL  ' || CHR(10) ||

'SELECT  ' || CHR(10) ||

'  -1001                           source_id,  ' || CHR(10) ||

'  -1004                           trans_object_type_id,  ' ||
CHR(10) ||
'  OPP.LEAD_ID                        trans_object_id,  ' || CHR
(10) ||
'  STATUS.OPP_OPEN_STATUS_FLAG                       OPEN_FLAG,
' || CHR(10) ||
'  SITE.PARTY_SITE_ID                 trans_detail_object_id,  '
|| CHR(10) ||
'  PARTY.PARTY_ID                  squal_num01,  ' || CHR(10) ||

'  SITE.PARTY_SITE_ID                                squal_num02,
' || CHR(10)
||

'  PARTY.PARTY_ID                  squal_num03,  ' || CHR(10) ||

'  OPP.LEAD_ID                      squal_num04,  ' || CHR(10) ||

'  PARTY.EMPLOYEES_TOTAL                             squal_num05,
' || CHR(10)
||

'  ORGP.CURR_FY_POTENTIAL_REVENUE        squal_num06,  ' || CHR
(10) ||
'  OPP.SOURCE_PROMOTION_ID             squal_num40,  ' || CHR(10)
||
'  OPP.TOTAL_AMOUNT                                  squal_num41,
' || CHR(10)
||
```

```
'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)       squal_curc01,  ' ||
CHR(10) ||
'  upper(OPP.CURRENCY_CODE)               squal_curc07,  ' ||
CHR(10) ||
'  upper(substr(PARTY.party_name,1,1))       squal_fc01,  ' ||
CHR(10) ||
'  upper(PARTY.PARTY_NAME)             squal_char01,  ' || CHR
(10) ||
'  upper(LOC.CITY)                  squal_char02,  ' || CHR(10)
||
'  upper(LOC.COUNTY)                     squal_char03,  ' || CHR
(10) ||
'  upper(LOC.STATE)                     squal_char04,  ' || CHR
(10) ||
'  upper(LOC.PROVINCE)                  squal_char05,  ' || CHR
(10) ||
'  upper(LOC.POSTAL_CODE)              squal_char06,  ' || CHR(10)
||
'  upper(LOC.COUNTRY)                  squal_char07,  ' || CHR(10)
||
'  upper(CNTPNT.PHONE_AREA_CODE)            squal_char08,  ' ||
CHR(10) ||
'  upper(PARTY.CATEGORY_CODE)
squal_char09,  ' || CHR(10)
||

'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)            squal_char11,  ' || CHR
(10) ||
'  upper(OPP.CHANNEL_CODE)
squal_char40,  ' || CHR(10)
||

'  upper(OPP.STATUS)                                  squal_char41
' || CHR(10)
||

'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(OPP.budget_status_code)
Q9103_BUDGET_STATUS ' ||
'FROM   AS_LEADS_ALL OPP,  ' || CHR(10) ||

'       HZ_CONTACT_POINTS CNTPNT,  ' || CHR(10) ||

'       HZ_PARTY_SITES SITE,  ' || CHR(10) ||

'       HZ_LOCATIONS LOC,  ' || CHR(10) ||

'       HZ_PARTIES PARTY,  ' || CHR(10) ||

'       HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'    AS_STATUSES_B STATUS  ' || CHR(10) ||

'WHERE  CNTPNT.OWNER_TABLE_NAME(+) = ''HZ_PARTY_SITES''  ' || CHR
(10) ||
'AND    SITE.PARTY_SITE_ID = CNTPNT.OWNER_TABLE_ID(+)  ' || CHR
(10) ||
'AND    CNTPNT.PRIMARY_FLAG(+) = ''Y''  ' || CHR(10) ||

'AND    CNTPNT.STATUS(+) <> ''I''  ' || CHR(10) ||
```

```
'AND    CNTPNT.contact_point_type(+)=''PHONE''  ' || CHR(10) ||

'AND    OPP.ADDRESS_ID = SITE.PARTY_SITE_ID  ' || CHR(10) ||

'AND    SITE.LOCATION_ID = LOC.LOCATION_ID  ' || CHR(10) ||

'AND    SITE.STATUS = ''A''  ' || CHR(10) ||

'AND    OPP.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    OPP.status = STATUS.status_code  ' || CHR(10) ||

'AND    STATUS.opp_flag = ''Y''  ';
```

- l_batch_incr_sql :=

```
l_batch_incr_sql :=

'SELECT  ' || CHR(10) ||

'  -1001              source_id,  ' || CHR(10) ||

'  -1004                       trans_object_type_id,  ' || CHR
(10) ||
'  OPP.lead_id         trans_object_id,  ' || CHR(10) ||

'  to_number(NULL)            trans_detail_object_id,  ' || CHR
(10) ||
'  STATUS.OPP_OPEN_STATUS_FLAG       OPEN_FLAG,  ' || CHR(10) ||

'  PARTY.PARTY_ID              squal_num01,  ' || CHR(10) ||

'  to_number(null)         squal_num02,  ' || CHR(10) ||

'  PARTY.PARTY_ID      squal_num03,  ' || CHR(10) ||

'  opp.lead_id          squal_num04,  ' || CHR(10) ||

'  PARTY.EMPLOYEES_TOTAL    squal_num05,  ' || CHR(10) ||

'  ORGP.CURR_FY_POTENTIAL_REVENUE squal_num06,  ' || CHR(10) ||

'  OPP.SOURCE_PROMOTION_ID    squal_num40,  ' || CHR(10) ||

'  OPP.TOTAL_AMOUNT        squal_num41,  ' || CHR(10) ||

'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY) squal_curc01,  ' || CHR
(10) ||
'  upper(OPP.CURRENCY_CODE)    squal_curc07,  ' || CHR(10) ||

'  upper(substr(PARTY.party_name,1,1)) squal_fc01,  ' || CHR(10)
||
'  upper(PARTY.PARTY_NAME)   squal_char01,  ' || CHR(10) ||

'  TO_CHAR(NULL)         squal_char02,  ' || CHR(10) ||

'  TO_CHAR(NULL)         squal_char03,  ' || CHR(10) ||

'  TO_CHAR(NULL)         squal_char04,  ' || CHR(10) ||

'  TO_CHAR(NULL)         squal_char05,  ' || CHR(10) ||

'  TO_CHAR(NULL)                     squal_char06,  ' || CHR(10)
||
'  TO_CHAR(NULL)         squal_char07,  ' || CHR(10) ||

'  upper(PARTY.PRIMARY_PHONE_AREA_CODE) squal_char08,  ' || CHR
(10) ||
'  upper(PARTY.CATEGORY_CODE)        squal_char09,  ' || CHR(10)
||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)  squal_char11,  ' || CHR(10) ||

'  upper(OPP.CHANNEL_CODE)   squal_char40,  ' || CHR(10) ||

'  upper(OPP.STATUS)                 squal_char41  ' || CHR(10)
||
'  ,UPPER(PARTY.party_type)
```

```
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(OPP.budget_status_code)
Q9103_BUDGET_STATUS ' ||
'FROM   AS_LEADS OPP,  ' || CHR(10) ||

'       HZ_PARTIES PARTY,  ' || CHR(10) ||

'       HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'    AS_STATUSES_B STATUS,  ' || CHR(10) ||

'    AS_CHANGED_ACCOUNTS_ALL ACHNG  ' || CHR(10) ||

'WHERE  OPP.ADDRESS_ID IS NULL  ' || CHR(10) ||

'AND    OPP.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    OPP.status = STATUS.status_code  ' || CHR(10) ||

'AND    STATUS.opp_flag = ''Y''  ' || CHR(10) ||

'AND    achng.lead_id = opp.lead_id  ' || CHR(10) ||

'AND    achng.request_id = l_REQUEST_ID UNION ALL  ' || CHR(10)
||
'SELECT  ' || CHR(10) ||

'-1001               source_id,  ' || CHR(10) ||

'-1004               trans_object_type_id,  ' || CHR(10) ||

'OPP.LEAD_ID            trans_object_id,  ' || CHR(10) ||

'SITE.PARTY_SITE_ID         trans_detail_object_id,  ' || CHR(10)
||
'STATUS.OPP_OPEN_STATUS_FLAG             OPEN_FLAG,  ' || CHR(10)
||
'PARTY.PARTY_ID          squal_num01,  ' || CHR(10) ||

'SITE.PARTY_SITE_ID                      squal_num02,  ' || CHR
(10) ||
'PARTY.PARTY_ID          squal_num03,  ' || CHR(10) ||

'OPP.LEAD_ID              squal_num04,  ' || CHR(10) ||

'PARTY.EMPLOYEES_TOTAL                   squal_num05,  ' || CHR
(10) ||
'ORGP.CURR_FY_POTENTIAL_REVENUE   squal_num06,  ' || CHR(10) ||

'OPP.SOURCE_PROMOTION_ID     squal_num40,  ' || CHR(10) ||

'OPP.TOTAL_AMOUNT                        squal_num41,  ' || CHR
(10) ||
'upper(ORGP.PREF_FUNCTIONAL_CURRENCY) squal_curc01,  ' || CHR(10)
||
'upper(OPP.CURRENCY_CODE)    squal_curc07,  ' || CHR(10) ||

'upper(substr(PARTY.party_name,1,1))  squal_fc01,  ' || CHR(10)
||
```

```
'upper(PARTY.PARTY_NAME)     squal_char01,  ' || CHR(10) ||

'upper(LOC.CITY)          squal_char02,  ' || CHR(10) ||

'upper(LOC.COUNTY)         squal_char03,  ' || CHR(10) ||

'upper(LOC.STATE)         squal_char04,  ' || CHR(10) ||

'upper(LOC.PROVINCE)         squal_char05,  ' || CHR(10) ||

'upper(LOC.POSTAL_CODE)     squal_char06,  ' || CHR(10) ||

'upper(LOC.COUNTRY)         squal_char07,  ' || CHR(10) ||

'upper(CNTPNT.PHONE_AREA_CODE)   squal_char08,  ' || CHR(10) ||

'upper(PARTY.CATEGORY_CODE)              squal_char09,  ' || CHR
(10) ||
'upper(party.sic_code_type||'': ''||party.sic_code) squal_char10,
' || CHR(10)
||

'UPPER(PARTY.DUNS_NUMBER_C)    squal_char11,  ' || CHR(10) ||

'upper(OPP.CHANNEL_CODE)                 squal_char40,  ' || CHR
(10) ||
'upper(OPP.STATUS)                       squal_char41  ' || CHR
(10) ||
'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(OPP.budget_status_code)
Q9103_BUDGET_STATUS ' ||
'FROM   AS_LEADS_ALL OPP,  ' || CHR(10) ||

'       HZ_CONTACT_POINTS CNTPNT,  ' || CHR(10) ||

'       HZ_PARTY_SITES SITE,  ' || CHR(10) ||

'       HZ_LOCATIONS LOC,  ' || CHR(10) ||

'       HZ_PARTIES PARTY,  ' || CHR(10) ||

'       HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'    AS_STATUSES_B STATUS,  ' || CHR(10) ||

'    AS_CHANGED_ACCOUNTS_ALL ACHNG  ' || CHR(10) ||

'WHERE  CNTPNT.OWNER_TABLE_NAME(+) = ''HZ_PARTY_SITES''  ' || CHR
(10) ||
'AND    SITE.PARTY_SITE_ID = CNTPNT.OWNER_TABLE_ID(+)  ' || CHR
(10) ||
'AND    CNTPNT.PRIMARY_FLAG(+) = ''Y''  ' || CHR(10) ||

'AND    CNTPNT.STATUS(+) <> ''I''  ' || CHR(10) ||

'AND    CNTPNT.contact_point_type(+)=''PHONE''  ' || CHR(10) ||

'AND    OPP.ADDRESS_ID = SITE.PARTY_SITE_ID  ' || CHR(10) ||

'AND    SITE.LOCATION_ID = LOC.LOCATION_ID  ' || CHR(10) ||

'AND    SITE.STATUS = ''A''  ' || CHR(10) ||

'AND    OPP.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||
```

```
'AND    party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND    party.STATUS = ''A''  ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  '
|| CHR(10) ||
'AND    OPP.status = STATUS.status_code  ' || CHR(10) ||

'AND    STATUS.opp_flag = ''Y''  ' || CHR(10) ||

'AND    achng.lead_id = opp.lead_id  ' || CHR(10) ||

'AND    achng.request_id = l_REQUEST_ID  ';


l_batch_dea_sql :=

NULL;
```

- l_batch_dea_sql := NO
- l_incr_reassign_sql :=

```
l_incr_reassign_sql :=

'SELECT  ' || CHR(10) ||

'  -1001                  source_id,  ' || CHR(10) ||

'  -1004                       trans_object_type_id,  ' || CHR
(10) ||
'  OPP.lead_id             trans_object_id,  ' || CHR(10) ||

'  STATUS.OPP_OPEN_STATUS_FLAG           OPEN_FLAG,  ' || CHR(10)
||
'  to_number(NULL)                trans_detail_object_id,  ' ||
CHR(10) ||
'  PARTY.PARTY_ID               squal_num01,  ' || CHR(10) ||

'  to_number(null)             squal_num02,  ' || CHR(10) ||

'  PARTY.PARTY_ID          squal_num03,  ' || CHR(10) ||

'  opp.lead_id              squal_num04,  ' || CHR(10) ||

'  PARTY.EMPLOYEES_TOTAL        squal_num05,  ' || CHR(10) ||

'  ORGP.CURR_FY_POTENTIAL_REVENUE   squal_num06,  ' || CHR(10) ||

'  OPP.SOURCE_PROMOTION_ID        squal_num40,  ' || CHR(10) ||

'  OPP.TOTAL_AMOUNT            squal_num41,  ' || CHR(10) ||

'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)  squal_curc01,  ' || CHR
(10) ||
'  upper(OPP.CURRENCY_CODE)        squal_curc07,  ' || CHR(10) ||

'  upper(substr(PARTY.party_name,1,1)) squal_fc01,  ' || CHR(10)
||
'  upper(PARTY.PARTY_NAME)    squal_char01,  ' || CHR(10) ||

'  TO_CHAR(NULL)             squal_char02,  ' || CHR(10) ||

'  TO_CHAR(NULL)             squal_char03,  ' || CHR(10) ||

'  TO_CHAR(NULL)             squal_char04,  ' || CHR(10) ||

'  TO_CHAR(NULL)             squal_char05,  ' || CHR(10) ||

'  TO_CHAR(NULL)                          squal_char06,  ' || CHR
(10) ||
'  TO_CHAR(NULL)             squal_char07,  ' || CHR(10) ||

'  upper(PARTY.PRIMARY_PHONE_AREA_CODE) squal_char08,  ' || CHR
(10) ||
'  upper(PARTY.CATEGORY_CODE)            squal_char09,  ' || CHR
(10) ||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)   squal_char11,  ' || CHR(10) ||

'  upper(OPP.CHANNEL_CODE)    squal_char40,  ' || CHR(10) ||

'  upper(OPP.STATUS)                     squal_char41  ' || CHR
(10) ||
'  ,UPPER(PARTY.party_type)
```

```
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(OPP.budget_status_code)
Q9103_BUDGET_STATUS ' ||
'FROM   AS_LEADS OPP,  ' || CHR(10) ||

'       HZ_PARTIES PARTY,  ' || CHR(10) ||

'       HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'    AS_STATUSES_B STATUS,  ' || CHR(10) ||

'    ( select distinct ACC.lead_id lead_id  ' || CHR(10) ||

'         from (select distinct terr_id terr_id  ' || CHR(10) ||

'               from JTY_CHANGED_TERRS  ' || CHR(10) ||

'               where tap_request_id = l_request_id) CHG_TERR,  '
|| CHR(10) ||
'              AS_ACCESSES_ALL ACC,  ' || CHR(10) ||

'              AS_TERRITORY_ACCESSES TERR_ACC  ' || CHR(10) ||

'         where CHG_TERR.terr_id = TERR_ACC.territory_id  ' ||
CHR(10) ||
'         and TERR_ACC.access_id = acc.access_id  ' || CHR(10) ||

'         and acc.lead_id is not null  ' || CHR(10) ||

'         and acc.sales_lead_id is null )  ACHNG  ' || CHR(10) ||

'WHERE  OPP.ADDRESS_ID IS NULL  ' || CHR(10) ||

'AND  OPP.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND  party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND  party.STATUS = ''A''  ' || CHR(10) ||

'AND  nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  ' ||
CHR(10) ||
'AND OPP.status = STATUS.status_code  ' || CHR(10) ||

'AND STATUS.opp_flag = ''Y''  ' || CHR(10) ||

'AND achng.lead_id = opp.lead_id UNION ALL  ' || CHR(10) ||

'SELECT  ' || CHR(10) ||

'  -1001                      source_id,  ' || CHR(10) ||

'  -1004                      trans_object_type_id,  ' || CHR
(10) ||
'  OPP.LEAD_ID                   trans_object_id,  ' || CHR(10)
||
'  STATUS.OPP_OPEN_STATUS_FLAG                  OPEN_FLAG,  ' ||
CHR(10) ||
'  SITE.PARTY_SITE_ID            trans_detail_object_id,  ' ||
CHR(10) ||
'  PARTY.PARTY_ID             squal_num01,  ' || CHR(10) ||

'  SITE.PARTY_SITE_ID                          squal_num02,  '
|| CHR(10) ||
'  PARTY.PARTY_ID             squal_num03,  ' || CHR(10) ||
```

```
'  OPP.LEAD_ID                  squal_num04,  ' || CHR(10) ||

'  PARTY.EMPLOYEES_TOTAL                          squal_num05,  '
|| CHR(10) ||
'  ORGP.CURR_FY_POTENTIAL_REVENUE    squal_num06,  ' || CHR(10)
||
'  OPP.SOURCE_PROMOTION_ID         squal_num40,  ' || CHR(10) ||

'  OPP.TOTAL_AMOUNT                               squal_num41,  '
|| CHR(10) ||
'  upper(ORGP.PREF_FUNCTIONAL_CURRENCY)   squal_curc01,  ' || CHR
(10) ||
'  upper(OPP.CURRENCY_CODE)            squal_curc07,  ' || CHR
(10) ||
'  upper(substr(PARTY.party_name,1,1))   squal_fc01,  ' || CHR
(10) ||
'  upper(PARTY.PARTY_NAME)         squal_char01,  ' || CHR(10) ||

'  upper(LOC.CITY)              squal_char02,  ' || CHR(10) ||

'  upper(LOC.COUNTY)                 squal_char03,  ' || CHR(10)
||
'  upper(LOC.STATE)                 squal_char04,  ' || CHR(10)
||
'  upper(LOC.PROVINCE)             squal_char05,  ' || CHR(10) ||

'  upper(LOC.POSTAL_CODE)          squal_char06,  ' || CHR(10) ||

'  upper(LOC.COUNTRY)              squal_char07,  ' || CHR(10) ||

'  upper(CNTPNT.PHONE_AREA_CODE)        squal_char08,  ' || CHR
(10) ||
'  upper(PARTY.CATEGORY_CODE)                     squal_char09,  '
|| CHR(10) ||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10,  ' ||
CHR(10) ||

'  UPPER(PARTY.DUNS_NUMBER_C)        squal_char11,  ' || CHR(10)
||
'  upper(OPP.CHANNEL_CODE)                        squal_char40,  '
|| CHR(10) ||
'  upper(OPP.STATUS)                              squal_char41  '
|| CHR(10) ||
'  ,UPPER(PARTY.party_type)
Q9010_PARTY_TYPE  ' ||
'  ,UPPER(OPP.budget_status_code)
Q9103_BUDGET_STATUS ' ||
'FROM   AS_LEADS_ALL OPP,  ' || CHR(10) ||

'       HZ_CONTACT_POINTS CNTPNT,  ' || CHR(10) ||

'       HZ_PARTY_SITES SITE,  ' || CHR(10) ||

'       HZ_LOCATIONS LOC,  ' || CHR(10) ||

'       HZ_PARTIES PARTY,  ' || CHR(10) ||

'       HZ_ORGANIZATION_PROFILES ORGP,  ' || CHR(10) ||

'    AS_STATUSES_B STATUS,  ' || CHR(10) ||

'    ( select distinct ACC.lead_id lead_id  ' || CHR(10) ||

'         from (select distinct terr_id terr_id  ' || CHR(10) ||
```

```
'               from JTY_CHANGED_TERRS  ' || CHR(10) ||

'               where tap_request_id = l_request_id) CHG_TERR,  '
|| CHR(10) ||
'              AS_ACCESSES_ALL ACC,  ' || CHR(10) ||

'              AS_TERRITORY_ACCESSES TERR_ACC  ' || CHR(10) ||

'         where CHG_TERR.terr_id = TERR_ACC.territory_id  ' ||
CHR(10) ||
'         and TERR_ACC.access_id = acc.access_id  ' || CHR(10) ||

'         and acc.lead_id is not null  ' || CHR(10) ||

'         and acc.sales_lead_id is null )  ACHNG  ' || CHR(10) ||

'WHERE  CNTPNT.OWNER_TABLE_NAME(+) = ''HZ_PARTY_SITES''  ' || CHR
(10) ||
'AND  SITE.PARTY_SITE_ID = CNTPNT.OWNER_TABLE_ID(+)  ' || CHR(10)
||
'AND  CNTPNT.PRIMARY_FLAG(+) = ''Y''  ' || CHR(10) ||

'AND  CNTPNT.STATUS(+) <> ''I''  ' || CHR(10) ||

'AND  CNTPNT.contact_point_type(+)=''PHONE''  ' || CHR(10) ||

'AND  OPP.ADDRESS_ID = SITE.PARTY_SITE_ID  ' || CHR(10) ||

'AND  SITE.LOCATION_ID = LOC.LOCATION_ID  ' || CHR(10) ||

'AND  SITE.STATUS = ''A''  ' || CHR(10) ||

'AND  OPP.CUSTOMER_ID = PARTY.PARTY_ID  ' || CHR(10) ||

'AND  party.party_id = orgp.party_id(+)  ' || CHR(10) ||

'AND  party.STATUS = ''A''  ' || CHR(10) ||

'AND  nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate  ' ||
CHR(10) ||
'AND  OPP.status = STATUS.status_code  ' || CHR(10) ||

'AND  STATUS.opp_flag = ''Y''  ' || CHR(10) ||

'AND  achng.lead_id = opp.lead_id  ';
```

2. Modify the R12OpportunityTT.sql file with the appropriate package parameters to make the SQL meta-data change.

```
JTY_TRANS_USG_PGM_SQL_PKG.Insert_Row(

p_source_id => -1001

,p_trans_type_id => -1004

,p_program_name => 'SALES/OPPORTUNITY PROGRAM'

,p_version_name => '12-JAN-06: CMA BUDGET STATUS ADDED'

,p_real_time_sql => l_real_time_sql

,p_batch_total_sql => l_batch_total_sql

,p_batch_incr_sql => l_batch_incr_sql

,p_batch_dea_sql => l_batch_dea_sql

,p_incr_reassign_sql => l_incr_reassign_sql

,p_use_total_for_dea_flag => null

,p_enabled_flag => 'Y'

,retcode => retcode

,errbuf => errbuf);

dbms_output.put_line('SALES/OPPORTUNITY PROGRAM retcode : ' ||
retcode);
dbms_output.put_line('SALES/OPPORTUNITY PROGRAM errbuf : ' ||
errbuf);
```

3. Run the R12OpportunityTT.sql file.

```
SQL> @R12OpportunityTT.sql /
```

Create the Custom Matching Attribute

Create the CMA. The basis of this script is to populate all the necessary information in the following tables:

JTF_SEEDED_QUAL_ALL_B

JTF_SEEDED_QUAL_ALL_TL

JTF_QUAL_USGS_ALL

1. Ensure that there is not an existing CMA record for the unique matching attribute ID that you will use for the new CMA.

```
set serveroutput ON SIZE 999999
DECLARE
    l_real_time_sql VARCHAR2(32000) := NULL;
    l_batch_total_sql VARCHAR2(32000) := NULL;
    l_batch_incr_sql VARCHAR2(32000) := NULL;
    l_batch_dea_sql VARCHAR2(32000) := NULL;
    l_incr_reassign_sql VARCHAR2(32000);
         retcode VARCHAR2(250);
           errbuf VARCHAR2(1000);

      TYPE namesarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE descarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE langarray IS VARRAY(5) OF VARCHAR2(50);
      names namesarray;
      descript descarray;
      lang langarray;
      total INTEGER;
 BEGIN
    BEGIN
        DELETE FROM jtf_seeded_qual_all_b
        WHERE seeded_qual_id = -9103;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_seeded_qual_all_tl
        WHERE seeded_qual_id = -9103;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_qual_usgs_all
        WHERE qual_usg_id = -9103;
            EXCEPTION
                  WHEN no_data_found THEN
                    NULL;
            END;
      BEGIN
```

2. Create the meta-data for the CMA. Follow the comments provided in the script to enter the relevant CMA information.

```
/* Provide CMA details in at least one language, such as 'US'*/
/* To create a CMA in multi-language, provide those details in
'names', 'descript' and 'lang' as an array*/

            names := namesarray('Opportunity Budget Status (CMA)';
            descript := descarray('Custom Matching Attribute for
Opportunity Budget Status');
            lang := langarray('US');
            total := names.count;

     FOR i in 1 .. total LOOP
/* The following meta-data will create the custom matching attribute
for the Budget Status.*/
       JTY_CUST_QUAL_PKG.Create_qual(
     p_seeded_qual_id     => -9103,
/* This number should correspond to the number used in the alias for
the attribute*/
     p_name               => names(i),
     p_description        => descript(i),
     p_language           => lang(i),

     ,/* Sales Usage: FK to
       JTF_SOURCES_ALL.SOURCE_ID*/
     p_source_id          => -1001

    ,/* Sales Transaction Type: FK to
        JTF_QUAL_TYPE_USGS_ALL.QUAL_TYPE_USG_ID*/
     p_trans_type_id      => -1004

    ,p_enabled_flag       => 'N'

    ,/* QUAL_RELATION_FACTOR
     This needs to be the next prime number greater than
     the value from the following SQL:

     SELECT MAX(qual_relation_factor)
     FROM jtf_qual_usgs_all

     The first 1000 primes can be found at:
     http://primes.utm.edu/lists/small/1000.txt

     Custom qualifiers should start at the 303rd prime
     which is 1999. This is to keep 1st to 302nd primes
     for product development to seed qualifiers.
     */
     p_qual_relation_factor       => 2027


    /* The following set of meta-data setups determine
    ** the behaviour of the matching attribute in
    ** HTML and Excel UIs: how it is displayed; what
    ** is the LOV SQL; and, what comparison operators
    ** are supported.
    */

    ,/* CONVERT_TO_ID_FLAG: displayed qualifier value is a CHAR
        but stored qualifier value is stored as an internal id  */
     p_convert_to_id_flag         => 'N'

    ,/* DISPLAY_TYPE: display type on UI (NUMERIC/CHAR) */
     p_display_type               => 'CHAR'
    ,/* LOV SQL */
    p_html_lov_sql1              =>
     ''
```

```
    ,p_html_lov_sql2              => null
    ,p_html_lov_sql3              => null
    ,p_display_sql1               => null
    ,p_display_sql2               => null
    ,p_display_sql3               => null
    ,p_hierarchy_type             => null

    ,/* Is the "=" operator supported? */
    p_equal_flag                  => 'Y'
    ,/* Is the "LIKE" operator supported? */
    p_like_flag                   => 'N'
    ,/* Is the "BETWEEN" operator supported? */
    p_between_flag                => 'N'

    /* Values table */
    /* matching attribute comparison operator */
    ,p_comparison_operator        => 'q9103_cp'
    ,p_low_value_char             => 'q9103_low_value_CHAR'
    ,p_high_value_char            => null
    ,p_low_value_char_id          => null
    ,p_low_value_number           => null
    ,p_high_value_number          => null
    ,p_interest_type_id           => null
    ,p_primary_interest_code_id   => null
    ,p_sec_interest_code_id       => null
    ,p_value1_id                  => null
    ,p_value2_id                  => null
    ,p_value3_id                  => null
    ,p_value4_id                  => null
    ,p_first_char                 => null
    ,p_currency_code              => null


    ,/* Transaction Type Alias/_TRANS table column mapping */
     p_qual_col1                  => 'Q9103_BUDGET_STATUS'

    ,/* TCA Classification for derivation */
     p_alias_rule1                => null

    ,/* OP_EQL FOR BATCH-MODE ASSIGNMENT:
     ** Since the AS_SALES_LEADS.STATUS_CODE value is not passed to
the
     ** assignment API, we must the PK (SALES_LEAD_ID) to get the
     ** STATUS_CODE value from AS_SALES_LEADS and compare to the
     ** territory qualifier value
     */
     p_op_eql                     =>
      ' (A.Q9103_BUDGET_STATUS = B.q9103_low_value_CHAR AND B.
q9103_cp = ''='')'
    ,p_op_like                    => null
    ,p_op_between                 => NULL
    ,p_op_common_where            => null

    ,p_real_time_select           =>
    'SELECT DISTINCT A.trans_object_id, A.trans_detail_object_id, A.
txn_date, B.terr_id, B.absolute_rank, B.top_level_terr_id, B.
num_winners'
    ,p_real_time_from             => null
    ,p_real_time_where            =>
    'WHERE '  || CHR(10) ||
    '  B.q9103_low_value_CHAR = A.Q9103_BUDGET_STATUS ' || CHR(10)
||
    '  AND B.q9103_cp = ''='' ' || CHR(10) ||
    '  AND B.source_id = -1001 ' || CHR(10) ||
    '  AND A.txn_date between B.start_date and B.end_date'
```

```
    ,retcode                      => retcode
    ,errbuf                       => errbuf);

  dbms_output.put_line('Retcode : ' || retcode);
  dbms_output.put_line('Errbuf : ' || errbuf);

  ad_morg.replicate_seed_data(NULL, 'JTF', NULL);

EXIT WHEN retcode=2;
END LOOP;
END;
COMMIT;

exception
  when others then
    dbms_output.put_line('Retcode : ' || retcode);
    dbms_output.put_line('Errbuf : ' || errbuf);
    raise;
end;
/
```

3. Run this script.

CMA for Oracle Collections, Customer: Customer Party Type

This is an example of creating a custom matching attribute for Sales using the Oracle Collections customer transaction and the customer party type attribute.

Modify Script 1

Modify Script 1, page A-96 to retrieve the relevant customer transaction type SQL meta-data.

1. Set the source ID corresponding to the usage (Sales or Collections). The source ID -1600 corresponds to the Collections usage. To find the desired usage ID query JTF_SOURES_ALL.

   ```
   p_source_id NUMBER := -1600;
   ```

2. Set the transaction type ID. The ID -1602 corresponds to the Customer transaction type ID. To find the desired transaction type ID, query JTF_QUAL_TYPES_ALL

   ```
   p_trans_type_id NUMBER := -1602;
   ```

3. Run the script and save the output to a file. Running this command will save the Customer transaction type SQL meta-data to the "R12CustomerTT.sql" file.

   ```
   SQL>spool <output filename - e.g., "d:\R12CustomerTT.sql">
   SQL> set serveroutput on size 999999
   SQL> @Script1.sql /
   ```

Modify Transaction SQL Meta-Data

Modify the transaction type SQL meta-data (R12CustomerTT.sql) to get the CMA's

corresponding transaction attribute value.

1. Add the party_type attribute to each of the following SQLs:
   - l_real_time_sql :=

```
'SELECT ' || CHR(10) ||

'  party.party_id
trans_object_id, ' ||
CHR(10) ||

'  null
trans_detail_object_id, ' ||
CHR(10) ||

'  TO_CHAR(NULL)                                        squal_char02,
' || CHR(10)
||

'  TO_CHAR(NULL)                                        squal_char06,
' || CHR(10)
||

'  TO_CHAR(NULL)                                        squal_char04,
' || CHR(10)
||

'  TO_CHAR(NULL)                                        squal_char05,
' || CHR(10)
||

'  TO_CHAR(NULL)                                        squal_char03,
' || CHR(10)
||

'  TO_CHAR(NULL)                                        squal_char07,
' || CHR(10)
||

'  party.party_id                                       squal_num01,
' || CHR(10) ||
'  TO_NUMBER(NULL)                                      squal_num02,
' || CHR(10) ||
'  upper(party.primary_phone_area_code)                 squal_char08,
' || CHR(10)
||

'  upper(party.party_name)                              squal_char01,
' || CHR(10)
||

'  party.party_id                                       squal_num03,
' || CHR(10) ||
'  party.employees_total                                squal_num05,
' || CHR(10) ||
'  upper(party.category_code)                           squal_char09,
' || CHR(10)
||

'  party.party_id                                       squal_num04,
' || CHR(10) ||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10, ' || CHR(10)
||

'  orgp.CURR_FY_POTENTIAL_REVENUE                       squal_num06,
' || CHR(10) ||
'  upper(orgp.PREF_FUNCTIONAL_CURRENCY)                 squal_curc01,
' || CHR(10)
||
```

```
'  upper(party.duns_number_c)                          squal_char11,
' || CHR(10)
||

'  l_txn_date                                          txn_date ' ||
CHR(10) ||
'  ,UPPER(party.party_type)
Q9600_PARTY_TYPE  ' ||
'FROM ' || CHR(10) ||

'  HZ_PARTIES party, ' || CHR(10) ||

'  HZ_ORGANIZATION_PROFILES ORGP ' || CHR(10) ||

'WHERE party.party_type IN (''PERSON'',''ORGANIZATION'') ' || CHR
(10) ||
'AND   party.status=''A'' ' || CHR(10) ||

'AND   party.party_id = orgp.party_id(+) ' || CHR(10) ||

'AND   nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate ' ||
CHR(10) ||
'AND   party.party_id = l_trans_object_id1 UNION ALL SELECT ' ||
CHR(10) ||
'  party.party_id
trans_object_id, ' ||
CHR(10) ||

'  addr.party_site_id
trans_detail_object_id, ' ||
CHR(10) ||

'  upper(LOC.city)                                     squal_char02,
' || CHR(10)
||

'  upper(LOC.postal_code)                              squal_char06,
' || CHR(10)
||

'  upper(LOC.state)                                    squal_char04,
' || CHR(10)
||

'  upper(LOC.province)                                 squal_char05,
' || CHR(10)
||

'  upper(LOC.county)                                   squal_char03,
' || CHR(10)
||

'  upper(LOC.country)                                  squal_char07,
' || CHR(10)
||

'  party.party_id                                      squal_num01,
' || CHR(10) ||
'  addr.party_site_id                                  squal_num02,
' || CHR(10) ||
'  upper(phone.phone_area_code)                        squal_char08,
' || CHR(10)
||

'  upper(party.party_name)                             squal_char01,
```

```
' || CHR(10)
||

'  party.party_id                                    squal_num03,
' || CHR(10) ||
'  party.employees_total                             squal_num05,
' || CHR(10) ||
'  upper(party.category_code)                        squal_char09,
' || CHR(10)
||

'  party.party_id                                    squal_num04,
' || CHR(10) ||
'  upper(party.sic_code_type||'': ''||party.sic_code)
squal_char10, ' || CHR(10)
||

'  orgp.CURR_FY_POTENTIAL_REVENUE                    squal_num06,
' || CHR(10) ||
'  upper(orgp.PREF_FUNCTIONAL_CURRENCY)              squal_curc01,
' || CHR(10)
||

'  upper(party.duns_number_c)                        squal_char11,
' || CHR(10)
||

'  l_txn_date                                        txn_date ' ||
CHR(10) ||
'  ,UPPER(party.party_type)
Q9600_PARTY_TYPE  ' ||
'FROM ' || CHR(10) ||

'  HZ_CONTACT_POINTS phone, ' || CHR(10) ||

'  HZ_PARTIES party, ' || CHR(10) ||

'  HZ_PARTY_SITES addr, ' || CHR(10) ||

'  HZ_LOCATIONS LOC, ' || CHR(10) ||

'  HZ_ORGANIZATION_PROFILES ORGP ' || CHR(10) ||

'WHERE  phone.owner_table_name(+) = ''HZ_PARTY_SITES'' ' || CHR
(10) ||
'AND    phone.primary_flag(+) = ''Y'' ' || CHR(10) ||

'AND    phone.status(+) = ''A'' ' || CHR(10) ||

'AND    phone.contact_point_type (+) = ''PHONE'' ' || CHR(10) ||

'AND    addr.party_site_id = phone.owner_table_id(+) ' || CHR(10)
||
'AND    addr.party_id=party.party_id ' || CHR(10) ||

'AND    party.party_id = orgp.party_id(+) ' || CHR(10) ||

'AND    nvl(orgp.effective_end_date(+),sysdate + 1) > sysdate '
|| CHR(10) ||
'AND    party.party_type IN (''PERSON'',''ORGANIZATION'') ' ||
CHR(10) ||
'AND    party.status=''A'' ' || CHR(10) ||

'AND    LOC.location_id = addr.location_id ' || CHR(10) ||
```

```
'AND    addr.status=''A'' ' || CHR(10) ||

'AND    party.party_id = l_trans_object_id1 ' || CHR(10) ||

'AND    addr.party_site_id = nvl(l_trans_object_id2, addr.
party_site_id) ';
l_batch_total_sql :=

'SELECT /*+ parallel(ORGPRO) parallel(CNTPNT) parallel(X)
use_hash(ORGPRO CNTPNT
X) */ ' || CHR(10) ||

'-1600 source_id, ' || CHR(10) ||

'-1601 trans_object_type_id, ' || CHR(10) ||

'X.party_id  TRANS_OBJECT_ID, ' || CHR(10) ||

'X.party_site_id TRANS_DETAIL_OBJECT_ID, ' || CHR(10) ||

'X.party_site_id SQUAL_NUM02, ' || CHR(10) ||

'X.party_id SQUAL_NUM04, ' || CHR(10) ||

'UPPER(CNTPNT.phone_area_code) SQUAL_CHAR08, ' || CHR(10) ||

'UPPER(X.category_code) SQUAL_CHAR09, ' || CHR(10) ||

'UPPER(ORGPRO.pref_functional_currency)  SQUAL_CURC01, ' || CHR
(10) ||
'UPPER(X.city) SQUAL_CHAR02, ' || CHR(10) ||

'ORGPRO.curr_fy_potential_revenue SQUAL_NUM06, ' || CHR(10) ||

'UPPER(X.country) SQUAL_CHAR07, ' || CHR(10) ||

'UPPER(X.county) SQUAL_CHAR03, ' || CHR(10) ||

'X.party_id SQUAL_NUM01, ' || CHR(10) ||

'UPPER(X.party_name_substring) SQUAL_FC01, ' || CHR(10) ||

'UPPER(X.party_name) SQUAL_CHAR01, ' || CHR(10) ||

'X.employees_total SQUAL_NUM05, ' || CHR(10) ||

'UPPER(X.postal_code) SQUAL_CHAR06, ' || CHR(10) ||

'UPPER(X.province) SQUAL_CHAR05, ' || CHR(10) ||

'X.party_id SQUAL_NUM03, ' || CHR(10) ||

'UPPER(X.sic_code_type||'': ''||X.sic_code) SQUAL_CHAR10, ' ||
CHR(10) ||
'UPPER(X.state)  SQUAL_CHAR04, ' || CHR(10) ||

'UPPER(X.duns_number_c) SQUAL_CHAR11 ' || CHR(10) ||

',UPPER(x.party_type)                          Q9600_PARTY_TYPE
' ||
'from HZ_ORGANIZATION_PROFILES ORGPRO , ' || CHR(10) ||

'HZ_CONTACT_POINTS CNTPNT, ' || CHR(10) ||

'(select /*+ parallel(PARTY) parallel(SITE) parallel(LOC)
```

```
use_hash(SITE)
use_hash(PARTY) use_hash(LOC) */ ' || CHR(10) ||

'SITE.party_site_id party_site_id,  Site.party_id party_id, ' ||
CHR(10) ||
'LOC.city city,  LOC.country country, ' || CHR(10) ||

'LOC.county county,  LOC.state state, ' || CHR(10) ||

'LOC.province province,  LOC.postal_code postal_code, ' || CHR
(10) ||
'PARTY.employees_total employees_total, ' || CHR(10) ||

'PARTY.sic_code_type sic_code_type, ' || CHR(10) ||

'PARTY.sic_code sic_code, ' || CHR(10) ||

'upper(substr(PARTY.party_name,1,1)) party_name_substring, ' ||
CHR(10) ||
'upper(PARTY.party_name) party_name, ' || CHR(10) ||

'PARTY.category_code category_code, ' || CHR(10) ||

'''HZ_PARTY_SITES'' owner_table_name, ' || CHR(10) ||

'SITE.party_site_id owner_table_id, ' || CHR(10) ||

'PARTY.duns_number_c ' || CHR(10) ||
',PARTY.party_type party_type  ' || CHR(10) ||

'from HZ_PARTY_SITES SITE, ' || CHR(10) ||

'HZ_LOCATIONS LOC, HZ_PARTIES PARTY where ' || CHR(10) ||

'SITE.status = ''A'' and SITE.party_id = PARTY.party_id and ' ||
CHR(10) ||
'PARTY.party_type in (''PERSON'', ''ORGANIZATION'') and PARTY.
status = ''A'' and
' || CHR(10) ||

'LOC.location_id = SITE.location_id ' || CHR(10) ||

'union all ' || CHR(10) ||

'select /*+ parallel(PARTY) */ to_number(NULL) party_site_id , '
|| CHR(10) ||
'PARTY.party_id party_id , ' || CHR(10) ||
'PARTY.party_type party_type,  ' || CHR(10) ||

'to_char(NULL) city ,  to_char(NULL) country , ' || CHR(10) ||

'to_char(NULL) county , to_char(NULL) state , ' || CHR(10) ||

'to_char(NULL) province ,  to_char(NULL) postal_code , ' || CHR
(10) ||
'PARTY.employees_total employees_total, ' || CHR(10) ||

'PARTY.sic_code_type sic_code_type, ' || CHR(10) ||

'PARTY.sic_code sic_code, ' || CHR(10) ||

'upper(substr(PARTY.party_name,1,1)) party_name_substring, ' ||
CHR(10) ||
'upper(PARTY.party_name) party_name, ' || CHR(10) ||
```

```
'PARTY.category_code category_code, ' || CHR(10) ||

'''HZ_PARTIES'' owner_table_name, ' || CHR(10) ||

'PARTY.party_id owner_table_id, ' || CHR(10) ||

'PARTY.duns_number_c ' || CHR(10) ||

'from HZ_PARTIES PARTY ' || CHR(10) ||

'where PARTY.party_type in (''PERSON'', ''ORGANIZATION'') and '
|| CHR(10) ||
'PARTY.status = ''A'' ) X , ' || CHR(10) ||

'AR_TRX_BAL_SUMMARY arba, ' || CHR(10) ||

'hz_cust_accounts Hzc ' || CHR(10) ||

'WHERE ' || CHR(10) ||

'CNTPNT.owner_table_name(+) = X.owner_table_name and ' || CHR(10)
||
'CNTPNT.owner_table_id(+) = X.owner_table_id and ' || CHR(10) ||

'CNTPNT.status(+)=''A'' and CNTPNT.primary_flag(+)=''Y'' and ' ||
CHR(10) ||
'CNTPNT.contact_point_type(+)=''PHONE'' and ' || CHR(10) ||

'ORGPRO.party_id(+) = X.party_id and nvl(ORGPRO.
effective_end_date(+),sysdate+1)
> sysdate ' || CHR(10) ||

'AND arba.CUST_ACCOUNT_ID = hzc.CUST_ACCOUNT_ID and hzc.party_id
= x.party_id
';
```

- l_batch_total_sql :=
- l_batch_incr_sql := NULL
- l_batch_dea_sql := NULL
- l_incr_reassign_sql := NULL

2. Modify the R12CustomerTT.sql file with the appropriate package parameters to make the SQL meta-data change.

```
JTY_TRANS_USG_PGM_SQL_PKG.Insert_Row(

p_source_id => -1600

,p_trans_type_id => -1601

,p_program_name => 'COLLECTIONS/CUSTOMER PROGRAM'

,p_version_name => '16-JAN-06: CMA PARTY TYPE ADDED'

,p_real_time_sql => l_real_time_sql

,p_batch_total_sql => l_batch_total_sql

,p_batch_incr_sql => l_batch_incr_sql

,p_batch_dea_sql => l_batch_dea_sql

,p_incr_reassign_sql => l_incr_reassign_sql

,p_use_total_for_dea_flag => null

,p_enabled_flag => 'Y'

,retcode => retcode

,errbuf => errbuf);

dbms_output.put_line('COLLECTIONS/CUSTOMER PROGRAM retcode : ' ||
retcode);
dbms_output.put_line('COLLECTIONS/CUSTOMER PROGRAM errbuf : ' ||
errbuf);
COMMIT;

END;

/
```

3. Run the R12CustomerTT.sql file.

```
SQL> @R12CustomerTT.sql /
```

Create the Custom Matching Attribute

Create the CMA. The basis of this script is to populate all the necessary information in the following tables:

JTF_SEEDED_QUAL_ALL_B

JTF_SEEDED_QUAL_ALL_TL

JTF_QUAL_USGS_ALL

1. Ensure that there is not an existing CMA record for the unique matching attribute ID that you will use for the new CMA.

```
set serveroutput ON SIZE 999999
DECLARE
    l_real_time_sql VARCHAR2(32000) := NULL;
    l_batch_total_sql VARCHAR2(32000) := NULL;
    l_batch_incr_sql VARCHAR2(32000) := NULL;
    l_batch_dea_sql VARCHAR2(32000) := NULL;
    l_incr_reassign_sql VARCHAR2(32000);
         retcode VARCHAR2(250);
           errbuf VARCHAR2(1000);

      TYPE namesarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE descarray IS VARRAY(5) OF VARCHAR2(500);
      TYPE langarray IS VARRAY(5) OF VARCHAR2(50);
      names namesarray;
      descript descarray;
      lang langarray;
      total INTEGER;
 BEGIN
    BEGIN
        DELETE FROM jtf_seeded_qual_all_b
        WHERE seeded_qual_id = -9600;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_seeded_qual_all_tl
        WHERE seeded_qual_id = -9600;
    EXCEPTION
        WHEN no_data_found THEN
          NULL;
    END;
    BEGIN
        DELETE FROM jtf_qual_usgs_all
        WHERE qual_usg_id = -9600;
            EXCEPTION
                  WHEN no_data_found THEN
                    NULL;
            END;
      BEGIN
```

2. Create the meta-data for the CMA. Follow the comments provided in the script to enter the relevant CMA information.

```
/* Provide CMA details in at least one language, such as 'US'*/
/* To create a CMA in multi-language, provide those details in
'names', 'descript' and 'lang' as an array*/

            names := namesarray('Customer Party Type (CMA)');
            descript := descarray('Customer Party Type (CMA)');
            lang := langarray('US');
            total := names.count;

     FOR i in 1 .. total LOOP
/* The following meta-data will create the custom matching attribute
for the Customer Party Type.*/
       JTY_CUST_QUAL_PKG.Create_qual(
     p_seeded_qual_id     => -9600,
/* This number should correspond to the number used in the alias for
the attribute*/
     p_name               => names(i),
    ,p_description        => descript(i),
    ,p_language           => lang(i),

     ,/* Sales Usage: FK to
     JTF_SOURCES_ALL.SOURCE_ID*/
     p_source_id          => -1600

    ,/* Sales Lead Transaction Type: FK to
        JTF_QUAL_TYPE_USGS_ALL.QUAL_TYPE_USG_ID*/
     p_trans_type_id      => -1601

    ,p_enabled_flag       => 'N'

    ,/* QUAL_RELATION_FACTOR
     This needs to be the next prime number greater than
     the value from the following SQL:

     SELECT MAX(qual_relation_factor)
     FROM jtf_qual_usgs_all

     The first 1000 primes can be found at:
     http://primes.utm.edu/lists/small/1000.txt

     Custom qualifiers should start at the 303rd prime
     which is 1999. This is to keep 1st to 302nd primes
     for product development to seed qualifiers.
     */
     p_qual_relation_factor       => 2011


    /* The following set of meta-data setups determine
    ** the behaviour of the matching attribute in
    ** HTML and Excel UIs: how it is displayed; what
    ** is the LOV SQL; and, what comparison operators
    ** are supported.
    */

    ,/* CONVERT_TO_ID_FLAG: displayed qualifier value is a CHAR
        but stored qualifier value is stored as an internal id  */
     p_convert_to_id_flag         => 'N'

    ,/* DISPLAY_TYPE: display type on UI (NUMERIC/CHAR) */
     p_display_type               => 'CHAR'
    ,/* LOV SQL */
    p_html_lov_sql1              =>
     ' SELECT a.meaning col1_value, a.lookup_code col2_value ' ||
CHR(10) ||
```

```
' FROM ar_lookups a ' || CHR(10) ||
     ' WHERE a.lookup_type = ''PARTY_TYPE'' ' || CHR(10) ||
     ' AND a.lookup_code IN (''ORGANIZATION'', ''PERSON'') ' || CHR
(10) ||
     ' ORDER BY col1_value '

    ,p_html_lov_sql2             => null
    ,p_html_lov_sql3             => null
    ,p_display_sql1              => null
    ,p_display_sql2              => null
    ,p_display_sql3              => null
    ,p_hierarchy_type            => null

    ,/* Is the "=" operator supported? */
    p_equal_flag                 => 'Y'
    ,/* Is the "LIKE" operator supported? */
    p_like_flag                  => 'N'
    ,/* Is the "BETWEEN" operator supported? */
    p_between_flag               => 'N'

    /* Values table */
    /* matching attribute comparison operator */
    ,p_comparison_operator       => 'q9600_cp'
    ,p_low_value_char            => 'q9600_low_value_CHAR'
    ,p_high_value_char           => null
    ,p_low_value_char_id         => null
    ,p_low_value_number          => null
    ,p_high_value_number         => null
    ,p_interest_type_id          => null
    ,p_primary_interest_code_id  => null
    ,p_sec_interest_code_id      => null
    ,p_value1_id                 => null
    ,p_value2_id                 => null
    ,p_value3_id                 => null
    ,p_value4_id                 => null
    ,p_first_char                => null
    ,p_currency_code             => null


    ,/* Transaction Type Alias/_TRANS table column mapping */
     p_qual_col1                 => 'Q9600_PARTY_TYPE'

    ,/* TCA Classification for derivation */
     p_alias_rule1               => null

    ,/* OP_EQL FOR BATCH-MODE ASSIGNMENT:
     ** Since the AS_SALES_LEADS.STATUS_CODE value is not passed to
the
     ** assignment API, we must the PK (SALES_LEAD_ID) to get the
     ** STATUS_CODE value from AS_SALES_LEADS and compare to the
     ** territory qualifier value
     */
     p_op_eql                    =>
      ' (A.Q9600_PARTY_TYPE = B.q9600_low_value_CHAR AND B.q9600_cp
= ''='')'
    ,p_op_like                   => null
    ,p_op_between                => NULL
    ,p_op_common_where           => null

    ,p_real_time_select          =>
    'SELECT DISTINCT A.trans_object_id, A.trans_detail_object_id, A.
txn_date, B.terr_id, B.absolute_rank, B.top_level_terr_id, B.
num_winners'
    ,p_real_time_from            => null
    ,p_real_time_where           =>
```

```
'WHERE '  || CHR(10) ||
    '  B.q9600_low_value_CHAR = A.Q9600_PARTY_TYPE ' || CHR(10) ||
    '  AND B.q9600_cp = ''='' ' || CHR(10) ||
    '  AND B.source_id = -1600 ' || CHR(10) ||
    '  AND A.txn_date between B.start_date and B.end_date'


    ,retcode                      => retcode
    ,errbuf                       => errbuf);

  dbms_output.put_line('Retcode : ' || retcode);
  dbms_output.put_line('Errbuf : ' || errbuf);

  ad_morg.replicate_seed_data(NULL, 'JTF', NULL);

EXIT WHEN retcode=2;
END LOOP;
END;
COMMIT;

exception
  when others then
    dbms_output.put_line('Retcode : ' || retcode);
    dbms_output.put_line('Errbuf : ' || errbuf);
    raise;
end;
/
```

3. Run this script.

Script 1

```
DECLARE

  CURSOR csr_real_time_sql (
                        lp_source_id       NUMBER
                      , lp_trans_type_id   NUMBER
                      , lp_program_name    VARCHAR2
                      , lp_sql_type        VARCHAR2 ) IS
   SELECT real_time_sql meta_data_SQL
   FROM jty_trans_usg_pgm_sql
   WHERE source_id = lp_source_id
     AND trans_type_id = lp_trans_type_id
     AND program_name LIKE lp_program_name
     AND lp_sql_type = 'real_time_sql';

  CURSOR csr_batch_total_sql (
                        lp_source_id       NUMBER
                      , lp_trans_type_id   NUMBER
                      , lp_program_name    VARCHAR2
                      , lp_sql_type        VARCHAR2 ) IS
   SELECT batch_total_sql meta_data_SQL
   FROM jty_trans_usg_pgm_sql
   WHERE source_id = lp_source_id
     AND trans_type_id = lp_trans_type_id
     AND program_name LIKE lp_program_name
     AND lp_sql_type = 'batch_total_sql';

  CURSOR csr_batch_incr_sql (
                        lp_source_id       NUMBER
                      , lp_trans_type_id   NUMBER
                      , lp_program_name    VARCHAR2
                      , lp_sql_type        VARCHAR2 ) IS
   SELECT batch_incr_sql meta_data_SQL
   FROM jty_trans_usg_pgm_sql
   WHERE source_id = lp_source_id
     AND trans_type_id = lp_trans_type_id
     AND program_name LIKE lp_program_name
     AND lp_sql_type = 'batch_incr_sql';

  CURSOR csr_batch_dea_sql (
                        lp_source_id       NUMBER
                      , lp_trans_type_id   NUMBER
                      , lp_program_name    VARCHAR2
                      , lp_sql_type        VARCHAR2 ) IS
   SELECT batch_dea_sql meta_data_SQL
   FROM jty_trans_usg_pgm_sql
   WHERE source_id = lp_source_id
     AND trans_type_id = lp_trans_type_id
     AND program_name LIKE lp_program_name
     AND lp_sql_type = 'batch_dea_sql';


  CURSOR csr_incr_reassign_sql (
                        lp_source_id       NUMBER
                      , lp_trans_type_id   NUMBER
                      , lp_program_name    VARCHAR2
                      , lp_sql_type        VARCHAR2 ) IS
   SELECT incr_reassign_sql meta_data_SQL
   FROM jty_trans_usg_pgm_sql
   WHERE source_id = lp_source_id
     AND trans_type_id = lp_trans_type_id
     AND program_name LIKE lp_program_name
     AND lp_sql_type = 'incr_reassign_sql';
```

```
  CURSOR csr_related_trans_types (
                lp_source_id      NUMBER
              , lp_trans_type_id  NUMBER ) IS
  SELECT qtd.qual_type_id
       --, qt.name
       , pgm.program_name
       --, pgm.version_name
  FROM jty_trans_usg_pgm_sql pgm
     , jtf_qual_type_denorm_v qtd
     , jtf_qual_types_all qt
     , jtf_qual_type_usgs_all qtu
     , jtf_sources_all u
  WHERE pgm.enabled_flag = 'Y'
    AND pgm.source_id = qtu.source_id
    AND pgm.trans_type_id = qtu.qual_type_id
    AND qtd.qual_type_id = qt.qual_type_id
    AND qt.qual_type_id = qtu.qual_type_id
    AND qtu.source_id = u.source_id
    AND u.source_id = lp_source_id
    AND qtd.related_id = lp_trans_type_id
  ORDER BY qtd.qual_type_id, pgm.program_name;

  v_clob           CLOB := NULL;
  v_clob_length    NUMBER := 0;

  v_line_start     NUMBER := 1;
  v_line_end       NUMBER := 0; -- End of the current line which will be
read
  v_line_length    NUMBER := 0; -- Length of current line which will be
read
  v_line           VARCHAR2(32767);
  v_last_line      VARCHAR2(1) := 'N';

  v_line_number    NUMBER := 0;
  p_line_number    NUMBER := 0;

  p_source_id      NUMBER := -1001;
  p_trans_type_id  NUMBER := -1002;
  p_version_name   VARCHAR2(60) := SYSDATE ||  ': CMA PARTY TYPE ADDED';
  v_source_id      NUMBER := NULL;
  v_trans_type_id  NUMBER := NULL;
  v_program_name   VARCHAR2(60) := NULL;
  v_sql_type       VARCHAR2(60) := NULL;

  v_sql_number     NUMBER := 1;

BEGIN

  dbms_output.put_line('DECLARE');
  dbms_output.put_line('');
  dbms_output.put_line('  l_real_time_sql varchar2(32000) := NULL; ');
  dbms_output.put_line('  l_batch_total_sql varchar2(32000) := NULL; ');
  dbms_output.put_line('  l_batch_incr_sql varchar2(32000) := NULL; ');
  dbms_output.put_line('  l_batch_dea_sql varchar2(32000) := NULL; ');
  dbms_output.put_line('  l_incr_reassign_sql varchar2(32000) := NULL;
');
  dbms_output.put_line('');
  dbms_output.put_line('  retcode varchar2(250); ');
  dbms_output.put_line('  errbuf  varchar2(1000); ');
  dbms_output.put_line('');
  dbms_output.put_line('BEGIN');
  dbms_output.put_line('');


  FOR my_csr IN csr_related_trans_types(p_source_id, p_trans_type_id)
```

```
LOOP

        v_source_id := p_source_id;
        v_trans_type_id := my_csr.qual_type_id;
        v_program_name := my_csr.program_name;
        v_sql_number := 1;

        dbms_output.put_line('/* ' || v_program_name || ' */');

        WHILE (v_sql_number <= 5) LOOP

          v_line_start     := 1;
          v_line_end       := 0;
          v_line_length    := 0;
          v_clob           := NULL;
          v_line           := NULL;

           /* Get Real Time SQL */
           IF (v_sql_number = 1) THEN

             v_sql_type := 'real_time_sql';
             OPEN csr_real_time_sql(v_source_id, v_trans_type_id,
v_program_name, v_sql_type);
             FETCH csr_real_time_sql INTO v_clob;
             CLOSE csr_real_time_sql;

           /* Get Batch Total Mode SQL */
           ELSIF (v_sql_number = 2) THEN

             v_sql_type := 'batch_total_sql';
             OPEN csr_batch_total_sql(v_source_id, v_trans_type_id,
v_program_name, v_sql_type);
             FETCH csr_batch_total_sql INTO v_clob;
             CLOSE csr_batch_total_sql;

           /* Get Batch Incremental Mode SQL */
           ELSIF (v_sql_number = 3) THEN

             v_sql_type := 'batch_incr_sql';
             OPEN csr_batch_incr_sql(v_source_id, v_trans_type_id,
v_program_name, v_sql_type);
             FETCH csr_batch_incr_sql INTO v_clob;
             CLOSE csr_batch_incr_sql;

            /* Get Batch Mode Date Effective SQL */
           ELSIF (v_sql_number = 4) THEN

             v_sql_type := 'batch_dea_sql';
             OPEN csr_batch_dea_sql(v_source_id, v_trans_type_id,
v_program_name, v_sql_type);
             FETCH csr_batch_dea_sql INTO v_clob;
             CLOSE csr_batch_dea_sql;



            /* Get Batch Mode Date Effective SQL */
           ELSIF (v_sql_number = 5) THEN

             v_sql_type := 'incr_reassign_sql';
             OPEN csr_incr_reassign_sql(v_source_id, v_trans_type_id,
v_program_name, v_sql_type);
             FETCH csr_incr_reassign_sql INTO v_clob;
             CLOSE csr_incr_reassign_sql;

           END IF;
```

```
v_clob_length := DBMS_LOB.GETLENGTH(v_clob);

           dbms_output.put_line('');
           dbms_output.put_line('');
           dbms_output.put_line('l_' || v_sql_type || ' := ');
           --dbms_output.put_line('START: v_line_start='||TO_CHAR
(v_line_start));
           --dbms_output.put_line('LENGTH: v_clob_length='||TO_CHAR
(v_clob_length));

           IF (v_clob_length = 0) THEN
             v_line := 'NULL;';
           END IF;

           dbms_output.put_line(v_line);

          /* Write the CLOB to file in chunks, each chunk
          ** is delimited by the new-line character
          */
          WHILE (v_line_start <= v_clob_length) LOOP

            v_line_number := v_line_number + 1;

            IF (p_line_number > 0 AND
                v_line_number = p_line_number) THEN
               EXIT;
            END IF;

            /* find position of next new-line character */
            v_line_end := DBMS_LOB.INSTR(v_clob, CHR(10), v_line_start,
1);
            --dbms_output.put_line('v_line_end='||TO_CHAR(v_line_end));

            IF (v_line_end = 0) THEN
              /* Last new-line reached so remaining part of CLOB to be
              ** output is from the current line start point to the
              ** end of the CLOB.
              */
              --dbms_output.put_line('Last new line reached at ' ||
TO_CHAR(v_line_start));
              v_line_end := v_clob_length + 1;
              --dbms_output.put_line('v_line_end='||TO_CHAR
(v_line_end));

              v_last_line := 'Y';

            END IF;

            /* +1 as we want to include new-line
            ** character in the output
            */
            v_line_length := v_line_end - v_line_start + 1;
            --dbms_output.put_line('LENGTH: v_line_length='||TO_CHAR
(v_line_length));

            v_line := TO_CHAR( DBMS_LOB.SUBSTR(v_clob, v_line_length-1,
v_line_start));

            --v_line := REPLACE(v_line, ''', ''''') ;

            v_line := REPLACE(v_line, '''', ''''''');

            IF (v_last_line = 'N') THEN
              dbms_output.put_line('''' || v_line || ' '' ');
            ELSE
              dbms_output.put_line('''' || v_line || ' '';');
```

```
END IF;

            v_line_start := v_line_start + v_line_length;

            v_last_line := 'N';

          END LOOP;

          v_sql_number := v_sql_number + 1;

          dbms_output.put_line('');
          dbms_output.put_line('');
          --dbms_output.put_line('Number of lines = '||TO_CHAR
(v_line_number));
          --dbms_output.put_line('VERY END: v_clob_position='||TO_CHAR
(v_line_start));


        END LOOP; /* WHILE (v_sql_number <= 5) LOOP */

        dbms_output.put_line('');
        dbms_output.put_line('');
        dbms_output.put_line('  JTY_TRANS_USG_PGM_SQL_PKG.Insert_Row(
');
        dbms_output.put_line('   p_source_id => ' || p_source_id );
        dbms_output.put_line('  ,p_trans_type_id => ' || my_csr.
qual_type_id );
        dbms_output.put_line('  ,p_program_name => ''' || my_csr.
program_name || ''' ');
        dbms_output.put_line('  ,p_version_name => ''' || p_version_name
|| ''' ');
        dbms_output.put_line('  ,p_real_time_sql => l_real_time_sql ');
        dbms_output.put_line('  ,p_batch_total_sql => l_batch_total_sql
');
        dbms_output.put_line('  ,p_batch_incr_sql => l_batch_incr_sql
');
        dbms_output.put_line('  ,p_batch_dea_sql => l_batch_dea_sql ');
        dbms_output.put_line('  ,p_incr_reassign_sql =>
l_incr_reassign_sql ');
        dbms_output.put_line('  ,p_use_total_for_dea_flag => null ');
        dbms_output.put_line('  ,p_enabled_flag => ''Y'' ');
        dbms_output.put_line('  ,retcode => retcode ');
        dbms_output.put_line('  ,errbuf => errbuf); ');
        dbms_output.put_line('');
        dbms_output.put_line('');
        dbms_output.put_line('dbms_output.put_line(''' || my_csr.
program_name || ' retcode : '' || retcode);');
        dbms_output.put_line('dbms_output.put_line(''' || my_csr.
program_name || ' errbuf : '' || errbuf);');
        dbms_output.put_line('');

  END LOOP; /* OPEN csr_related_trans_types */

  dbms_output.put_line('');
  dbms_output.put_line('');
  dbms_output.put_line('COMMIT;');
  dbms_output.put_line('');
  dbms_output.put_line('END;');
  dbms_output.put_line('/');


END;
```

Index

A

account merge, 6-3
alignment, 1-3
  set up, 5-10
APIs
  Create Geography, 5-12
attributes
  enable, 5-6
  sales, 5-6

C

Calculate Territory Alignment Metrics
  concurrent program, 5-8, 5-11
concurrent programs, 3-1, 3-2, 5-8, 5-11
custom matching attributes
  account example, A-1
  Collections example, A-83
  create, 5-3
  lead example, A-16
  opportunity example, A-61
  proposal example, A-52
  quote example, A-37
  script 1, A-96

D

dependencies, 2-1, 2-1
Dun & Bradstreet Data, 5-10

E

Excel, 2-1, 5-12
export, 2-1
  alignments, 1-3
  set up, 5-12

F

features, 1-4

G

geography, 1-3
  API, 5-12
  load data, 5-12

I

implementation
  process, 3-1

M

matching attributes
  custom, 5-3
  enable, 5-6
  sales, 5-6
matching rules, 4-19

N

named accounts, 1-2
Named Account Territory Post Processing
  concurrent program, 5-8
non-sales territories, 1-3

O

overview, 1-1

P

party merge, 6-3
planning, 4-2
  example, 4-5
profile options, 6-1
proxy user, 5-11

R

rank, 4-4, 4-16, 4-17

S

sales
  matching attributes, 5-6
security, 4-2, 5-1
Synchronize Territory Assignment Rules, 3-1, 3-2
, 5-8
system profile options, 5-9, 5-11

T

territory assignment program, 5-10
Territory Assignment Program (TAP), 1-1
Trading Community Architecture, 2-1
  merge, 6-3

V

value added tax, 5-3

W

web adi
  prerequisites, 6-2
Web ADI, 2-1, 5-12