# Oracle® Service

Implementation Guide

Release 11*i*

February 2002

Part No. A92138-03

ORACLE®

Oracle Service Implementation Guide, Release 11*i*

Part No. A92138-03

# Contents

# Send Us Your Comments

**Oracle Service Implementation Guide, Release 11*i***

**Part No. A92138-03**

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

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

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

- CRM Content Development Manager
  Oracle Corporation
  500 Oracle Parkway
  Redwood Shores, CA 94065
  U.S.A.

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

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

# Preface

## Audience for This Guide

Welcome to Release 11*i* of the *Oracle Service Implementation Guide*.

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

- The principles and customary practices of your business area.
- Oracle Service

  If you have never used Oracle Service, Oracle suggests you attend one or more of the Oracle Service training classes available through Oracle University.
- The Oracle Applications graphical user interface.

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

See Other Information Sources for more information about Oracle Applications product information.

## Documentation Accessibility

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

visit the Oracle Accessibility Program Web site at
`http://www.oracle.com/accessibility/`.

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

**Accessibility of Links to External Web Sites in Documentation** This documentation may contain links to Web sites of other companies or organizations that Oracle Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.

## How To Use This Guide

This document contains the information you need to implement the Charges and Counters components of the Oracle Service application.

- Chapter 1 provides an overview of the Charges application and details of its implementation steps.
- Chapter 2 provides an overview of the Counters application and details of its implementation steps.
- Appendix A provides the Charges application programming interfaces (APIs).

## Typographic Conventions

In examples, an implied carriage return occurs at the end of each line, unless otherwise noted. You must press the Return key at the end of a line of input.

The following conventions are used in this manual:

| Convention | Meaning |
|---|---|
| .<br>.<br>. | Vertical ellipsis points in an example mean that information not directly related to the example has been omitted. |
| . . . | Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted |

| Convention | Meaning |
|---|---|
| **boldface text** | Boldface type in text indicates a term defined in the text, the glossary, or in both locations. |
| < > | Angle brackets enclose user-supplied names. |
| [ ] | Brackets enclose optional clauses from which you can choose one or none. |

# Other 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 Service.

If this guide refers you to other Oracle Applications documentation, use only the Release 11i versions of those guides.

## Online Documentation

All Oracle Applications documentation is available online (HTML or PDF). Online help patches are available on MetaLink.

## Related Documentation

Oracle Service shares business and setup information with other Oracle Applications products. Therefore, you may want to refer to other product documentation when you set up and use Oracle Service.

You can read the documents online by choosing Library from the expandable menu on your HTML help window, by reading from the Oracle Applications Document Library CD included in your media pack, or by using a Web browser with a URL that your system administrator provides.

If you require printed guides, you can purchase them from the Oracle Store at http://oraclestore.oracle.com.

## Documents Related to All Products

### Oracle Applications User's Guide

This guide explains how to enter data, query, run reports, and navigate using the graphical user interface (GUI) available with this release of Oracle Service (and any other Oracle Applications products). This guide also includes information on

setting user profiles, as well as running and reviewing reports and concurrent processes.

You can access this user's guide online by choosing "Getting Started with Oracle Applications" from any Oracle Applications help file.

## Documents Related to This Product

### Oracle Service Concepts and Procedures

This guide provides instructions for using the components of the Oracle Service application including Charges and Counters.

## Installation and System Administration

### Oracle Applications Concepts

This guide provides an introduction to the concepts, features, technology stack, architecture, and terminology for Oracle Applications Release 11*i*. It provides a useful first book to read before an installation of Oracle Applications. This guide also introduces the concepts behind Applications-wide features such as Business Intelligence (BIS), languages and character sets, and Self-Service Web Applications.

### Installing Oracle Applications

This guide provides instructions for managing the installation of Oracle Applications products. In Release 11*i*, much of the installation process is handled using Oracle Rapid Install, which minimizes the time to install Oracle Applications, the Oracle8 technology stack, and the Oracle8*i* Server technology stack by automating many of the required steps. This guide contains instructions for using Oracle Rapid Install and lists the tasks you need to perform to finish your installation. You should use this guide in conjunction with individual product user's guides and implementation guides.

### Oracle Applications Supplemental CRM Installation Steps

This guide contains specific steps needed to complete installation of a few of the CRM products. The steps should be done immediately following the tasks given in the Installing Oracle Applications guide.

### Upgrading Oracle Applications

Refer to this guide if you are upgrading your Oracle Applications Release 10.7 or Release 11.0 products to Release 11*i*. This guide describes the upgrade process and lists database and product-specific upgrade tasks. You must be either at Release 10.7

(NCA, SmartClient, or character mode) or Release 11.0, to upgrade to Release 11*i*. You cannot upgrade to Release 11*i* directly from releases prior to 10.7.

### Maintaining Oracle Applications

Use this guide to help you run the various AD utilities, such as AutoUpgrade, AutoPatch, AD Administration, AD Controller, AD Relink, License Manager, and others. It contains how-to steps, screenshots, and other information that you need to run the AD utilities. This guide also provides information on maintaining the Oracle applications file system and database.

### Oracle Applications System Administrator's Guide

This guide provides planning and reference information for the Oracle Applications System Administrator. It contains information on how to define security, customize menus and online help, and manage concurrent processing.

### Oracle Alert User's Guide

This guide explains how to define periodic and event alerts to monitor the status of your Oracle Applications data.

### Oracle Applications Developer's Guide

This guide contains the coding standards followed by the Oracle Applications development staff. It describes the Oracle Application Object Library components needed to implement the Oracle Applications user interface described in the *Oracle Applications User Interface Standards for Forms-Based Products*. It also provides information to help you build your custom Oracle Forms Developer 6*i* forms so that they integrate with Oracle Applications.

### Oracle Applications User Interface Standards for Forms-Based Products

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

## Other Implementation Documentation

### Multiple Reporting Currencies in Oracle Applications

If you use the Multiple Reporting Currencies feature to record transactions in more than one currency, use this manual before implementing Oracle Service. This

manual details additional steps and setup considerations for implementing Oracle Service with this feature.

### Multiple Organizations in Oracle Applications

This guide describes how to set up and use Oracle Service with Oracle Applications' Multiple Organization support feature, so you can define and support different organization structures when running a single installation of Oracle Service.

### Oracle Workflow Guide

This guide explains how to define new workflow business processes as well as customize existing Oracle Applications-embedded workflow processes.You also use this guide to complete the setup steps necessary for any Oracle Applications product that includes workflow-enabled processes.

### Oracle Applications Flexfields Guide

This guide provides flexfields planning, setup and reference information for the Oracle Service implementation team, as well as for users responsible for the ongoing maintenance of Oracle Applications product data. This manual also provides information on creating custom reports on flexfields data.

### Oracle eTechnical Reference Manuals

Each eTechnical Reference Manual (eTRM) contains database diagrams and a detailed description of database tables, forms, reports, and programs for a specific Oracle Applications product. This information helps you convert data from your existing applications, integrate Oracle Applications data with non-Oracle applications, and write custom reports for Oracle Applications products. Oracle eTRM is available on Metalink

### Oracle Manufacturing APIs and Open Interfaces Manual

This manual contains up-to-date information about integrating with other Oracle Manufacturing applications and with your other systems. This documentation includes APIs and open interfaces found in Oracle Manufacturing.

### Oracle Order Management Suite APIs and Open Interfaces Manual

This manual contains up-to-date information about integrating with other Oracle Manufacturing applications and with your other systems. This documentation includes APIs and open interfaces found in Oracle Order Management Suite.

### Oracle Applications Message Reference Manual

This manual describes Oracle Applications messages. This manual is available in HTML format on the documentation CD-ROM for Release 11i.

### Oracle CRM Application Foundation Implementation Guide

Many CRM products use components from CRM Application Foundation. Use this guide to correctly implement CRM Application Foundation.

## Training and Support

### Training

Oracle offers training courses to help you and your staff master Oracle Service and reach full productivity quickly. You have a choice of educational environments. You can attend courses offered by Oracle University at any one 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's 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 Oracle Service 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 Oracle8*i* server, and your hardware and software environment.

### Oracle*MetaLink*

Oracle*MetaLink* is your self-service support connection with web, telephone menu, and e-mail alternatives. Oracle supplies these technologies for your convenience, available 24 hours a day, 7 days a week. With Oracle*MetaLin*k, you can obtain information and advice from technical libraries and forums, download patches, download the latest documentation, look at bug details, and create or update TARs. To use MetaLink, register at (http://metalink.oracle.com).

**Alerts:** You should check Oracle*MetaLink* alerts before you begin to install or upgrade any of your Oracle Applications. Navigate to the Alerts page as follows:

Technical Libraries/ERP Applications/Applications Installation and Upgrade/Alerts.

**Self-Service Toolkit:** You may also find information by navigating to the Self-Service Toolkit page as follows: Technical Libraries/ERP Applications/Applications Installation and Upgrade.

## Do Not Use Database Tools to Modify Oracle Applications Data

***Oracle STRONGLY RECOMMENDS that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle Applications 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 Applications data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle Applications tables are interrelated, any change you make using Oracle Applications can update many tables at once. But when you modify Oracle Applications data using anything other than Oracle Applications, 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 Applications.

When you use Oracle Applications to modify your data, Oracle Applications automatically checks that your changes are valid. Oracle Applications 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.

## About Oracle

Oracle Corporation develops and markets an integrated line of software products for database management, applications development, decision support, and office automation, as well as Oracle Applications, an integrated suite of more than 160 software modules for financial management, supply chain management, manufacturing, project systems, human resources and customer relationship management.

Oracle products are available for mainframes, minicomputers, personal computers, network computers and personal digital assistants, allowing organizations to

integrate different computers, different operating systems, different networks, and even different database management systems, into a single, unified computing and information resource.

Oracle is the world's leading supplier of software for information management, and the world's second largest software company. Oracle offers its database, tools, and applications products, along with related consulting, education, and support services, in over 145 countries around the world.

# 1

# Implementing Charges

This topic group provides descriptions of the setup and configuration tasks required to implement the application successfully.

## 1.1 Overview of Charges

### 1.1.1 Definition of Charges

Charges is a component of the Oracle Service application and is used by many Oracle applications such as Oracle Field Service and Oracle Depot Repair. See "Understanding Charges" in *Oracle Service Concepts and Procedures* for more information.

Typical business needs that Charges supports are as follows:

- Create orders and shipments
- Create return orders (RMA)
- Bill customers for any services provided to them

Charges has the following basic features and functions:

- Create and view charge lines (orders, returns, and billing)
- Create estimated charge lines
- Roll up charge lines (labor, material, expenses) into a defined item (labor, material, and expenses)
- Associate charge lines with an existing order
- Apply contract terms and conditions to charge lines
- View coverage information for a contract

- Submit charge lines to Order Management through Quoting - Forms
- Achieve multi-currency compliance

### 1.1.2 New Items in Charges Implementation for This Release

New items in Charges implementation for this release include the following:

- Profile option to default the business process in the Charges window from the service request type
- Profile option to set the default price list in the Charges window
- Additional window for transaction billing types to associate Order Management header types and line type by operating unit
- Setup window to define how a billing operating unit is defaulted in Charges

### 1.1.3 Architecture of Charges

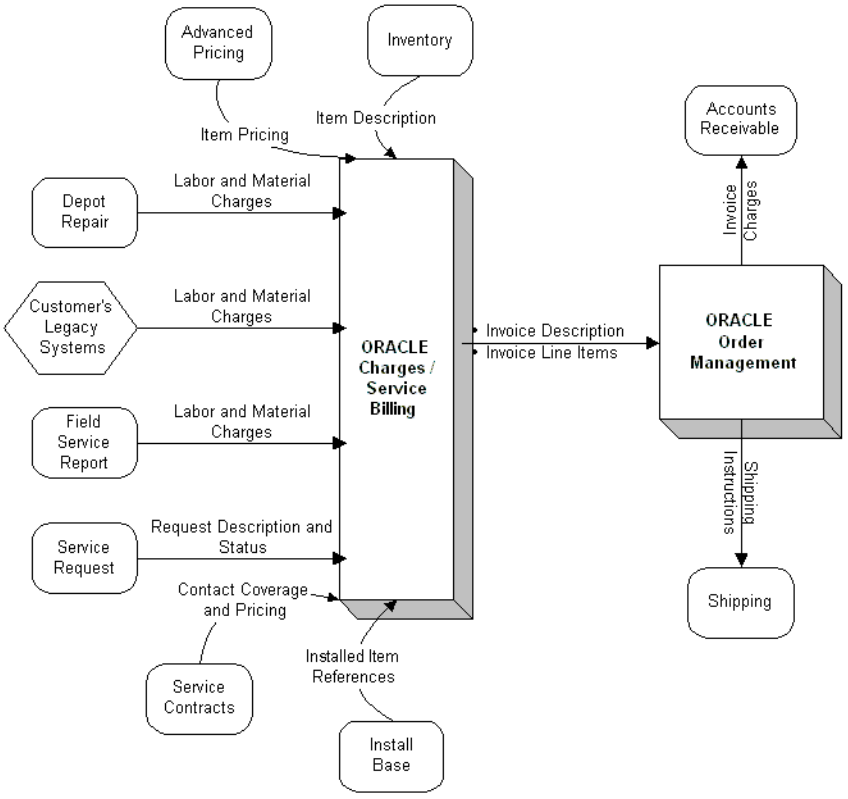
Charges screens remain based on the forms technology stack.

Charges integrates with the following Oracle applications:

- Support
- Field Service
- Depot Repair
- Contracts
- Quoting - Forms
- Inventory
- Receivables
- Install Base
- Order Management

The following figure illustrates the architecture and data flow for Charges.

***Figure 1–1 Architecture and Data Flow for Charges***




Charges integrates with Order Management for returns, shipments, and billing through Quoting - Forms. When you book an order (shipment or return), Quoting - Forms passes the order to Oracle Order Management, an application that determines how the order is filled and shipped. Quoting - Forms then gives you a confirmation number which is displayed in the Charges form in the Line Number field. After the invoice cycle is complete, the invoice number and the invoice date are also displayed on the Others tabbed page of the Charges window in the Invoice Number and Invoice Date fields respectively.

Service Request and Depot Repair integrate directly with Charges, whereas Field Service communicates with Charges through the Charges APIs.

### 1.1.4 Charges Users

Charges has no seeded users, and defining users is not applicable to Charges.

## 1.2 Prerequisites

As a prerequisite to the setup of Charges, confirm the setup of all applications at your site that will use Charges, such as Oracle Field Service.

## 1.3 Implementation and Setup

### 1.3.1 Comprehensive Setup List

This is the comprehensive list of steps to verify that Charges is set up correctly:

1. Set up system administration
2. Define key flexfields
3. Confirm the setup of General Ledger
4. Define responsibilities
5. Define users
6. Confirm the setup of Inventory
7. Confirm the setup of Order Management
8. Confirm the setup of Advanced Pricing
9. Confirm the setup of Service Contracts
10. Confirm the setup of Service Request
11. Define inventory items
12. Define units of measure
13. Define line types
14. Define line categories
15. Define price lists
16. Define return reason codes
17. Define currency conversion types

18. Define business processes
19. Define transaction types and relate them to billing types and Order Management header and line types
20. Confirm the setup of the universal work queue
21. Confirm the setup of Contracts
22. Define rules for defaulting the charge line operating unit

## 1.3.2 Setup Checklist

This guide focuses on only certain setup steps for types and codes that are required for transaction processing in Charges. Here is the list of these steps, which is composed of the headings of each step that is detailed in subsequent sections:

1. Confirm the Setup of Inventory Items (Required)
2. Confirm the Setup of Units of Measure (Required)
3. Confirm the Setup of Line Types (Required)
4. Confirm the Setup of the Price List (Required)
5. Confirm the Setup of Line Categories (FND_LOOKUPS) (Required)
6. Confirm the Setup of Return Reason Codes (Required)
7. Confirm the Setup of Currencies (Required)
8. Confirm the Setup of the Functional Currency (Required)
9. Confirm the Setup of Currency Conversion Types (Required)
10. Confirm the Setup of Currency Conversion Rates (Required)
11. Confirm the Setup of Coverage Templates (Required)
12. Confirm the Setup of Contracts (Required)
13. Confirm the Setup of Business Processes (Required)
14. Confirm the Setup of Order Categories (FND_LOOKUPS) (Required)
15. Confirm the Setup of Transaction Types, Transaction Billing Types, and Associated Order Management Mappings (Required)
16. Confirm the Setup of the Field Service Report (Required)
17. Confirm the Setup of Install Base Instance Statuses (Required)

18. Confirm the Setup of Install Base Transaction Types (Required)
19. Confirm the Proper Setup of Defaulting Rules for Billing Operating Units (Required)
20. Define a Manually Overridable Modifier in Oracle Pricing Manager (Optional)

## 1.3.3 Setup Steps

This section contains information about the setup of types and codes that are required for transaction processing in Charges.

### 1.3.3.1 Confirm the Setup of Inventory Items (Required)

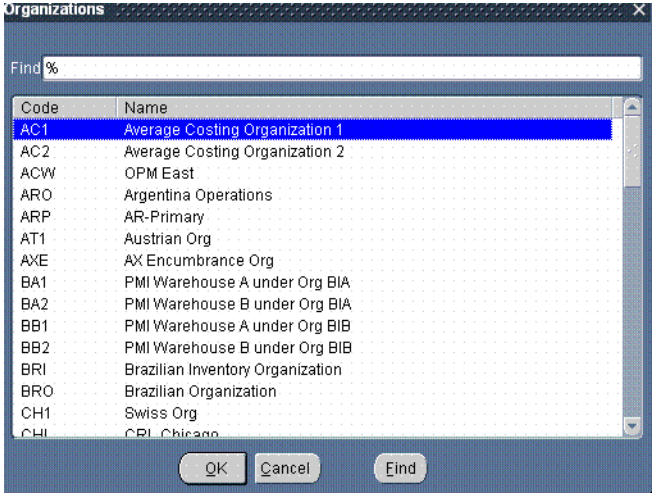
#### Navigation

Forms mode: Order Management Super User responsibility: (N) Items > Master Items

#### Procedures

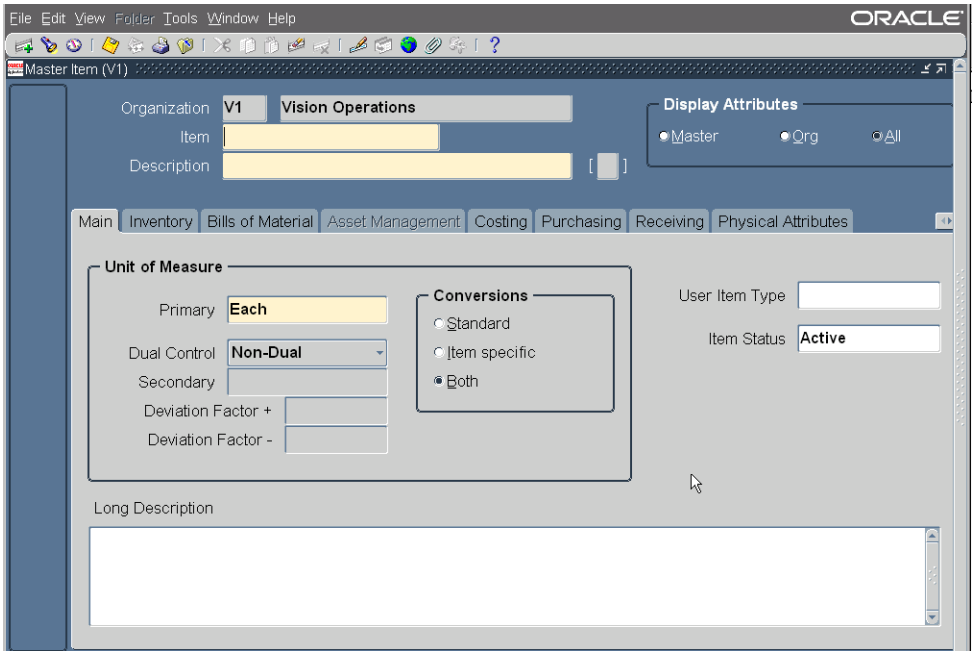
By navigating to Master Items, the Organizations window appears so that you can pick an organization for which you are defining items.

***Figure 1–2 Organizations Window***



Clicking Find brings up a populated Master Item window that can be used to confirm and define an item and its attributes using various tabs. Some tabs are shown when the window opens. More tabs can be accessed by clicking the arrow at the end of the tabs.

***Figure 1–3 Master Item Window***




The following table shows examples of master items with different attributes.

***Table 1–1 Examples of Master Items***

| Item Description | Serviceable Flag | Service Flag | Shippable Flag | Billing Type | Costed | Priced | IB Trackable Flag | Stockable Flag |
|---|---|---|---|---|---|---|---|---|
| Products Repaired/ Upgraded | Y | N | Y | M | N | N | Y | Y |
| Parts used in the Depot/WIP | Y or N | N | Y or N | M | Y | N | N | Y |

***Table 1–1 Examples of Master Items***

| Item Description | Serviceable Flag | Service Flag | Shippable Flag | Billing Type | Costed | Priced | IB Trackable Flag | Stockable Flag |
|---|---|---|---|---|---|---|---|---|
| Parts Charges | N | N | N | M | N | Y | Y | N |
| Labor Charges | N | N | N | L | N | Y | N | N |
| Expense Charges | N | N | N | E | N | Y | N | N |
| Service Charges (for example, Major Overhaul, 50,000 Click Service, Oil Change, PM) | N | N | N | L | N | Y | N | N |

### References

*Oracle Inventory User's Guide*

### 1.3.3.2 Confirm the Setup of Units of Measure (Required)

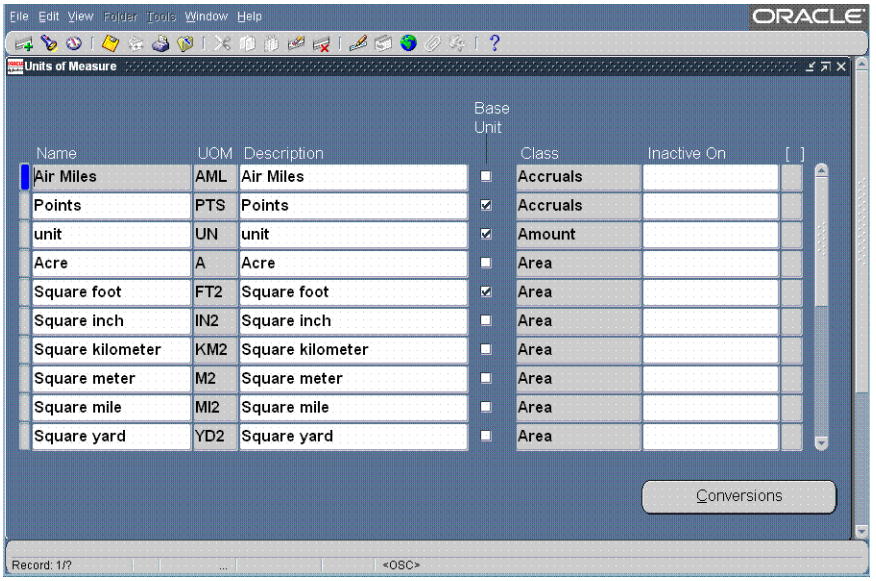
### Navigation

Forms mode: Order Management Super User responsibility: (N) Setup > UOM > Units

### Procedures

You can use the Units of Measure window to confirm and set up the units of measure.

*Figure 1–4 Units of Measure Window*




## References

*Oracle Inventory User's Guide*

### 1.3.3.3 Confirm the Setup of Line Types (Required)

## Navigation

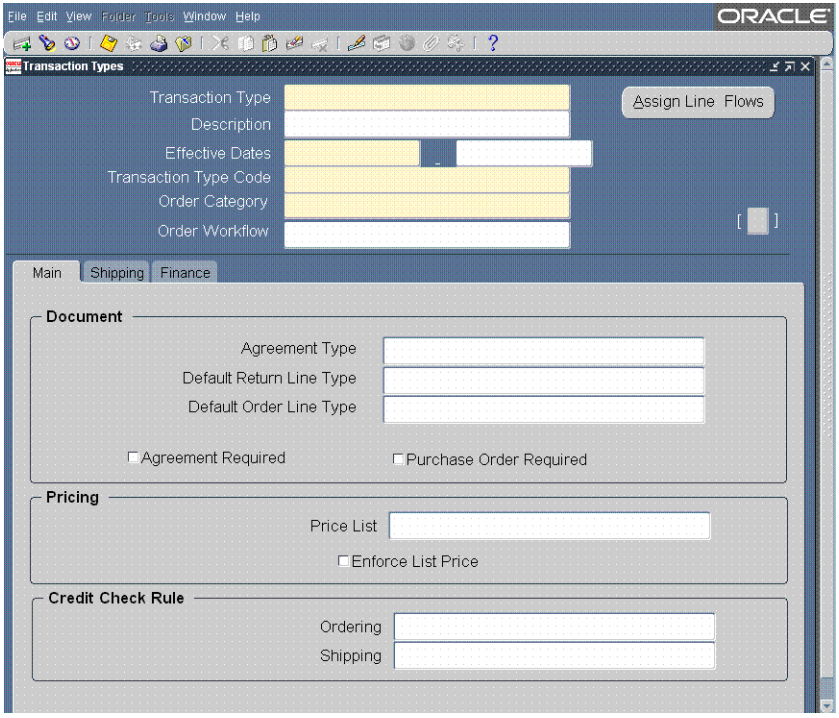
Forms mode: Order Management Super User responsibility: (N) Setup > Transaction Types > Define

## Procedures

You enter a line type on each charge line in the Charges window. The line category is used to populate the charge line's category (which you cannot enter or change) and a price list (which you can change).

In the Transaction Types window, the field Transaction Type Code has two values, Order and Line. You use this window to define line types.

*Figure 1–5 Transaction Types Window for Order Management*




If the transaction type code is of type Order, then the values for order category can have three values, Order, Return, and Mixed. If the transaction type code is of type Line, then the values for order category can have two values, Order and Return.

The Charges window creates Order Management order headers using the order header category for the order header type that is mapped to the transaction billing type.

The following table provides a sample of line types.

*Table 1–2 Line Types*

| Line Types | Line Category | Price List |
|---|---|---|
| Shipment | Order | Corporate |
| Return | Return | Corporate |
| Charge | Order | Corporate |

## References

- *Oracle Order Management Suite Implementation Manual*
- *Oracle Order Management User's Guide*

### 1.3.3.4 Confirm the Setup of the Price List (Required)

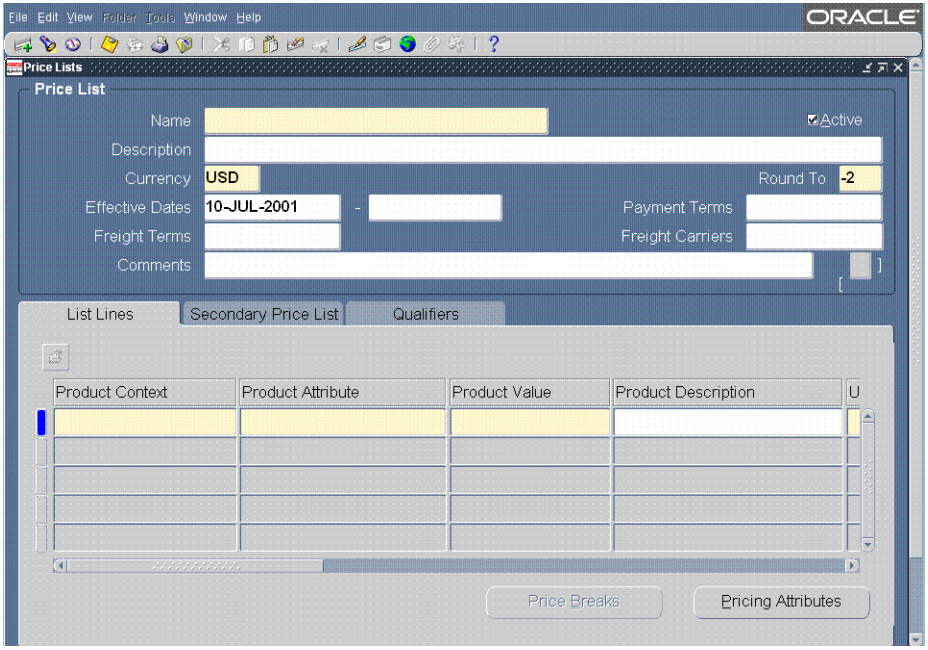
## Navigation

Forms mode: Order Management Super User > Pricing > Price Lists > Price List Setup

## Procedures

You can confirm or define items to the price list by selecting Order Management Super User: (N) Pricing > Price Lists.

***Figure 1–6 Price Lists Window***



### References

- *Oracle Order Management Suite Implementation Manual*
- *Oracle Order Management User's Guide*

### 1.3.3.5 Confirm the Setup of Line Categories (FND_LOOKUPS) (Required)

### Navigation

Forms mode: Order Management Super User: Setup > Quick Codes > Order Management

### Procedures

Query for a line category in the User Name field. The following table shows typical line (order) categories for Order Management.

***Table 1–3 Line Categories for Order Management***

| Order Category | Description |
|---|---|
| Order | Sales |
| Return | Return |

### References

- *Oracle Order Management Suite Implementation Manual*
- *Oracle Order Management User's Guide*

### 1.3.3.6 Confirm the Setup of Return Reason Codes (Required)

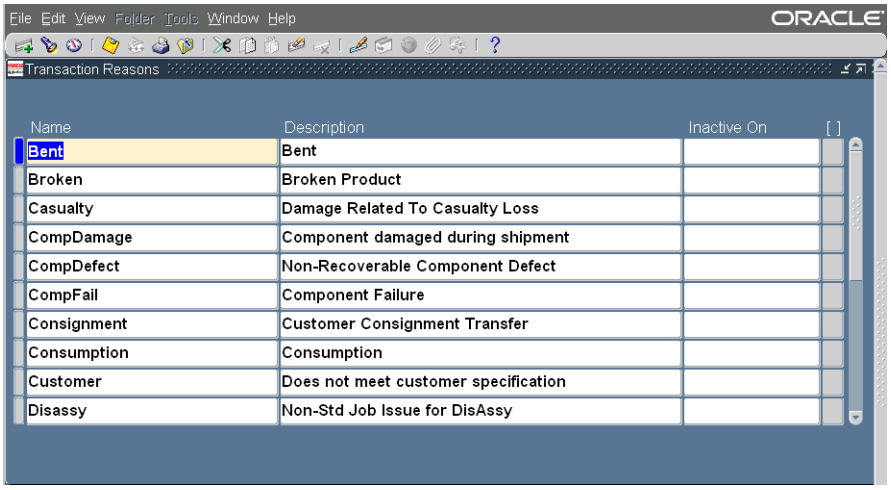
### Navigation

Forms mode: Inventory responsibility: (N) Setup > Transactions > Reasons

### Procedures

Confirm and define return reason codes as necessary.

***Figure 1–7 Transaction Reasons Window (for Return Reason Codes)***




## References

- *Oracle Order Management Suite Implementation Manual*
- *Oracle Order Management User's Guide*

### 1.3.3.7 Confirm the Setup of Currencies (Required)

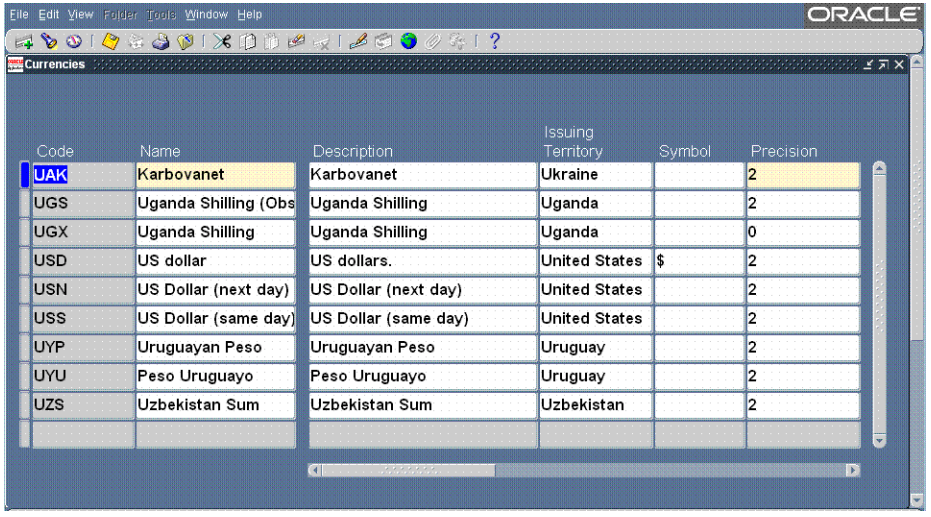
## Navigation

Forms mode: General Ledger responsibility: (N) Setup > Currencies > Define

## Procedures

Use the Currencies window to define the various currencies that you want to use in business transactions.

***Figure 1–8 Currencies Window***




## References

*Oracle General Ledger User Guide*

### 1.3.3.8 Confirm the Setup of the Functional Currency (Required)

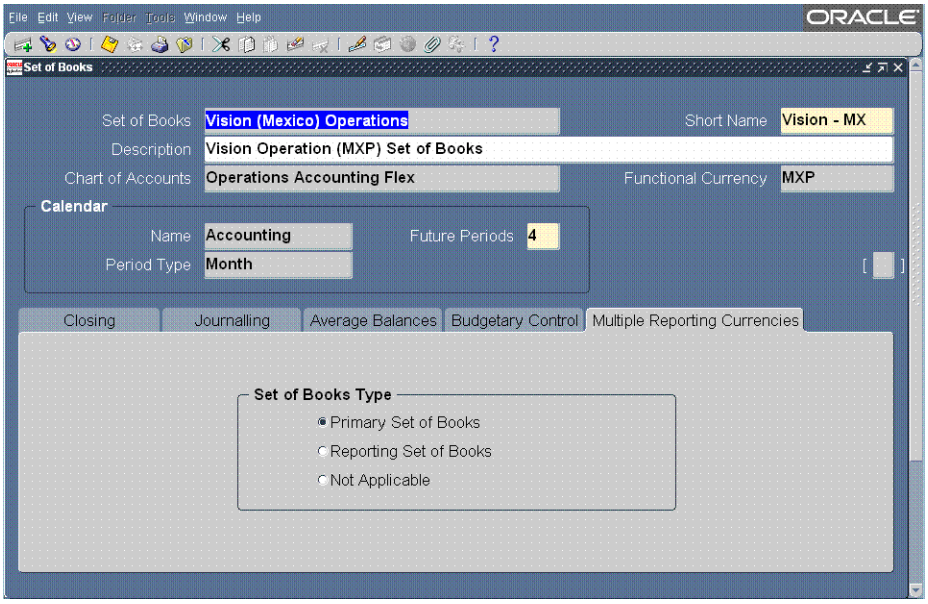
## Navigation

Forms mode: General Ledger responsibility: (N) Setup > Financials > Books > Define

## Procedures

Confirm or define the functional currency is defined while defining the set of books.

*Figure 1–9 Set of Books Window*




## References

*Oracle General Ledger User Guide*

### 1.3.3.9 Confirm the Setup of Currency Conversion Types (Required)

The currency conversion rate and type are used to calculate the charges in currency that is not the functional currency.

The Currency Conversion Rate field and Type field in the Currency and Conversions region on the Charges window are enabled when the currency chosen in the currency field is not a functional currency.

## Navigation

Forms mode: General Ledger responsibility: (N) Setup > Currency > Rates > Types

## References

*Oracle General Ledger User Guide*

### 1.3.3.10 Confirm the Setup of Currency Conversion Rates (Required)

The Currency Conversion rate field and Type field in the Currency and Conversions region box on the Charges form are enabled when the currency chosen in the currency field is not a functional currency.

#### Navigation

Forms mode: General Ledger responsibility: (N) Setup > Currency > Rates > Daily or Period or Historical

#### References

*Oracle General Ledger User Guide*

### 1.3.3.11 Confirm the Setup of Coverage Templates (Required)

You can select a contract and an associated coverage on the Charges window for each charge line.

#### Navigation

Forms mode: Service Contracts Manager responsibility: (N) Coverage Templates

#### Procedures

Confirm and define the coverage templates as required.

#### References

*Oracle Contracts for Service Concepts and Procedures*

### 1.3.3.12 Confirm the Setup of Contracts (Required)

A service contract is associated with a service request and in turn with Charges. The service contract can be defined at the following levels: party, account, system, customer product, and inventory item. A contract and an associated coverage can be selected on the charges window for each charge line.

#### Navigation

Forms mode: Service Contracts Manager responsibility: (N) Launch Contracts > New on the Tools menu. The Create a New Contract window appears.

### Procedures

Use the Oracle Contracts window and the Create a New Contract window to confirm or define related contracts. To be used by Charges, contracts must be have the status of Active.

### References

- *Oracle Contracts for Service Concepts and Procedures*
- *Oracle Service Concepts and Procedures*

## 1.3.3.13 Confirm the Setup of Business Processes (Required)

A business process groups transaction types so that you can restrict transaction type availability. For example, for the field engineers you can define a business process called Field Service that allows field engineers only the transaction types relevant to them.

The business process is entered in the Charges window, where it is used to restrict the list of values for the transaction type. Business process is synonymous with the transaction groups in Oracle Service 3i. The Business Process Transactions window is used to define business processes and associate transaction types with them.

### Navigation

Forms mode: Customer Support responsibility: (N) Setup > Installed Base > Business Processes

### Procedures

In this step you confirm or define necessary business processes. Later, using this window you will associate a transaction type with each business process.

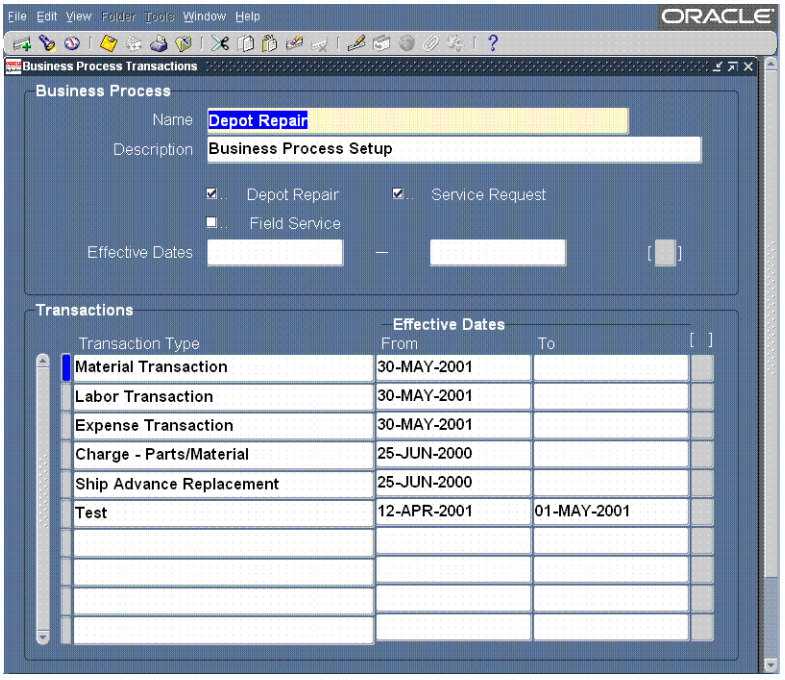
A sample of business processes and associated transaction types follows:

***Table 1–4 Sample of a Business Process and Associated Transaction Type***

| Business Process | Order Type | Service Request Flag | Depot Repair Flag | Field Service Flag | Business Process Transaction Types |
|---|---|---|---|---|---|
| Depot Repair/Returns/ Claims | | Y | Y | Y | All Transaction Types recommended later in this chapter |

This code is entered in the Charges window, where it is used to create the LOV for the Transaction Type. It is defaulted from the service request type. When a transaction type is associated with a business process, all transaction billing types for it are available.

***Figure 1–10 Business Process Transactions Window***




## References

*Oracle Install Base Implementation Guide*

### 1.3.3.14 Confirm the Setup of Order Categories (FND_LOOKUPS) (Required)

The order category attached to the line type is used to populate the charge line's category. You cannot enter or change the order category in the Charges window.

## Navigation

Forms mode: Order Management Super User responsibility: Setup > Quick Codes > Order Management

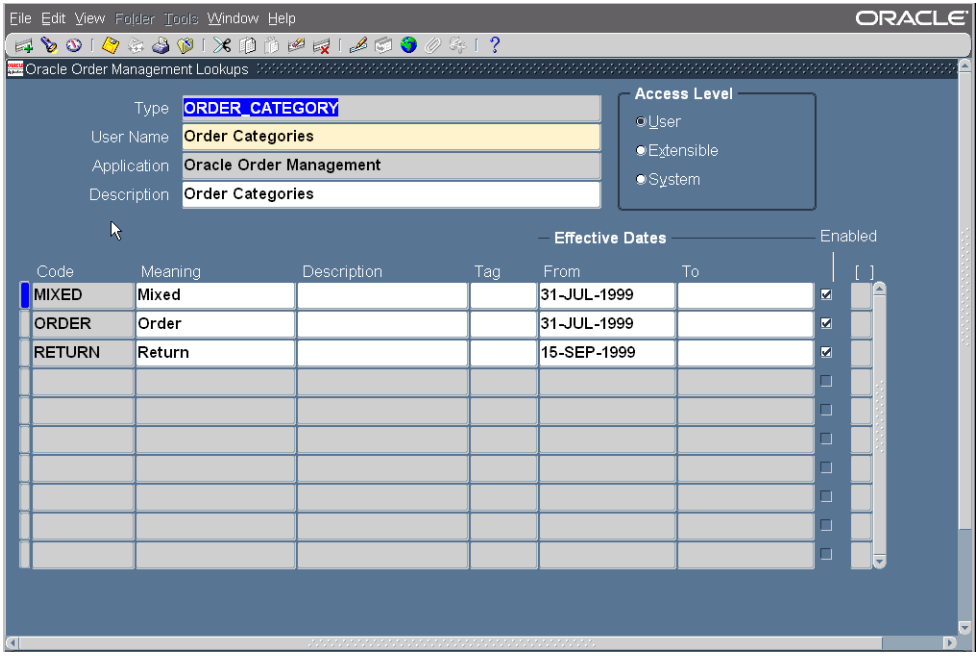
Procedures

Confirm or define the order categories in the User Name field. The following table shows typical order categories for Order Management.

***Table 1–5 Order Categories for Order Management***

| Order Category | Description |
|---|---|
| Order | Sales |
| Return | Return |
| Mixed | Mixed |

***Figure 1–11 Order Management Lookups for Order Categories***



### 1.3.3.15 Confirm the Setup of Transaction Types, Transaction Billing Types, and Associated Order Management Mappings (Required)

#### Navigation

Forms mode: Customer Support responsibility: (N) Setup > Installed Base > Transaction Billing Types

#### Selective Field Descriptions

##### Line Category Code

Line Category Code can be either `Order` or `Return`. This is set at the level of transaction type and prevents a single transaction type from being an order in one operating unit and a return in another.

##### Order Management Header & Line Types

When a charge line is submitted to Order Management, the transaction billing type and the operating unit are used to retrieve an Order Management header type and line type from the setup. These are used to submit the order.

##### Procedures

Use the Transactions Billing Type window to confirm or define business transactions and associated billing types, Order Management header types, and line types.

---

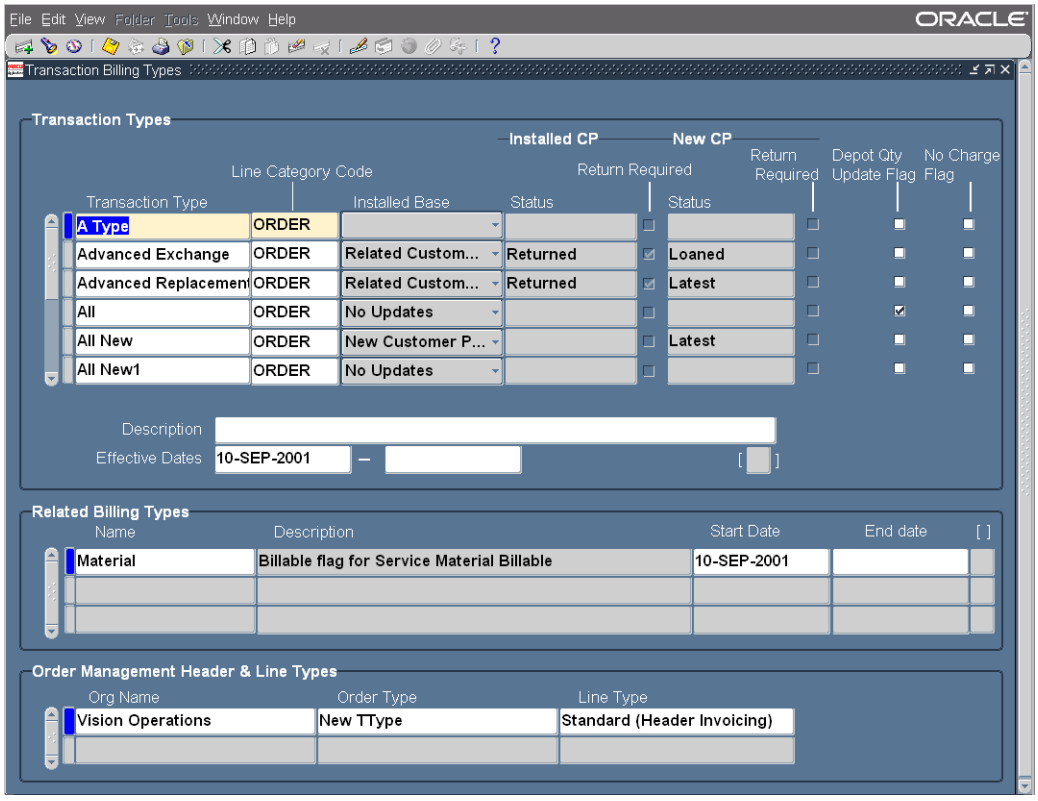
**Note:** This release does not fully support user-defined billing types because they do not work in the Field Service report.

Define material transaction billing types that are used in the Field Service report with Order Management line types that have workflows that do not include a shipping step. This prevents Order Management from shipping an inventory item in addition to one that a field service engineer has already installed.

Similarly, Oracle recommends that transaction types used in Field Service for material is mutually exclusive with transaction types used for material in Service Request and Depot Repair.

---

***Figure 1–12 Transaction Billing Types Window***




For this release, the attributes that control how Install Base is updated have been moved to the setup for subtypes.

Here is more information about billing types from FND_LOOKUPS:

- M: Material
- L: Labor
- E: Expense

Currently the Item Master window permits assignment of only these codes.

### 1.3.3.16 Confirm the Setup of the Field Service Report (Required)

This step is required only if you have installed Oracle Field Service.

## References

*Oracle Field Service Implementation Guide*

### 1.3.3.17 Confirm the Setup of Install Base Instance Statuses (Required)

## Navigation

Forms mode: Customer Support responsibility: (N) Setup > Installed Base > Instance Status

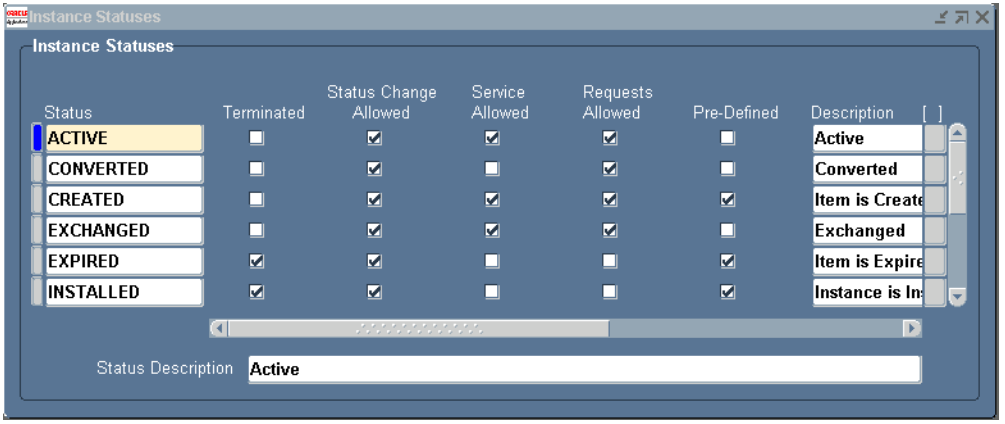
## Procedures

Refer to the latest *Oracle Install Base Implementation Guide* for the current table of Install Base status codes. Use the Instance Statuses window to confirm and define these codes and the related flags.

Here is more information about the flags:

- Status Change Allowed not selected: Install Base update programs do not update the status code for the Install Base item.
- Incident Allowed not selected: Users cannot open a Service Request against the Install Base item.
- Service Order Allowed not selected: Order Management prevents sale of a contract or extended warranty for the Install Base item

***Figure 1–13 Instance Statuses Window***



## References

*Oracle Install Base Implementation Guide*

### 1.3.3.18 Confirm the Setup of Install Base Transaction Types and Sources (Required)

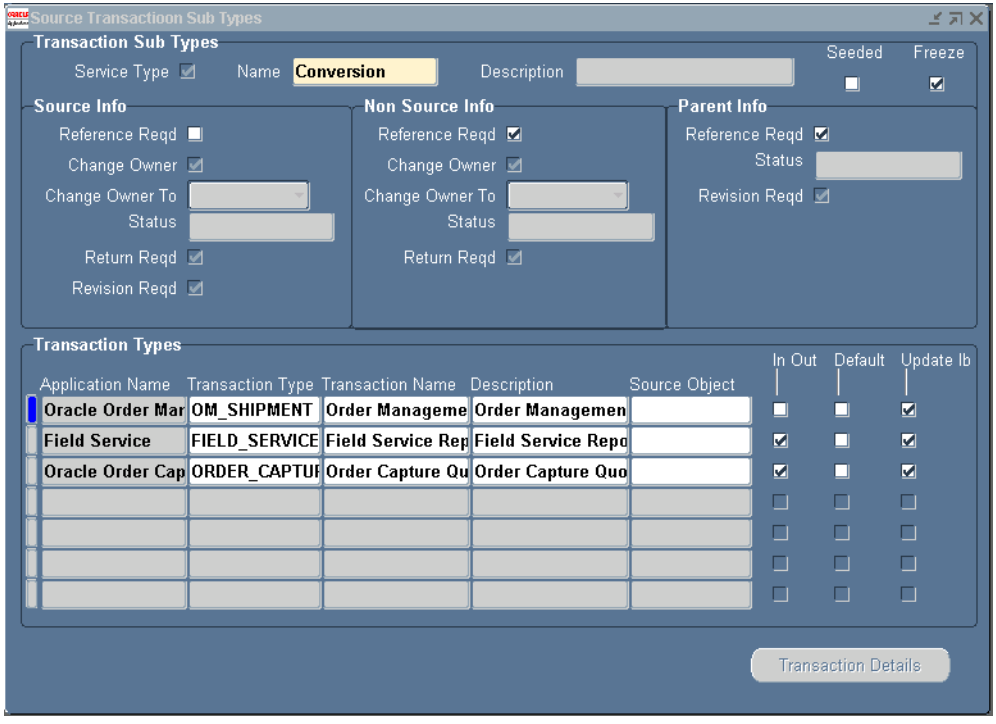
## Navigation

Forms mode: Oracle Installed Base Admin responsibility: (N) Setups > Installed Base Transaction Types

## Procedures

Confirm the setup of Install Base transaction types and subtypes and their sources.

***Figure 1–14 Source Transactions Sub Types Window***



### References

*Oracle Install Base Implementation Guide*

## 1.3.3.19 Confirm the Proper Setup of Defaulting Rules for Billing Operating Units (Required)

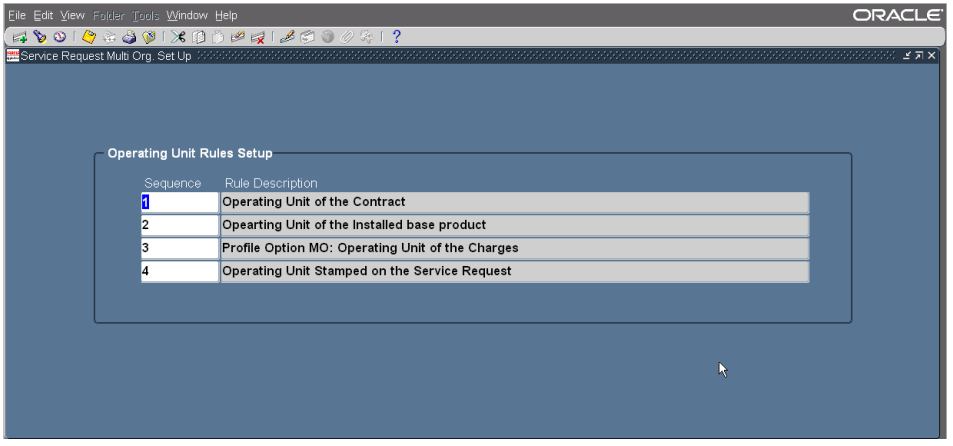
### Navigation

Forms mode: Customer Support responsibility: (N) Setup > Service Requests > Multi Org Setup

### Procedures

Use the Service Request Multi Org Set Up window to set up the defaulting rules for the Charges operating unit. With this window you can select which rules to use and their precedence.

***Figure 1–15 Service Request Multi Org Set Up Window***




Typically your system administrator uses this window to assign a sequence to the four available rules. Based on these rules, an operating unit is derived for use in a calling window, currently from Charges and Field Service. The following table describes the rules.

*Table 1–6 Multi-Org Rules*

| Rule Name | Description |
|---|---|
| Operating Unit of the Contract | If a contract is associated with the service request, then the operating unit of the contract is used. |
| Operating Unit of the Installed Base Product | If an Install Base item is associated with the service request, then the operating unit of the Install Base item is used. |
| Profile Option MO: Operating Unit of the Charge | The operating unit set in this profile is used. |
| Operating Unit Stamped on the Service Request | The operating unit associated with the service request is used. |

Based on the sequence associated with these rules, the appropriate operating unit appears in the Charges window and the Field Service window.

### Example

If a user has set sequence 1 for rule b and 2 for rule d, then the program to derive the operating unit first checks if rule b returns any operating unit. If yes, it returns that operating unit to the calling program. If no, it checks if rule d returns any operating unit. If none of the rules returns any operating unit, then the program returns null.

## 1.3.3.20 Define a Manually Overridable Modifier in Oracle Pricing Manager (Optional)

If you will be adding freight charges in Order Management and modifying the Calculate Price flag in Order Management, then you need to complete the following additional setup. If you do not modify the Calculate Price flag in Order Management, then you do not need to complete these steps.

### Navigation

Forms mode: Oracle Pricing Manager responsibility: (N) Modifiers > Modifier Setup

### Procedures

1. In the Define Modifier window, add a modifier that has the following attributes:
   - Header level Automatic flag cleared
   - Header level Type set to Discount List

- Line level Modifier Type set to Discount
- Application Method set to Percent
- No qualifiers or items attached to it

For details on modifiers, see the *Oracle Advanced Pricing Users Guide*.

2. Set the profile Service: Default Manual Modifier to the modifier created in step 1.

### References

*Oracle Advanced Pricing User's Guide*

## 1.4 Profile Options

### Steps

1. Select the forms mode and the System Administrator responsibility.
2. Navigate Profile > System.
3. Query for the name of the desired profile option for Charges.
4. Confirm or set profile options for Charges according to the following table.

***Table 1–7 Profile Options for Charges***

| Profile Option Name | Possible Values | User | Rest | App | Site | Functionality |
|---|---|---|---|---|---|---|
| Service: Allow Charge Operating Unit Update | Yes and No. | x | x | x | x | Controls whether the operating unit on the charge line can be updated. |
| Service: Charges - Default Business Process from SR | Yes and null. | | | | | If this profile option is set to `Yes`, then the Charges window defaults the business process from the service request type. If the profile option is not set, then the Charges window does not default the business process. |
| Service: Charges -Default Price List | | x | x | x | x | Sets the default price list in the Charges window |

| Profile Option Name | Possible Values | User | Rest | App | Site | Functionality |
|---|---|---|---|---|---|---|
| Service: Default Expense Item | Any item in the Item Master (MTL_ SYSTEM_ITEMS) with Material_ Billable_Flag = "E" | x | x | x | x | Specifies the expense item to be used for charge lines when the Roll-up checkbox is selected. |
| Service: Default Labor Item | Any item in the Item Master (MTL_ SYSTEM_ITEMS) with Material_ Billable_Flag = "L" | x | x | x | x | Specifies the expense item to be used for charge lines when the Roll-up checkbox is selected. |
| Service: Default Manual Modifier | Any modifier defined in the modifier setup of Pricing Manager | | | | | Controls the ability to add freight charges in Order Management when modifying the Calculate Price flag. |
| Service: Default Material Item | Any item in the Item Master (MTL_ SYSTEM_ITEMS) with Material_ Billable_Flag = "M" or null | x | x | x | x | Specifies the expense item to be used for charge lines when the Roll-up checkbox is selected. |

# 2

# Implementing Counters

This topic group provides descriptions of the setup and configuration tasks required to implement the application successfully.

When you implement Counters, you mostly use the same windows that an end user uses to run the application. During implementation, however, the emphasis tends to be on designing the counter and group templates for end users.

## 2.1 Overview of the Counters Application

Using the Counters application you can define counters to track the usage of a customer's product or service and execute business processes that are based on the usage information. Common everyday objects that can have counters are automobile odometers, gas and electric meters, and office equipment such as photocopier machines. Counters provide a mechanism for tracking new product warranties, service contracts, support agreements, and similar business needs.

Service providers rely on counts that originate from time, distance, or usage to closely track the usage of a product or service and to monitor these status counts to:

- Manage product warranties and external warranties
- Manage service, rental, and leasing contracts
- Know when the time has come to reorder or ship parts
- Determine billing cycles and rate change
- Trigger product preventive maintenance or overhauls
- Calculate product reliability
- Perform lifecycle tracking

For example, a company that services photocopiers can use counters to determine how often service must be performed on each piece of equipment, or to know when consumable items such as toner cartridges and paper must be replenished at the customer's site. Counters can provide the trigger mechanism for a contract or rental agreement to alert the service provider that an upcoming event such as a preventive maintenance or overhaul is coming due, to automatically adjust a price formula or rate table change, or even signal a contract renewal date or expiration date.

Other types of service counters can be set up to track the activities of a customer support contract to monitor the total number of calls or time spent by the support agent on each call or to escalate those calls that have reached a critical status.

The Counters application supports the following Service applications:

- Customer Care
- Install Base
- Service Contracts
- Depot Repair
- Field Service
- Mobile Field Service
- Spares Management
- Logistics
- Advanced Service Online

The Service applications can create a unique set of counters and templates that are required to satisfy their individual business needs for each product or service residing in their Install Base. Updates must be performed regularly using a standard set of business practices incorporated by each service provider.

### 2.1.1 New Features in Counters for This Release

The Counters application is fully integrated with the Oracle e-Business Suite and provides the following new features in this release:

***Table 2–1 New Features in Counters***

| Product Feature | Functionality |
|---|---|
| Counters HTML window | Display and capture counter readings through *i*Support. A profile option and APIs enable this access. |

*Table 2–1 New Features in Counters*

| Product Feature | Functionality |
|---|---|
| Calculate a delta value | Calculate delta values to display net usage |
| Set up counter property types and values for property LOVs | Link counter readings to a source reader |
| Retire counters | Program to retire counters that have past their expiration date |
| Delete counters | APIs to delete a complete counter instance |
| Use a templates to set up counters | Manual instantiation of counters created from a template |

## 2.1.2 Definitions

Here are definitions of some basic terms used in Counters.

*Table 2–2*

| Feature Name | Function |
|---|---|
| Physical Counter | An incremental electro-mechanical or software device built into a product to track equipment usage. These counters are tangible and can rollover to a specified value, be reset to any value, or replaced if broken during normal operation. |
| Formula Counter | An intangible or derived counter that runs inside a software application program, keeping track of usage of a product or service. A common logical counter is the number of service telephone calls made by a customer against a particular product or service. Logical counters can be added or combined with other counters to form a new counter. |
| Counter Template | A template that is defined for groups of counters and instantiated in Install Base customer products or contract lines. A template can be defined for a single counter or a group containing one or many counters. |

*Table 2–2*

| Feature Name | Function |
|---|---|
| Counter Group | Can be associated with a product or service and is permitted to have only one associated counter group. Group templates can also be applied with multiple Install Base products or contract lines. After a counter group is instantiated, it can be modified for a single instance or for all instances.<br><br>A single counter group can be defined for each Install Base customer product or contract line to automatically instantiate a counter, or be manually instantiated by selection from a predefined counter or counter group template. |

## 2.1.3 Typical Tasks Associated with Counters

### Capture Counter Readings

Counter readings can be entered manually using the Capture Counters window. This window is available from Service Request, Depot Repair Orders, Field Service, Service Contracts, and Contracts Authoring windows.

### Capture Miscellaneous Reading Adjustments

Miscellaneous reading adjustments can also be captured for counters. Miscellaneous reading types can be defined using the Misc. Reading Types LOV Lookups window to describe any kind of adjustment. For example, it can be a service counter used to adjust a technician's copy usage during a PM Service or repair of a customer's photocopier machine.

### View the Counter History Log

The Counters application creates a new instance each time a reading event has been performed and saves the results in the counter history log. The log contains a sequential list of before and after counter captures with time and date stamp, and, if selected, the property source reference id. To view the log, launch the Capture window and then click the View History Reading button.

### Install Base Life Cycle Tracking

Counters uses Install Base to manage and track reading history related to an item, service, or contract throughout the active lifecycle. If a transfer of ownership occurs, the reading history appears with the record.

### Reset Counters

Certain service providers require that a counter reset is performed to re-establish a new starting point for a product or service. A counter reset can be performed from the Counter Capture window, and each reset event will be recorded in the history file.

### Modify Counters

Counter groups and counters can be modified after instantiation with a product or service. To modify a counter instance, navigate to the Counters Setup window from the application Navigator window. See Using Counters for more information.

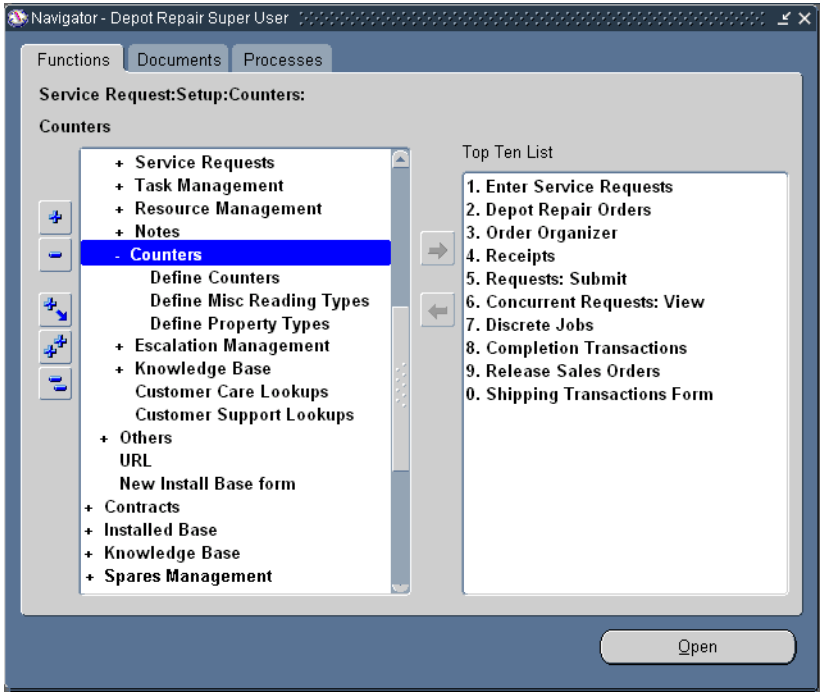
## 2.2 The Basics of Counters

### 2.2.1 Accessing Counters

You can access the Counters windows as follows:

1. Using the Forms mode, log in with the Customer Support or Depot Repair Super User responsibility.
2. Navigate to Counters.

***Figure 2–1 Navigator Window***




The following Counters setup windows are accessible from the Navigator:

***Table 2–3 Counters Functions from the Navigator***

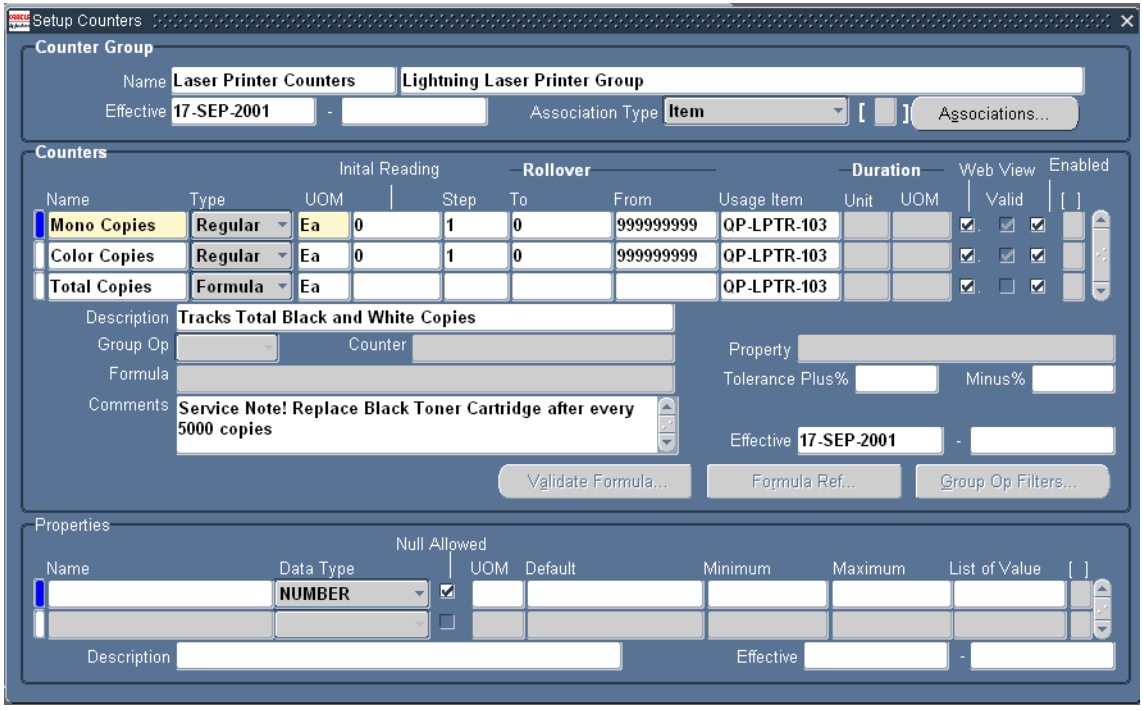
| Navigator Item | Functional Description |
|---|---|
| Define Counters | View and define counter group templates and counters. |
| Define Misc. Reading Types | View and define miscellaneous reading types for LOV lookups. |
| Define Property Types | View and define and counter property types and values for LOV lookups. |

You access the main window of Counters, Setup Counters, by choosing Define Counters from the Navigator.

## 2.2.2 The Main Window of Counters

Setup Counters is the main window of Counters.

***Figure 2–2 Setup Counters Window***




The following table provides a description of the Setup Counters window:

***Table 2–4 Description of the Setup Counters Window***

| Feature | Function |
|---|---|
| Counter Group Region | Defines the counter template used for an associated product, service, or service contract line. |
| Association Type LOV | Defines if the counter group is associated with a BOM (Bill of Material), item, or service. |
| Associations Button | Displays an Associations window to link the specified counter group to a product, service, or usage item. |

*Table 2–4 Description of the Setup Counters Window*

| Feature | Function |
|---|---|
| Counters Region | Defines the individual counters contained in the counter group. |
| Counters Types | Displays an LOV drop down list of counter types of Regular, Group Function, Formula, or Time Based. |
| Validate Formula Button | Validates the entered formula as a valid SQL expression for a formula counter. |
| Formula Ref. Button | Displays a window to allow definition of the references used in the formula and to indicate if a reading type uses a current or delta reading. |
| Properties Region | Defines additional counter reading properties for regular counters that are data types of Number, Date, or Character. |

### 2.2.3 Explanations and Defaults

The following table contains explanations and defaults used in Charges.

*Table 2–5 Explanations and Defaults*

| | |
|---|---|
| Usage items for a counter | The list of values for Usage Item is based on master items that have been identified with the value Usage Items. This flag is set in the Master Items window on the Service tabbed page for serviceable items. See Implementing Inventory for more information. |
| Use of SUM and COUNT in formula counters. | SUM totals all reading history for the counter and provides an accumulated total of the counter readings. COUNT counts the total number of readings that have occurred prior to resetting the counter. For example, take a counter that had three readings and was reset after each reading. If the three readings are 100, 100, and 100, then SUM returns 300, and COUNT returns 3. |

## 2.3 Prerequisites

As prerequisites to the setup of Counters, perform the following:

- Confirm the setup of all applications at your site that will use Counters.
- Confirm the setup of inventory items. Refer to the *Oracle Inventory User's Guide*.

- Confirm the setup of unit of measure. Refer to the *Oracle Inventory User's Guide.*

## 2.4 Setup Checklist

1. Required. Define counter group templates. Refer to Section 2.5.
2. Required. Confirm the setup of user-definable counter property lookups. Refer to Section 2.7.
3. Optional. Define regular counters. Refer to Section 2.8.
4. Optional. Define group function counters. Refer to Section 2.9.
5. Optional. Define formula counters. Refer to Section 2.10.
6. Optional. Define time-based counters Refer to Section 2.11.
7. Required. Confirm the setup of the concurrent program for the time-based Counters engine. Refer to Section 2.12
8. Required. Define profile options. Refer to Section 2.13.
9. Required. Instantiate Counters. Refer to Section 2.14

## 2.5 Defining a Counter Group

Use this procedure to define a counter group.

### Steps

1. Enter a Counter Group Name and Description.
2. Select a Start Date from the Calendar LOV to activate the counter group.
3. Select an end Date from the Calendar LOV or leave it blank for no expiration.
4. Select an Association Type from LOV to associate the Counter Group with a type.

   Seeded types are BOM (Bill of Material), Item, and Service. Association Types - BOM and Item are Master Items that have been identified as Serviceable Products and share the same list of values, whereas Service is based on Master Items that are identified as Support Services. See Implementing Inventory for more information.
5. Click Associations to launch the window.
6. Select an item from the LOV drop-down list.

7. Enter a Description.
8. Click OK to return to the Setup window.
9. From the toolbar, click the Save icon to update and save the counter group.

## 2.6 Understanding Property Type and Values LOV Lookups

Counter property types and values LOV lookups are identifiers that link the source or person performing a counter reading update. As part of the capture process, the reader enters the new counter reading and appropriate property value to link his ID to the new capture. See Defining Counter Property LOV Types for defining these types and their values in the Counters application.

### Business Case

A service provider sets up two (2) property types and values sets to identify his authorized service employees and customers performing counter updates on his service products in the field.

### Example 1: Property Type and Value Codes for Seattle Mfg. Authorized Employees

In this example the service provider sets up a property type lookup table containing values for all service employees that are authorized to perform a counter reading.

Product: ABC Copier

Counter: Total Copies

Capture: 2765

Reader: Adams, Mr. Brian

Property Code: CS_CTR1

Description: Authorized Employees

Application: Oracle Service

***Table 2–6 Value Set for the CS_CTR1 Property Code***

| Value Code | Meaning | Description |
|---|---|---|
| Depot_Tech. | Authorized Reader | Seattle Mfg |
| Depot_Mgr. | Authorized Reader | Seattle Mfg |
| Support_Tech. | Authorized Reader | Seattle Mfg |

*Table 2–6 Value Set for the CS_CTR1 Property Code*

| Value Code | Meaning | Description |
|---|---|---|
| F/S_Eng. | Authorized Reader | Seattle Mfg |

### Example 2: Property Type/ Value Codes for Business World Authorized Employees

In this example the service provider sets up a property type lookup table containing values for Customer: Business World, and list of employees authorized to perform a counter reading.

Product: XYZ Printer

Counter: Total Copies

Capture: 1743

Reader: Beaulie, Mr. Andre

Property Code: CS_CTR2

Description: Authorized Customers

Application: Oracle Service

*Table 2–7 Value Set for the CS_CTR2 Property Code*

| Value Code | Meaning | Description |
|---|---|---|
| Plant_Mgr. | Authorized Reader | Business World |
| Office_Mgr. | Authorized Reader | Business World |
| Maint_Mgr. | Authorized Reader | Business World |
| Maint_Tech. | Authorized Reader | Business World |

## 2.7 Defining Counter Property LOV Types

Counters provides a setup window called CS Counter Property LOV Types Lookups to define a standard set of values that are used when setting up counters. Property types and values are used to associate and identify the source of a counter capture. In order to use this feature, you must first define property types and values for property LOV lookups. Use this procedure to do so.

## Prerequisites

None.

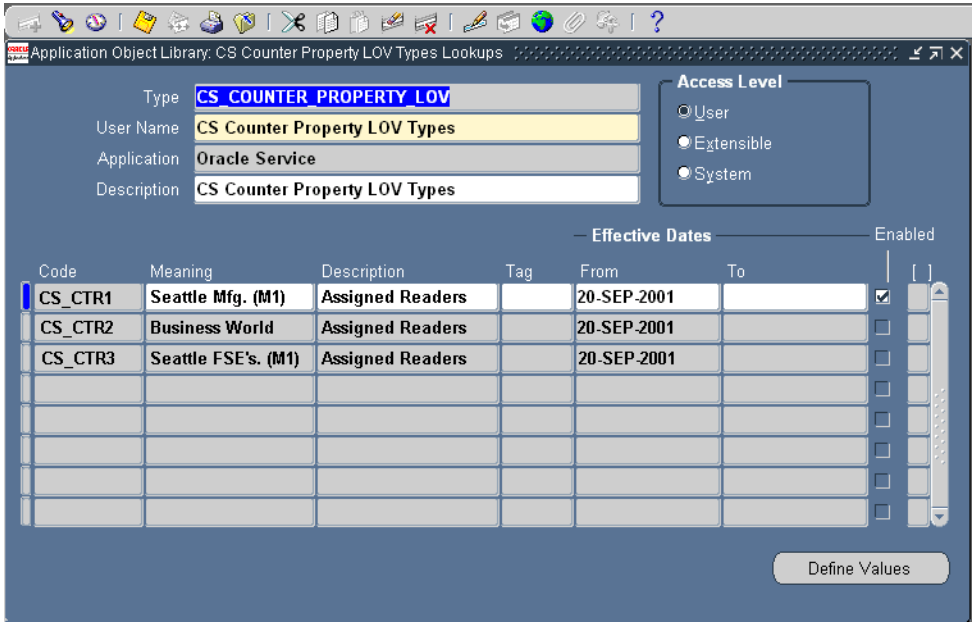
## Steps

### Define New Property Types

1. Navigate Service Request > Setup > Counters > Define Property Types.

   The CS Counter Property LOV Types Lookups window appears with seeded header detail.

***Figure 2–3 Window for CS Counter Property LOV Types Lookups***




2. Define a Property Type by completing the following:
   a. Enter a Code such as `CS_CTR1`.
   b. Enter a Meaning such as `Seattle Mfg. (M1)`.
   c. Enter a Description such as `Assigned Readers`.
   d. Leave the Tag field blank.

e. Select a from date. The application automatically supplies one .

f. Select a to (end) date. Leave this blank if no expiration occurs.

g. Confirm that the Enabled checkbox is selected.

3. Repeat Step 2a through 2g for each new property type code that you wish to add. Choose a different meaning for each code to avoid any duplication errors.

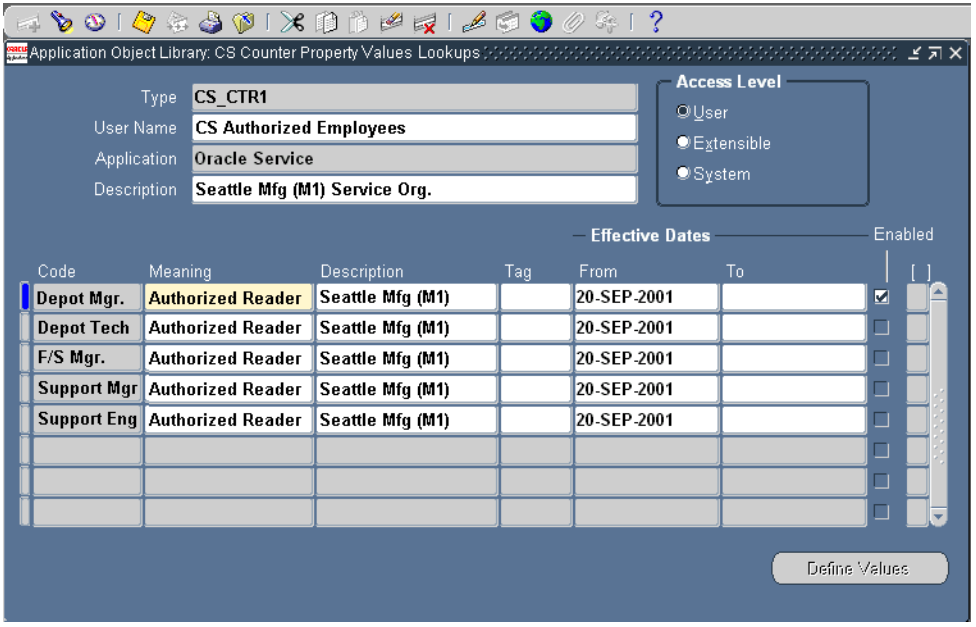
4. From the toolbar menu, click the Save icon to update and save the record.

5. Set focus on the code line for which you want to create a values table and proceed to the next step.

### Define Values for Property Types

6. Click Define Values to launch the CS Counter Property Values Lookups window with the header details auto-populated into the window.

***Figure 2–4 Window for CS Counter Property Values Lookups***




7. Define a value by completing the following:

a. Enter a Code such as `Depot Mgr`.

b. Enter a Meaning such as `Authorized Reader.`

c. Enter a Description such as `Seattle Mfg. (M1).`

d. Leave the Tag field blank.

e. Select a from date. The application automatically supplies one.

f. Select a to (end) date. Leave this blank if no expiration occurs.

g. Confirm that the Enabled checkbox is selected.

8. Repeat steps 7a through 7g for each new code value that you wish to add to the table. Choose a different meaning for each code to avoid any duplication errors.

9. From the toolbar menu, click the Save icon to update and save the record.

   The newly created values table is now linked to the Property Type Code line.

10. Click the X located in the top right-hand corner of the window to close it and return to the property types setup window.

#### Defining Additional Value Sets for Property Types

11. To create another values table, set focus on the next Code line located on the CS Counter Property LOV Types Lookup window and repeat steps 6 through 10.

## 2.8 Defining a Regular Counter (Physical)

Use this procedure to define counter types that are described as physical or regular counters, which typically are found in tangible objects such as automobiles, gas meters, and photocopier machines. A regular counter can also be classified as a logical counter. For example, consider a service agent who wants to track the number of support calls that he receives each day. The agent can set up a logical counter of type Regular and manually increment this counter at the completion of every support call.

### Prerequisites

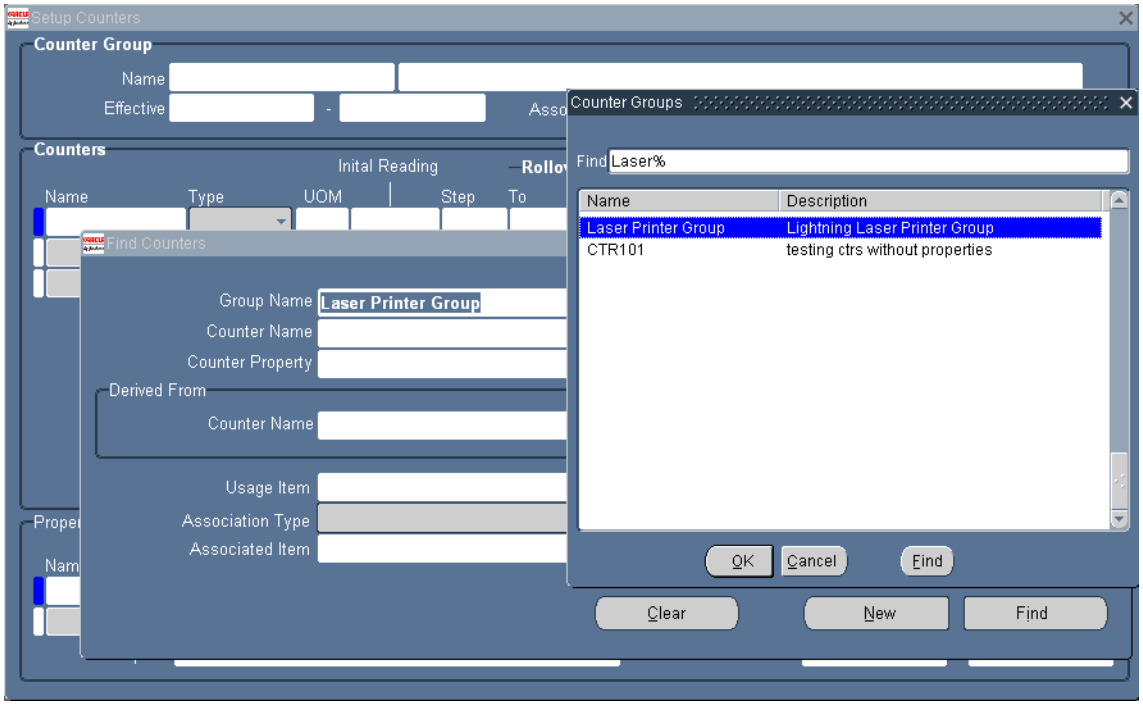
None.

### Steps

1. From the Navigator Menu list, double-click Service Request.
2. Double-click Setup from the Service Request submenu list.
3. Double-click Counters from the Setup submenu list.

4. Click Define Counters to launch the Find Counters window, and then click New.

***Figure 2–5 Find Counters Window***




5. Complete the Counter Group information.
6. Define the Counter Name and select Regular for Type.
7. Select the Unit of Measure (UOM) = Ea.
8. Enter the initial reading of the counter.

   For example, a new automobile may have an initial reading on the odometer of 000022 miles accrued during testing and delivery.
9. Enter the counter step number.

   For example, if you want the counter to be incremented by one for each reading, then enter 1 for the Step.

10. Enter the rollover to and from counter numbers.

    For example, a counter on an automobile may start at 000000 and count up to 999999 before it rolls over to 000000 or if reset, starts with the Rollover To count of 000000.

11. Select a Usage Item from the LOV.

12. Confirm that the Web View checkbox is selected allow the counter to be viewed from the its HTML page from the *i*Support window.

13. Confirm that the Enabled checkbox is selected to enable counter updates.

14. Identify a counter description or comments in the provided fields.

15. Select an effective start and end date from the Calendar LOV. Do not select an end date if no expiration is required.

16. Identify a tolerance percentage, both plus and minus.

    For example, if a counter is used by Service Contracts, determine a range in which the counter needs to be updated.

17. In the lower region of Properties window, click the List of Values LOV button to display the drop-down list. Then select a property type to link to this counter.

    For example, you can select a property type called Assigned Counter Resources, which contains a list of resource names assigned to perform monthly counter updates.

18. Click the Default LOV button to display the LOV list, and then select a value that you wish to use as a default in the Counter Capture window.

19. Type the name assigned to the counter property.

20. Select a Data Type of CHAR (Character).

21. Confirm that the Null Allowed checkbox is selected if a null condition is allowed.

22. Click UOM (Unit of Measure) LOV button to display a list. Select Ea.

23. Optionally, enter a description for the property type being defined.

24. Enter an effective start and end date to enable property type setup. Do not enter an end date if no expiration is required.

25. From the toolbar menu, click the Save icon to update and save the record.

## 2.9 Defining a Group Function Counter

Use this procedure to define counters of the type Group Function. With this type you can derive counters using SUM and COUNT. Group function counters can be used in formula counters.

### Explanation of SUM and COUNTERS

SUM totals all reading history for a counter and provides an accumulated total of the counter readings. COUNT counts the total number of readings that have occurred prior to resetting a counter. For example, consider a counter that had three readings and was reset after each reading. If the three readings are 100, 100, 100, then SUM returns 300, and COUNT returns 3.

### Prerequisites

None.

### Steps

1. From the Navigator, double-click Service Request.
2. Double-click Setup from the Service Request submenu.
3. Double-click Counters from the Setup submenu.
4. Click Define Counters to launch the Counters Find window. Then click New to launch the Counters Setup window.
5. Complete the information in the Counter Group region.
6. Define the counter name and select Group Function for Type.
7. Select a UOM (Unit of Measure).Example: Ea. (Each)
8. Select the usage item for this counter.
9. Identify counter description and effective dates.
10. Select the Group Operation of either SUM or COUNT.
11. Identify the Tolerance Plus and Minus % for the counter.

    Tolerance is used by Oracle Service Contracts Events to determine the range within which these counters must be updated.
12. Enter any additional comments desired.
13. Click Group Operation Filter to launch the Counter Filter window.

14. Complete the counter filter information for the properties.
15. From the toolbar menu, click the Save icon to update and save the record.

## 2.10 Defining a Formula Counter

Use this procedure to define a counter of the type Formula. With formula counters you can use simple or complex mathematical formulas to derive a new counter value. This procedure is stated in the context of a specific business case and example steps that include regular counters.

### Business Case

A service provider sells and leases laser printers and wants to track the copy usage for each printer residing in his Install Base. The service provider creates a counter group with individual counters to track the usage for each product to determine how often to perform his PM service tasks, and also to determine how often to ship out consumables such as toner kits and paper.

### Example Task

Create a counter group that contains three counters to track usage for these printers. One of these counters is a formula counter. Complete the following steps:

#### High-Level Steps

1. Create a counter group with the name Laser Printer Group.
2. Define a regular counter with the name MonoCopies.
3. Define a regular counter with the name ColorCopies.
4. Define a formula counter with the name TotalCopies.

### Prerequisites

None.

### Step

1. From the Navigator Menu, double-click Service Request.
2. Double-click Setup from the Service Request submenu list.
3. Double-click Counters from the Setup submenu list.

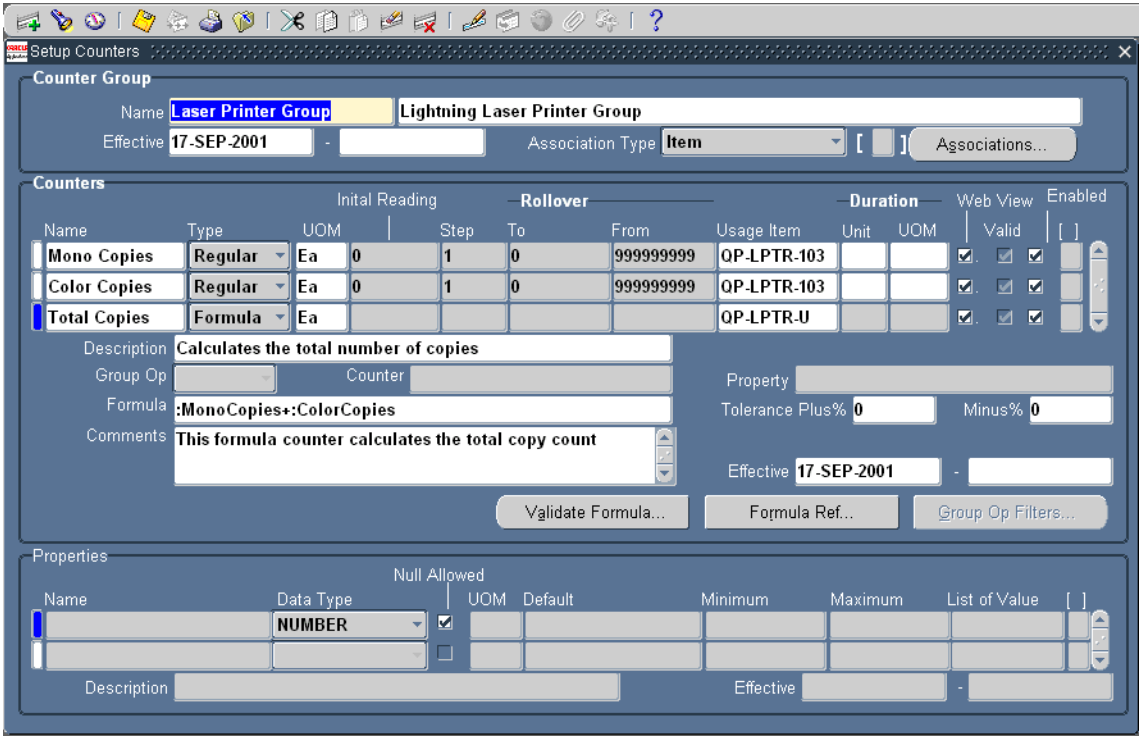
4. Click Define Counters to launch the Counters Find window, and then click New.
5. Complete the Counter Group information described in the example.
6. Create two regular counters, one called MonoCopies, and one called ColorCopies. See Defining a Regular Counter for more information.
7. Create a third counter called TotalCopies, and then select Counter Type > Formula.
8. Select `Ea` for the UOM (Unit of Measure).
9. Select the usage item for this counter.
10. Identify an additional counter description and select the start and end dates from the LOV Calendar. Leave the End Date field blank if no expiration is required.
11. Enter the formula that you wish to use for the counter. The syntax is any valid expression in standard SQL. Use your own formula reference names preceded by the ':' symbol.

    To create a formula that adds MonoCopies to Color Copies to derive a new total called TotalCopies, enter the formula as follows:

    ```
    :MonoCopies+:ColorCopies
    ```

    You must enter a colon [:] in front of a counter name with no spaces between any of the names and math symbols

***Figure 2–6 Setup Counters Window for a Formula Counter***




12. Click Validate Formula to validate the newly entered formula. If the formula is correct, the following message appears: `Formula is Valid`.
13. In the event of an error, check that each regular counter name used in the formula is spelled correctly, a colon [:] is displayed before each counter name, and all spaces are eliminated in the formula statement.
14. Enter the % Plus and Minus Tolerance. For example: Plus%: 10, Minus%: 10. Tolerance is used by Oracle Service Contracts Events to determine the range in which the counter should be updated.
15. Enter any desired comments into the Comment field.
16. Click Formula Reference to launch the Formula Reference window.
17. Select a reading type of Current or Delta for the regular counter being defined in the formula counter.

18. Optionally, select a counter name from the LOV to map to the formula reference name for each of the counters.
19. Optionally, select an item from the LOV to map the formula reference name for each of the Counters.
20. Click OK to update and close the Formula Reference window.
21. From the toolbar, click the Save icon to update and save the record.

## 2.11 Defining a Time-Based Counter

A time-based counter can be defined in hours, days, weeks, months, or years to keep track of events that occur in time. Time-based counters can be used to track such events as a contract expiration or preventive maintenance service that is due on a customer product. Time-based counters must be updated periodically by running a concurrent program called Time Based Counters Engine to increment each counter and automatically expire a counter that has past its expiration date.

### Prerequisites

None

### Steps

1. From the Navigator, double-click Service Request.
2. Double-click Setup from the Service Request submenu.
3. Double-click Counters from the Setup submenu.
4. Click Define Counters to launch the Counters Find window. Then click New to launch the Counters Setup window.
5. Complete the information in the Counter Group region.
6. Create a counter of type Time Based.
7. Select the unit of measure (UOM). The selected UOM should be for a measure of a unit of time such as Hours, Days, and Weeks.
8. Select the Usage Item for this counter.
9. Identify counter description, effective dates, and comments. there are no properties for time-based counters.
10. From the toolbar menu, click the Save icon to update and save the counter record.

## 2.12 Setting the Time Based Counters Engine

The Time Based Counters Engine concurrent program must be set to run periodically to advance and update all active time-based counters and to expire those counters that have expired their effective end dates.

## 2.13 Setting Profile Options

The only profile option for Counters is CS Counter Webview Allowed? and has a value of Yes or No. Set the option to Yes if you wish to enable counter views from *i*Support.

## 2.14 Instantiating Counters from a Template

The Counters application permits manual instantiation of counters for an Install Base product, service, or service contract line using APIs to execute the manual instantiation process. For more information about this topic, consult the guides for the application using the Counters application. Use this procedure to instantiate a counter from a template.

### Prerequisites

Confirm setup of Counter Groups/ Counter templates before attempting to use this functionality.

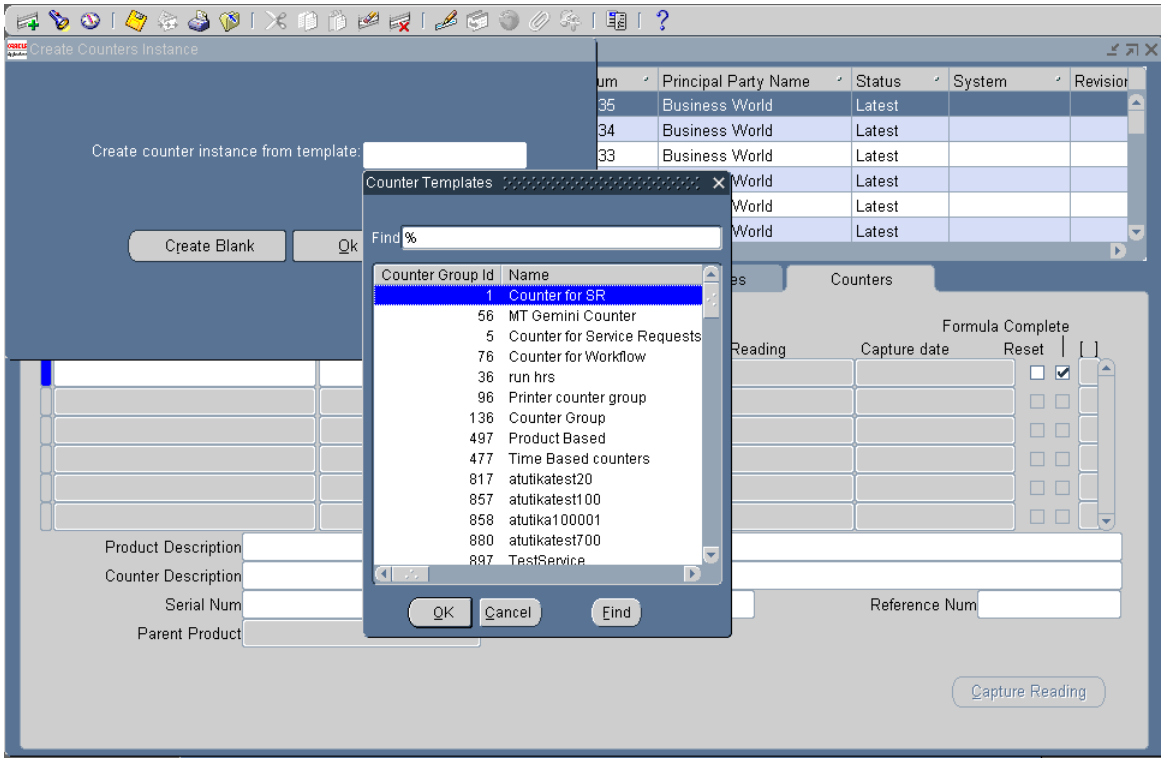
### Steps

1. Navigate Install Base > View Customer Products.

   The Find Products window appears.

2. Enter the customer's account number and/or product.
3. Click Find to start the query and launch the Install Base Product Summary/ Details window.
4. Select a product from the Products table by focusing the mouse pointer on the item that you want to display on the tabbed pages.
5. Click the Counters tab to display its page.

***Figure 2–7 Windows from the Install Base Product Summary Window***




6. From the toolbar, choose Tools > Counters Setup.
7. Click Counters Setup to launch the Create Counters Instance window.
8. Click Create Counter Instance from the Template LOV button to display the Counter Templates find window.
9. Click the Counter Group ID or Name to select an entry, and click OK to populate the selection.
10. Click OK on the Create Counters Instance window to start the query.

    The Setup Counters window launches with the template details populated.
11. Confirm setup details and effective start date to activate the counter group and counters being used for the selected product or service.

12. From the toolbar, click the Save icon to update and save the record.
13. Close the Setup Counters window by clicking on the X located in the top right hand corner of the window to return to the Install Base Product and Summary window.
14. Confirm that counters were instantiated for the selected Install Base product or service.

# A

# Charges Public APIs

The Charges module is the central location for billing within the Service modules. It is used by Support, Depot Repair, and Field Service Report. These modules create Charge lines in the Charges schema. The user can then navigate to the Charges form, add pricing information such as price list and contract discounts, and then submit the line to Order Management for shipping and/or invoicing.

## A.1 Charges Public Packages

The APIs described below allow custom code to create, update, and delete Charge lines which then can be processed using the existing Charges Business flow.

CS_Charge_Details_PUB, see Table 4-1, "Charges APIs".

The following table describes the public APIs discussed in this chapter.

***Table A–1 Charges APIs***

| Name | Description |
|---|---|
| Create Charge Details | This procedure creates a charge line. |
| Update Charge Details | This procedure updates a charge line that has not been submitted to Order Management. |
| Delete Charge Details | This procedure deletes a charge line that has not been submitted to Order Management. |

## A.2 Package CS_Charge_Details_PUB

This package contains all of the Charges public APIs.

- Create_Charge_Details

- Update_Charge_Details
- Delete_Charge_Details

## A.2.1 Create_Charge_Details

This procedure allows the caller to create Charge lines for a given Service Request.

### Procedure Specification

```
PROCEDURE Create_Charge_Details(
        p_api_version            IN   NUMBER,
        p_init_msg_list          IN   VARCHAR2 := FND_API.G_FALSE,
        p_commit                 IN   VARCHAR2 := FND_API.G_FALSE,
        p_validation_level       IN   NUMBER   :=
                                        FND_API.G_VALID_LEVEL_FULL,
        x_return_status         OUT   VARCHAR2,
        x_msg_count             OUT   NUMBER,
        x_object_version_number OUT   NUMBER,
        x_msg_data              OUT   VARCHAR2,
        x_estimate_detail_id    OUT   NUMBER,
        x_line_number           OUT   NUMBER,
        p_resp_appl_id           IN   NUMBER   := NULL,
        p_resp_id                IN   NUMBER   := NULL,
        p_user_id                IN   NUMBER   := NULL,
        p_login_id               IN   NUMBER   :=
                                        FND_API.G_MISS_NUM,
        p_transaction_control    IN   VARCHAR2 := FND_API.G_TRUE,
        p_Charges_Rec            IN   Charges_Rec_Type
        ) ;
```

### Current Version

```
110.0
```

### Parameter Descriptions

The following table describes the IN parameters associated with this API.

***Table A–2 Create_Charge_Details IN Parameters***

| Parameter | Data Type | Required | Descriptions and Validations |
|---|---|---|---|
| p_api_version | NUMBER | Yes | Parameter for the version of API. If this does not match the version of the API being called, it will return an unexpected error. |
| p_init_msg_list | VARCHAR2 | Yes | FND_API.G_FALSE or FND_API.G_TRUE. True indicates that the API will initialize the message stack. If false, the calling program must initialize the message stack using FND_MSG_PUB.initialize. This is used in the case of multiple API calls. |
| p_commit | VARCHAR2 | Yes | Indicates if the API should commit its results. |
| p_validation_level | NUMBER | | Parameter for controlling the validation level in the API. Values are FND_API.G_ VALID_LEVEL_FULL and FND_API.G_ VALID_LEVEL_NONE. |
| p_resp_appl_id | NUMBER | | ID used to identify the Application. |
| p_resp_id | NUMBER | | ID used to identify the Responsibility. |
| p_user_id | NUMBER | | ID used to identify the Application User. If this is not passed, it will assume -1 (anonymous user). |
| p_login_id | NUMBER | | ID used to identify the Login Session ID. |
| p_transaction_control | VARCHAR2 | | Used by API to do transaction Control. This should be set to 'F' (False) if API is called from triggers, as triggers do not allow Transaction Control in procedures. Otherwise it should always be 'T' (True). |
| p_Charges_Rec | Charges_Rec_ Type | Yes | Used to pass details of the Charges row that you wish to create. |

The following table describes the OUT parameters associated with this API.

***Table A–3 Create_Charge_Details Out Parameters***

| Parameter | Data Type | Descriptions |
|---|---|---|
| x_return_status | VARCHAR2 | Return Status of API. The values are:<br>FND_API.RET_STS_SUCCESS is Success<br>FND_API.RET_STS_ERROR is Expected Error<br>FND_API.RET_STS_UNEXP_ERROR is Unexpected Error |
| x_msg_count | NUMBER | Number of messages returned by the API. |
| x_object_version_number | NUMBER | Used for row locking in HTML applications. |
| x_msg_data | VARCHAR2 | Contains the msg_data generated by API. |
| x_estimate_detail_id | NUMBER | The unique Identifier for the line created by the API |
| x_line_number | NUMBER | The line number created by the API. |

## A.2.2 Update_Charge_Details

This procedure allows the caller to create Update existing Charge lines for a given Service Request.

### Procedure Specification

```
PROCEDURE Update_Charge_Details(
       p_api_version            IN   NUMBER,
       p_init_msg_list          IN   VARCHAR2 := FND_API.G_FALSE,
       p_commit                 IN   VARCHAR2 := FND_API.G_FALSE,
       p_validation_level       IN   NUMBER   :=
                                       FND_API.G_VALID_LEVEL_FULL,
       x_return_status         OUT   VARCHAR2,
       x_msg_count             OUT   NUMBER,
       x_object_version_number OUT   NUMBER,
       x_msg_data              OUT   VARCHAR2,
       p_resp_appl_id           IN   NUMBER   := NULL,
       p_resp_id                IN   NUMBER   := NULL,
       p_user_id                IN   NUMBER   := NULL,
       p_login_id               IN   NUMBER   :=
                                       FND_API.G_MISS_NUM,
       p_transaction_control    IN   VARCHAR2 := FND_API.G_TRUE,
       p_Charges_Rec            IN   Charges_Rec_Type
       ) ;
```

### Current Version

110.0

### Parameter Descriptions

The following table describes the IN parameters associated with this API.

***Table A–4 Update_Charge_Details IN Parameters***

| Parameter | Data Type | Required | Descriptions and Validations |
|---|---|---|---|
| p_api_version | NUMBER | Yes | Parameter for the version of API. If this does not match the version of the API being called, it will return an unexpected error. |
| p_init_msg_list | VARCHAR2 | Yes | FND_API.G_FALSE or FND_API.G_TRUE. True indicates that the API will initialize the message stack. If false, the calling program must initialize the message stack using FND_MSG_PUB.initialize. This would be used in the case of multiple API calls. |
| p_commit | VARCHAR2 | Yes | Indicates if the API should commit its results. |
| p_validation_level | NUMBER | | Parameter for controlling the validation level in the API. Values are FND_API.G_VALID_LEVEL_FULL and FND_API.G_VALID_LEVEL_NONE. |
| p_resp_appl_id | NUMBER | | ID used to identify the Application. |
| p_resp_id | NUMBER | | ID used to identify the Responsibility. |
| p_user_id | NUMBER | | ID used to identify the Application User. If this is not passed, it will assume -1 (anonymous user). |

***Table A–4 Update_Charge_Details IN Parameters***

| Parameter | Data Type | Required | Descriptions and Validations |
|---|---|---|---|
| p_login_id | NUMBER | | ID used to identify the Login Session ID. |
| p_transaction_control | VARCHAR2 | | Used by API to do transaction Control. This should be set to 'F' (False) if API is called from triggers, as triggers do not allow Transaction Control in procedures. Otherwise it should always be 'T' (True). |
| p_Charges_Rec | Charges_Rec_ Type | Yes | Used to pass details of the Charges row that you wish to create. |

The following table describes the OUT parameters associated with this API.

***Table A–5 Update_Charge_Details Out Parameters***

| Parameter | Data Type | Descriptions |
|---|---|---|
| x_return_status | VARCHAR2 | Return Status of API. The values are:<br>FND_API.RET_STS_SUCCESS is Success<br>FND_API.RET_STS_ERROR is Expected Error<br>FND_API.RET_STS_UNEXP_ERROR is Unexpected Error |
| x_msg_count | NUMBER | Number of messages returned by the API. |
| x_object_version_ number | NUMBER | Used for row locking in HTML applications. |
| x_msg_data | VARCHAR2 | Contains the msg_data generated by API. |

## A.2.3 Delete_Charge_Details

This procedure allows the caller to delete Charge lines for a given Service Request.

### Procedure Specification

```
PROCEDURE Delete_Charge_Details(
        p_api_version            IN   NUMBER,
        p_init_msg_list          IN   VARCHAR2 := FND_API.G_FALSE,
        p_commit                 IN   VARCHAR2 := FND_API.G_FALSE,
        p_validation_level       IN   NUMBER   :=
                                        FND_API.G_VALID_LEVEL_FULL,
```

```
x_return_status          OUT   VARCHAR2,
x_msg_count              OUT   NUMBER,
x_msg_data               OUT   VARCHAR2,
p_transaction_control     IN   VARCHAR2 := FND_API.G_TRUE,
p_estimate_detail_id      IN   NUMBER   := NULL
) ;
```

### Current Version

110.0

### Parameter Descriptions

The following table describes the IN parameters associated with this API.

***Table A–6 Delete_Charge_Details IN Parameters***

| Parameter | Data Type | Required | Descriptions and Validations |
|---|---|---|---|
| p_api_version | NUMBER | Yes | Parameter for the version of API. If this does not match the version of the API being called, it will return an unexpected error. |
| p_init_msg_list | VARCHAR2 | Yes | FND_API.G_FALSE or FND_API.G_TRUE. True indicates that the API will initialize the message stack. If false, the calling program must initialize the message stack using FND_MSG_PUB.initialize. This would be used in the case of multiple API calls. |
| p_commit | VARCHAR2 | Yes | Indicates if the API should commit its results. |
| p_validation_level | NUMBER | | Parameter for controlling the validation level in the API. Values are FND_API.G_VALID_LEVEL_FULL and FND_API.G_VALID_LEVEL_NONE. |
| p_transaction_control | VARCHAR2 | | Used by API to do transaction Control. This should be set to 'F' (False) if API is called from triggers, as triggers do not allow Transaction Control in procedures. Otherwise it should always be 'T' (True). |
| p_estimate_detail_id | NUMBER | Yes | Used to pass detail ID of the Charges row to be deleted. |

The following table describes the OUT parameters associated with this API.

***Table A–7 Delete_Charge_Details Out Parameters***

| Parameter | Data Type | Descriptions |
|---|---|---|
| x_return_status | VARCHAR2 | Return Status of API. The values are:<br>FND_API.RET_STS_SUCCESS is Success<br>FND_API.RET_STS_ERROR is Expected Error<br>FND_API.RET_STS_UNEXP_ERROR is Unexpected Error |
| x_msg_count | NUMBER | Number of messages returned by the API. |
| x_msg_data | VARCHAR2 | Contains the msg_data generated by API. |

# A.3 Data Structure Specifications

The following data structure is used in CS_Charge_Details_PUB:

- Charges_Rec_Type

## A.3.1 Charges_Rec_Type

This data structure is used to pass the details of the Charges line to be created or updated to the Charges API.

### Record Specification

```
TYPE Charges_Rec_Type IS RECORD
(
 estimate_detail_id        NUMBER        := FND_API.G_MISS_NUM,
 original_source_number    VARCHAR2(200) := FND_API.G_MISS_CHAR,
 purchase_order_num        VARCHAR2(50)  := FND_API.G_MISS_CHAR,
 price_list_id             NUMBER        := FND_API.G_MISS_NUM,
 original_source_id        NUMBER        := FND_API.G_MISS_NUM,
 original_source_code      VARCHAR2(10)  := FND_API.G_MISS_CHAR,
 source_number             VARCHAR2(200) := FND_API.G_MISS_CHAR,
 source_id                 NUMBER        := FND_API.G_MISS_NUM,
 source_code               VARCHAR2(10)  := FND_API.G_MISS_CHAR,
 txn_billing_type_id       NUMBER        := FND_API.G_MISS_NUM,
 incident_id               NUMBER        := FND_API.G_MISS_NUM,
 business_process_id       NUMBER        := FND_API.G_MISS_NUM,
 contract_id               NUMBER        := FND_API.G_MISS_NUM,
 coverage_id               NUMBER        := FND_API.G_MISS_NUM,
 coverage_txn_group_id     NUMBER        := FND_API.G_MISS_NUM,
 coverage_bill_rate_id     NUMBER        := FND_API.G_MISS_NUM,
```

```
exception_coverage_used   VARCHAR2(1)   := FND_API.G_MISS_CHAR,
currency_code             VARCHAR2(15)  := FND_API.G_MISS_CHAR,
invoice_to_org_id         NUMBER        := FND_API.G_MISS_NUM,
ship_to_org_id            NUMBER        := FND_API.G_MISS_NUM,
inventory_item_id_in      NUMBER        := FND_API.G_MISS_NUM,
quantity_required         NUMBER        := FND_API.G_MISS_NUM,
unit_of_measure_code      VARCHAR2(3)   := FND_API.G_MISS_CHAR,
after_warranty_cost       NUMBER        := FND_API.G_MISS_NUM,
customer_product_id       NUMBER        := FND_API.G_MISS_NUM,
reference_number          NUMBER        := FND_API.G_MISS_NUM,
inventory_item_id_out     NUMBER        := FND_API.G_MISS_NUM,
serial_number_out         VARCHAR2(50)  := FND_API.G_MISS_CHAR,
installed_cp_return_by_date  DATE       := FND_API.G_MISS_DATE,
new_cp_return_by_date     DATE          := FND_API.G_MISS_DATE,
order_header_id           NUMBER        := FND_API.G_MISS_NUM,
original_system_reference VARCHAR2(50)  := FND_API.G_MISS_CHAR,
interface_to_oe_flag      VARCHAR2(1)   := FND_API.G_MISS_CHAR,
no_charge_flag            VARCHAR2(1)   := FND_API.G_MISS_CHAR,
add_to_order_flag         VARCHAR2(1)   := FND_API.G_MISS_CHAR,
rollup_flag               VARCHAR2(1)   := FND_API.G_MISS_CHAR,
line_category_code        VARCHAR2(6)   := FND_API.G_MISS_CHAR,
line_type_id              NUMBER        := FND_API.G_MISS_NUM,
return_reason_code        VARCHAR2(30)  := FND_API.G_MISS_CHAR,
order_line_id             NUMBER        := FND_API.G_MISS_NUM,
selling_price             NUMBER        := FND_API.G_MISS_NUM,
serial_number             VARCHAR2(50)  := FND_API.G_MISS_CHAR,
est_tax_amount            NUMBER        := FND_API.G_MISS_NUM,
attribute1             VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute2             VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute3             VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute4             VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute5             VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute6             VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute7             VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute8             VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute9             VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute10            VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute11            VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute12            VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute13            VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute14            VARCHAR2(150)    := FND_API.G_MISS_CHAR,
attribute15            VARCHAR2(150)    := FND_API.G_MISS_CHAR,
context                VARCHAR2(30)     := FND_API.G_MISS_CHAR,
pricing_context        VARCHAR2(30)     := FND_API.G_MISS_CHAR,
pricing_attribute1     VARCHAR2(150)    := FND_API.G_MISS_CHAR,
```

```
pricing_attribute2     VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute3     VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute4     VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute5     VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute6     VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute7     VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute8     VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute9     VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute10    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute11    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute12    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute13    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute14    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute15    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute16    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute17    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute18    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute19    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute20    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute21    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute22    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute23    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute24    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute25    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute26    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute27    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute28    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute29    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute30    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute31    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute32    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute33    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute34    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute35    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute36    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute37    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute38    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute39    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute40    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute41    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute42    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute43    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute44    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute45    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute46    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
```

```
pricing_attribute47    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute48    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute49    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute50    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute51    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute52    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute53    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute54    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute55    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute56    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute57    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute58    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute59    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute60    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute61    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute62    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute63    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute64    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute65    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute66    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute67    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute68    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute69    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute70    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute71    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute72    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute73    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute74    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute75    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute76    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute77    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute78    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute79    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute80    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute81    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute82    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute83    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute84    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute85    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute86    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute87    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute88    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute89    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute90    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
pricing_attribute91    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
```

```
 pricing_attribute92    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
 pricing_attribute93    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
 pricing_attribute94    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
 pricing_attribute95    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
 pricing_attribute96    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
 pricing_attribute97    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
 pricing_attribute98    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
 pricing_attribute99    VARCHAR2(150)    := FND_API.G_MISS_CHAR,
 pricing_attribute100   VARCHAR2(150)    := FND_API.G_MISS_CHAR,
 org_id                 NUMBER           := FND_API.G_MISS_NUM,
 item_revision             VARCHAR2(3)   := FND_API.G_MISS_CHAR
) ;
```

### Parameter Descriptions

The following table describes the parameters associated with this data structure.

***Table A–8 Charges_Rec_Type Parameters***

| Parameter | Data Type | Description |
|---|---|---|
| estimate_detail_id | NUMBER | Unique identifier of a charge line. Required for calls to Update Charge Details |
| original_source_number | VARCHAR2(200) | Original source number of the charges line. For example, if charges line is created from the Service Request form, then it is incident_ number. |
| purchase_order_num | VARCHAR2(50) | Required if the Payment Type on the Order Type requires it. |
| price_list_id | NUMBER | Unique identifier of the Price List on the charge line. If not passed, then the price list will be defaulted from the contract and the 'Charge: Default Price List' profile option. |
| original_source_id | NUMBER | Original source ID of charges line. For example, if the Charges line is created from a Service Request, then it will be incident_id. |
| original_source_code | VARCHAR2(10) | Original source code of the charges line. The possible values are SR (Service Request), DR (Depot Repair), and SD (Field Service Report). |
| source_number | VARCHAR2(200) | Latest source number of the charges line. For example, if the charges line currently has a source of the Service Request form, then it is incident_number. |

***Table A–8 Charges_Rec_Type Parameters***

| Parameter | Data Type | Description |
|---|---|---|
| source_id | NUMBER | Latest source ID of the charges line. For example, if the charges line currently has a source of the Service Request form, then it is incident_id. |
| source_code | VARCHAR2(10) | Latest source code of the charges line. The possible values are SR (Service Request), DR (Depot Repair), and SD (Field Service Report). |
| txn_billing_type_id | NUMBER | The ID of the Transaction Billing Type for the charges line being created.<br>Required for Creates. Optional for Updates. |
| incident_id | NUMBER | The Incident ID of the parent Service Request for the Charge line being created.<br>Required for Creates. Not used for updates. |
| business_process_id | NUMBER | The ID of the Business Process for the charges line being created.<br>Required for Creates. Optional for Updates. |
| contract_id | NUMBER | Contract ID for the Charges line being created. Optional. If not passed on creates, the contract will default from the Service Request if the contract on the Service Request covers the Business Process passed to the API. |
| coverage_id | NUMBER | Coverage ID of the Charges line being created. |
| coverage_txn_group_id | NUMBER | Transaction Group ID of the Charges line being created. |
| coverage_bill_rate_id | NUMBER | Billing Rate ID used to calculate contract billing rate for labor lines. The is an ID of OKC_K_LINES_B. |
| exception_coverage_used | VARCHAR2(1) | Not Used. |
| currency_code | VARCHAR2(15) | Currency to be used for Billing. If not passed it will be defaulted from the Service Request. |
| invoice_to_org_id | NUMBER | Bill to site id from TCA. This is defaulted from the Service Request if not passed. |
| Ship_to_org_id | NUMBER | Ship to site id from TCA. This are defaulted from the Service Request if not passed. |
| Inventory_item_id_in | NUMBER | Inventory Item to be used on the Charge Line. |

***Table A–8 Charges_Rec_Type Parameters***

| Parameter | Data Type | Description |
|---|---|---|
| quantity_required | NUMBER | Quantity of the item to be used on the Charge line. |
| unit_of_measure_code | VARCHAR2(3) | Unit Of Measure code of the item. Optional. If this is not passed, it defaults from the primary unit of measure of the Item. It is a reference to MTL_UNITS_OF_MEASURE. |
| After_warranty_cost | NUMBER | This is the final amount of the Charge line. |
| customer_product_id | NUMBER | This is the ID of the Installed Base instance.<br>The installed base record will be identified based on the passed values of:<br>customer_product_id<br>reference_number<br>inventory_item_id_out *and* serial_number_out in that order. |
| reference_number | NUMBER | This is used to locate the Installed Base instance. See customer_product_id for the search rules. |
| inventory_item_id_out | NUMBER | Inventory Item used to uniquely select an Installed Base instance. This is used in conjunction with serial_number_out. See customer_product_id for the search rules. |
| serial_number_out | VARCHAR2(50) | Serial number used to uniquely select an Installed Base instance. This is used in conjunction with inventory_item_id_out. See customer_product_id for the search rules. |
| installed_cp_return_by_ date | DATE | not used. |
| new_cp_return_by_date | DATE | Date that the new Installed Base instance should be returned. Used in the case of loaners. |
| order_header_id | NUMBER | Order Header ID of an existing order to add this charge line too. This line will not be added until it is submitted from the Charges UI. This option is controlled by the add_to_order_flag. |
| original_system_reference | VARCHAR2(50) | Not used. |
| interface_to_oe_flag | VARCHAR2(1) | Flag to indicate whether this line should be submitted to OM when the 'Submit' button is pressed on the Charges UI. |

***Table A–8 Charges_Rec_Type Parameters***

| Parameter | Data Type | Description |
|---|---|---|
| no_charge_flag | VARCHAR2(1) | Indicates that this line should have a zero charge. |
| add_to_order_flag | VARCHAR2(1) | 'Y' or 'N'. Indicates that this charge line should be added to a specific existing order (based on order_header_id) when submitted to Order Management. |
| rollup_flag | VARCHAR2(1) | 'Y' or 'N'. Indicates that when the line is submitted to Order Management, it will be submitted with the Inventory Item based on one of the rollup Inventory Item profile options. |
| line_category_code | VARCHAR2(6) | Not used. |
| line_type_id | NUMBER | Not used. |
| return_reason_code | VARCHAR2(30) | The Return Reason code in Order Management. Required when the Transaction Type of the Charge line has a line category code of RETURN. |
| order_line_id | NUMBER | Not used. |
| selling_price | NUMBER | Not used. |
| serial_number | VARCHAR2(50) | Serial Number on the charge line. This field is for information only and has no validation. |
| est_tax_amount | NUMBER | Not used. |
| attribute1<br>attribute2<br>attribute3<br>.<br>.<br>.<br>attribute15 | VARCHAR2(150)<br>VARCHAR2(150)<br>VARCHAR2(150)<br>.<br>.<br>.<br>VARCHAR2(150) | Descriptive flex field columns on the Charges table. |
| context | VARCHAR2(30) | Context for the Charges descriptive flex field. |
| pricing_context | VARCHAR2(30) | Context for the Pricing flex field. |

*Table A–8 Charges_Rec_Type Parameters*

| Parameter | Data Type | Description |
|---|---|---|
| pricing_attribute1<br>pricing_attribute2<br>pricing_attribute3<br>.<br>.<br>.<br>pricing_attribute100 | VARCHAR2(150)<br>VARCHAR2(150)<br>VARCHAR2(150)<br>.<br>.<br>.<br>VARCHAR2(150) | Pricing attributes used for identifying the product on the price list. |
| Org_id | NUMBER | The bill from Operating Unit of the Charge line. Optional. If this is not passed, it will be defaulted. |
| Item_revision | VARCHAR2(3) | Revision code for Inventory Item. Only required on material Order lines where item is revision controlled. If not supplied it will default to null. |

# A.4 Messages and Notifications

The status messages associated with the Charges public APIs are from the following category:

Error messages

## A.4.1 CS_Charge_Details_PUB

The following table describes a lists of error messages and notifications that the Charges API can generate. Note that not all messages are returned by all APIs.

***Table A–9 API Messages***

| Number | Type | Name | Text |
|---|---|---|---|
| n/a | E | CS_CHG_CANNOT_ INSERT | Service Request Status does not allow creation of Charge lines |
| n/a | E | CS_API_ALL_MISSING_ ORG_ID | API Programming Error (&API_NAME): The parameter ORG_ID is required in a multi-org environment. |
| n/a | E | CS_API_ALL_VALUE_ TRUNCATED | API Programming Warning (&API_NAME): The parameter &TRUNCATED_PARAM was truncated because the character value (&VAL_LEN) is longer than the defined width of the VARCHAR2 column (&DB_LEN). |
| n/a | E | CS_API_ALL_CANT_ LOCK_RECORD | API Programming Error (&API_NAME): This API failed because it was called with the NOWAIT option and it cannot lock the appropriate record. |
| n/a | E | CS_API_CHG_CANT_ UPD_DET_PARAM | API Programming Error(&API_NAME): The API cannot update the Column '&COLUMN_NAME' with Value '&VALUE' as this line has been submitted to Order Entry or the items have been rolled up |
| n/a | E | CS_API_CHG_RATE_ UOM_USED | API Programming Message(&API_NAME): The Unit of Measure Code passed was ignored. The rate type defined UOM was used. |
| n/a | E | CS_API_CHG_CANT_ ROLLUP_ITEM | API Programming Error (&API_NAME): The API cannot rollup the item because the transaction type chosen does not allow it |
| n/a | E | CS_API_CHG_CANT_ DELETE_DET | API Programming Error(&API_NAME): The API cannot delete this charge line as this line has been submitted to Order Entry |
| n/a | E | CS_API_CHG_ITEM_ MISMATCH | API Programming Error(&API_NAME): The item was rejected as the item billing type does not match the transaction type |
| n/a | E | CS_API_CHG_MISS_ ORD_PARAM | API Programming Error(&API_NAME): The Add to Order requires passing of either parameter '&PARAM1' or parameter 'PARAM2' |
| n/a | E | CS_API_CHG_DATE_ IGNORED | API Programming Error(&API_NAME): The API ignored the parameter '&DATE_PARAM'.\n\<br>Reason: Transaction Type requires it. |

***Table A–9 API Messages***

| Number | Type | Name | Text |
|---|---|---|---|
| n/a | E | CS_API_CHG_DATE_ REQUIRED | API Programming Error(&API_NAME): The API requires the parameter '&DATE_PARAM' as the Transaction Type requires it |
| n/a | E | CS_API_CHG_RATE_ IGNORED | API Programming Error(&API_NAME): The API ignored the parameter '&IGNORED_PARAM'.\n\<br>Reason: This parameter is only needed with Labor Transactions and with the Contracts Model. |
| n/a | E | CS_API_CHG_CANT_ CALC_QTY | API Programming Error(&API_NAME): The API could not calculate the line quantity as this Quantity was not passed |
| n/a | E | CS_API_CHG_MISC | API Programming Error(&API_NAME): The API could not process as it required the definition of a contract and coverage which was missing. |
| n/a | E | CS_API_SR_DESC_FLEX_ ERROR | API Programming Error (&API_NAME): An error occurred when validating the descriptive flexfield. \n\<br>Additional information:&DESC_FLEX_MSG |
| n/a | E | CS_API_ALL_INVALID_ ARGUMENT | API Programming Error (&API_NAME): The value \"&VALUE\" for parameter &PARAMETER is invalid. |
| n/a | E | CS_API_SR_KEY_FLEX_ ERROR | API Programming Error (&API_NAME): An error occurred when validating the key flexfield. \n\<br>Additional information:&KEY_FLEX_MSG |
| n/a | E | CS_API_ALL_NULL_ PARAMETER | API Programming Error (&API_NAME): The value for parameter &NULL_PARAM cannot be NULL |
| n/a | E | CS_API_ALL_PARAM_ IGNORED | API Programming Warning (&API_NAME): Parameter &IGNORED_PARAM was ignored. |
| n/a | E | CS_API_ALL_ DUPLICATE_VALUE | API Programming Error (&API_NAME): The given value parameter \"&DUPLICATE_VAL_PARAM\" cannot be resolved into a unique ID. |
| n/a | E | CS_API_ALL_SAME_ VAL_UPDATE | API Programming Warning (&API_NAME): The update was not performed because the value of \"&SAME_VAL_PARAM\" is the same as the value in the database. |

***Table A–9 API Messages***

| Number | Type | Name | Text |
|---|---|---|---|
| n/a | E | CS_API_ALL_MISSING_PARAM | API Programming Error (&API_NAME): The parameter &MISSING_PARAM is required and was not passed in. |
| n/a | E | CS_API_ALL_ORG_ID_IGNORED | API Programming Warning (&API_NAME): The parameter ORG_ID cannot be set because Multi-Org is not enabled |
| n/a | E | CS_CHG_NO_MULTI_ORG | API Programming Error(&API_NAME): Failure in Multi_Org API |