11*i* APS, Diagnostic and Troubleshooting Case Study

Student Guide

14462GC11

Edition 1.1

January 2000

D31170

ORACLE®

Author

Jeff Goulette

Technical Contributors and Reviewers

Christine Chen, Brett Fox, Roger Goossens, Jeff Goulette, Sridhar Hoskote, Shailesh Kumar, Nile Leach, Moshin Lee, Jignesh Mehta, Jim Rogers, Daniel Soosai, Sam Tupe, Daniel Weir, Paul Willner

This book was published using:

Oracle® Tutor™

Table of Contents

Preface

Profile

Prerequisites

- Introduction to Oracle: SQL and PL/SQL
- R11*i* Use Advanced Supply Chain Planning

Suggested Prerequisites

- R11*i* Define Flexfields
- Enterprise Database Administration Part 1A: Database Administration

How This Course Is Organized

11i APS, Diagnostic and Troubleshooting Case Study is an instructor-led course featuring lecture and hands-on exercises. Online demonstrations and written practice sessions reinforce the concepts and skills introduced.

Related Publications

Oracle Publications

- System release bulletins
- Installation and user's guides
- *read.me* files
- *Oracle Magazine*

Typographic Conventions

Typographic Conventions in Text

Convention	Element	Example
Bold italic	Glossary term (if there is a glossary)	The ***algorithm*** inserts the new key.
Caps and lowercase	Buttons, check boxes, triggers, windows	Click the Executable button. Select the Can't Delete Card check box. Assign a When-Validate-Item trigger to the ORD block. Open the Master Schedule window.
Courier new, case sensitive (default is lowercase)	Code output, directory names, filenames, passwords, pathnames, URLs, user input, usernames	Code output: `debug.set ('I", 300);` Directory: `bin` (DOS), `$FMHOME` (UNIX) Filename: Locate the `init.ora` file. Password: User `tiger` as your password. Pathname: Open `c:\my_docs\projects` URL: Go to `http://www.oracle.com` User input: Enter `300` Username: Log on as `scott`
Initial cap	Graphics labels (unless the term is a proper noun)	Customer address (*but* Oracle Payables)
Italic	Emphasized words and phrases, titles of books and courses, variables	Do *not* save changes to the database. For further information, see *Oracle7 Server SQL Language Reference Manual.* Enter `user_id@us.oracle.com`, where *user_id* is the name of the user.
Quotation marks	Interface elements with long names that have only initial caps; lesson and chapter titles in cross-references	Select "Include a reusable module component" and click Finish. This subject is covered in Unit II, Lesson 3, "Working with Objects."
Uppercase	SQL column names, commands, functions, schemas, table names	Use the SELECT command to view information stored in the LAST_NAME column of the EMP table.

Convention	Element	Example
Arrow	Menu paths	Select File—> Save.

Brackets	Key names	Press [Enter].
Commas	Key sequences	Press and release keys one at a time: [Alternate], [F], [D]
Plus signs	Key combinations	Press and hold these keys simultaneously: [Ctrl]+[Alt]+[Del]

Typographic Conventions in Code

Convention	Element	Example
Caps and lowercase	Oracle Forms triggers	`When-Validate-Item`
Lowercase	Column names, table names	`SELECT last_name` `FROM s_emp;`
	Passwords	`DROP USER scott` `IDENTIFIED BY tiger;`

	PL/SQL objects	`OG_ACTIVATE_LAYER`
		`(OG_GET_LAYER`
		`('prod_pie_layer'))`

Lowercase italic	Syntax variables	`CREATE ROLE` *`role`*
Uppercase	SQL commands and functions	`SELECT userid` `FROM emp;`

Typographic Conventions in Navigation Paths

This course uses simplified navigation paths, such as the following example, to direct you through Oracle Applications.

(N) Invoice > Entry > Invoice Batches Summary (M) Query > Find (B) Approve

This simplified path translates to the following:

1. (N) From the Navigator window, select Invoice > Entry > Invoice Batches Summary.
2. (M) From the menu, select Query > Find.
3. (B) Click the Approve button.

Notations :

(N) = Navigator

(M) = Menu

(T) = Tab

(I) = Icon

(H) = Hyperlink

(B) = Button

Typographical Conventions in Help System Paths

This course uses a "navigation path" convention to represent actions you perform to find pertinent information in the Oracle Applications Help System.

The following help navigation path, for example—

(Help) General Ledger > Journals > Enter Journals

—represents the following sequence of actions:

1. In the navigation frame of the help system window, expand the General Ledger entry.
2. Under the General Ledger entry, expand Journals.
3. Under Journals, select Enter Journals.
4. Review the Enter Journals topic that appears in the document frame of the help system window.

Getting Help

Oracle Applications provides you with a complete online help facility.

Whenever you need assistance, simply choose an item from the Help menu to pinpoint the type of information you want.

To display help for a current window:

1. Choose Window Help from the Help menu, click the Help button on the toolbar, or hold down the Control key and type 'h'.

   A web browser window appears, containing search and navigation frames on the left, and a frame that displays help documents on the right.

   The document frame provides information on the window containing the cursor. The navigation frame displays the top-level topics for your responsibility, arranged in a tree control.

2. If the document frame contains a list of topics associated with the window, click on a topic of interest to display more detailed information.

3. You can navigate to other topics of interest in the help system, or choose Close from your web browser's File menu to close help.

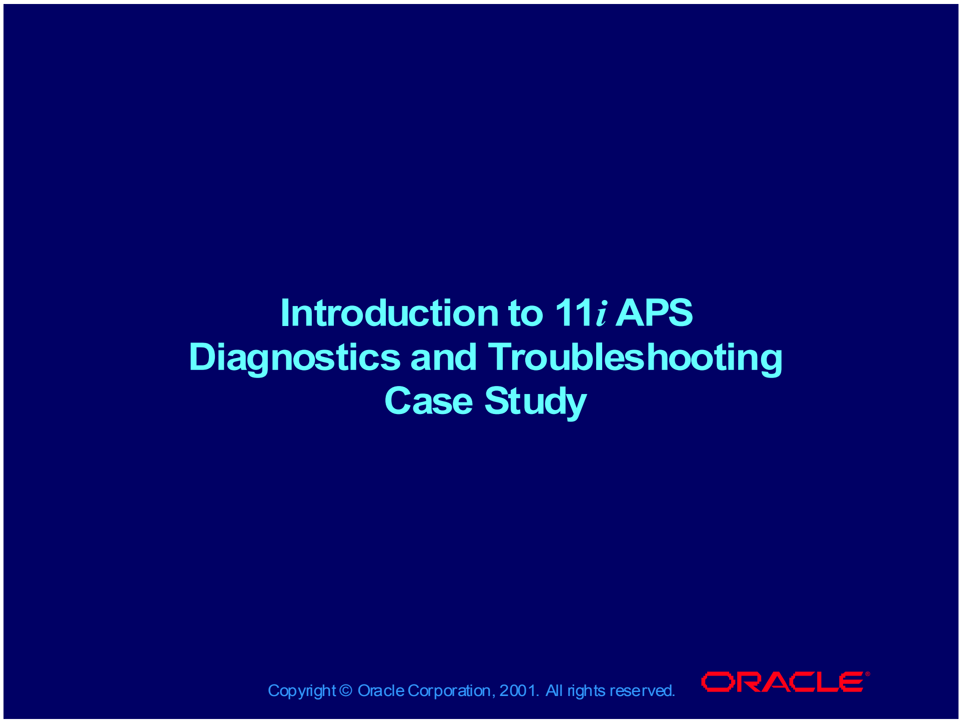
Searching for Help

You can perform a search to find the Oracle Applications help information you want. Simply enter your query in the text field located in the top-left frame of the browser window when viewing help, then click the adjacent Find button.

A list of titles, ranked by relevance and linked to the documents in question, is returned from your search in the right-hand document frame. Click on whichever title seems to best answer your needs to display the complete document in this frame. If the document doesn't fully answer your questions, use your browser's Back button to return to the list of titles and try another.

Introduction to 11*i* APS Diagnostics and Troubleshooting Case Study

Chapter 1

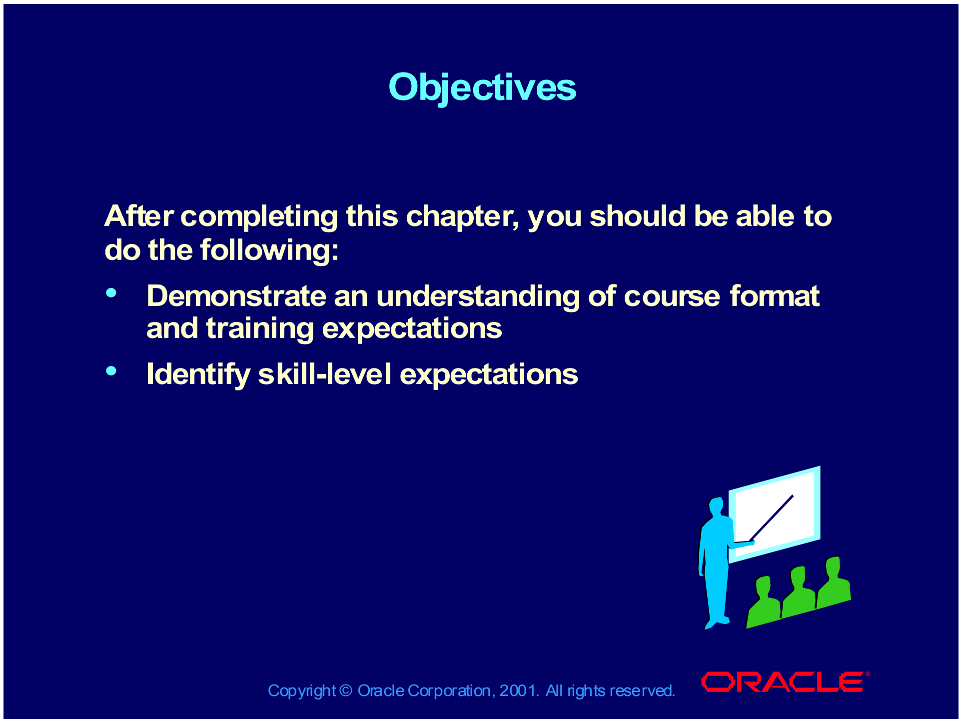

Introduction to 11*i* APS
Diagnostics and Troubleshooting
Case Study
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Agenda

Agenda

- Objectives
- APS technical prerequisites
- Course expectations
- Course content

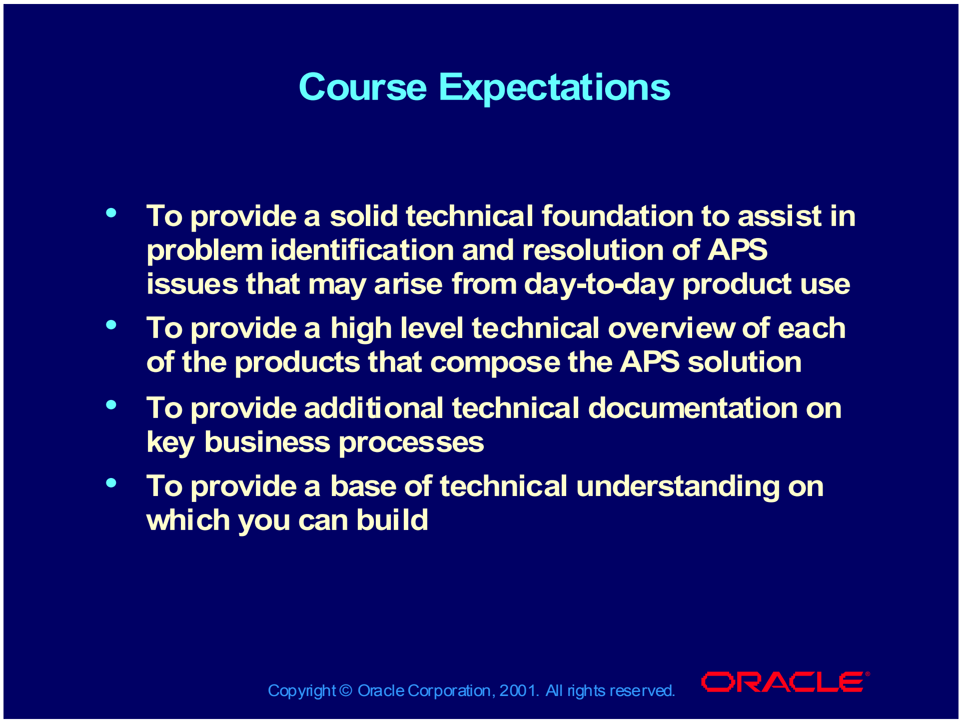
Objectives


Objectives
After completing this chapter, you should be able to do the following:
• Demonstrate an understanding of course format and training expectations
• Identify skill-level expectations
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

APS Technical Prerequisites

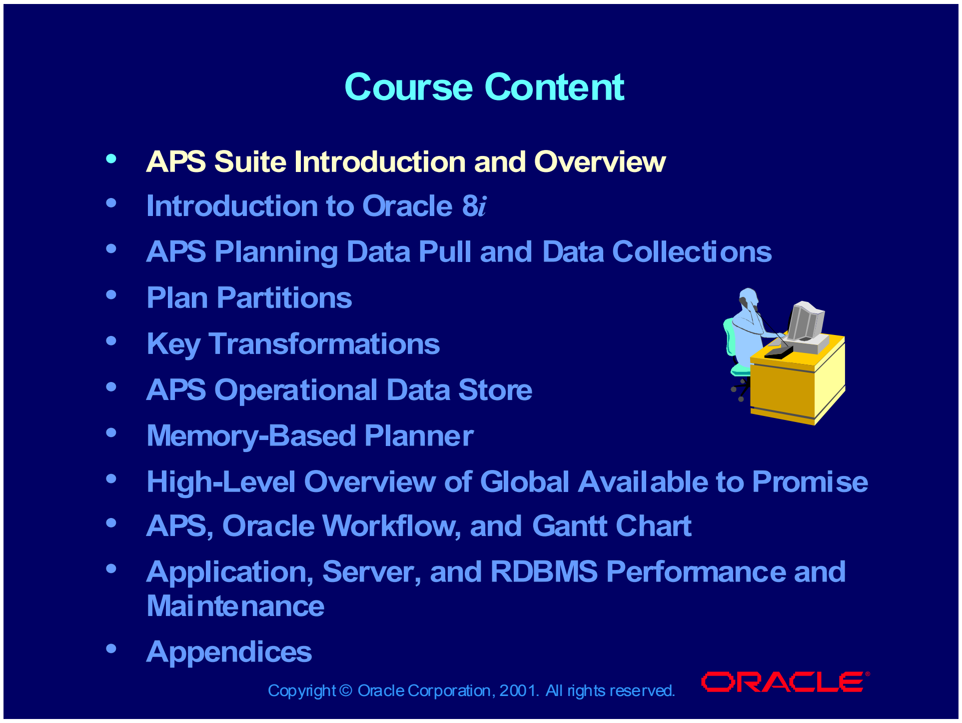
- Navigate Oracle Applications release 11*i*
- Understand basic planning terminology: forecasting, snapshot, and so on
- Understand basic planning functionality: optimization, constraints, and so on
- Have a working knowledge of Oracle Workflow and Oracle flexfields
- Have a working knowledge of SQL
- Understand the concepts of Oracle transaction processing and API principles
- Have a functional knowledge of Advanced Planning and Scheduling (APS)

ORACLE



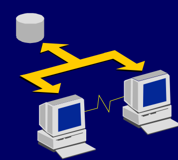
Course Expectations
• To provide a solid technical foundation to assist in problem identification and resolution of APS issues that may arise from day-to-day product use
• To provide a high level technical overview of each of the products that compose the APS solution
• To provide additional technical documentation on key business processes
• To provide a base of technical understanding on which you can build
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Course Content



Course Content

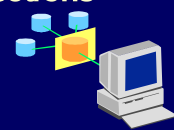
- APS Suite Introduction and Overview
- Introduction to Oracle 8*i*
- APS Planning Data Pull and Data Collections
- Plan Partitions
- Key Transformations
- APS Operational Data Store
- Memory-Based Planner
- High-Level Overview of Global Available to Promise
- APS, Oracle Workflow, and Gantt Chart
- Application, Server, and RDBMS Performance and Maintenance
- Appendices



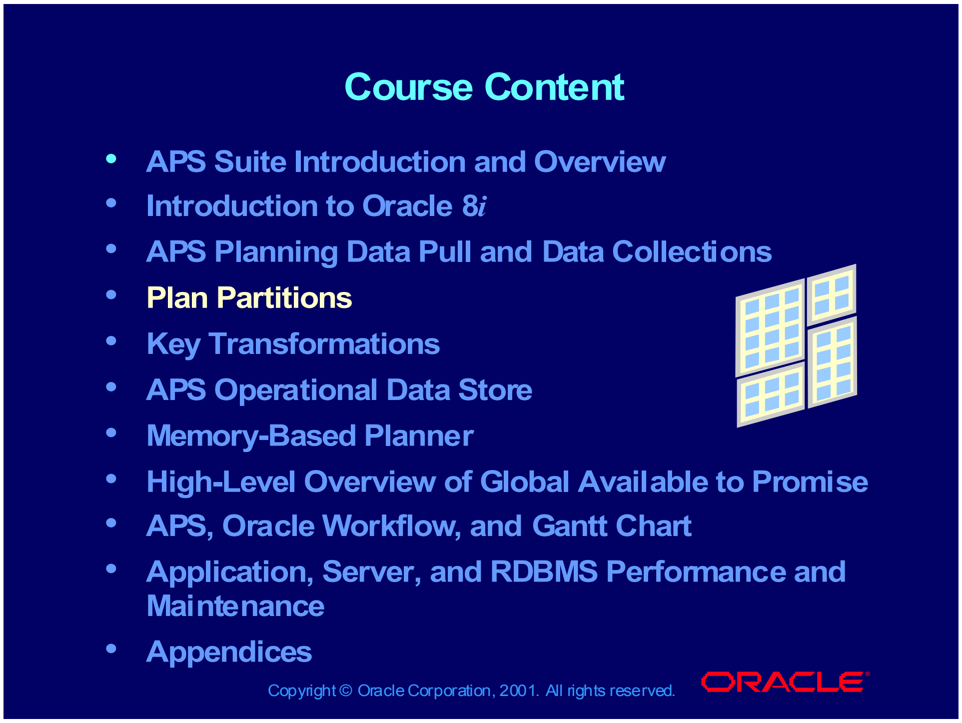
Course Content

Course Content

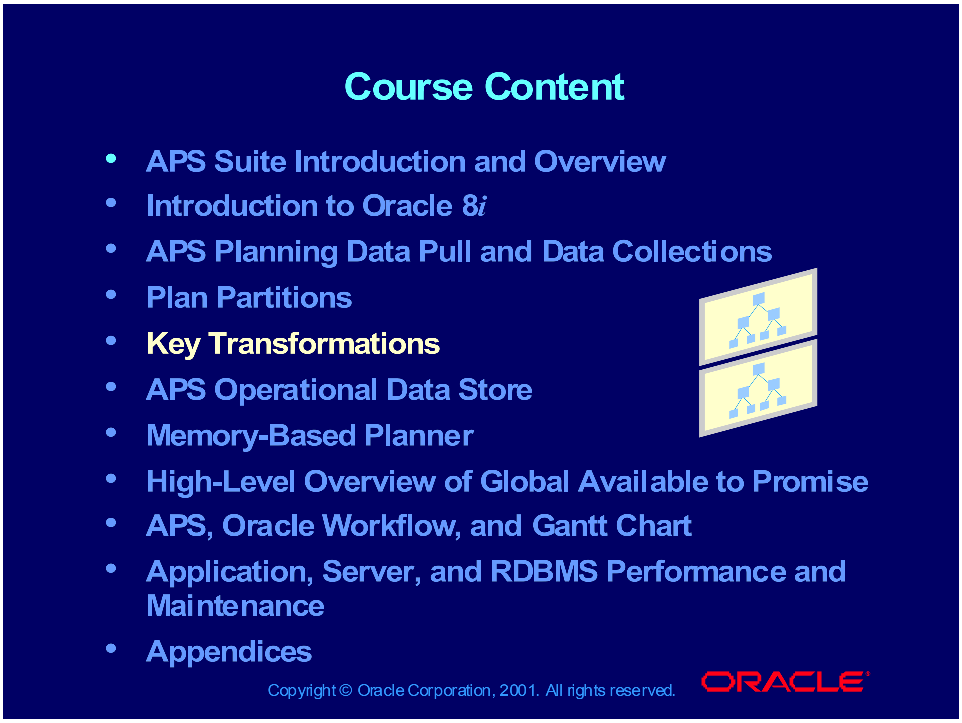
- APS Suite Introduction and Overview
- Introduction to Oracle 8*i*
- APS Planning Data Pull and Data Collections
- Plan Partitions
- Key Transformations
- APS Operational Data Store
- Memory-Based Planner
- High-Level Overview of Global Available to Promise
- APS, Oracle Workflow, and Gantt Chart
- Application, Server, and RDBMS Performance and Maintenance
- Appendices



ORACLE


Course Content
APS Suite Introduction and Overview
Introduction to Oracle 8i
APS Planning Data Pull and Data Collections
Plan Partitions
Key Transformations
APS Operational Data Store
Memory-Based Planner
High-Level Overview of Global Available to Promise
APS, Oracle Workflow, and Gantt Chart
Application, Server, and RDBMS Performance and Maintenance
Appendices
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

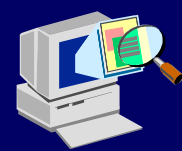
Course Content



Course Content

Course Content

- APS Suite Introduction and Overview
- Introduction to Oracle 8*i*
- APS Planning Data Pull and Data Collections
- Plan Partitions
- Key Transformations
- APS Operational Data Store
- Memory-Based Planner
- High-Level Overview of Global Available to Promise
- APS, Oracle Workflow, and Gantt Chart
- Application, Server, and RDBMS Performance and Maintenance
- Appendices



ORACLE

Course Content

Course Content

- APS Suite Introduction and Overview
- Introduction to Oracle 8*i*
- APS Planning Data Pull and Data Collections
- Plan Partitions
- Key Transformations
- APS Operational Data Store
- Memory-Based Planner
- High-Level Overview of Global Available to Promise
- APS, Oracle Workflow, and Gantt Chart
- Application, Server, and RDBMS Performance and Maintenance
- Appendices

ORACLE

Course Content

Course Content

- APS Suite Introduction and Overview
- Introduction to Oracle 8*i*
- APS Planning Data Pull and Data Collections
- Plan Partitions
- Key Transformations
- APS Operational Data Store
- Memory-Based Planner
- High-Level Overview of Global Available to Promise
- APS, Oracle Workflow, and Gantt Chart
- Application, Server, and RDBMS Performance and Maintenance
- Appendices

Course Content

- APS Suite Introduction and Overview
- Introduction to Oracle 8*i*
- APS Planning Data Pull and Data Collections
- Plan Partitions
- Key Transformations
- APS Operational Data Store
- Memory-Based Planner
- High-Level Overview of Global Available to Promise
- APS, Oracle Workflow, and Gantt Chart
- Application, Server, and RDBMS Performance and Maintenance
- Appendices

Course Content

- APS Suite Introduction and Overview
- Introduction to Oracle 8*i*
- APS Planning Data Pull and Data Collections
- Plan Partitions
- Key Transformations
- APS Operational Data Store
- Memory-Based Planner
- High-Level Overview of Global Available to Promise
- APS, Oracle Workflow, and Gantt Chart
- Application, Server, and RDBMS Performance and Maintenance
- Appendices

Course Content

- APS Suite Introduction and Overview
- Introduction to Oracle 8*i*
- APS Planning Data Pull and Data Collections
- Plan Partitions
- Key Transformations
- APS Operational Data Store
- Memory-Based Planner
- High-Level Overview of Global Available to Promise
- APS, Oracle Workflow, and Gantt Chart
- Application, Server, and RDBMS Performance and Maintenance
- Appendices

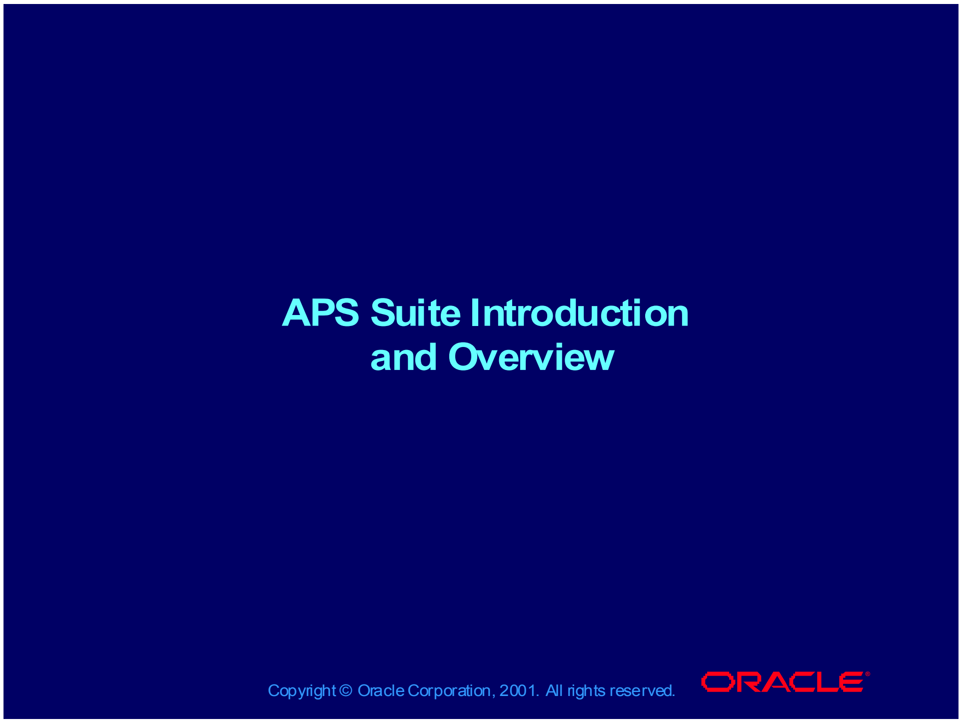
ORACLE

Topics Covered

- Optional maintenance topics
- MSC schema
- MSC programs
- Flexfields
- Key indicators
- Optional tracing hints
- API open interface
- OPM-APS integration

APS Suite Introduction and Overview

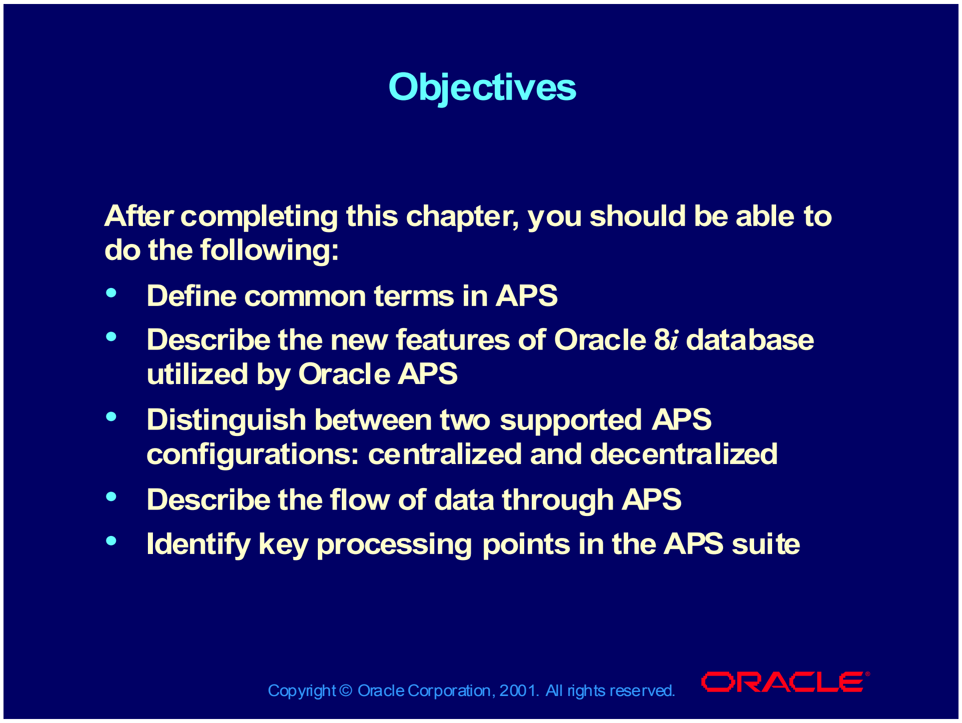
Chapter 2


APS Suite Introduction
and Overview
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Agenda

- Objectives
- Common naming conventions and definition of terms
- Oracle APS components
- APS environment architecture
- Oracle APS system configuration options
- Interface tables for legacy data import
- APS data transformation process

Objectives



Definition of Common APS Terminology

Database link

A database link connection allows a local database to access a remote database. This describes a decentralized configuration. If an error or a conflict occurs during this process, error messages are generated in the log files of processes affected by the error. Database links are used within APS to collect data from the source instance and to push planning recommendations back to the source instance.

Application Data Store (ADS)

This data store represents all of the Oracle Applications database objects that act as a data source. The ADS is used to build and maintain the Operational Data Store and the Planning Data Store (PDS). The ADS represents an instance of Oracle Applications acting as a potential source to Advanced Planning and Scheduling.

Operational Data Store (ODS)

The ODS is a part of the APS planning schema. The ODS is stored within the MSC_* database objects. The ODS is the destination of collected data from Oracle Application Data Stores ADS as well as legacy data. This acts as the input for the Memory Based Planner Snapshot.

Planning Data Store (PDS)

The PDS is a part of the APS planning schema. The PDS is stored within the MSC_*. Database objects. The PDS is used for Planning input and output.

Source Key

The internally generated sequence used as the primary key within the Application DataStore (ADS). For example, the VENDOR_ID corresponding to User Key Code ‘ABC’ in PO_VENDORS_ALL could be 244.

Local ID

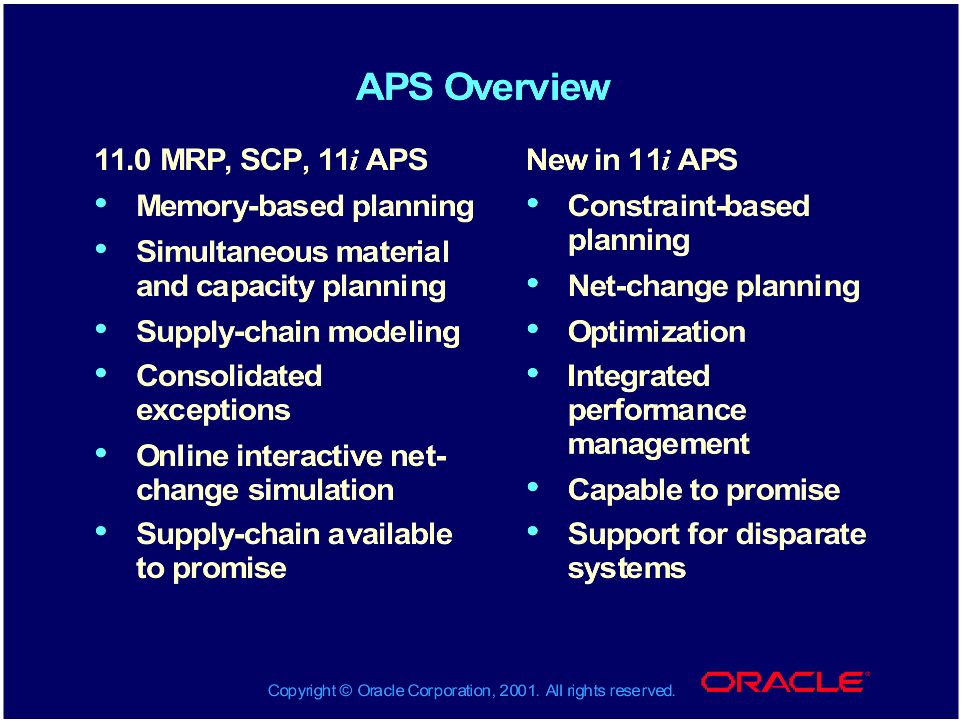
This represents the primary key used in Advanced Supply Chain Planning. Only global entities such as items, categories, suppliers and customers go through key transformation. One Local Id will be generated for the same User Key Code defined in Multiple Source instances.

User Key

A user defined unique key across the Applications instances (based on Flexfield set-ups). For example, ‘Segment1’ holds the Supplier code in PO_VENDORS_ALL.

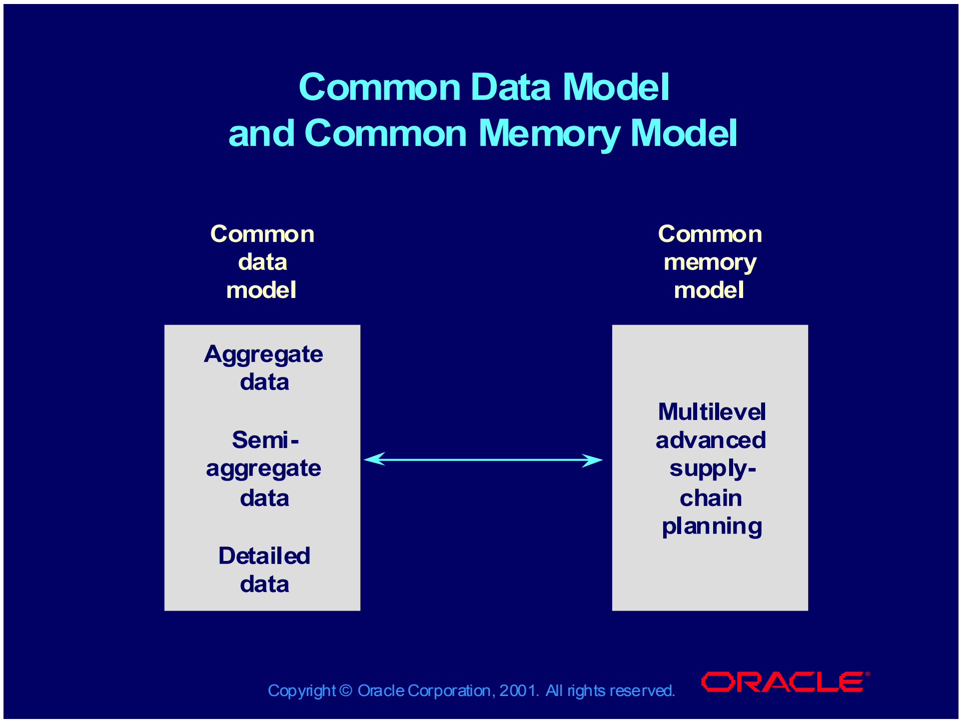
User Key Code

This represents the code or name corresponding to the Source data record. For example, the value in ‘Segment1’ for Supplier ‘ABC Supplier’ in PO_VENDORS_ALL is ‘ABC’ (Vendor Code).


APS Overview
11.0 MRP, SCP, 11*i* APS
• Memory-based planning
• Simultaneous material and capacity planning
• Supply-chain modeling
• Consolidated exceptions
• Online interactive net-change simulation
• Supply-chain available to promise
New in 11*i* APS
• Constraint-based planning
• Net-change planning
• Optimization
• Integrated performance management
• Capable to promise
• Support for disparate systems
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE®

Oracle Applications and APS

- Required
  - Oracle Inventory
  - Oracle Order Entry or Oracle Order Management (11*i*)
  - Oracle Bills of Material
  - Oracle MRP
- Optional
  - Oracle Work in Process
  - Oracle Purchasing
  - Oracle Business Intelligence System
  - Oracle Workflow Designer
  - Oracle Project Manufacturing
  - Oracle Flow Manufacturing



Common Data Model and Common Memory Model

APS and current Oracle Applications share common data for planning and execution. No additional setup is required to ensure successful data integration between Oracle Application releases.

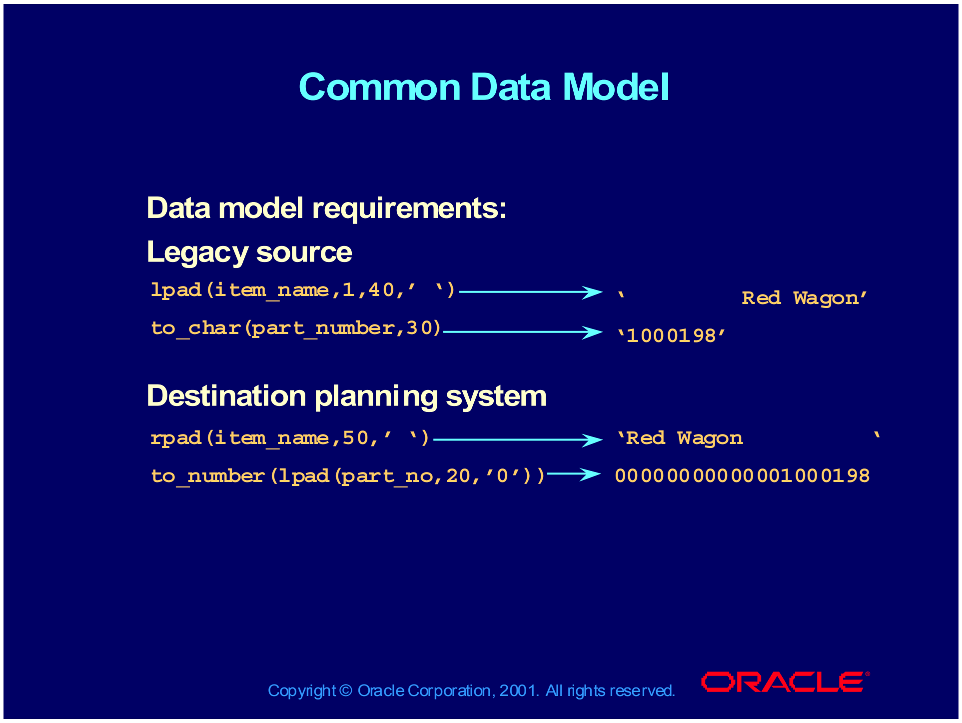
High-level planning and detailed scheduling share a common in-memory process. APS customers never see more than one model of bills, routings, items, bills of distribution, sourcing rules. The same sourcing rules that automatically choose supply sources in Oracle Purchasing and ATP are automatically reused by Oracle Advanced Supply Chain Planning to define the supply-chain sourcing relationships. No additional setup is required

This drastically reduces the time needed to implement the APS solution, leading to much faster return on investment.

- Reduces ongoing maintenance costs by eliminating redundant data entry
- Eliminates the need to maintain multiple representations of your business data

The common data model and single planning process result in lower total cost of ownership and higher returns on your investment.

Common Data Model

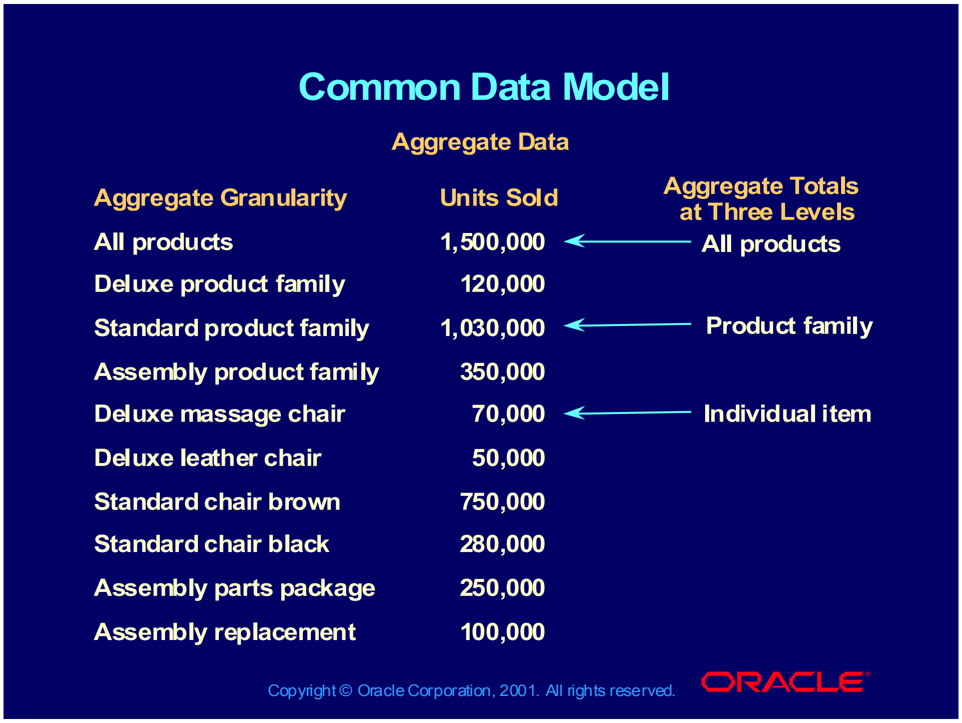


Common Data Model

In the example on the slide, the legacy system requires that the item name is left-padded with spaces, 40 characters long. The destination planning system requires that the item name is 50 characters long, right-padded with spaces.

The legacy system requires that the part number is 30 characters. The destination planning system requires the part number to be a number, 20 digits long, left-padded with zeros.

This mapping exercise is still required for legacy system integration into APS.

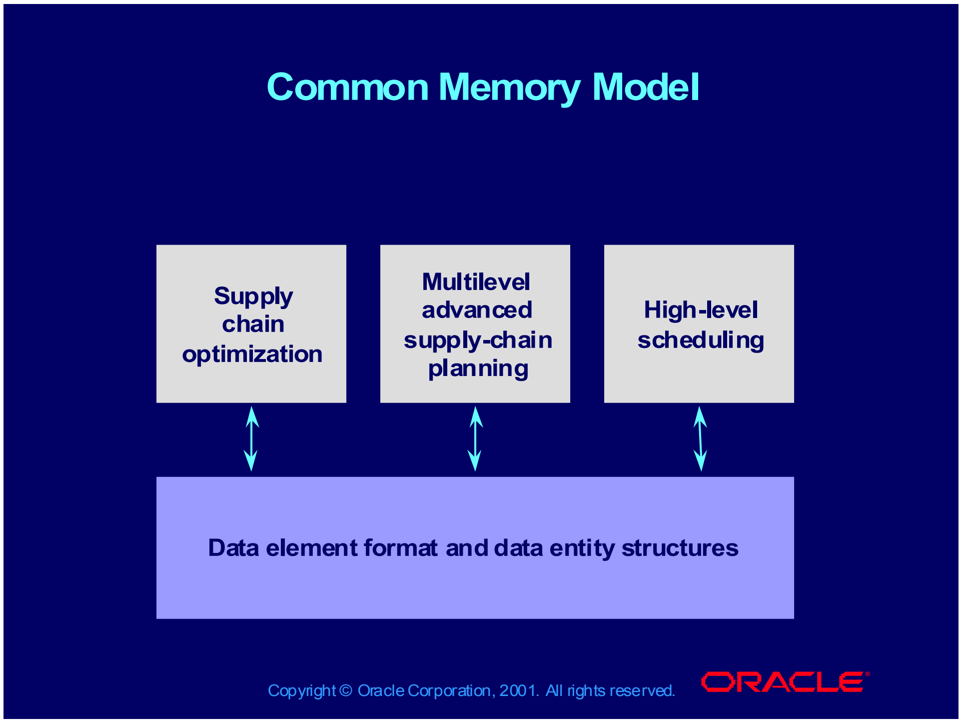


Common Data Model (continued)

In the example on the slide, three possible levels of aggregation for products are:

- Product level
- Product family level
- Item level

APS utilizes aggregate totals in planning to avoid having to calculate these totals during the execution of the advanced supply-chain planning.

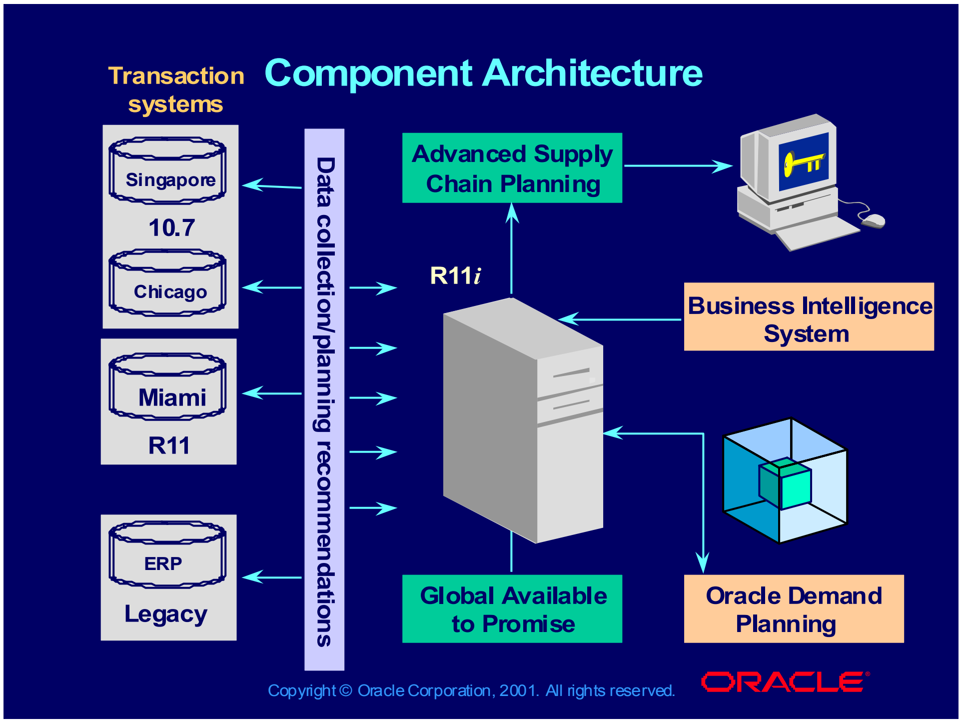


Common Memory Model

Because Oracle uses a common memory model, processing is reserved for planning purposes. A common memory model has the following effects:

- Performance improves because data conversion does not need to occur.
- Code maintenance is not required for changing base tables.
- Consistency of data improves across versions of Oracle Applications.
- Less intervention and translation of changing data models is required.

Component Architecture



Component Architecture

No Redundant Setup

The APS system uses transaction system (also called source system) information already set up in existing release 10.7 or later implementations.

Each of the instances feeding data to the Planning Server, needs to be registered with associated identification information, such as location, local time, and so on. Database links have to be created from the Planning Server to each of the Oracle Applications instances.

Each source instance can have its own refresh interval.

Planning Demand and Supply

To begin the planning cycle, Internet-based collaboration is used to collect demand information from supply chain partners and internal sources.

The demand planning process results in a Master Demand Schedule that is brought into the 11*i* Advanced Planning Server and eventually used as input into the Advanced Supply Chain Planning engine (ASCP)

ASCP then creates plans optimized to objectives such as inventory turnover, profit, on-time delivery, and resource utilization that are consistent with the key business performance indicator targets that you have established.

The Planner Workbench user interface provides graphical comparisons of exceptions and key performance indicators associated with alternative plans.

Key performance indicators:

- Inventory turns

- On-time delivery
- Margin percentage
- Utilization

The Planner Workbench also provides the ability to view exception message details and to quickly simulate alternative plans.

The planning control loop is closed with replanning, which updates order suggestions and Global ATP information based on execution performance.

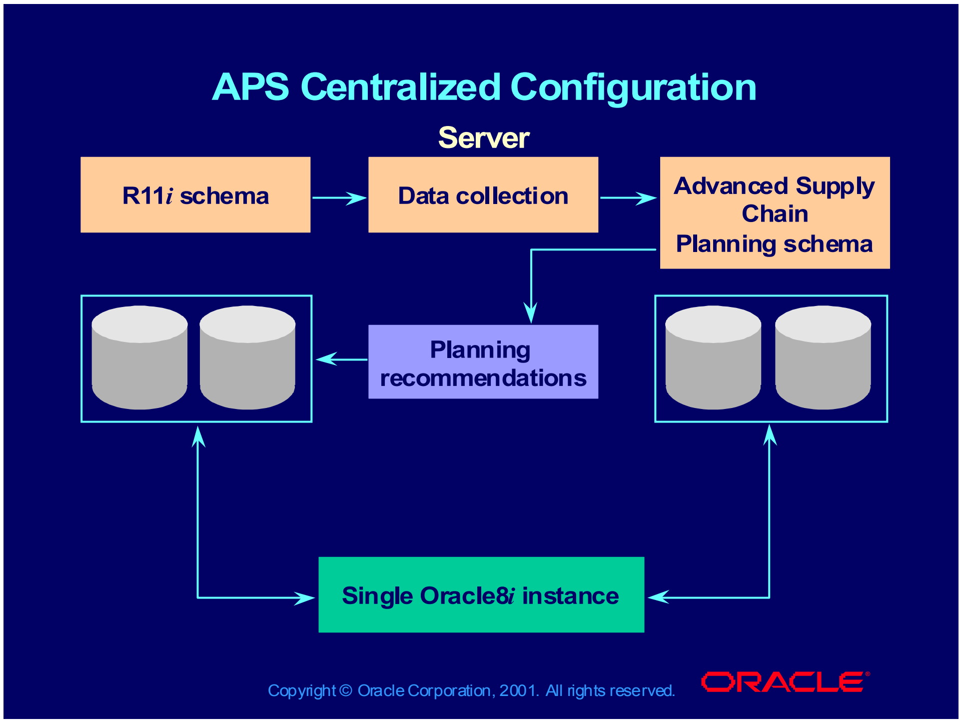
Executing Plans

ASCP creates an optimized sourcing plan consisting of suggested work orders and purchase orders. The supply chain sourcing plan forms the basis for Global ATP sales order promising.

Shop floor job, operation, and resource rescheduling decisions are implemented in the APS Manufacturing Scheduling module and automatically integrated to the Work in process module. These changes to the WIP supply information are collected during the next planning cycle.

There are several integrated modules in APS:

- Transactions systems are extracted from the critical planning path.
- Global Available to Promise becomes centrally available, reflecting several sources of supply and demand.
- Demand Planning supplies new accuracy and functionality in forecasting demand.
- Business Intelligence allows for honing manufacturing according to corporate objectives.
- Advanced Supply Chain Planning (ASCP) manipulates and presents data crunched by the engine.
- Multi-Org Planning Server delivers the common data model built on 8*i*.



APS Centralized Configuration

Advantages:

- Ability to generate single, cross-supply chain plan
- Low server count
- Low server acquisition cost
- Centralized server maintenance

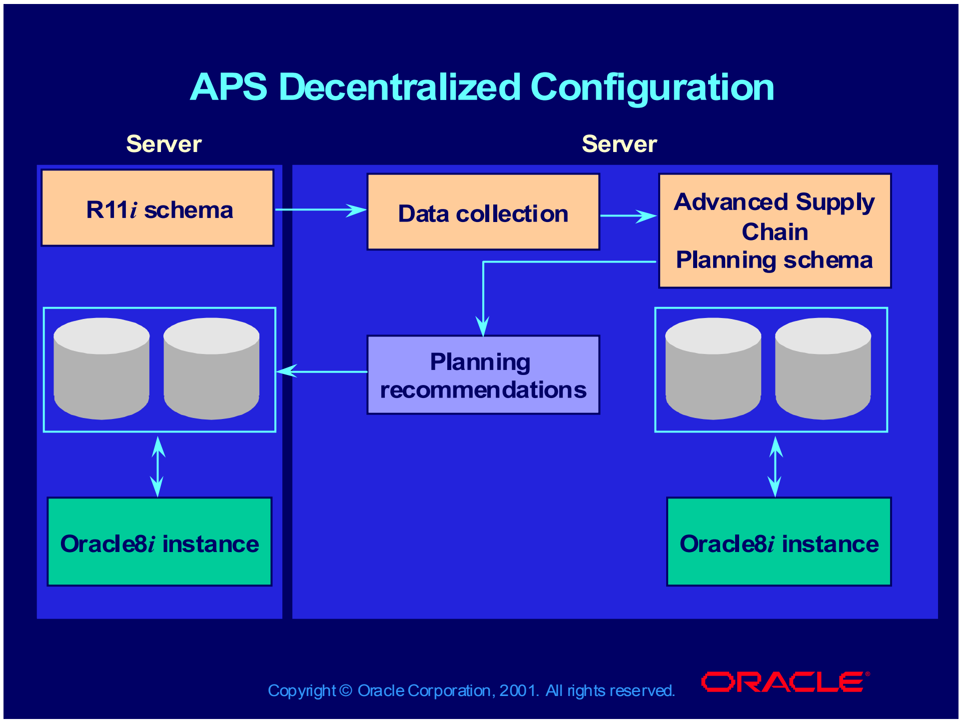
Disadvantages:

- Transaction processing competes for server resources with planning processing (for example, “what-if” planning simulations)
- The ability to replan on a timely basis may be affected by work schedule differences between the individual facilities and the central facility.

Additionally:

- Both MSC and Applications reside in the same database.
- No database link is required in this case.
- The two components can communicate through the planning object APIs and the interface tables defined in APS.
- This configuration is supported only with release 11*i* and later of the Applications.

APS Decentralized Configuration



APS Decentralized Configuration

The MSC works as an enterprise central planning server across several Applications instances.

Source application instances can be release 10.7, 11.0, or 11*i* of Oracle Applications.

The Oracle Applications instance on the Advanced Planning Server must be on release 11*i* of the Applications.

Advantages:

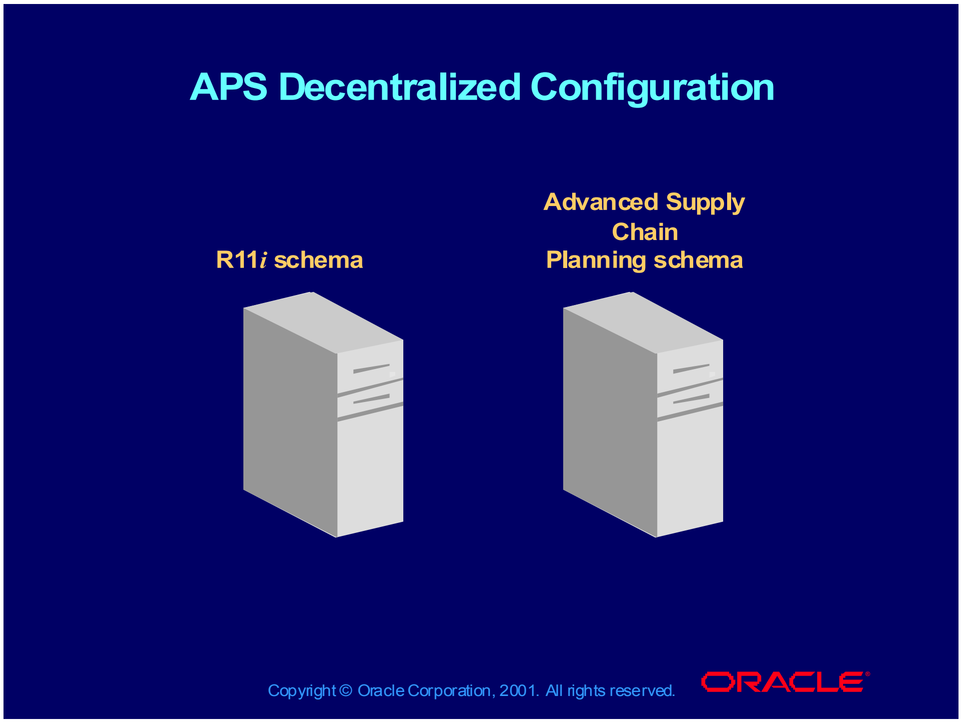
- The highest degree of problem decomposition and solution speed is attained.
- A central planning server generates a distribution requirements plan
- Remote servers generate manufacturing resource plans and detailed schedules for individual facilities

Disadvantages:

- Must perform multiple planning runs in order to generate a complete supply chain plan
- Highest server count
- Highest server acquisition cost
- Decentralized server maintenance requirements

Database links have to be created from the Advanced Planning Server to each of the Oracle Applications instances.

Database links have to be created from the Oracle Application instance to the Advanced Planning Server.

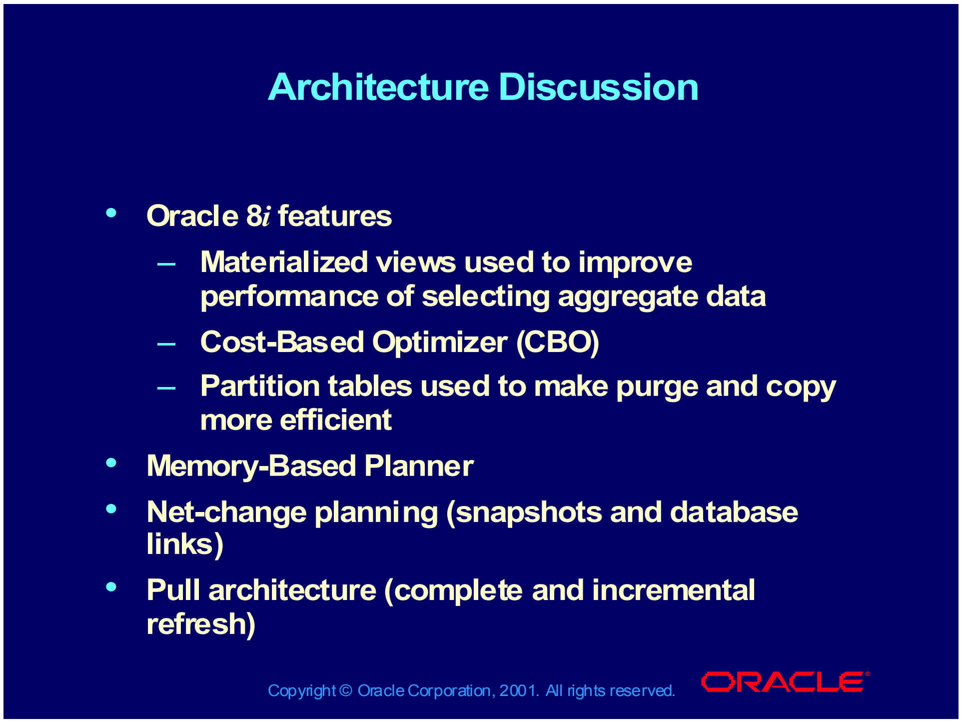


APS Decentralized Configuration (continued)

It is important to note that in a decentralized configuration, database maintenance activities must be performed for each MSC schema.

Architecture Discussion

- Internet-based, three-tier architecture (client, Forms server, database server)
- Modular architecture: decentralized and centralized configuration possible
- 10.7, 11, 11*i* legacy system support
- MSC is the schema/application name
- Integration with CRM suite through Order Management *i*store and Call Center; Sales and Marketing quote/lease information through Demand Planning
- Enhanced graphical user interface
- ILOG suite embedded into the APS system



Architecture Discussion

Partitioning affects the general performance of the Memory-Based Planner (MBP).

Partitioning enables improved performance.

Architecture Discussion

Architecture Discussion

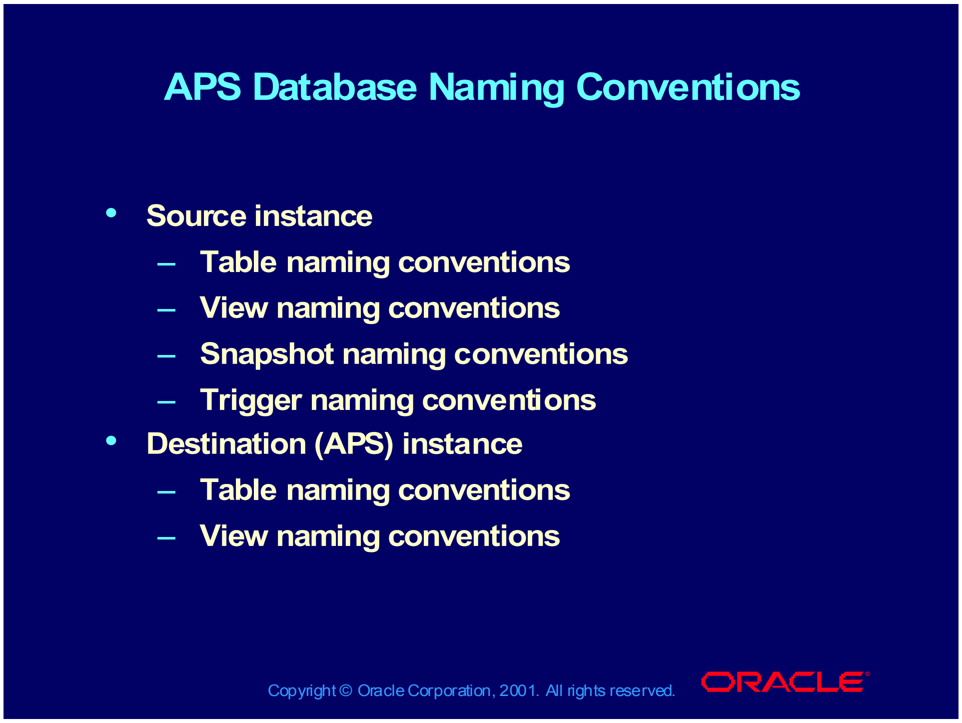
- Multiprocess collection architecture
- Data consolidation (across instances)
- Collection program components (data pull and ODS loader)
- Planner's Workbench with enhanced UI capabilities such as graphing
- Holistic optimization, planning, and scheduling
- Integrated performance management (integration with BIS and Workflow)
- Advanced simulation

Architecture Discussion

Architecture Discussion

- PL/SQL-based public API procedures and interface tables are used in the communication process between application instances and planning server.
- In a decentralized environment, the procedure calls are made using database links.
- If an error or a conflict occurs during the APS process, error messages are generated and stored in the concurrent process log files.
- Data is collected from the source instances into the staging tables on the APS instance (Planning Server) using a database link.

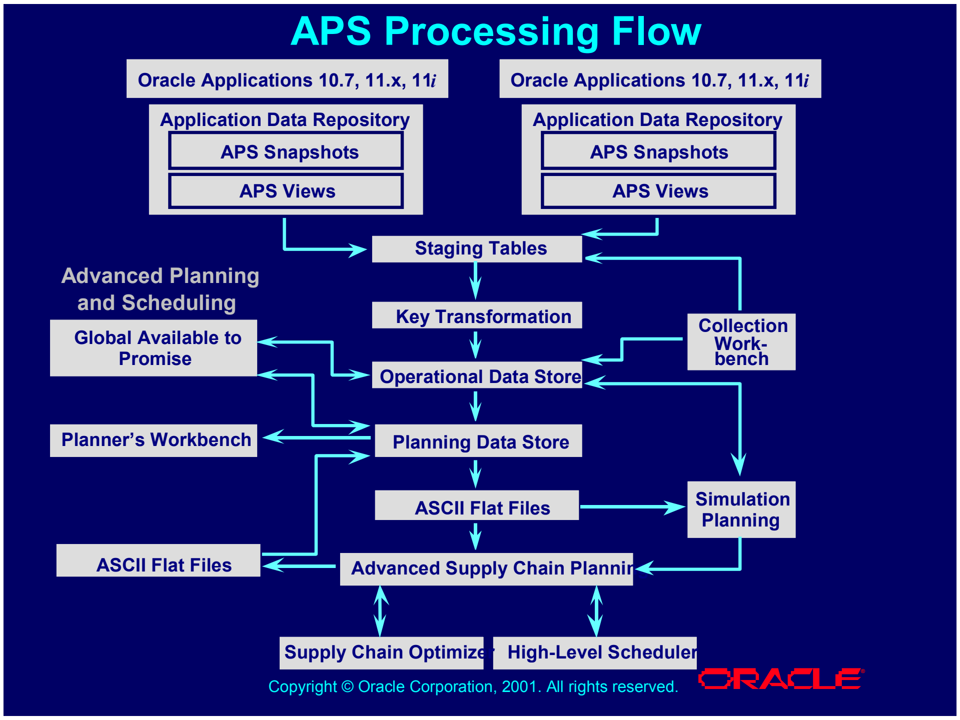
APS Database Naming Conventions



APS Database Naming Conventions

Planning Server	(ODS/PDS)
`MSC_xxx`	Planning table
`MSC_xxx_V`	Planning views
`MSC_xxx_LID`	Planning primary local ID tables
`MSC_SR_xxx`	Planning source tables
`MSC_SR_xxx_V`	Planning source views
`MSC_ST_xxx`	Planning Server staging table
Oracle Application source instances (ADS)	
`MRP_AP_XXX`	Source transaction tables
`MRP_AD_xxx`	Tables are used to trap and store deletions that occur in the source transaction tables
`MRP_SN_xxx_Tn`	Snapshot triggers
`MRP_AD_xxx`	Planning source transaction tables
`MRP_AD_xxx`	Planning source planning tables

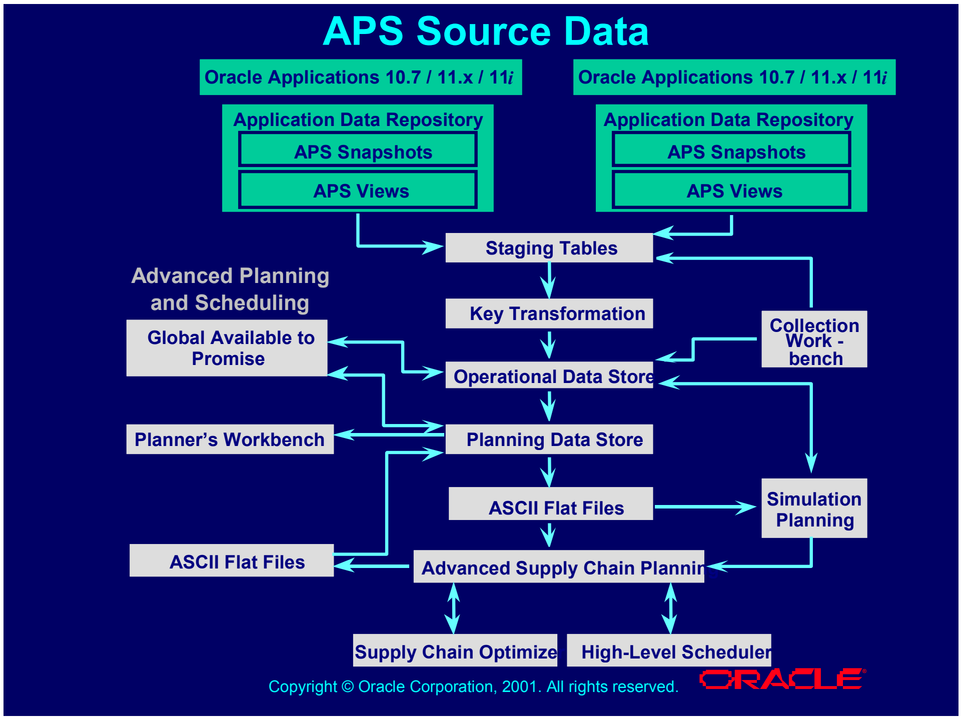
APS Processing Flow



Introduction to the APS Processing Flow

- Data is collected from the source online transaction processing (OLTP) instances schema into the snapshot tables residing in the source instance.
- The Planning Data Pull pulls data from the source instance and moves it to staging tables on the APS instance using database links, if required.
- The Planning Operation Load transforms the data in the staging tables and moves it into the Operational Data Store.
- The Memory-Based Planner uses the data from Operational Data Store and the Planning Data Store to generate the plan output for Advanced Supply Chain Planning.
- The Memory-Based Planner interacts with Supply Chain Optimizer and the High-Level Scheduler to deliver a plan that can be produced.
- Planning recommendations are then pushed back to the corresponding source instances using a database link. This is performed by autoreleasing discrete jobs and requisitions.
- Integration between 10.7, 11.x, and legacy systems is delivered using the open interface capabilities.
- ASCII flat files are loaded into Oracle objects by way of SQL*Loader.
- Global Available to Promise can be configured to use the APS Operational Data Store or the Planning Data Store

Note: APS = Advanced Supply Chain Planning + Oracle Demand Planning + Manufacturing Scheduling.



APS Source Data

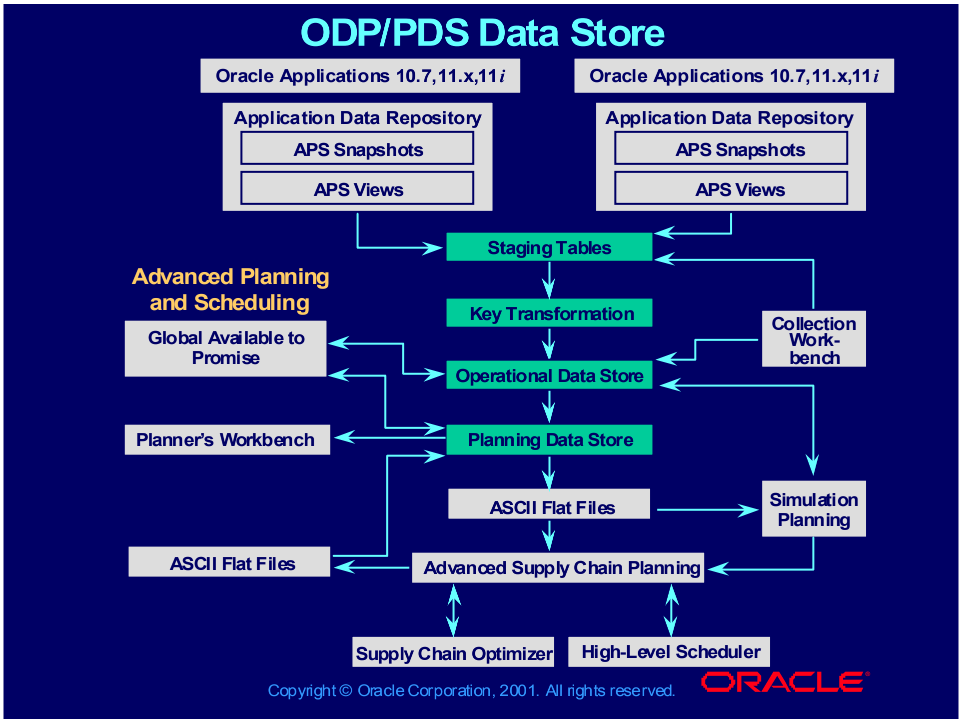
Application Data Store (ADS)

- This ADS is used to build and maintain the Planning Data Store in the APS system.
- The ADS represents an instance of Oracle Applications as a legacy system that will not have the seeded Oracle APS database objects.

`MRP_AP_XXX`	Source transaction tables
`MRP_AD_xxx_V`	Source transaction views
`MRP_SN_xxx_Tn`	Snapshot triggers
`MRP_AD_xxx`	Planning source transaction tables
`MRP_AD_xxx`	Planning source planning tables

Each of the instances feeding data to the Planning server needs to be registered with associated identification information such as location, local time, and so on.

Rows are populated in the `MRP_AD_xxx` and `MRP_AP_xxx` tables by database triggers (`MRP_SN_xxx_Tn`) defined on the corresponding database snapshots (`xxx_SN`). This activity occurs on the source instance.



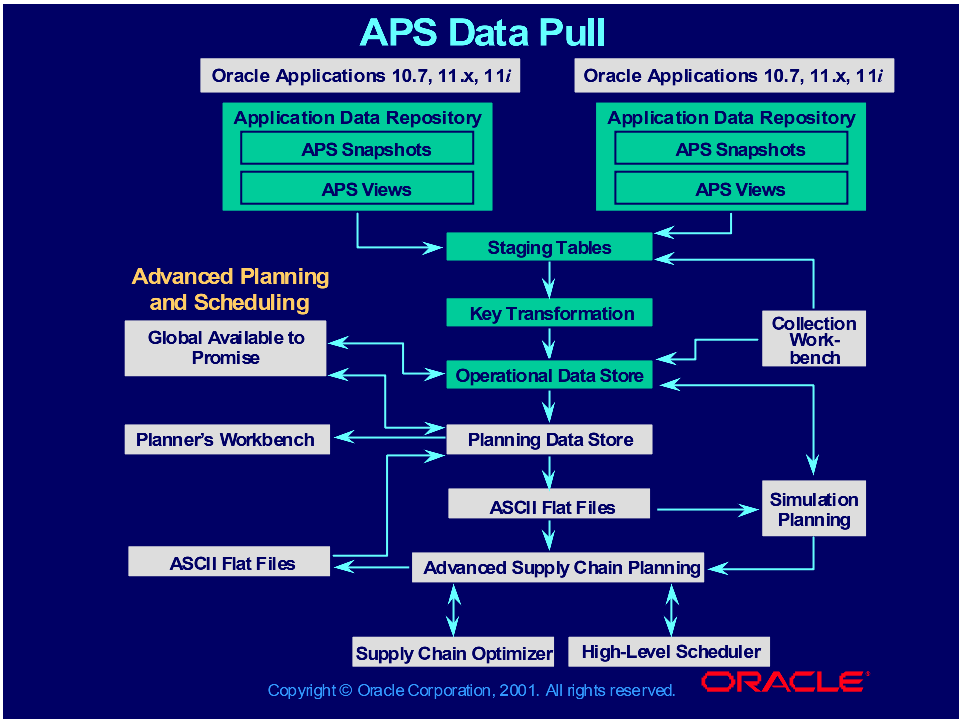
ODS/PDS Data Store

Operational Data Store (ODS)

- The Operational Data Store (ODS) is the part of the planning component that represents all the MSC tables that act as destination for the collected data from each of the Application Data Stores.
- This acts as a source of input to the memory-based snapshot portion of the planning process. ODS and the Planning Data Store, defined below, share the same MSC schema.
- The ODS contains plan-neutral data, which is data that has not been included in an Advanced Supply Chain Planning plan.

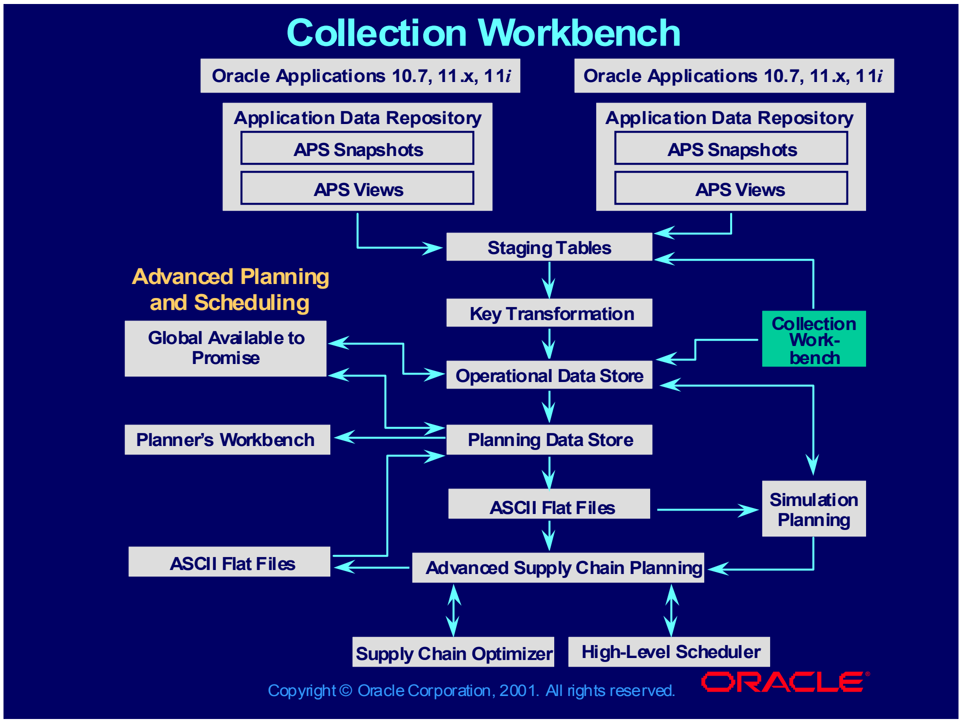
Planning Data Store (PDS)

- Planning Data Store (PDS) represents the Advanced Supply Chain Planning data in the MSC schema.
- The PDS acts as a source of input to the memory-based snapshot portion of the planning process. ODS and PDS share the same MSC schema.



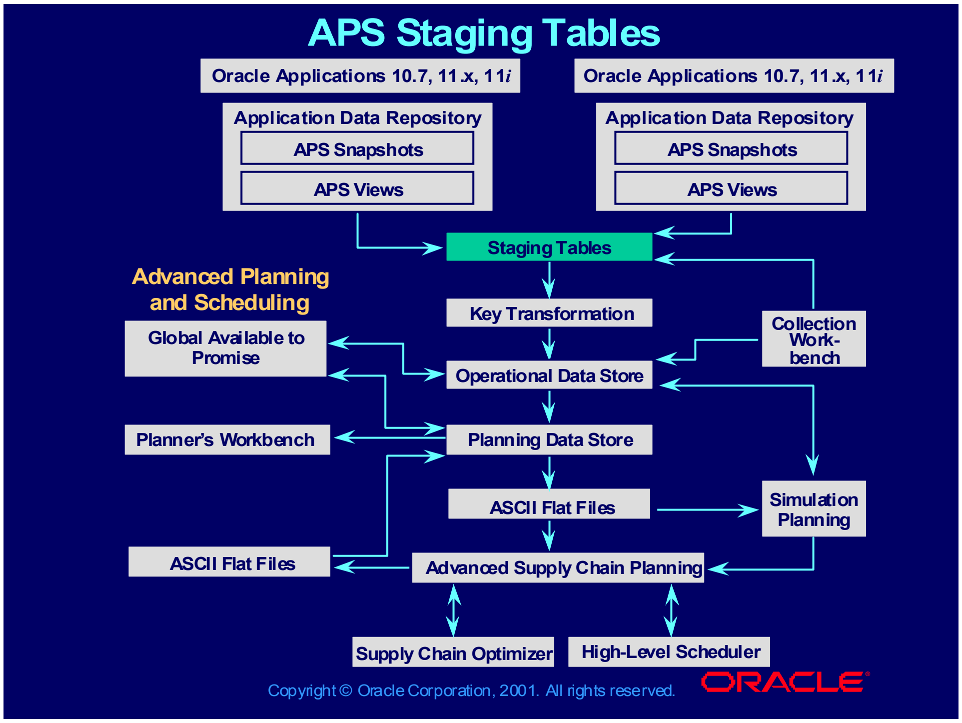
APS Data Pull

- The Planning Data Pull (PDP) is the act of pulling data from source Application Data Store database objects.
- The PDP is flexible and can be scheduled for an instance with different parameter values.
- The PDP can be performed to match business operations. If suppliers are updated every two weeks, pull the supplier data entity every two weeks.
- Data collection types;
  - Data Collection Complete or complete refresh is required after certain data changes. This will be covered in the PDP lesson.
  - Data Collection Incremental can be scheduled when desired, keeping planning data current.
  - Data Collection from legacy system is performed using the staging tables.
- Staging tables act as temporary storage for data destined for the Operational Data Store (ODS). It is here that a user-supplied data-cleansing tool can be used to scrub incoming data.
- As this data is loaded into the ODS, key transformation takes place. Data is extracted from the staging tables and the primary keys are transformed before the data is loaded into the ODS.
- The ODS acts as a temporary area, used for storage of data until needed in the Advanced Supply Chain Planning engine.



Collection Workbench

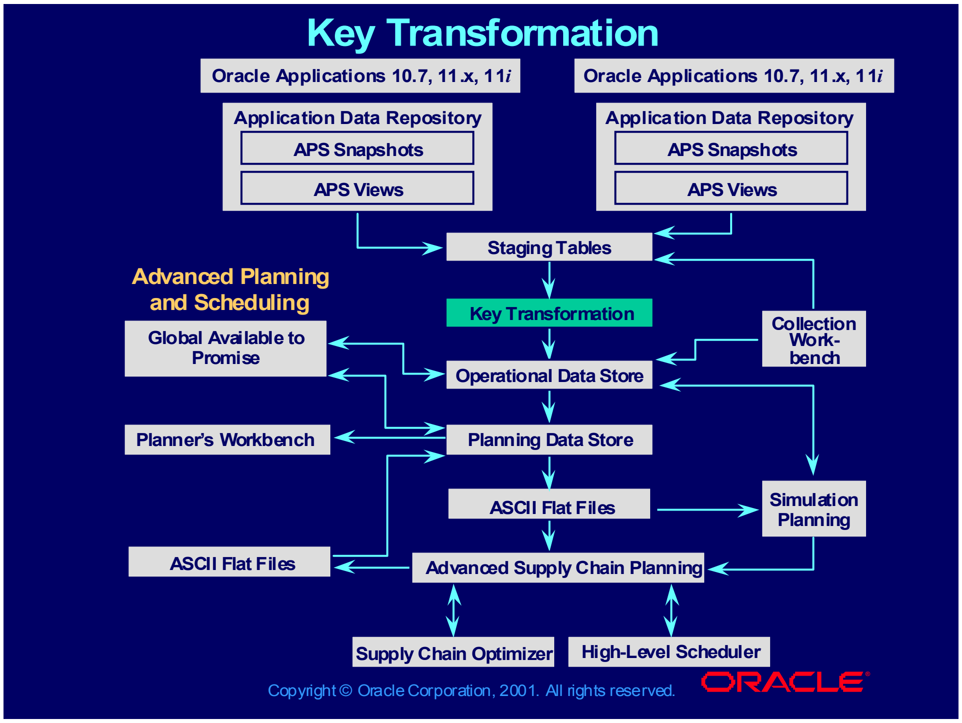
Use the Collection Workbench to verify collected data.



APS Staging Tables

- Staging tables can be loaded individually.
- Once the database is loaded, the data can be cleansed by user cleansing programs.
- The optional, user-supplied, data-cleansing program must be run at the command prompt or registered and entered into the PDP request set or ATP request set. This topic will be discussed in a later lesson.

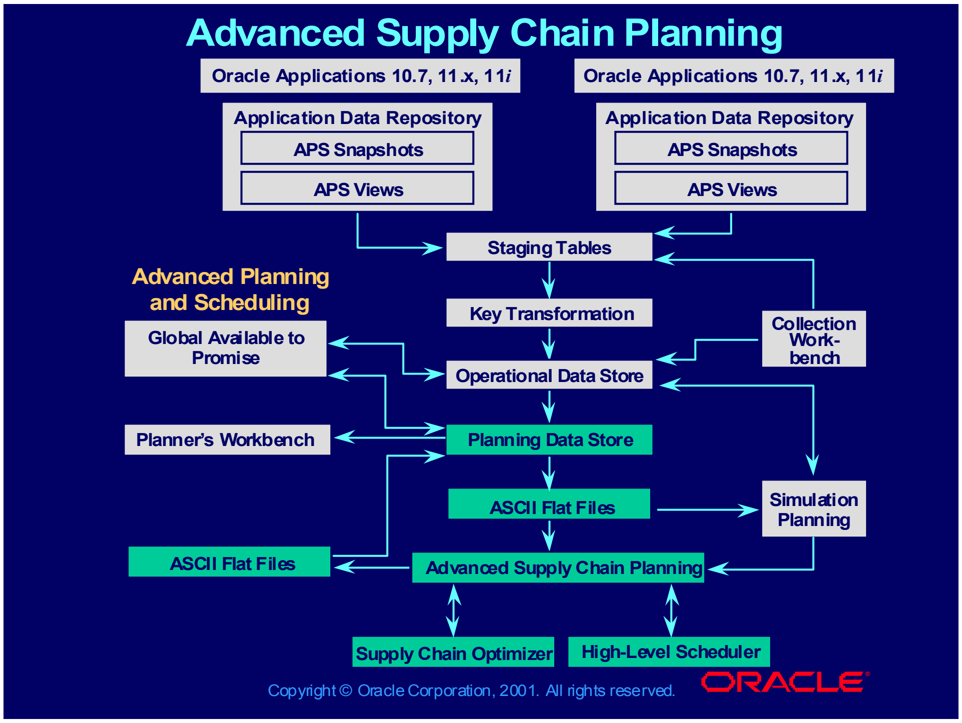
Key Transformation



Key Transformation

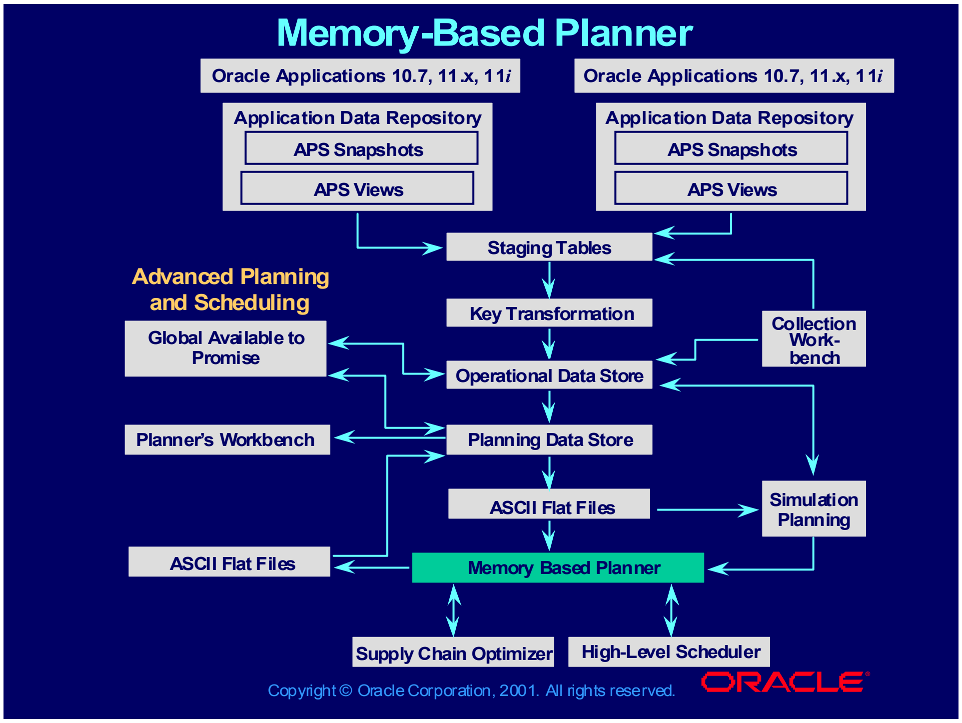
- Key Transformation must take place before the source data is made available to APS.
- The data types used in the primary key are `NUMBER` and `VARCHAR2`. Key transformation takes place only for items, categories, suppliers, and customers.
- The mapping of the transformation from the source ID to the local ID is stored in the mapping tables.
- Purge notes: The unused data in the mapping tables can be deleted only during the data purging process. There will be orphaned business entities as time progresses, and these are deleted at that time.

Advanced Supply Chain Planning



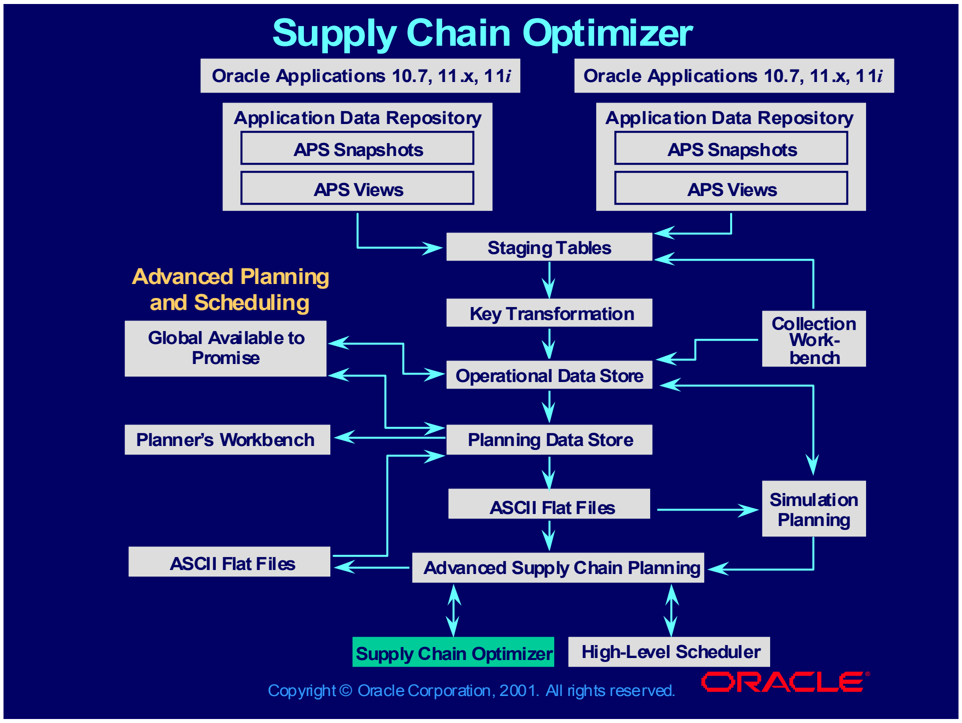
Advanced Supply Chain Planning

- The required planning data is extracted from the ODS and PDS.
- This data is then written to ASCII flat files.
- The ASCII flat files are loaded into the memory structures required for Advanced Supply Chain Planning.
- The Advanced Supply Chain Planning engine is made of three distinct modules that are compiled and linked as one executable. The modules of code are:
  - Oracle MBP-based planning and netting engine. A C-based concurrent program, the MBP, creates production and distribution plans for all organizations. The MBP identifies the constraint violations from the output of SCO. The MBP also performs detailed full pegging of supply to demand and generated exceptions based on user tolerance.
  - Supply Chain Optimizer (SCO), C++ using ILOG CPLEX libraries and includes. The SCO identifies constraint violations that were caused by possible relaxation of specific nonlinear constraints: quantity dependent lead times and actual BOM/routing effectivities based on such dynamic order starts.
  - High-Level Scheduler (HLS), C++ using ILOG solver and Scheduler includes and libraries. HLS schedules, loads, and repairs the output of planning to arrive at a balanced and executable schedule for each of the facilities planned.



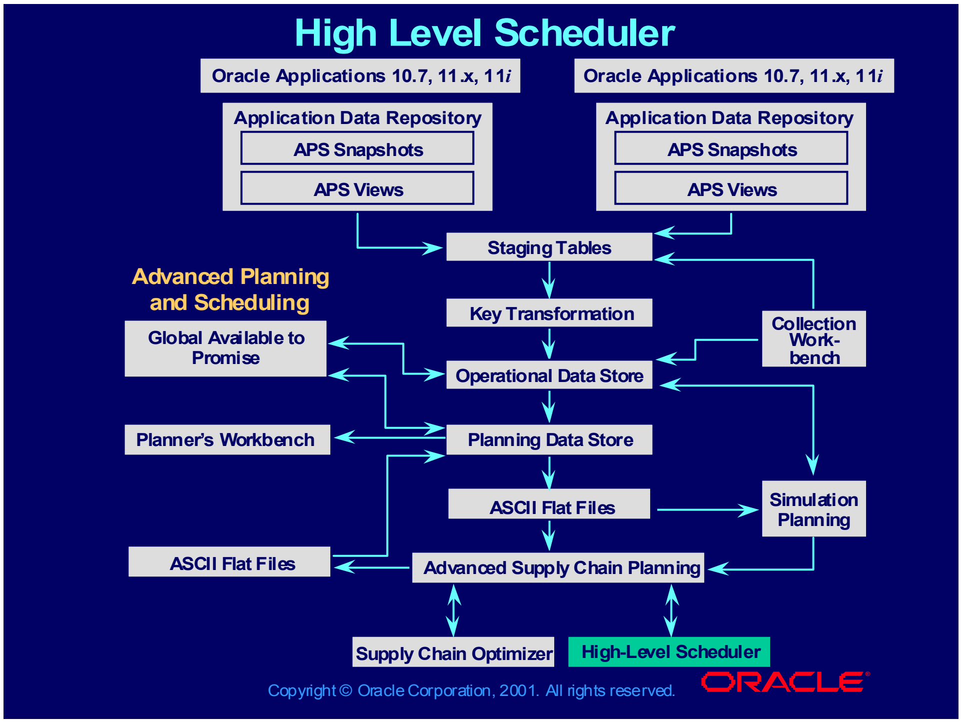
Memory Based Planner (MBP)

- The MBP engine is the centerpiece of the APS solution. It is implemented as an AOL concurrent program.
- The MBP allows replanning with minimal I/O because the data has been captured and stored in the ODS/PDS data stores.
- The responsibilities of the MBP include planning and netting.
- The results of the MBP are flushed into the database, ready for display and reporting.
- For more information on synchronization, refer to a later lesson dealing with MBP internals.



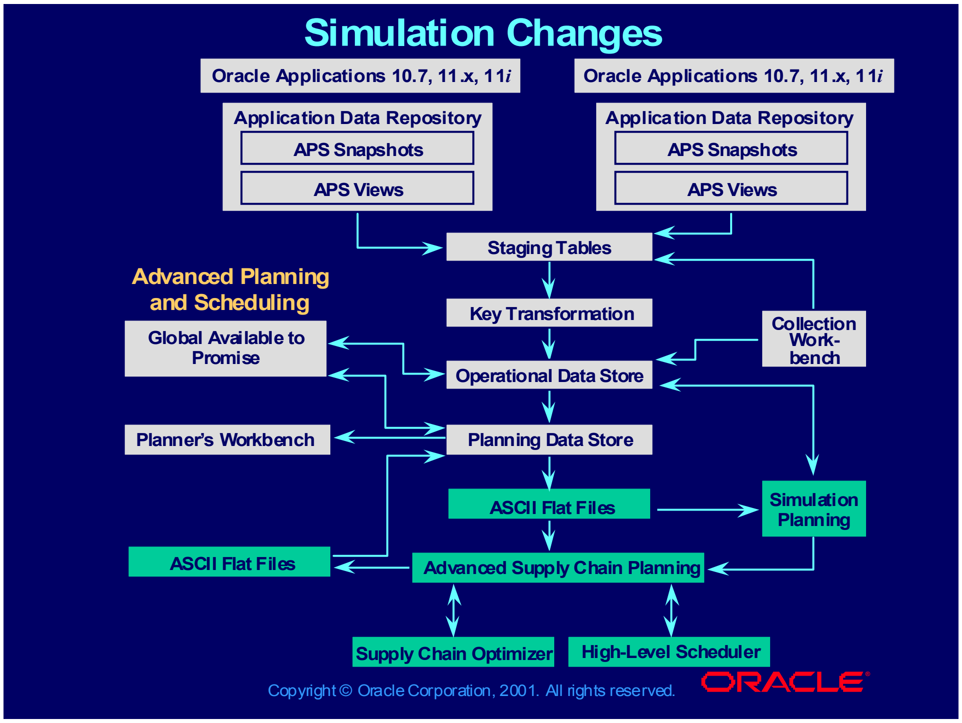
Supply Chain Optimizer (SCO)

- Uses ILOG solver and optimization technology
- Incorporates the use of optimization techniques to determine sourcing decisions and generate time-phased supplies to meet demand
- Optimizes plans to fit strategic objectives, producing optimal production and distribution plans for each organization
- Inputs results to the third-generation Memory-Based Planner



High-Level Scheduler (HLS)

- The HLS is a C++ library for solving problems in planning, allocation, and optimization, exploiting constraint programming and object-oriented programming.
- The current objectives for the HLS are demand priority and capacity constraints.



Simulation Changes

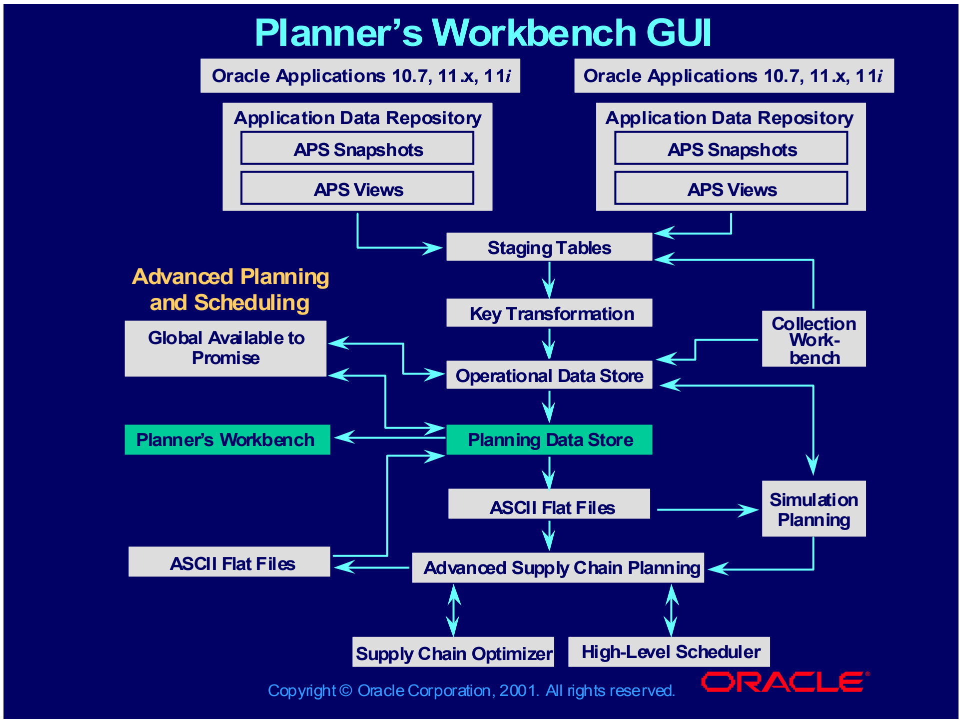
You are able to simulate various scenarios and changes, using the enhanced Planner's Workbench, and replan to see the impact of these changes on the current plan.

MBP simulations are effective for the five following types of changes:

- Supply
- Demand
- Resource requirements
- Supplier capacity
- Resource availability

The ASCII flat files used for storage of MBP data are reloaded into the MBP for simulation planning.

- These input tables must not be erased if simulation planning is to be done.
- More information regarding this process is available in the MBP internals lesson.

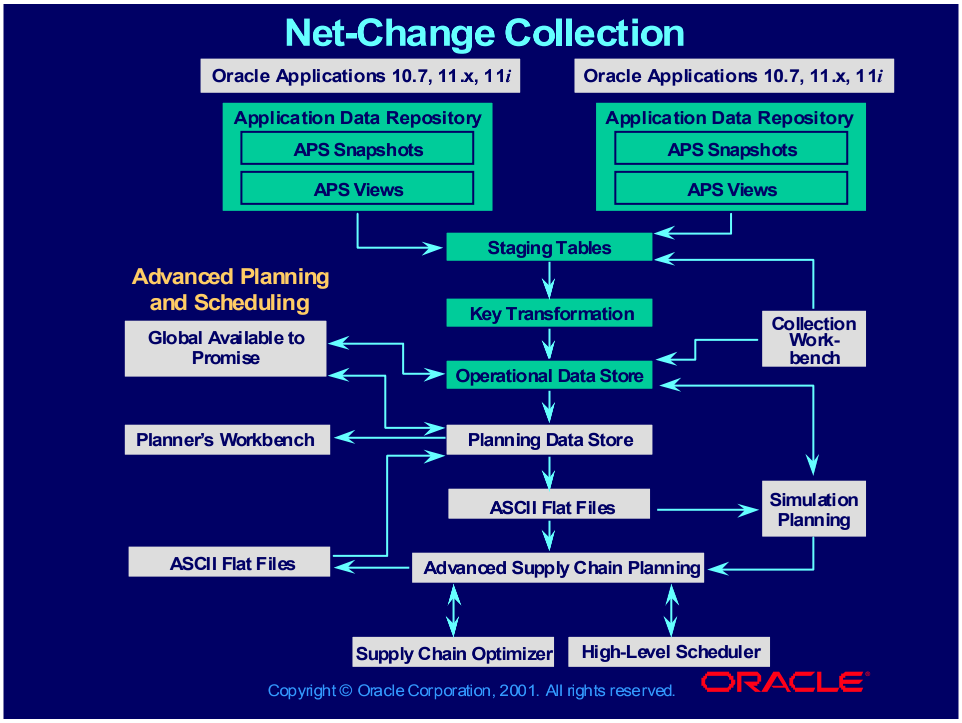


Planner's Workbench GUI

- The form name is `MSCFNSCW.fmb`
- The Gantt chart is used for displaying activities in a resource-centric and forms-centric view.
- The forms tree is used for pegging.
- The pivot table and graph widgets are used for horizontal plans and KPI graphs.
- Release purchasing requirements can be pushed back to the source instance.
- Release WIP requirements can be pushed back to the source instance.

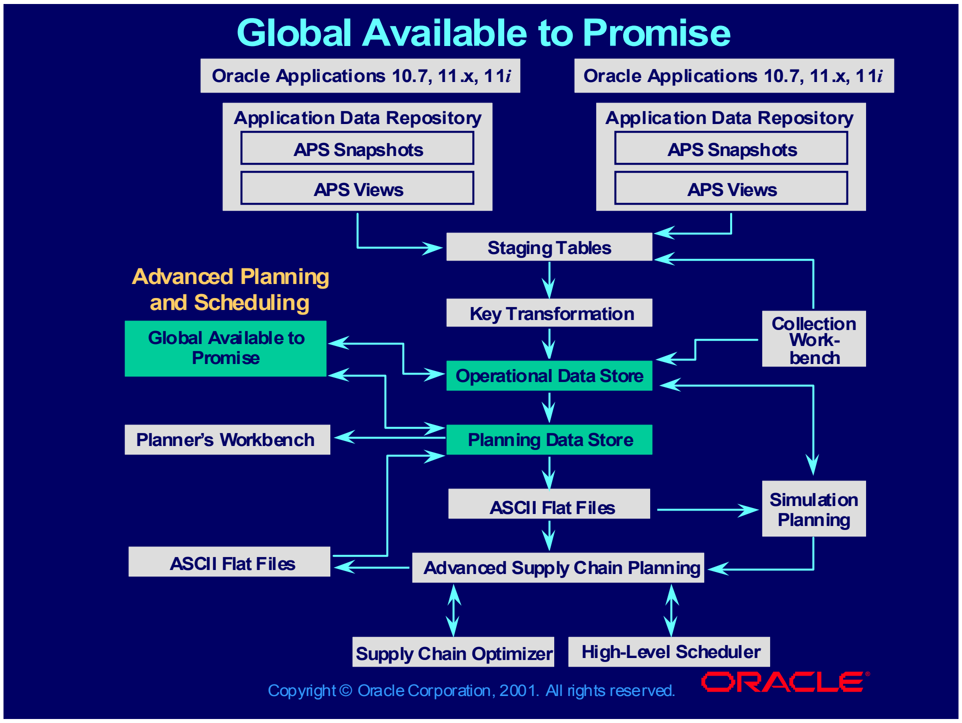
Possible manipulation in Workbench:

	Release	Reschedule/Cancel
WIP	YES	YES
Repetitive Schedule	YES	N/A
PO Requisition	YES	YES
PO	N/A	Workflow Enabled
Flow Manufacturing	NO	NO



Net-Change Collection

- Net changes reflect business data groups that are collected and applied to the ODS.
- Only the data that has been changed since the last collection is collected into APS.



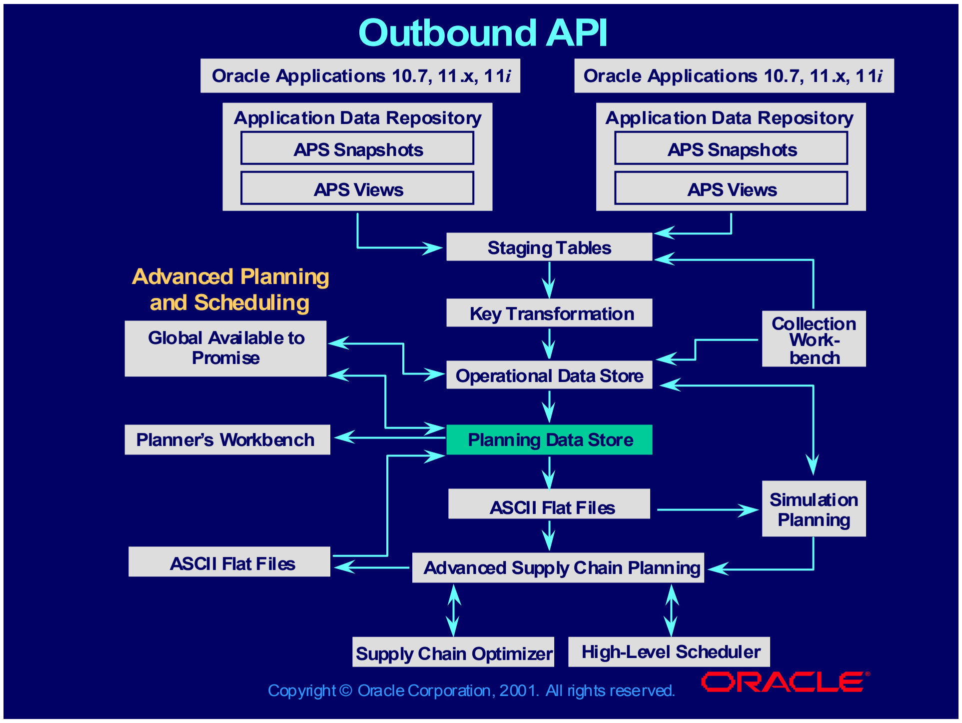
Global Available to Promise (GATP)

GATP information is derived from:

- Operational Data Store (ODS)
- Planning Data Store (PDS)

The GATP program:

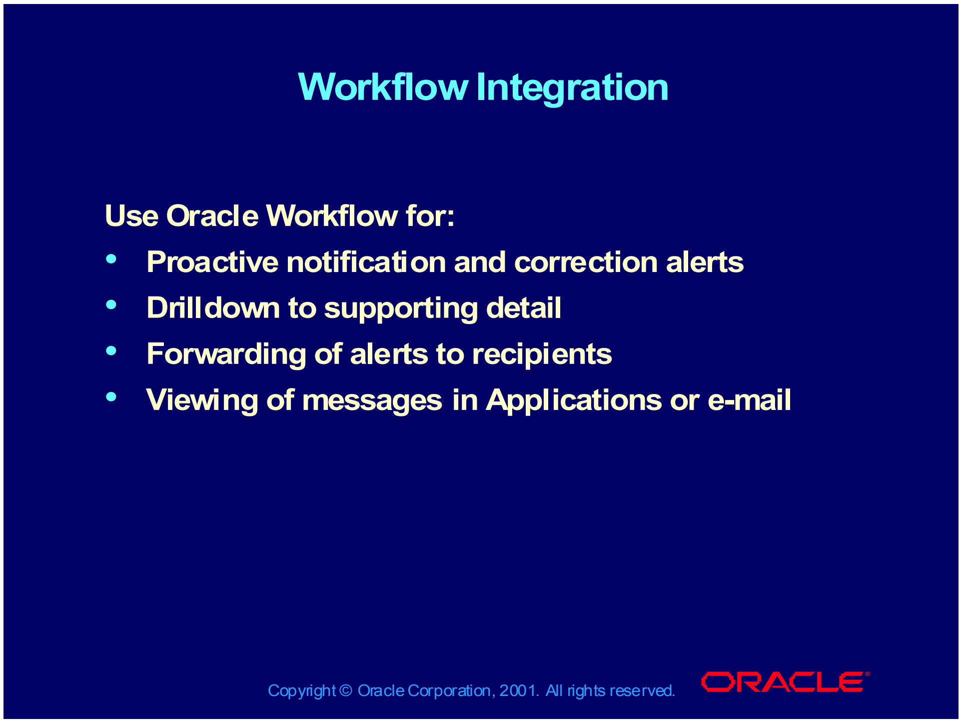
- Collects the existing item supply and item demand records from the execution system or ADS
- Utilizes the multithreaded capabilities of the Oracle database to process concurrent ATP request efficiently
- Uses the same technology as APS data pull



Outbound API

- MBP results are published back to the source OLTP instance by way of the release mechanisms in the Planner's Workbench.
- The customer can read from the PDS data store, populate ASCII flat files, and load this data back to the legacy system.

Workflow Integration



Workflow Integration

The new, powerful collaboration features of Oracle Supply Chain Planning enable you to automate and streamline your interactions with customers and suppliers.

Oracle APS contains built-in features from Oracle Workflow that allow:

- Proactive notification and corrective action alerts
- Drilldown to supporting detail
- Forwarding of alerts to proper recipients by way of Oracle Workflow after a plan is run, if the Production plan parameter is selected.
- Viewing of the messages either in Applications or e-mail.

Workflow Integration

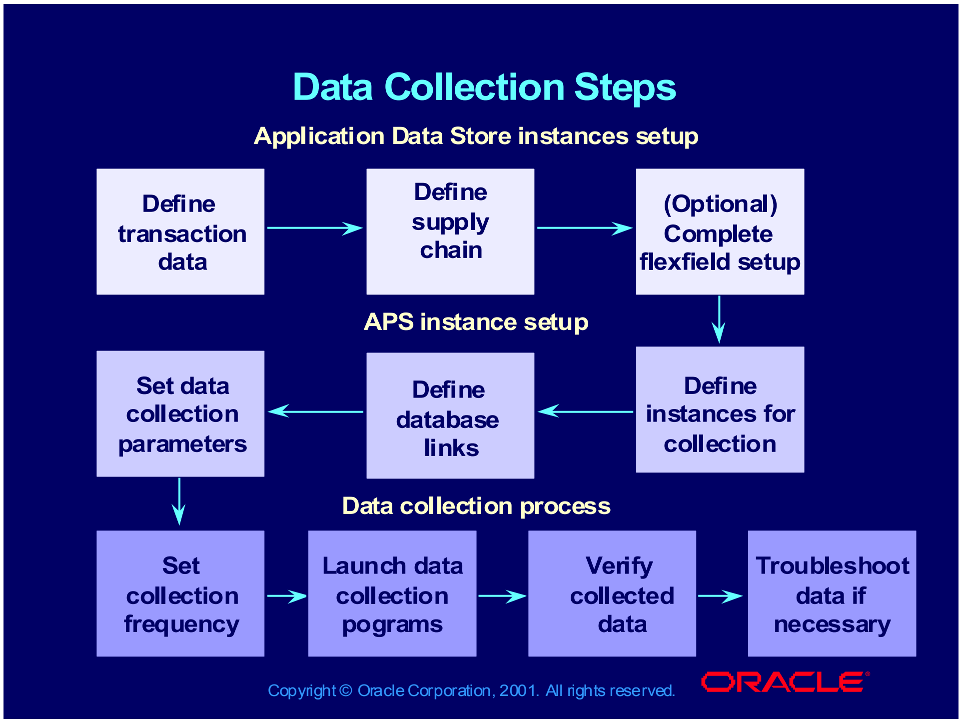
Workflow Integration

Workflow exception processes:

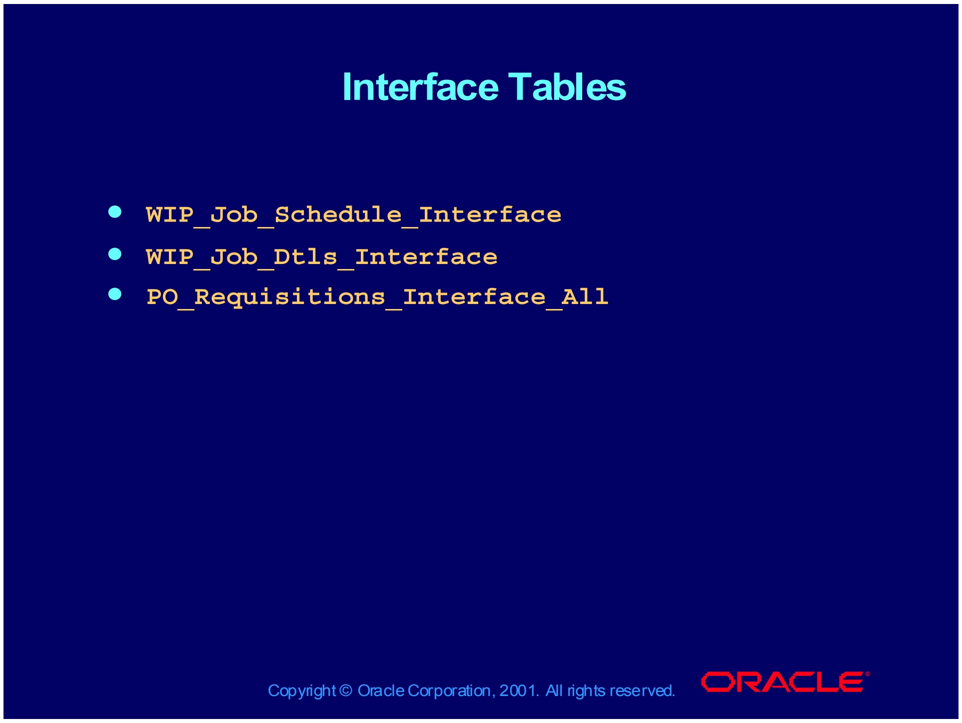
- **Process 1: Item Exception**
- **Process 2: Project Exception**
- **Process 3: Sales Order Exception**
- **Process 4: Supply Reschedule Exception**
- **Process 5: Supplier Capacity Constraint Exception**

ORACLE

Data Collection Steps



Interface Tables



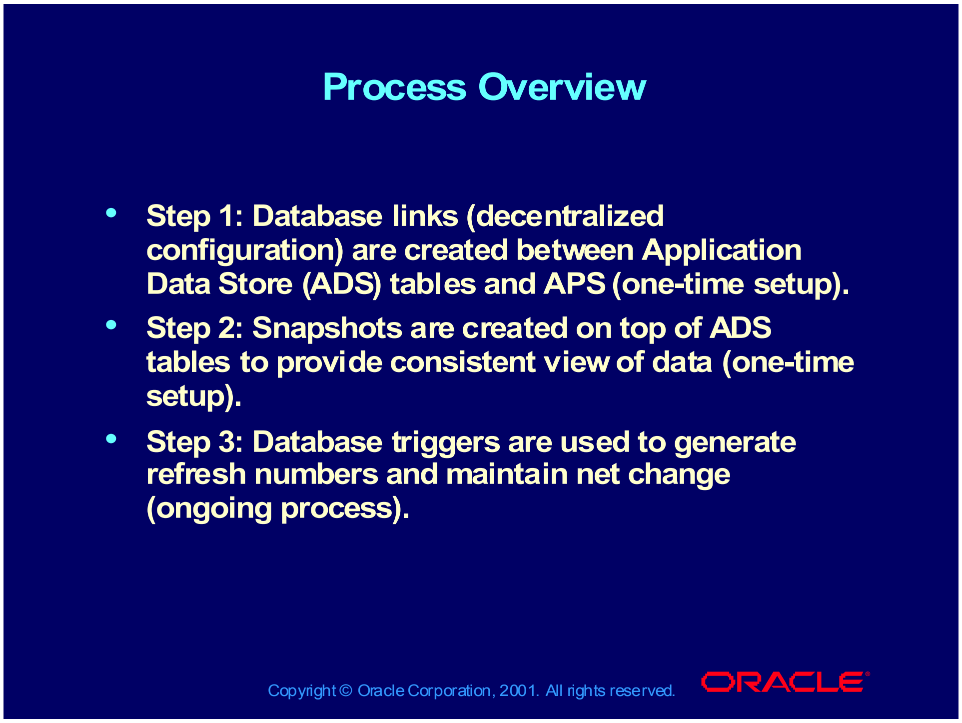
Interface Tables

- When releasing planned order recommendations are released to Purchasing, rows are inserted into the `PO_Requisitions_Interface_All` table residing in the Applications Data Store.
- When planned order recommendations are released to Work in Process, rows are inserted into the `WIP_Job_Schedule_Interface` table residing in the Applications Data Store.
- The `WIP_Jobs_Dtls_Interface` table contains the resource requirements and component requirements.
- There are no interface tables in APS.

Note: For the release of planned orders and purchase requisitions to succeed, the following setup must exist and be operational:

In the instance definition for the instance in question, there must be a valid database link in the Planning Database Link column, and this database link must be able to connect to the correct source instance.

Process Overview



Process Overview

A database link is a pointer that defines a one-way communication path from an Oracle database server to another database server. The link pointer is actually defined as an entry in a data dictionary table. A database link connection allows the local database to access the remote database in a decentralized configuration.

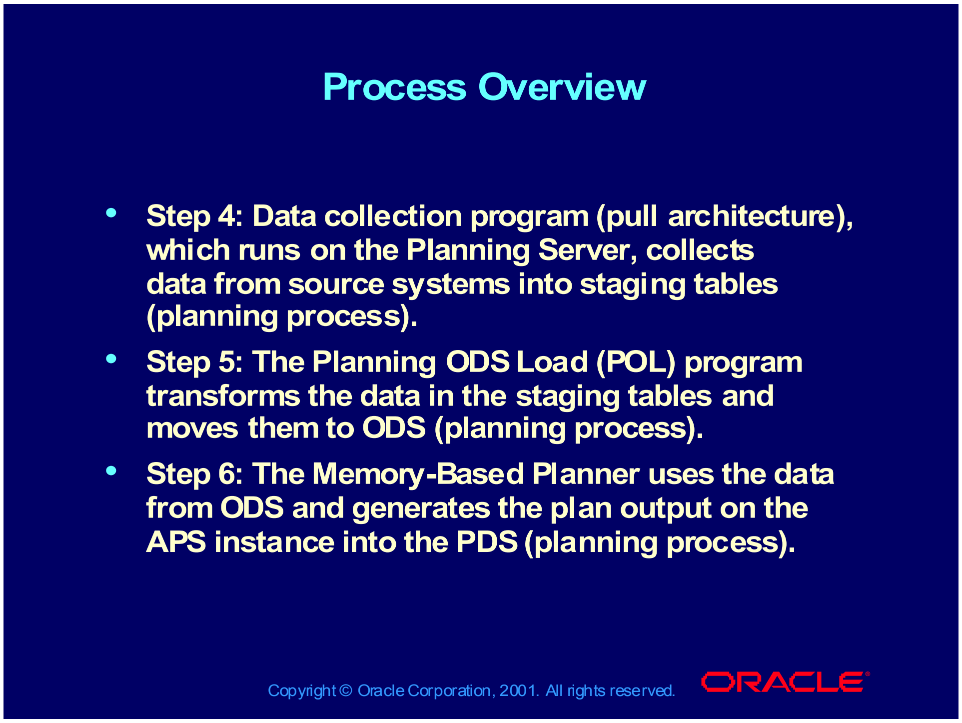
Step 1

On the source instance:

- The `MRP_AP_APPS_INSTANCES` tablecontains one database link that points to the destination APS server.
- `MRP_AP_xxx_V` and `MRP_AD_xxx_V` support the transformation of APS data to be used by the pull program.
- `MRP_SN_xx_TN` are triggers used to intercept deletions and store the relevant information regarding the deleting row’s primary key and refresh number to the table, `MRP_AD_xxx`.

On the destination instance:

- The `MSC_APPS_INSTANCES` table contains database link and instance information according to the instance ID.
- `MSC_ST_xx` tables store the data loaded by the pull program.



Process Overview (continued)

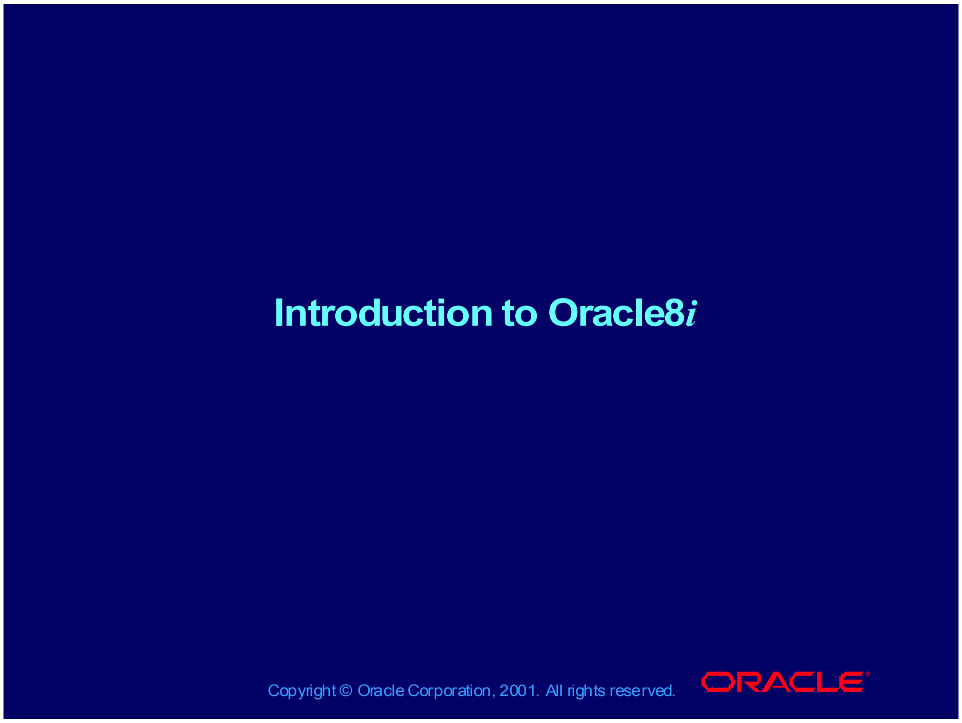
Step 4: Setup start date, end date, and refresh interval on Planning Server. Program name is `MSCPDP`

Step 5: Perform the ID transformation, if it is required, and load the data into the ODS. The executable concurrent programs are `MSCLMON` and `MSCLWOR` for the monitor and the worker.

Step 6: Oracle Advanced Supply Chain Planning: `MSCNEW`. Oracle Advanced Supply Chain with Constraint Option (includes license for ILOG components): `MSONEW`

Introduction to Oracle 8*i*

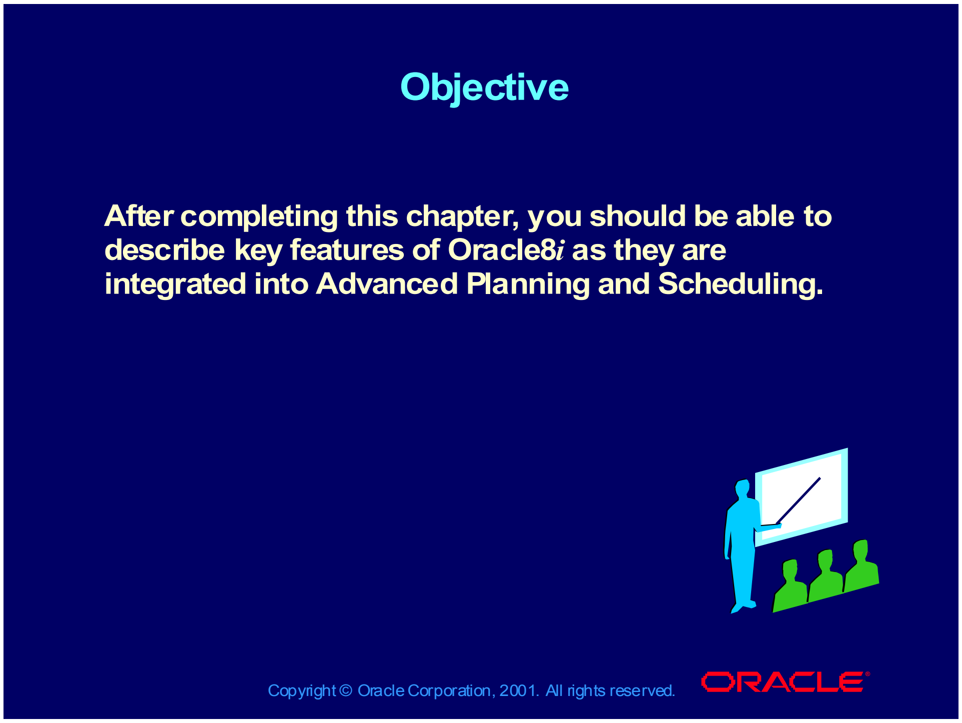
Chapter 3

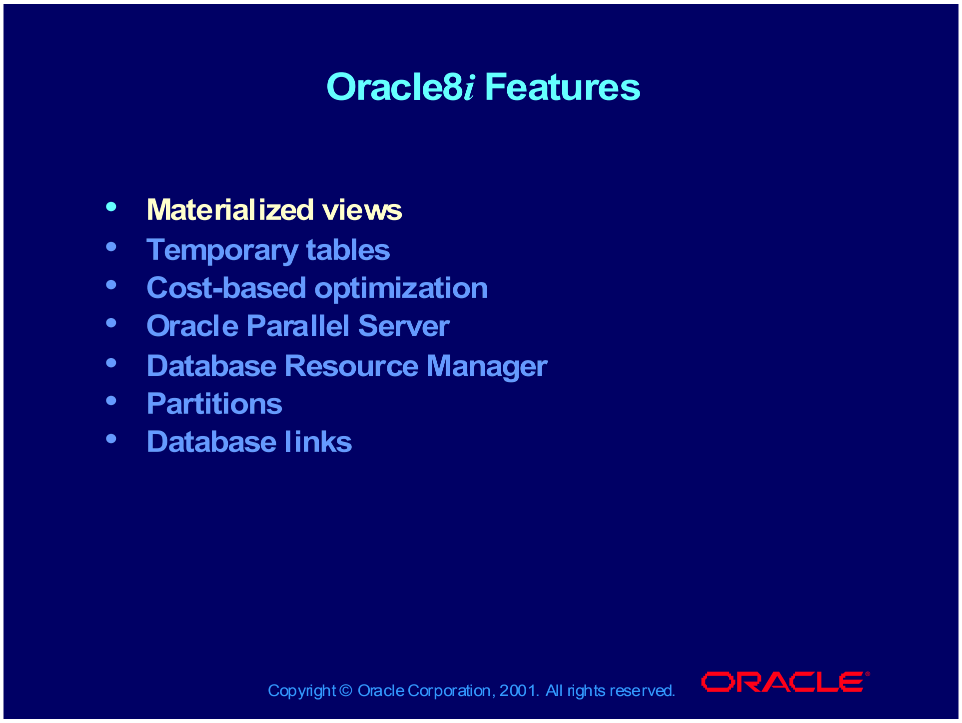

Introduction to Oracle8*i*
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Agenda

- Objective
- Oracle8*i* features

Objective


Objective
After completing this chapter, you should be able to describe key features of Oracle8*i* as they are integrated into Advanced Planning and Scheduling.
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE



Materialized Views

Materialized views are database objects that can be used to summarize, precompute, replicate, and distribute data. For example, materialized views can be used to precompute and store aggregated data such as sums and averages. When used in conjunction with the query-rewrite feature, materialized views can provide a very efficient method for query processing. Query-rewrite improves efficiency by retrieving the result from a materialized view instead of base tables mentioned in the SQL if the data in the materialized view is current.

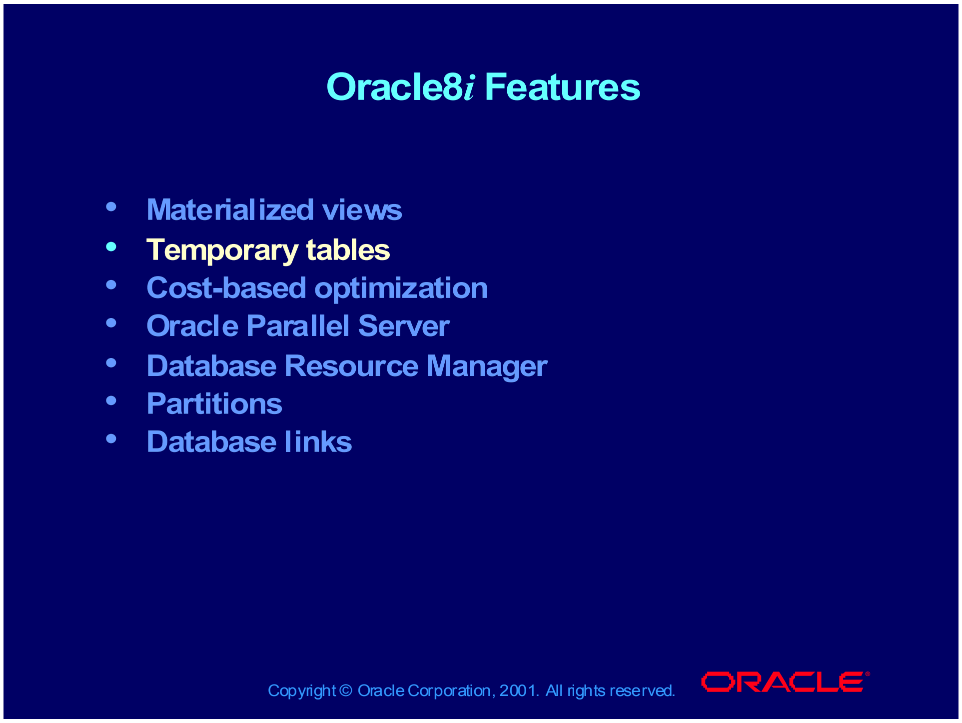
Cost-based optimization can make use of these materialized views to improve query performance by automatically recognizing when a materialized view can and should be used to perform a query. The cost-based optimizer automatically rewrites the SQL statement to make use of an existing materialized view.

Unlike regular views, materialized views store data and consume storage space and require frequent refreshing to capture the changes made to the master tables. Existence of materialized views can be transparent to the applications and end users or they can be directly accessed using the DML commands.

In a distributed environment, materialized views are also called snapshots. In the Oracle8*i* Server, materialized views are used to replicate data at distributed sites and to synchronize updates in a replicated database.

Both materialized views and function-based indexes are powerful because of the query optimizer’s ability to substitute precalculated values when executing a query against underlying tables.

Oracle8i Features



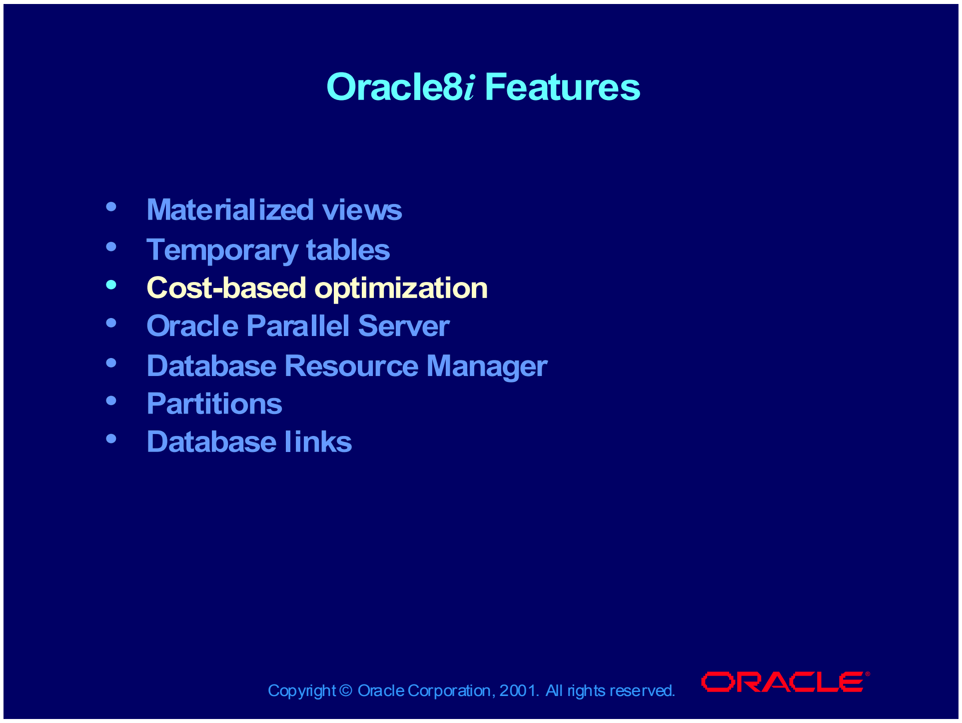
Temporary Tables

Temporary tables are a feature of the Oracle8*i* Server that is supported for release 11*i*. Temporary tables hold data that exists only for the duration of a transaction or session. Data in temporary tables is private to the session. Each session can see and modify only its own data.

The CREATE GLOBAL TEMPORARY TABLE statement creates a temporary table that can be transaction or session specific. For transaction-specific temporary tables, data exists for the duration of the transaction. For session-specific temporary tables, data exists for the duration of the session.

DML locks are not acquired on the data of the temporary tables. The LOCK statement has no effect on a temporary table, because each session has its own private data. Unlike permanent tables, SQL statements on temporary tables do not generate entries in redo logs during data manipulation.

Temporary tables provide a much cleaner way of handling temporary data requirements for many of the application product groups. Instead of creating permanent tables for short-term data requirements, they now can create temporary tables that can be dropped automatically once the transaction is completed or the session is ended. Since temporary tables are in-memory tables, any DML operation is very fast. Some of the release 11*i* product groups, such as General Ledger, and Inventory, use the temporary table feature.



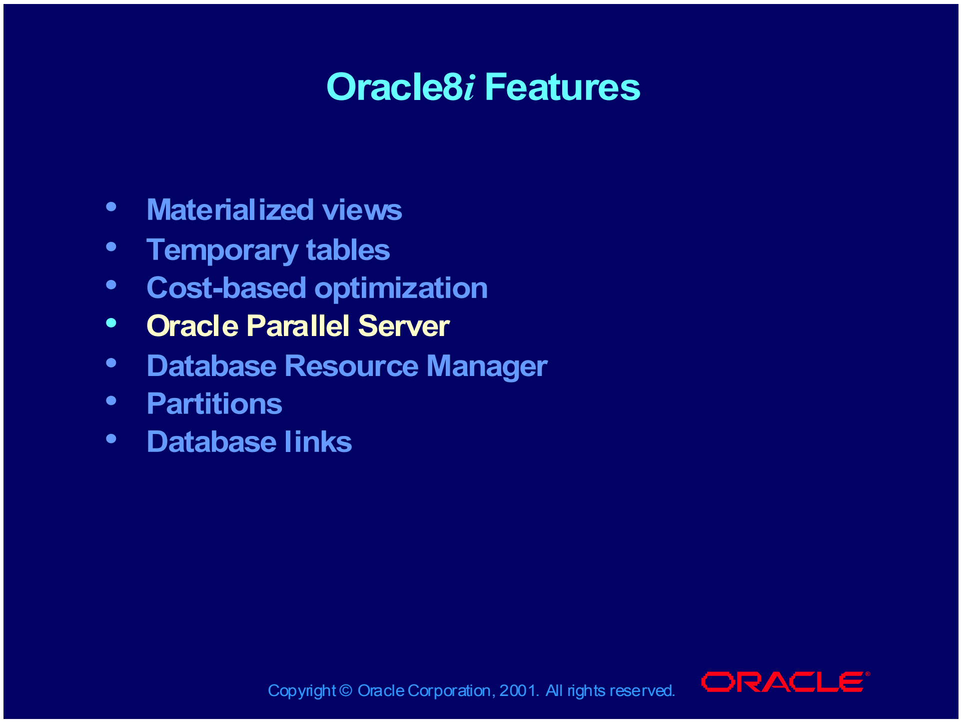
Cost-based Optimization (CBO)

Optimization is the process of choosing the most efficient way to execute a SQL statement. When a query is submitted for execution, Oracle's optimizer automatically calculates the most efficient way to execute the SQL statement based on either cost-based or rule-based optimization. Release 11*i* of Oracle Applications uses Cost Based optimization (CBO) instead of the rule-based optimization (RBO) technique used in earlier versions.

By default, the Oracle8*i* Server uses CBO for optimizing the SQL queries. Hence, the Applications 11*i* SQL statements were tuned to work with CBO. Release 11*i* no longer supports RBO.

The goal of the CBO approach is to achieve maximum throughput using minimal resources to process all data accessed by a SQL statement. In general, executing a SQL statement using CBO involves the following steps:

1. The optimizer generates a set of potential execution plans for the SQL statement based on its available access paths and hints.
2. The optimizer estimates the cost of each execution plan based on statistics in the data dictionary for the tables and indexes accessed by the SQL statement.
3. The optimizer compares the costs of the execution plans and chooses the one with the lowest cost.

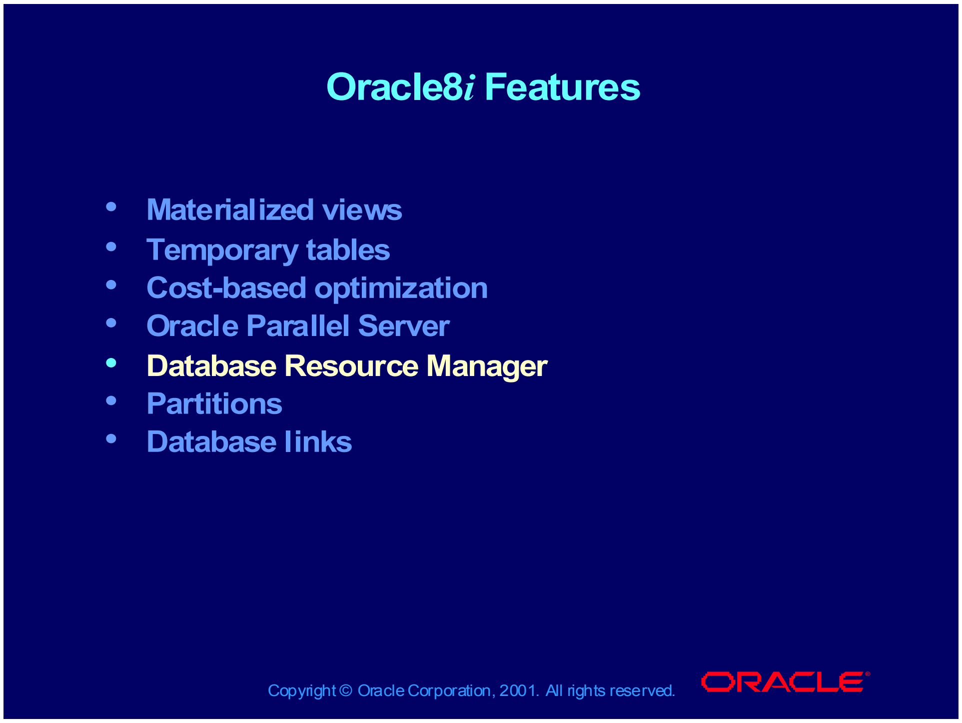


Oracle Parallel Server (OPS)

The OPS utilizes the processing power of multiple, interconnected computers all running Oracle8*i* and all connected to the same physical database. The OPS environment typically is deployed with a series of servers (nodes) that are put together to form a cluster.

Using the OPS, you can concurrently execute transactions against the same database from several nodes. Distributing database processes among multiple nodes results in faster and more efficient performance and provides application failover in case of failure on any node. The OPS automatically coordinates each node's access to shared data to provide data consistency and integrity.

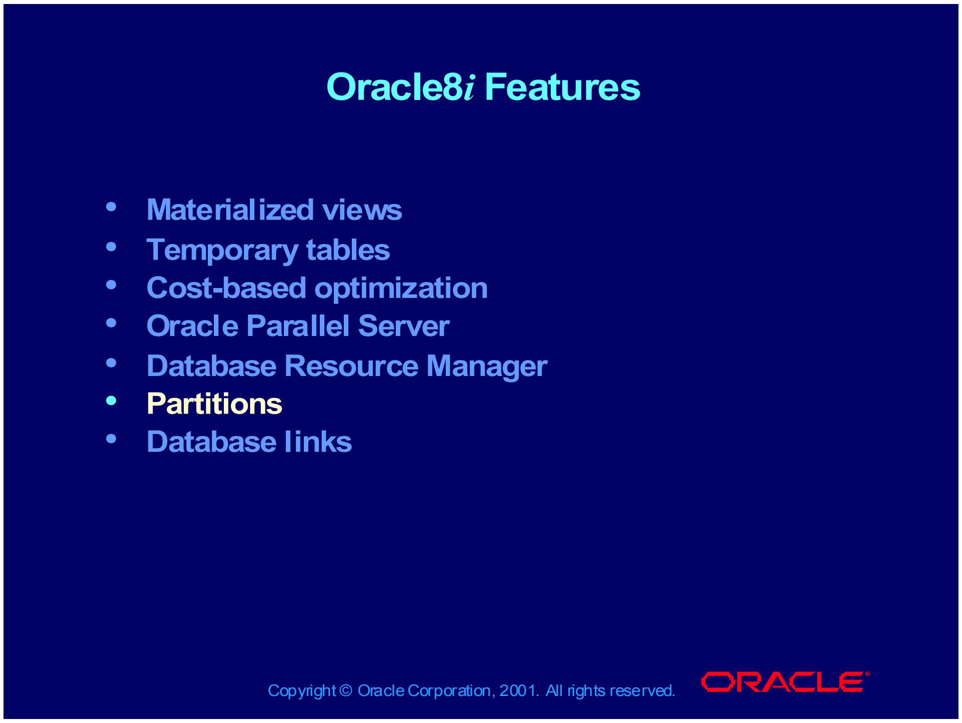
Although the OPS has been available with earlier releases of the Oracle database, it was not supported for use with earlier releases of Oracle Applications.



Database Resource Manager

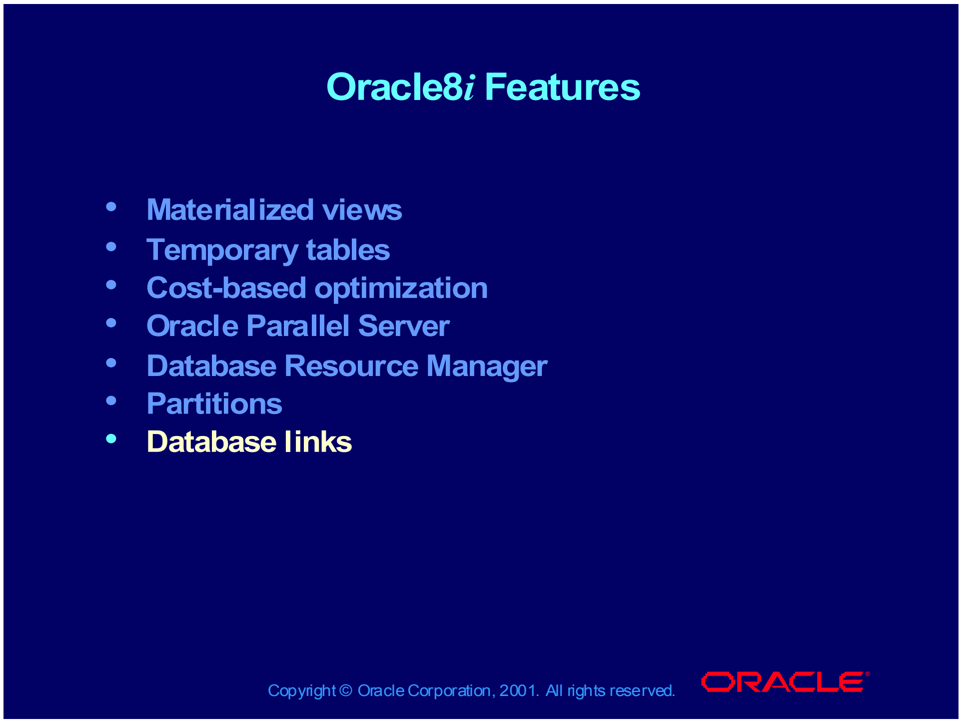
The Database Resource Manager is also a new feature of the Oracle8*i* Server. The database Resource Manager allows the database administrator to have more control over managing the resources than would normally be possible using the operating system resource management alone. Improved resource management enables better application performance and availability. Using the Database Resource Manager, the database administrator can:

- Guarantee certain groups of users a minimum amount of processing resources, regardless of the load or number of users in other groups on the system. For example, the DBA can guarantee resources to the users of a certain product group.
- Distribute available processing resources by allocating percentages of CPU time to different users and applications. For example, a higher priority may be given to GL batch processes.
- Limit the degree of parallelism that a set of users can use.
- Configure an instance to use a particular plan for allocating resources. A database administrator can dynamically change the plan, for example, from a daytime setup to a nighttime setup, without having to shut down and restart the instance.



Partitions

Partitioning is covered in a subsequent lesson.



Database Links

A database link is a schema object that causes ORACLE to connect to a remote database to an object there. A database link connection allows the local database to access the remote database in a distributed configuration. If an error or a conflict occurs during the process, error messages are generated in the log files. Database links are used in APS to collect data from the source instance and to push planning recommendations back to the source instance.

In other words, database links are used to define paths to related remote databases.

Distributed Configuration

- APS can run on a separate 11*i* instance or the same instance as OLATP.
- Data in the staging tables is striped by instance_id and organization_id.

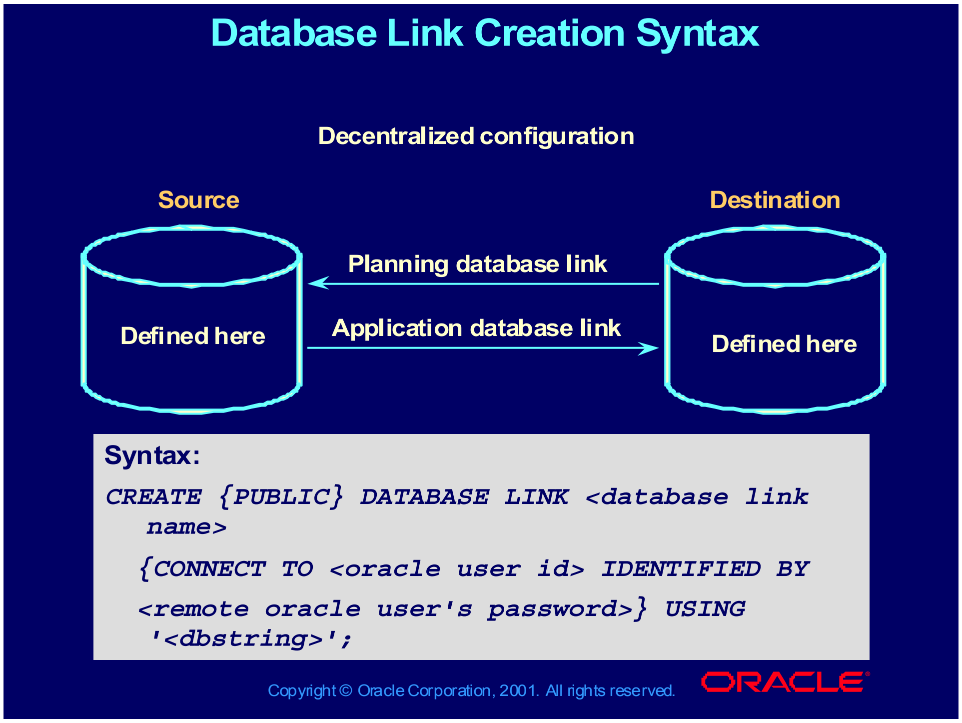
Centralized Configuration

Planning and OLTP take place in the same instance on the same server.

Additional References

- *Oracle Applications Concepts*, *Release 11i*, A82932-01
- *Oracle8i Concepts*, Volume 1, A67782-01
- *Oracle8i Concepts*, Volume 2, A67783-01

Database Link Creation Syntax



Database Links (continued)

The planning database link is defined by the database administrator on the APS planning (destination) instance. The application database link is defined by the database administrator on the transaction (source) instance. Both links are bidirectional.

The planning database link is used for data collection from the Application Data Store to the planning instance. When an action such as releasing a planned order or requisition occurs in the APS planning system, the data is “published” to the source instance. The first step in this process is to send a signal by way of the planning database link to the transaction instance. This initiates a remote procedure that pulls the planned order or requisition record from the planning instance to the transaction instance.

The application database link is used to complete the publishing process: the remote procedure that pulls the planned order or requisition record from the planning instance to the transaction instance does so using the application database link.

Example

A new Oracle Application Instance has been added to the corporate network. Link the new database to the Planning Server.

New database: mps07db1

Existing Planning Server database: mps07db3 using service name (alias) of "db3" and "db_name" of "mps07db3”

`Tnsnames.ora` entry:

```
db3=
(DESCRIPTION=(ADDRESS_LIST=  (ADDRESS= (PROTOCOL=TCP)
(HOST=mps07)  (PORT=1543) ) ) (CONNECT_DATA=
(SID=mps07db3) ) )
```

Create database link syntax:

```
CREATE [PUBLIC] DATABASE LINK mps07db3
CONNECT TO TC IDENTIFIED BY TIMEPRD
USING 'db3';
```

or

```
CREATE [PUBLIC] DATABASE LINK mps07db3
CONNECT TO TC IDENTIFIED BY TIMEPRD
USING 'db3.world';
```

Note: Omitting the keyword PUBLIC will automatically create private database links. APS requires public database links.

- Use the CONNECT TO option if you want to access the remote database with a different Oracle userid (that is, not the one currently logged in).
- <database link name> must correspond to the name of the database to which the database link refers if parameter is GLOBAL_NAMES=TRUE.
- <dbstring> is a valid SQL*Net connect string found in tnsnames.ora.

Test Example of Creating a Database Link:

Database global name: V734

tnsnames.ora alias: wally_V734.world

DBLINK TEST:

Local database: V804

Remote database: V734

First you need to create user XXX and grant privileges to XXX.

```
SVRMGR> create user XXX identified by XXX;
…..
Statement processed.
SVRMGR> connect XXX/XXX
Connected.
```

1. Connect to the V734 database and select from global_name to find out global_name and show parameter global to find out if global_names is set to TRUE.

```
SVRMGR> connect XXX
Password:
Connected.
SVRMGR> select * from global_name;
GLOBAL_NAME
-----------------------------------------------------
-------------------
V734.WORLD
1 row selected.

SVRMGR> show parameter global
NAME          TYPE       VALUE
-----------   -------    ----------------------------
global_names  boolean    TRUE
```

2. Find the `tnsnames.ora` file, which will show the service name (alias) for databases. For V734 you can see that the alias is wally_V734.world. `Tnsnanes.ora` for tiger 5 is located in the following directory:

```
 /u01/app/oracle/product/8.0.4/network/admin
View "tnsnames.ora" file:
….
wally_V734.world =
  (DESCRIPTION = (ADDRESS = (PROTOCOL= TCP)(Host=
138.1.102.165)
  (Port= 1521)) (CONNECT_DATA = (SID = V734))
….
```

3. If you create a database link named SHIRLEY and using alias from `tnsnames.ora`, you will receive an ORA-2085 error.

```
SVRMGR> connect XXX/XXX
Connected.
SVRMGR> CREATE [PUBLIC] DATABASE LINK SHIRLEY
     2> CONNECT TO XXX identified by XXX
     3> USING 'wally_V734.world';
Statement processed.

SVRMGR> SELECT * FROM GLOBAL_NAMES@SHIRLEY;
SELECT * FROM GLOBAL_NAME@SHIRLEY
                           *
```

```
ORA-02085: database link SHIRLEY.WORLD connects to
V734.WORLD
```

Cause: The database link attempted to connect to a database with a different name. The name of the database link must be the same as the name of the database.

Action: Create a database link with the same name as the database to which it connects.

See <Note:19367.1> OERR: ORA 2085 "database link %s connects to %s" for further explanation of the error and workaround involving `init<SID>.ora` parameter `GLOBAL_NAMES`.

4. Create another dblink naming the link the same as database name (V734) and make sure that you are using correct alias per tnsnames.ora (`'wally_V734.world'`).

```
SVRMGR> CREATE [PUBLIC] DATABASE LINK V734
     2> CONNECT TO XXX identified by XXX
     3> USING 'wally_V734.world';
Statement processed.
```

5. Select across DBLINK to V734 database is successful.

```
SVRMGR> select * from global_name@V734;
GLOBAL_NAME
--------------------------------------------------
V734.WORLD
1 row selected.
```

Solution Explanation: After you have a service name (alias) and a database name from tnsnames.ora, you will be able to successfully create your database link and do a select across the database link.

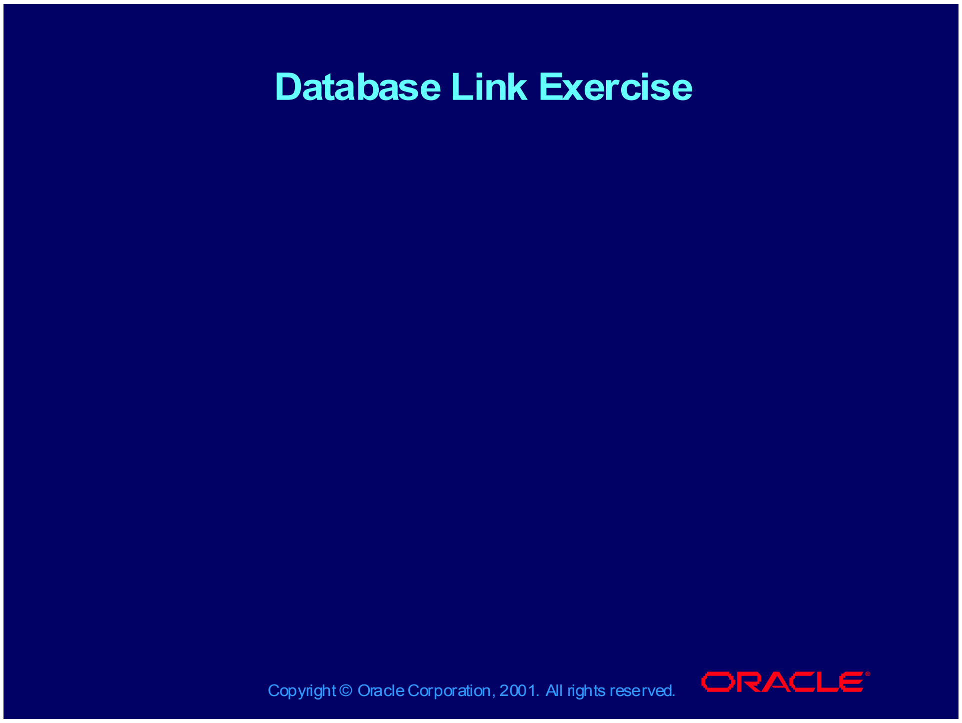
Attention Oracle8 Users: Note that Net8 also allows you to create a working database link when <dbstring> is not an alias defined in the `tnsnames.ora` file but a fully qualified connect string as in the following example:

```
create database link V805.NL.ORACLE.COM
connect to repadmin identified by repadmin using
'(description=(address=(protocol=tcp)(host=nldock0592
.nl.oracle.com)  (Port = 1521) ) (connect_data=
(sid=v805)))';
```

This can be very useful when the server side tnsnames.ora file does not have the same connect strings for database services as your local tnsnames.ora.

For example, if you connect scott/tiger@alias but a database link using 'alias' does not work, you can use the complete string as in the example.

Database Link Exercise



Database Link Exercise

1. Login to SQL*Plus on the SOURCE instance.
2. Verify that the database link pointing to the APS instance exists.

```
select  substr(M2A_DBLINK,1,20) dest_to_source,
            substr(A2M_DBLINK,1,20) source_to_dest
 from    mrp_ap_apps_instances;

 DEST_TO_SOURCE                SOURCE_TO_DEST
 --------------                ---------------
 to_dest_link
```

3. Now, we will issue a select, that will be used later to verify that our database link that points to the source is operational.

```
select  count(*)
 from    MFG_LOOKUPS;

 COUNT(*)
 --------------
           3550
```

4. Test the database link from the source to the destination.

```
select  count(*)
 from   msc_demands@SOURCE_TO_MSC db link name here;

 COUNT(*)
 -------------
            483
```

5. Login to SQL*Plus on the destination instance.

6. Determine what database links are set up at the destination instance.

```
select substr(M2A_DBLINK,1,20) DEST_TO_SOURCE,
            substr(A2M_DBLINK,1,20) SOURCE_TO_DEST
 from msc_apps_instances;

 DEST_TO_SOURCE             SOURCE_TO_DEST
 --------------             ---------------
 to_source_link
```

7. Test the database link back to the source. Does the count here equal the select from mfg_lookups, performed at the source, as returned above?

```
select  count(*)
 from    mfg_lookups@DEST_TO_SOURCE db link name
here;
```

APS Planning Data Pull and Data Collections

Chapter 4

APS Planning Data Pull and Data Collections

ORACLE

Agenda

Agenda

- Objectives
- Planning Data Pull (PDP) program
- PDP execution
- PDP report set
- PDP execution flow
- *_APPS_INSTANCES tables
- PDP performance
- PDP diagnostics
- PDP failure
- Staging tables

Agenda

Agenda

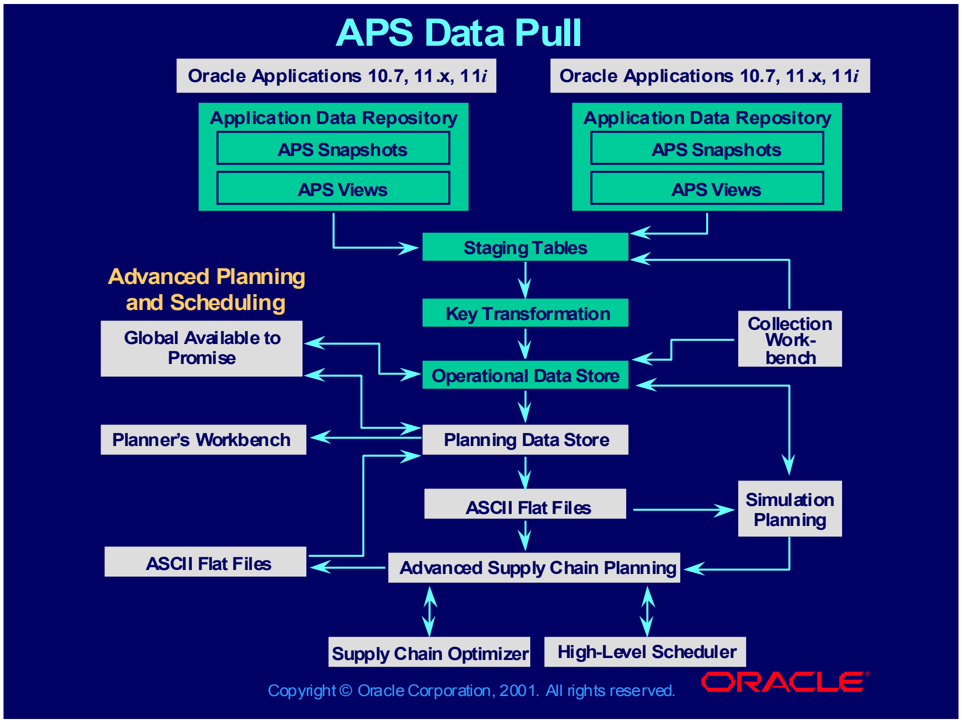
- PDP purging staging tables
- PDP capturing incremental changes
- Snapshots
- PDP complete refresh
- PDP incremental (fast) refresh
- PDP exercise

Objectives

After completing this chapter, should be able to do the following:

- Describe the data flow of the Planning Data Pull
- Identify the programs involved
- Describe the Planning Data Pull process

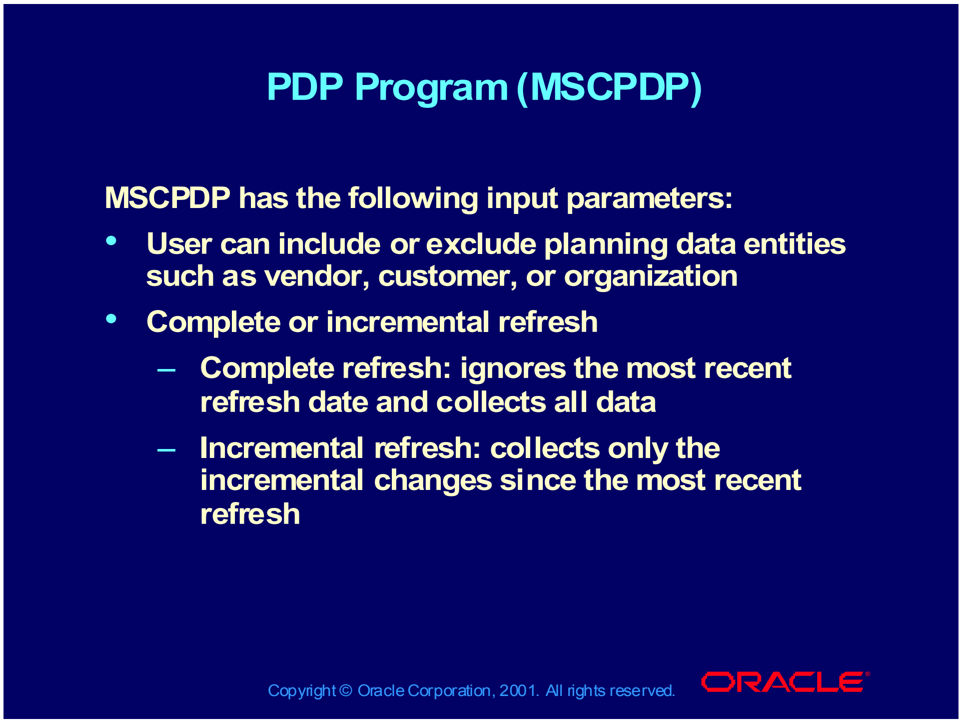
APS Data Pull



Introduction

- The Planning Data Pull (PDP) process is the act of pulling data from source Application Data Store database objects.
- The PDP process is flexible and can be scheduled for an instance with different parameter values.
- The PDP process can be performed to match business operations. If suppliers are updated every two weeks, pull the supplier data entity every two weeks.
- Data collection types:
  - Data Collection Complete, or complete refresh is required after certain data changes. This will be covered in the PDP lesson.
  - Data Collection Incremental can be scheduled when desired, keeping planning data current.
  - Data Collection from legacy system is performed using the ST tables.
- Staging tables act as temporary storage for data destined for the Operational Data Store (ODS). It is here that a user-supplied data-cleansing tool can be used to scrub incoming data.
- As this data is loaded into the ODS, Key Transformation takes place. Data is extracted from the staging tables and the primary keys are transformed before the data is loaded into the Operational Data Store (ODS).

- The ODS acts as a temporary area, used for storage of data until needed in the Advanced Supply Chain Planning engine stage.



MSCPDP Program

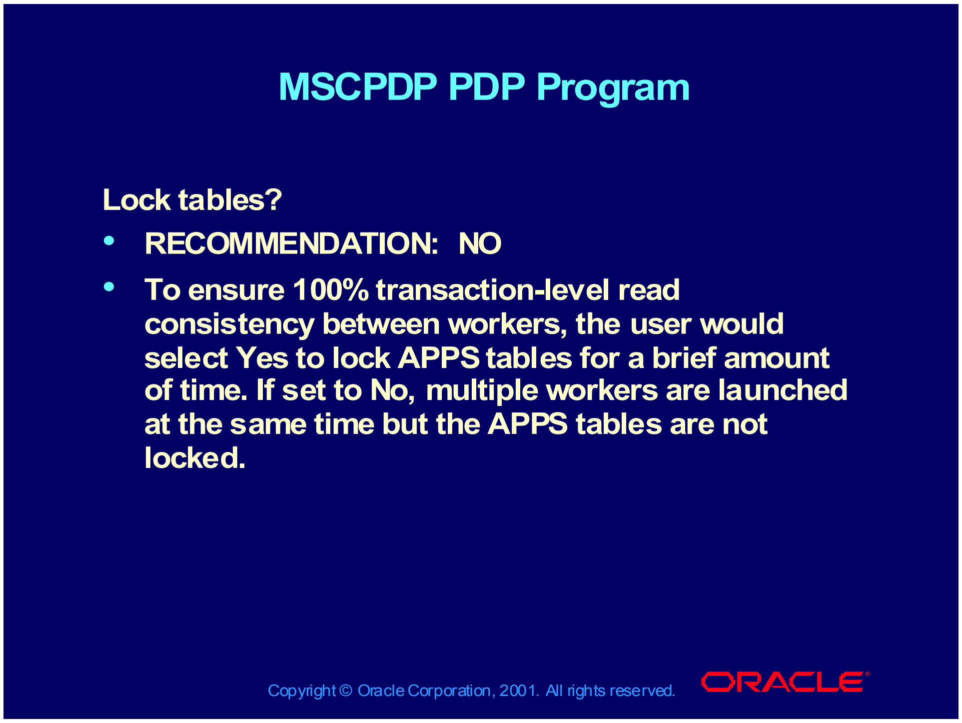
The following data entities can be selected from the collection program:

Pull Items	Pull Suppliers	Pull Customers
Pull BOM/Routing	Pull Reservations	Pull Sourcing Rules
Pull Work in Process	Pull Safety Stock	Pull Purchasing Supply
Pull On Hand	Pull UOM	Pull MDS
Pull MPS	Recalculate NRA	Recalculate Sourcing History

Complete refresh:

- The data in the ODS for the collected instance has `plan_id = -1`.
- If there is no corresponding record in the staging tables, the original data is flagged as deleted.
- In the complete refresh process, the original ODS data for this instance will be updated according to the data in the staging tables.
- Categories and buyer contacts are pulled along with items when Pull Items is enabled.
- If Complete Refresh = YES, all other parameters will be set to Yes as well.

MSCPDP PDP Program



MSCPDP Program (continued)

For ORA-00060 and ORA-00054 database messages, perform the following:

- Check your alert log for ORA-00060 messages.
- Using PL/SQL lock timer scripts, monitor the MSCPDP process as it is running.

Lock tables to ensure data consistency.

PDP Execution

Data collection is a two-step process:

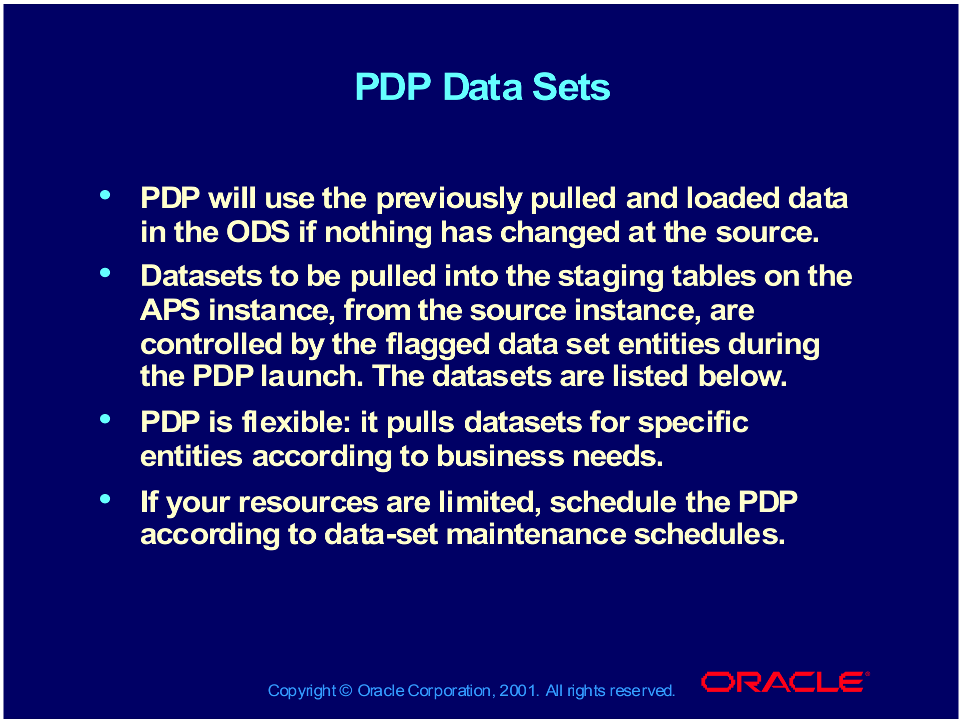
1. Pull data from `MRP_AP_xxx_V` and `MRP_AD_xxx_V` database views on the source
2. Move the planning data to staging tables on the APS instance.
   – Data collection requests are launched on the Planning (APS) instance.
   – The PDP concurrent request on the APS instance launches the PDP monitor process.

PDP Execution

The data collection process is required both in the decentralized and centralized configurations.

If you are running a two-step collection, you must run the Purge Staging Tables concurrent program

PDP Data Sets



PDP Data Sets

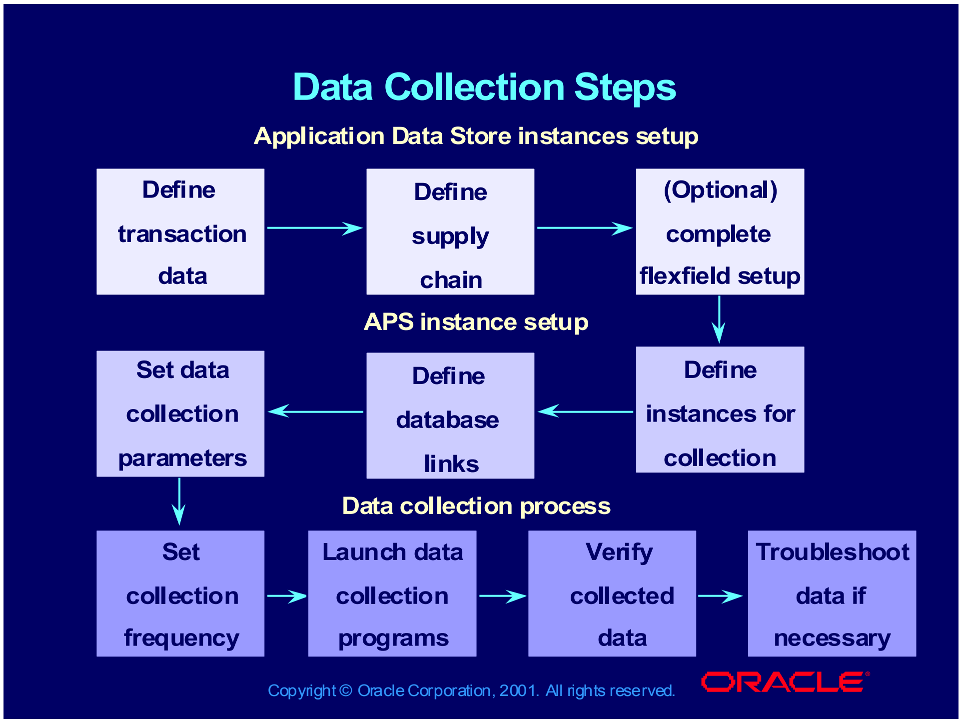
Data collection requests can be scheduled for an instance with different parameter values. For example:

- If sourcing rules are added or changed once a month, then schedule (as part of the planning or ATP data collection request) to pull these entities just once a month.
- If new suppliers or customers are added every two weeks, then pull these entities once every two weeks.

Multiple data collection requests can be scheduled to run against an instance with different parameters to run during different times. For example, the first Weekly Data Collection in the morning at 6 a.m. should pull all entities but subsequent data collections should pull data based on specified needs.

PDP Data Sets:

Pull Items	Pull Suppliers	Pull Customers
Pull BOM/Routing	Pull Reservations	Pull Sourcing Rules
Pull Work in Process	Pull Safety Stock	Pull Purchasing Supply
Pull On Hand	Pull UOM	Pull MDS
Pull MPS	Recalculate NRA	Recalculate Sourcing History



APS (Destination) Data

The Oracle APS setup for data collection involves three steps:

- Define instances.
- Establish database links.
- Specify parameters for the data to be collected from each instance for each named request set.

Data collection process:

- Different request sets can be run as soon as possible, at a scheduled time, at periodic intervals, or on specific days.
- Launch data collection programs across specified source instances: 10.7, 11.0, 11*i*, OPM.
- Use the Collection Workbench to verify that the intended data has been collected.
- If necessary, troubleshoot errors in data collection, and rerun the data collection program.

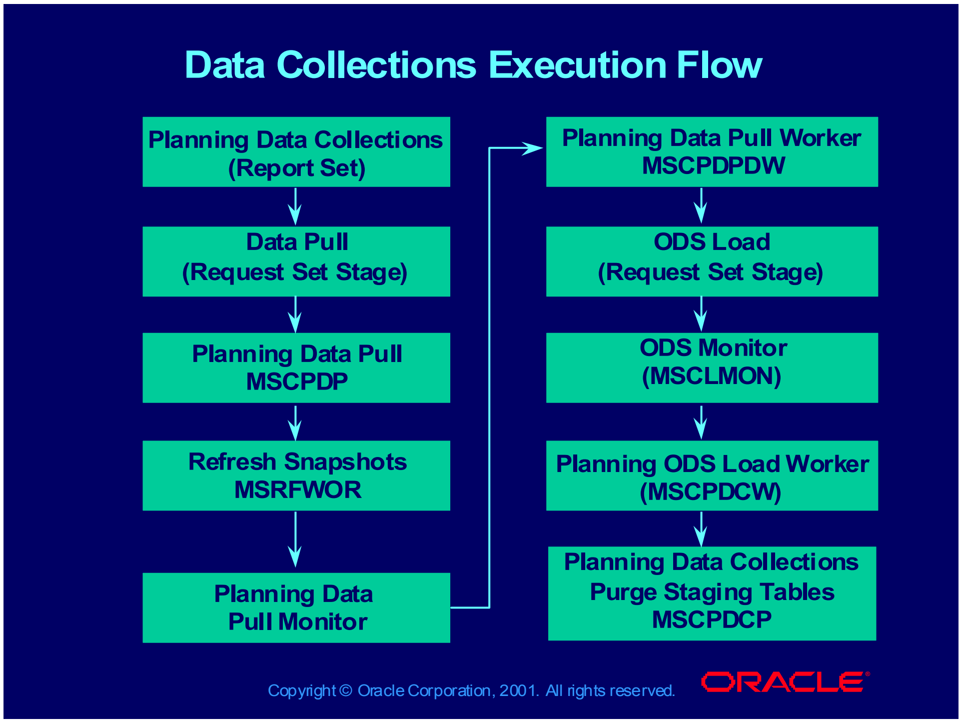
Planning Data Collection Report Set

The Planning Data Collection (Planning/ATP) report set includes two concurrent jobs that run sequentially:

- MSCPDP: Planning Data Pull
- MSCPDPW: Planning Data Pull Worker
  - Controlled by MSCPDP
  - Input parameter is number of workers (default is 1 if stand-alone, 2 if launched within request set)

Planning Data Collection Report Set

- Only one data collection request should be launched against a unique instance at any point of time.
- Multiple data collection requests against different instances can run at the same time.



Data Collections Execution Flow

- The Refresh Snapshot program, MSRFWOR, runs on the Application Data Store.
- The number of pull workers and snapshot workers is controlled by setting the number of workers in the Planning Collections data entry form.
- These processes use the standard manager.
- It is advised that you set the number of workers, in the standard manager, to 30 for the purposes of APS.

PDP Execution Flow

1. Launch the workers and monitor the process. The PDP program coordinates the source instance snapshot workers and monitors the executed tasks through the data collection process. The PDP monitor launches multiple PDP workers to load data into the staging tables on the APS instance.

ORACLE

PDP Execution Flow

The pull program improves performance by identifying and distributing data transformation and collection tasks that could be done in parallel. The source Refresh Snapshots and the APS Staging Tables Purge processes operate in parallel during the PDP.

To determine if PDP is executing at the planning instance:

- If your operating system is UNIX, at command prompt enter:

  `ps -ef | grep MSCPDP`

- If your operating system is NT, perform the following:

  Navigate to the Task Manager and look for MSCPDP.

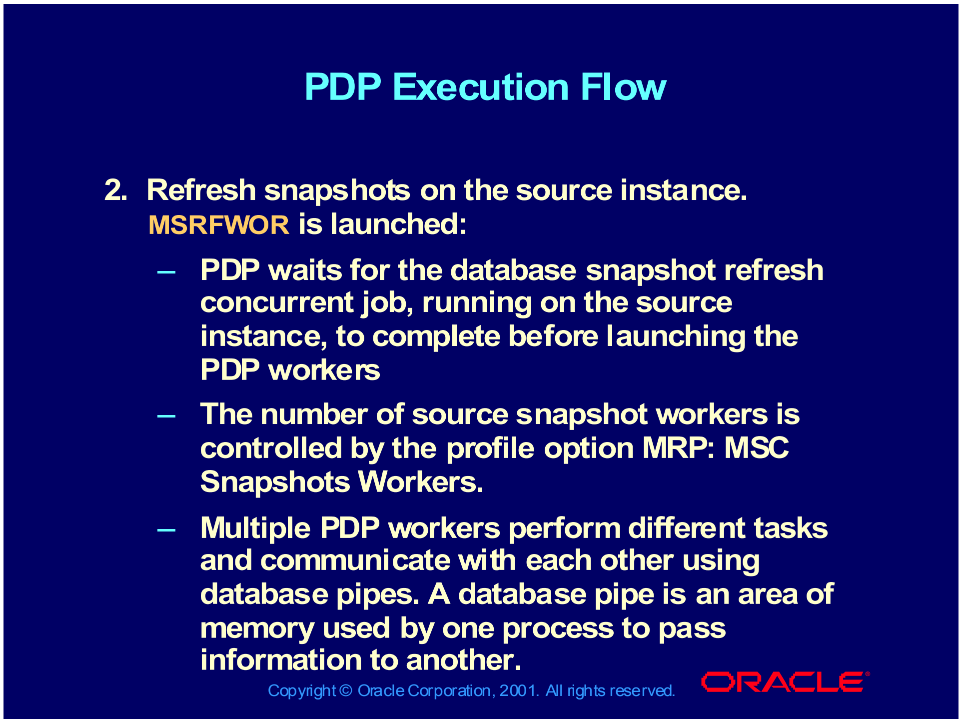
To determine if the snapshot is running on the source instance, execute the following:

- If your operating system is UNIX, at command prompt, enter:

  `ps -ef | grep MSRFWOR`

- If your operating system is NT, perform the following:

  Navigate to the Task Manager and look for MSRFWOR.



MSRFWOR Concurrent Program

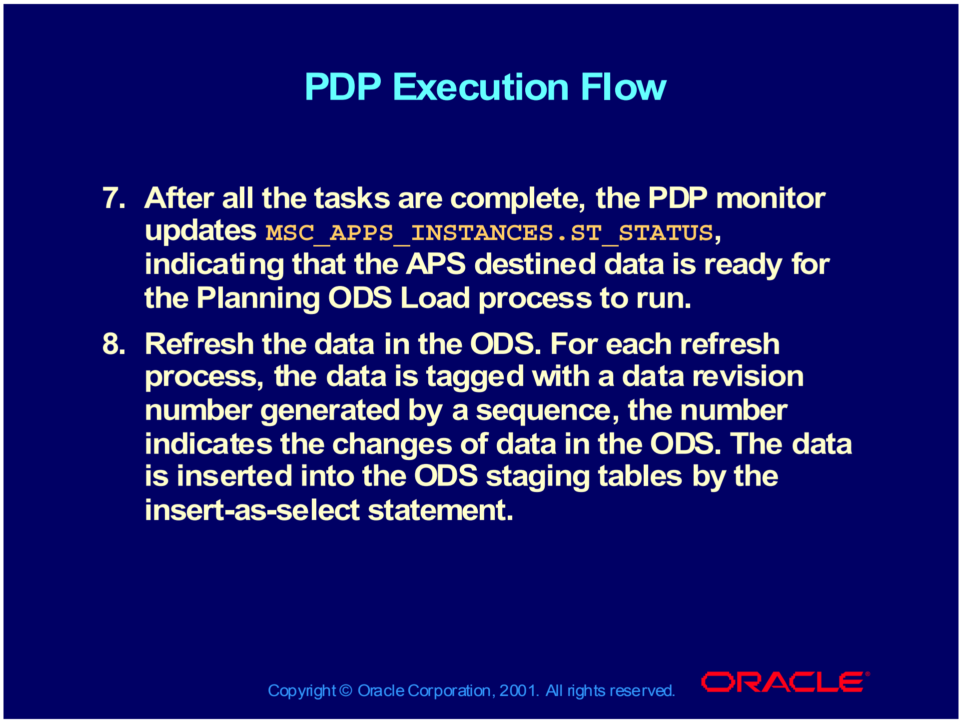
The executable concurrent programs are MSCLMON and MSRFWOR for the monitor and the worker.

PDP Execution Flow

1. Run Staging Table Purges (MSCPDCP) for the Instance ID on the APS instance.
2. The PDP workers will pull the Snapshot data into the APS instance (Planning Server).
   – The number of pull workers is controlled by the "No. of workers" parameter when the PDP is launched from data collection process.
   – Move data from ADS views (`MRP_AP_xxx_V` and `MRP_AD_xxx_V`) on the source instance into staging tables on the APS instance.

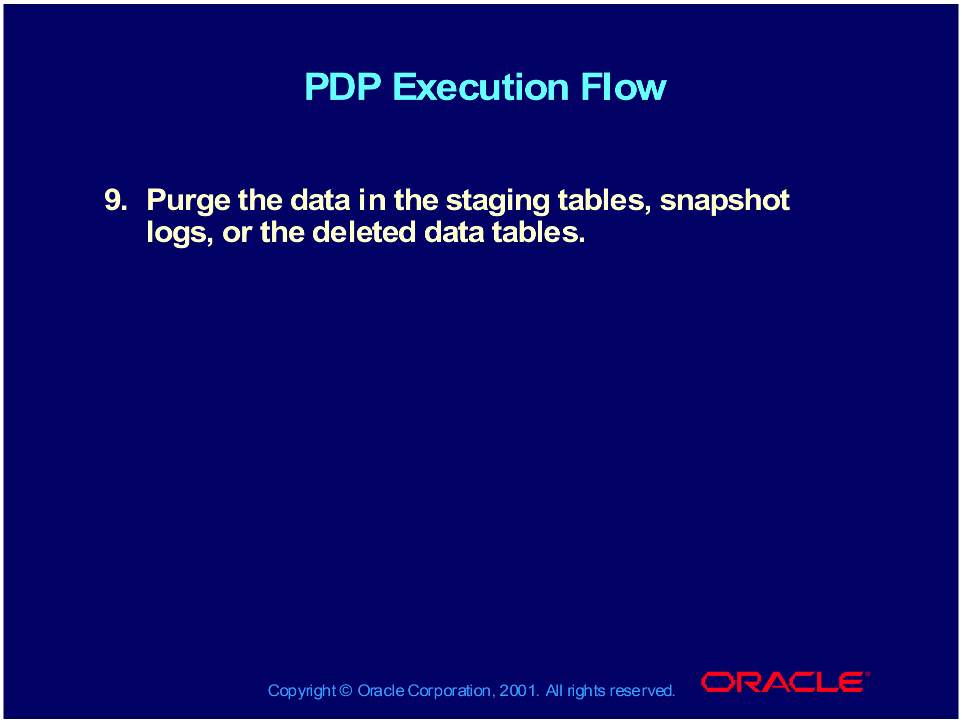
PDP Execution Flow

5. Launch optional data-cleansing tools provided by users. The pull program will launch the data cleansing program provided by the user
6. Launch the PL/SQL program on the Planning Server for ID transformation and ODS load. Process the transformed data according its status insert, update, or delete.



Staging tables on the APS instance have no integrity constraints; hence the data inserts are faster.

PDP Execution Flow

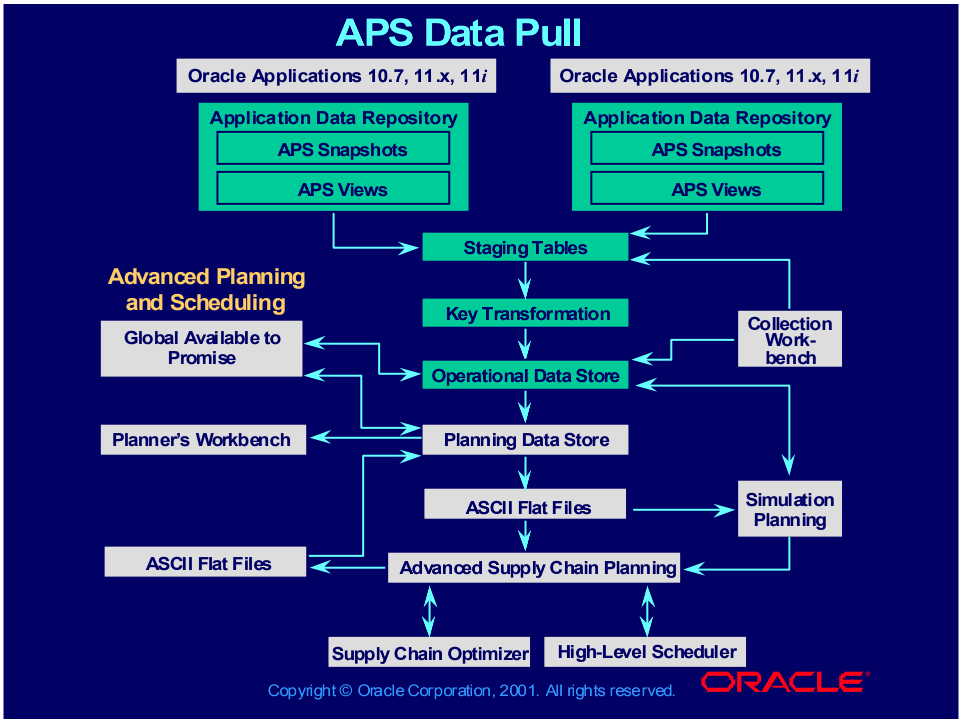


PDP Completion

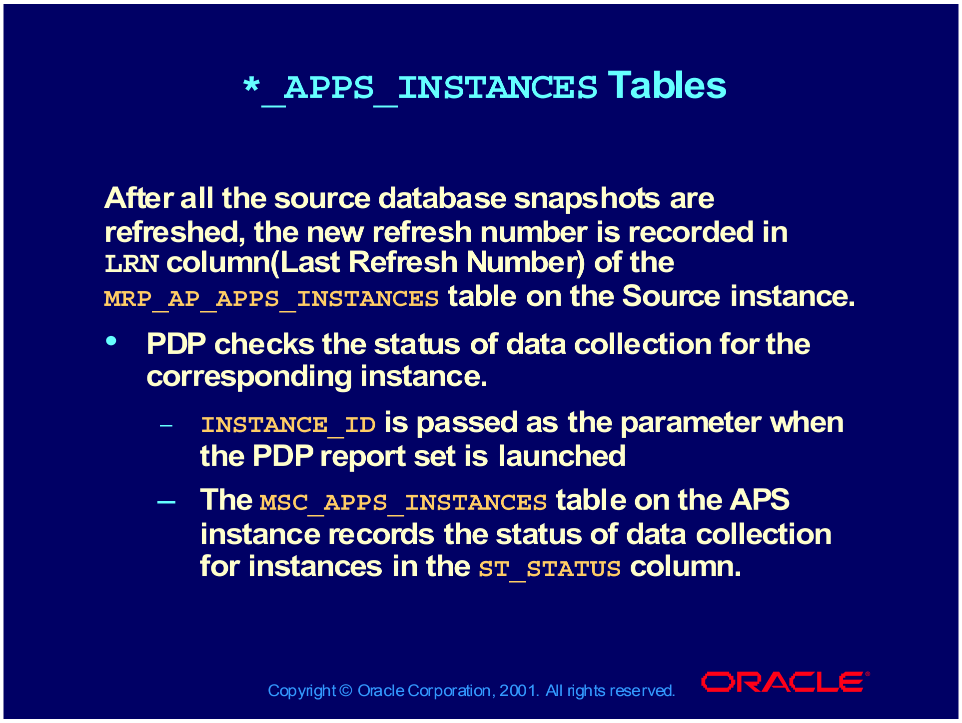
After all the tasks are completed, the PDP monitor stamps the Last Refresh Number, Refresh Type (Complete/Incremental) and the Last Refresh Id in table `MSC_APPS_INSTANCES` on the APS instance for the corresponding instance code.

Column `ST_STATUS` will be updated to 2 to reflect that it is ready for the Planning ODS Load process to run.

APS Data Pull



*_APPS_INSTANCES Tables



MRP_AP_APPS_INSTANCES Table

MRP_AP_APPS_INSTANCES resides on the data source instance and contains only one row, which stores the following details:

INSTANCE_ID

LRN (Last Refresh Number)

LRD (Last Refresh Date)

INSTANCE_CODE

SN_STATUS (Snapshot Status)

LAST_UPDATE_DATE

LAST_UPDATED_BY

CREATION_DATE

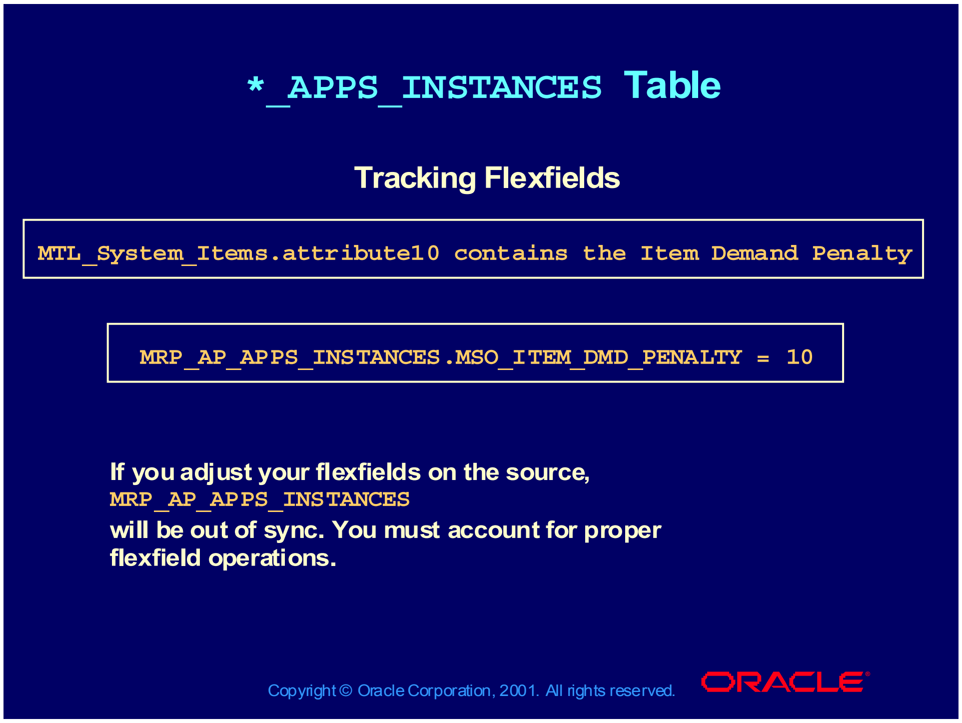
CREATED_BY

LAST_UPDATE_LOGIN

M2A_DBLINK (Database link from MSC (APS instance) to Apps (source))

A2M_DBLINK (Database link from Apps (source) to MSC (APS instance))

*_APPS_INSTANCES Table



MRP_AP_APPS_INSTANCES Table

The following columns in MRP_AP_APPS_INSTANCES are updated with the attribute values defined for the corresponding profiles on the source instance by the PDP process:

BOM_HOUR_UOM_CODE	MRP_MPS_CONSUMPTION
MRP_SHIP_ARRIVE_FLAG	CRP_SPREAD_LOAD NUMBER
MSO_ITEM_DMD_PENALTY	MSO_ITEM_CAP_PENALTY
MSO_ORG_DMD_PENALTY	MSC_ALT_OP_RES
MSC_ALT_RES_PRIORITY	MSC_SIMUL_RES_SEQ
MRP_BIS_AV_DISCOUNT	MRP_BIS_PRICE_LIST
MSC_DMD_PRIORITY_FLEX_NUM	

*_APPS_INSTANCES Table

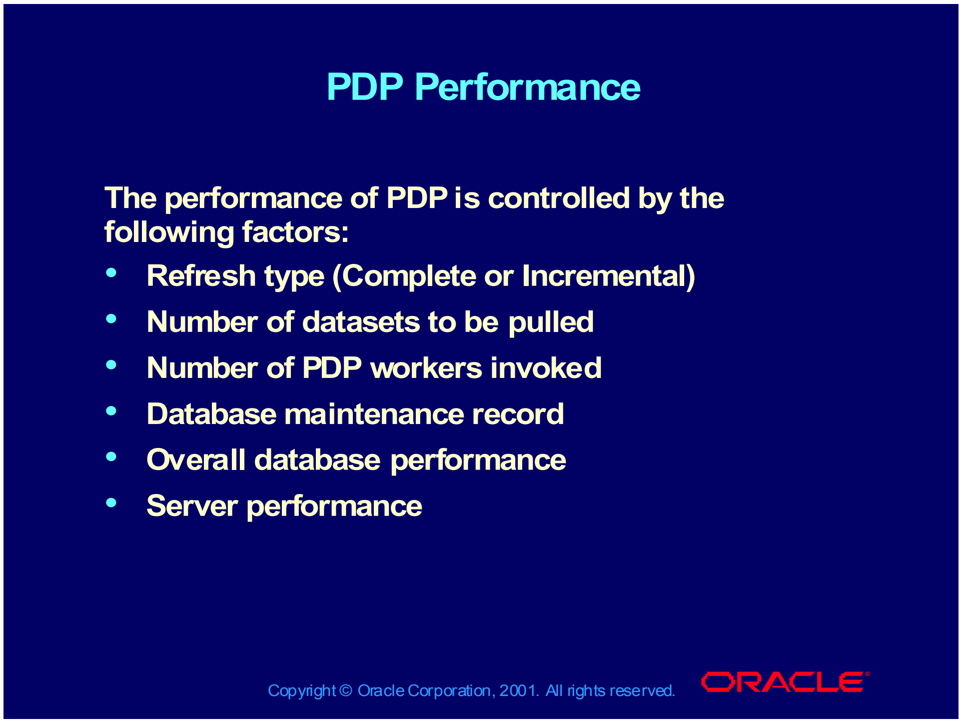
*_APPS_INSTANCES Table

MSC_APPS_INSTANCES.ST_STATUS:

- (0) Staging tables are empty for the particular instance. No instance data exists.
- (1) Planning data is being pulled from the source snapshots (complete refresh or Net Change) by the PDP process.
- (2) Data has already been pulled from the source snapshots and is ready to be collected by the Planning ODS Load process.
- (3) Data is being transformed from the staging tables and loaded to the Operational Data Store by the Planning ODS Load process.
- (4) Data is being purged from the staging tables.

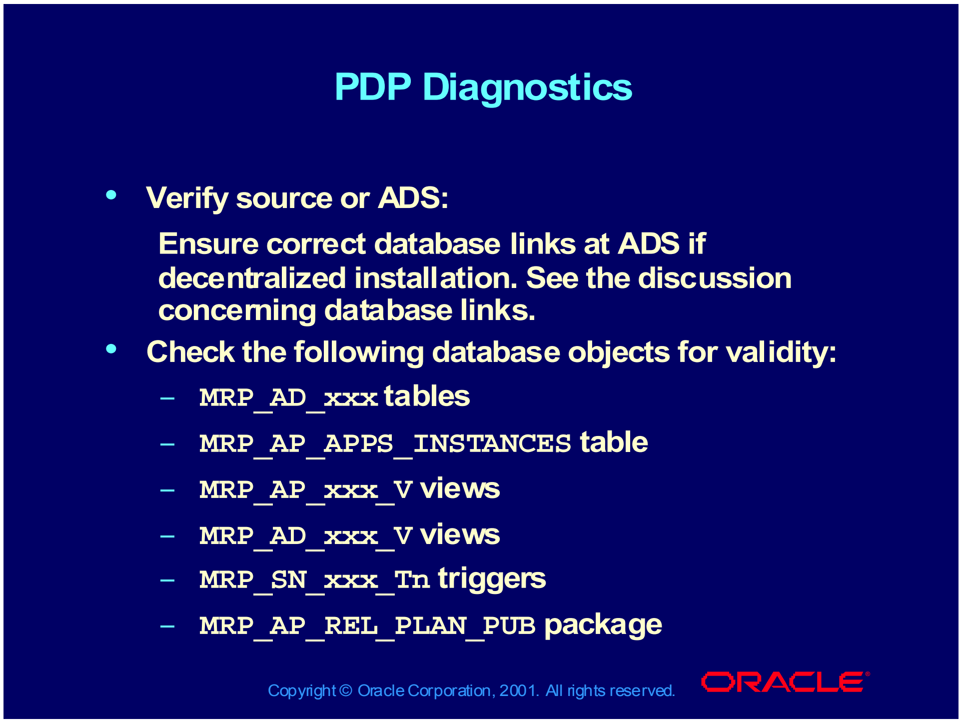
PDP Status

If your PDP process is not functioning properly, you can check the PDP status of the instance. A following exercise will demonstrate this query.



PDP Performance

Your overall performance methodology would address most of the above. If you need assistance developing a performance methodology, see the Enterprisewide Performance Guide at `http://metalink.oracle.com`



PDP Diagnostics

- Refer to the database link topic in the Oracle 8*i* lesson of this class.
- Do your source application default users have the correct privileges? Users MRP, WIP, BOM, APPS, any other application default user, and schema owner must have CREATE ANY VIEW and CREATE ANY SNAPSHOT privileges.

PDP Failure

PDP failure will occur if any of the following are true:

- The PDP process is already running for the instance:
  - UNIX: `ps -ef | grep MSCPDP`
  - NT: Go to Task Manager and look for `MSCPDP`
- APS staging tables are empty:

  `MSC_APPS_INSTANCES.ST_STATUS = 0`
- The PDP process has just pulled data into the staging tables and is waiting for the Planning ODS Load request to process the data:

  `MSC_APPS_INSTANCE.ST_STATUS = 1`

PDP Failure

- Planning ODS Load process is loading data for the instance. In this case:

  `MSC_APPS_INSTANCE.ST_STATUS = 2`

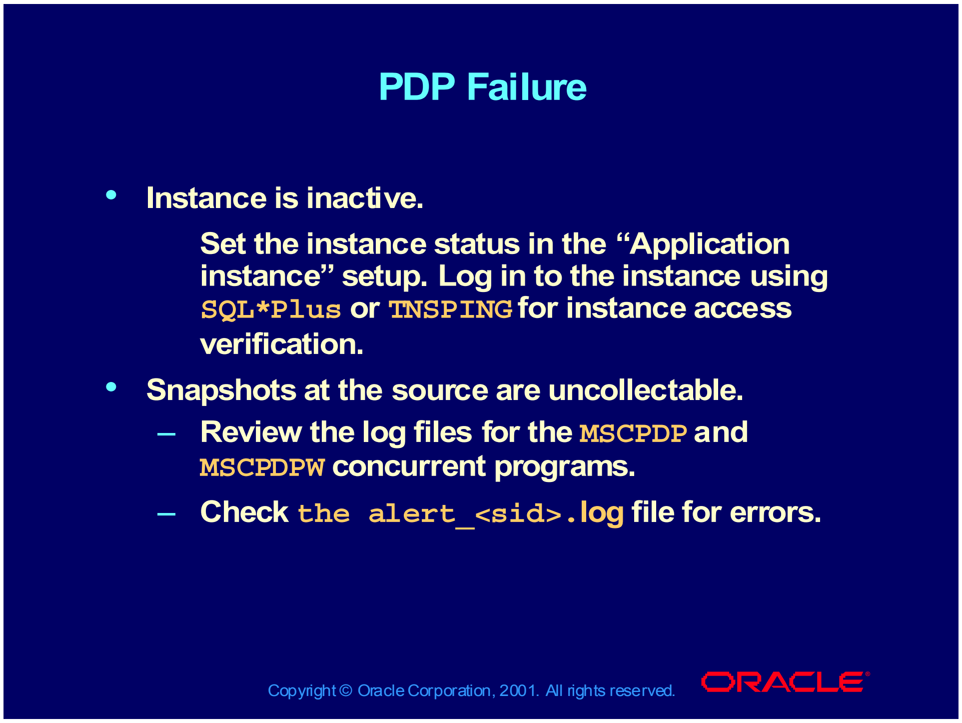
- Data is being transformed from the staging tables and loaded to the Operational Data Store by the Planning ODS Load process:

  `MSC_APPS_INSTANCE.ST_STATUS = 3`

- Data is being purged from the staging tables for the instance:

  `MSC_APPS_INSTANCE.ST_STATUS = 4`

PDP Failure



PDP Failure

Using any tool available, attempt a connection to the source instance. If the instance does not respond, contact your technical personnel.

Consider the following diagnostic steps:

- Run the Purge Staging Tables program with Validation = YES.
- Check the status of the `MSC_APPS_INSTANCES.STATUS` column. Reset, if necessary.
- Check for invalid packages on the source and destination instances.
- Refresh the source snapshots, if required. You can do this by running the commands labeled REFRESHING SOURCE SNAPSHOT.
- Snapshots consume disk space. It is conceivable that available space has expired at the source or destination instances. Check the `Alert_<Sid>.log file` for the sources and destination instances. If a database error occurs, the `Alert_<Sid>.log` file will contain the errors. Repair the database error and start at the beginning of these diagnostic steps. The alert log may only be visible by your technical staff. Contact your technical staff if you suspect that the database is the source of your errors.

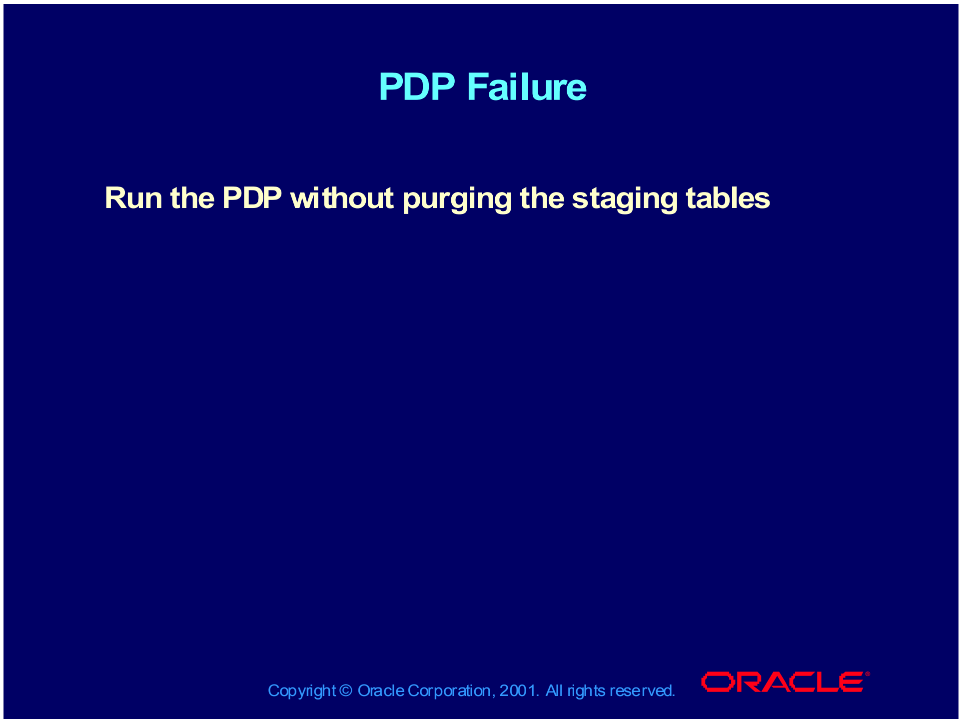
How to launch refreshing source snapshot

You can launch:

- Switch to the correct responsibility.
- NAV>View/Requests.

- Click Submit New Request.
- Select Single Request.
- Select Refresh Snapshot.
- Submit.

PDP Failure



PDP Failure (continued)

To debug the PDP, it may be necessary to run MSCPDP alone. This will prevent MSCPDCP from running, and as a consequence, the staging tables will not be purged. The end result is that your debugging activities will center around the data remaining in the staging tables.

Following is a navigational dialog for setting up a request set that will run only the MSCPDP:

System Administrator > Requests > set > Planning Data Collection MSCPDC

Define stages > Data Pull MSCPDP

To run the data pull by itself:

Go to System Administrator > Requests > set > Request Set Wizard

How would you like the request to run? Select Sequentially

Click Next.

What would you like the set to do when a request ends with the status Error?

Select Abort Processing.

Click Next.

Name your set (example: Test Data Pull).

Select Oracle Advanced Supply Chain Planning as the application.

Provide a description.

Click Next.

When would you like your request output files to print? Select option 2.

Click Next.

Select Planning Data Pull as the Program.

Click Finish.

Everything necessary to run will default in for you.

Go to System Administrator > Requests > Run > Request set > Select your request set.

Put your cursor in the Parameters field. The Parameters window will pop up. Fill in the appropriate parameters and close the window.

Submit the request.

Staging Tables

- Staging tables reside on the APS instance.
  Example: `MSC_ST_SYSTEM_ITEMS` or `MSC_ST_DEMANDS`
- Staging tables are used to store planning data destined for key transformation and finally the ODS.
- Data can be cleansed in staging tables.
- Staging tables need to be sized correctly.
- Depending on staging table activity, consider isolating the staging tables from ODS/PDS.

Staging Tables

The following entities are represented in the staging tables:

Bill of Materials	Bill of Resources	Calendar
Categories	Demands	Sales Orders
Hard Reservations	Items	Resources
Safety Stock	Schedule Designator	Sourcing
Subinventory	Supplier Capacity	Supply
Resource Requirements	Trading Partners	Unit Numbers
Projects	Parameters	Unit of Measures
ATP Rules	Planners	Demand Classes
Partner Contacts	Business Intelligence System	

For a complete listing of staging tables, refer to Appendix A.

PDP Purging Staging Tables

- Data in staging tables is usually purged after Planning ODS Load (POL) is successfully completed.
  - This pertains to Complete and Incremental refresh PDP modes.
- The PDP process will attempt to purge the already extracted data and any data orphaned as a result of a terminated PDP process.
  - MSCPDCP spawns MSCPDEL, the deletion program.

PDP Purging Staging Tables

Staging tables can be purged by running the Purge Staging Tables concurrent request.

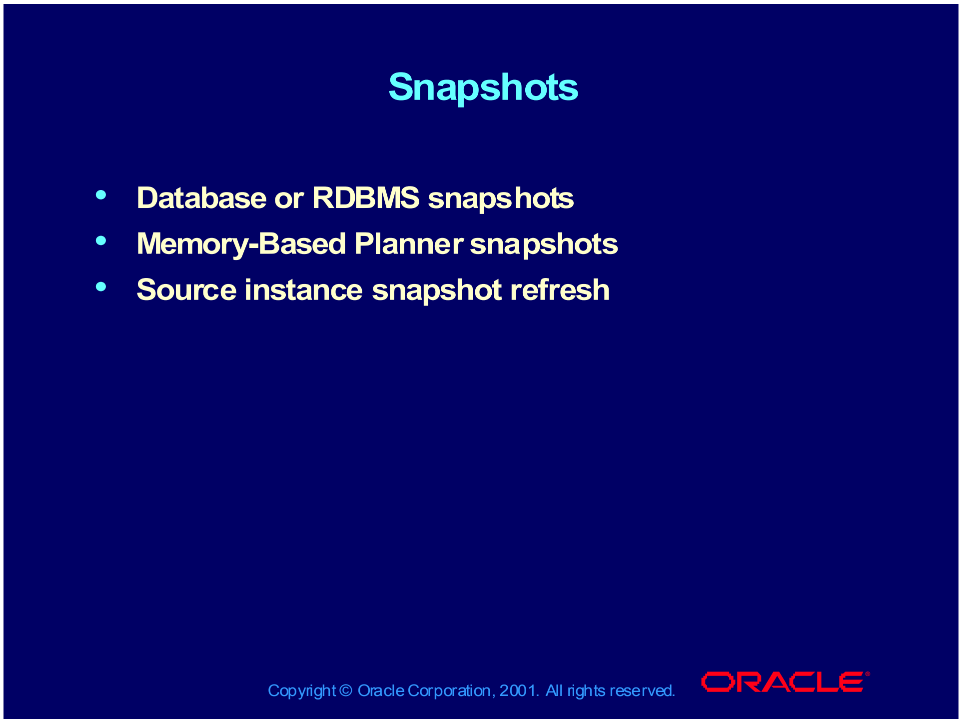
PDP Capturing Incremental Changes

PDP Capturing Incremental Changes

- The Refresh Snapshots process retrieves the `Last_Refresh_Number (LRN)` by fetching the value from `MRP_AP_APPS_INSTANCES.LRN`.
  - This value is the key to track incremental changes between each data collection.
  - All rows from views MRP_AP_xxx_V and MRP_AD_xxx_V in the source instance with Refresh Numbers (RN) <= LRN have already been transferred to APS instance during the previous data collection runs. This comparison occurs for both Complete or Incremental refreshes.

PDP Capturing Incremental Changes

- The RN used for each row during the refresh process is derived from column `Nextval` for sequence `MRP_AP_REFRESH_S`.
  - This refresh process (with the RN generation) occurs in both the PDP Complete and Incremental refreshes.
  - If the refresh type for the current PDP is Incremental, all rows from the `MRP_AP_xxx_V` and `MRP_AD_xxx_V` views on the source instance with RN > LRN will be pulled into the corresponding staging tables on the APS instance.
  - In case of a PDP Complete refresh, all rows will be transferred from the `MRP_AP_xxx_V` and `MRP_AD_xxx_V` views on the source instance to the corresponding staging tables on the APS instance.

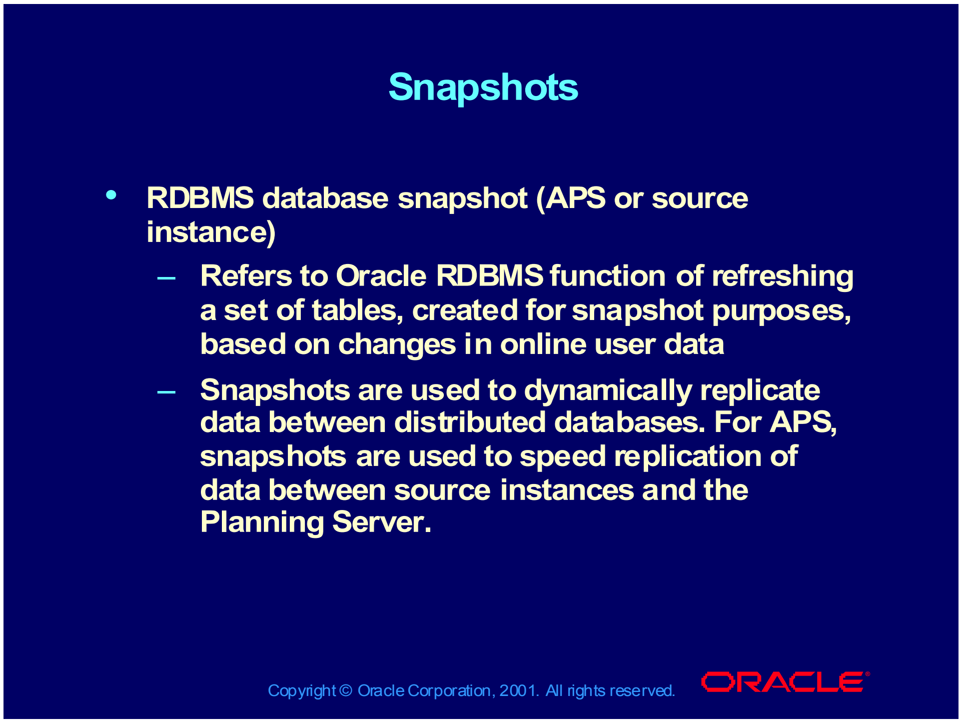


Snapshots

Oracle Applications database tables used by snapshot routine for APS:

Inventory	Bills Of Materical	WIP
mtl_demand	bom_bill_of_materials	wip_discrete_jobs
mtl_material_transactions_temp	bom_inventory_components	wip_flow_schedules
mtl_onhand_quantities	bom_operation_resources	wip_lines
mtl_supply	bom_operation_sequences	wip_operations
mtl_system_items (10.7,11.0)	bom_operational_routings	wip_repetitive_items
mtl_system_items_b (11.i)	bom_resource_changes	wip_repetitive_schedules
		wip_requirement_operations
PO/RCV	**MRP**	**OE**
po_supplier_item_capacity	mrp_schedule_dates	so_order_lines_all

For a complete listing of actual snapshot tables used during data collection from an OLTP, see Appendix B.


Snapshots
• RDBMS database snapshot (APS or source instance)
– Refers to Oracle RDBMS function of refreshing a set of tables, created for snapshot purposes, based on changes in online user data
– Snapshots are used to dynamically replicate data between distributed databases. For APS, snapshots are used to speed replication of data between source instances and the Planning Server.
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Snapshots

Snapshots

Two snapshot types exist:

- **Simple: Snapshot will correspond one to one with records in the master table.**
- **Complex: Snapshot will not correspond one to one with records in the master table. There may be records from multiple master tables.**

ORACLE

Snapshots (continued)

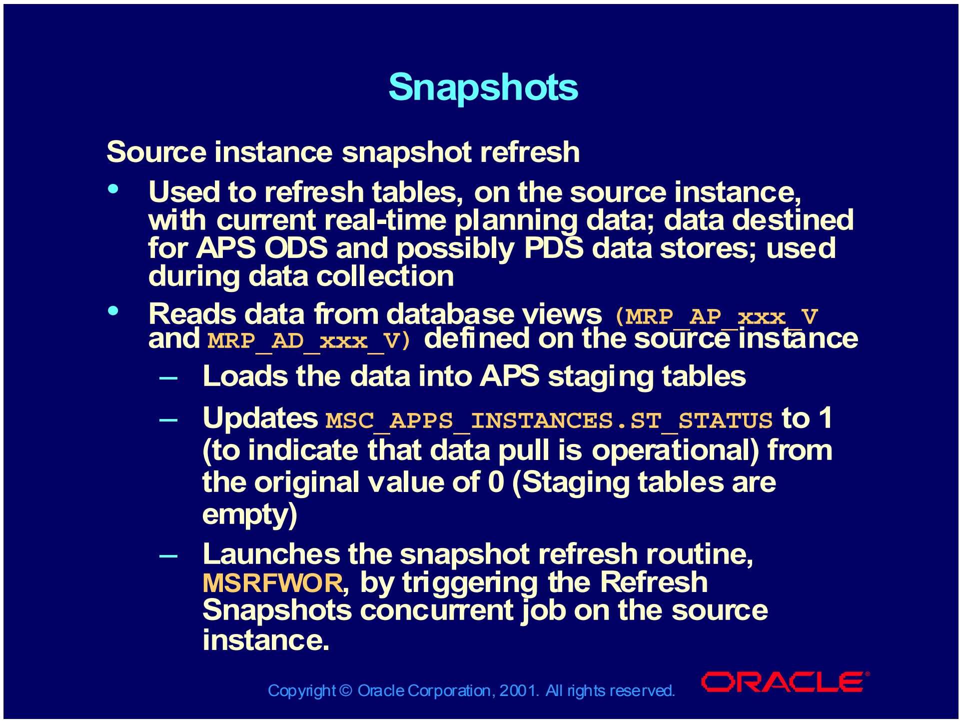
Oracle Application database tables used by snapshot routine for APS:

Inventory	Bills of Material	WIP
mtl_demand	bom_bill_of_materials	wip_discrete_jobs
mtl_material_transactions_temp	bom_inventory_components	wip_flow_schedules
mtl_onhand_quantities	bom_operation_resources	wip_lines
mtl_supply	bom_operation_sequences	wip_operations
mtl_system_items (10.7,11.0)	bom_operational_routings	wip_repetitive_items
mtl_system_items_b (11.i)	bom_resource_changes	wip_repetitive_schedules
		wip_requirement_operations
PO/RCV	**MRP**	**OE**
po_supplier_item_capacity	mrp_schedule_dates	so_order_lines_all

For a complete listing of actual snapshot tables used during data collection from an OLTP, see Appendix B.

Snapshots

- Memory-Based Planner snapshot (APS instance)
  - Process launched to purge, collect, and plan with the data extracted from online user tables
  - Process of collecting real-time planning data and inserting it into planning tables (used in APS as well as prior Oracle MBP applications)
  - Available in all Memory-Based Planner implementations
- Net change, in this environment, means a replan.



Snapshots (continued)

To launch this program:

1. Switch to the correct responsibility:
2. NAV>View/Requests
3. Click Submit New Request.
4. Select Single Request.
5. Select Refresh Snapshot.
6. Submit.

Snapshots

Snapshots

- **Refresh snapshot in centralized configuration**

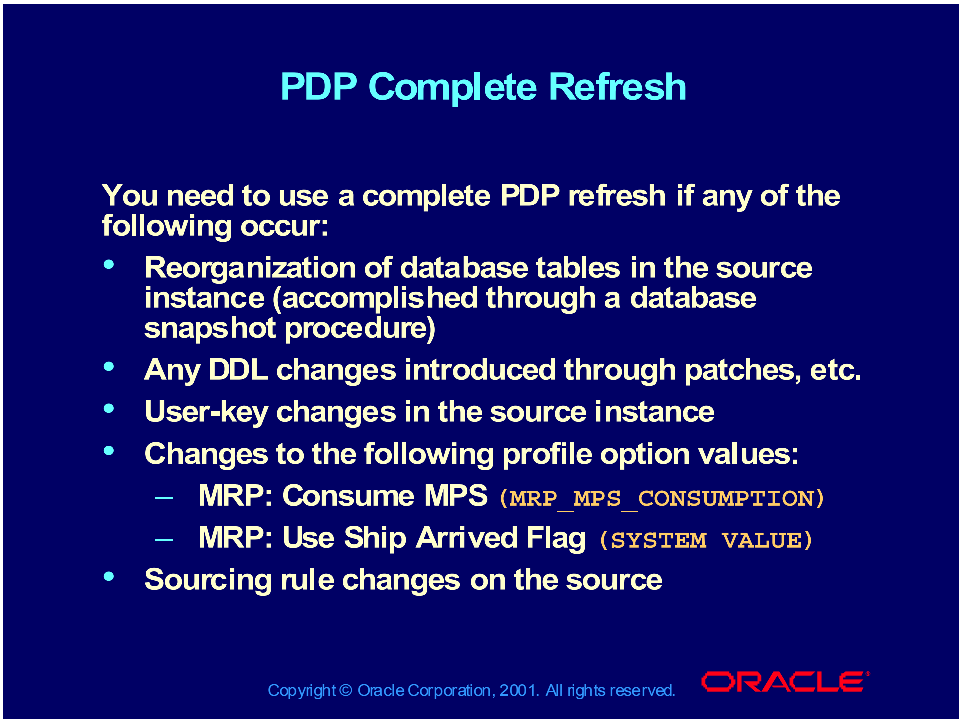
  PDP process executes the snapshot refresh routine on the same instance where it runs (database link is not required in this case).

- **Refresh snapshot in Decentralized configuration**

  PDP process (on the Planning Server) executes the snapshot refresh package (`MRPDELFS`) on the remote database (source instance) using the database link defined on the APS instance.

ORACLE

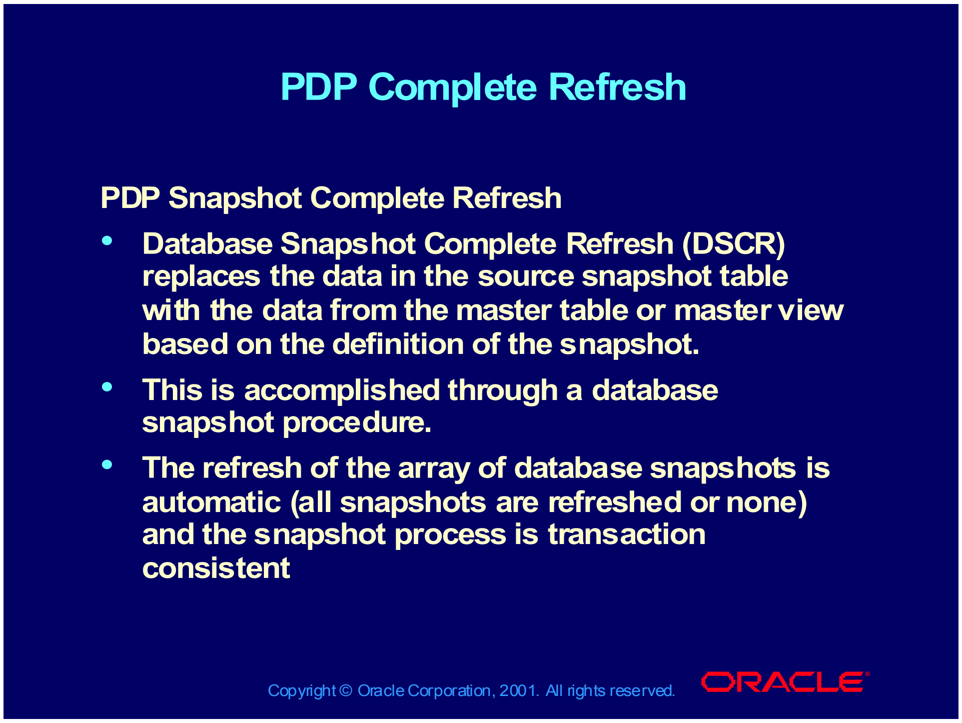
PDP Complete Refresh



PDP Complete Refresh

What PDP refresh type should I use?

- Incremental refresh is faster.
- Use Complete refresh when necessary.
- Analyze your tables using `MSCSTATS`. This program can be found on the APS instance under `$MSC_TOP`.



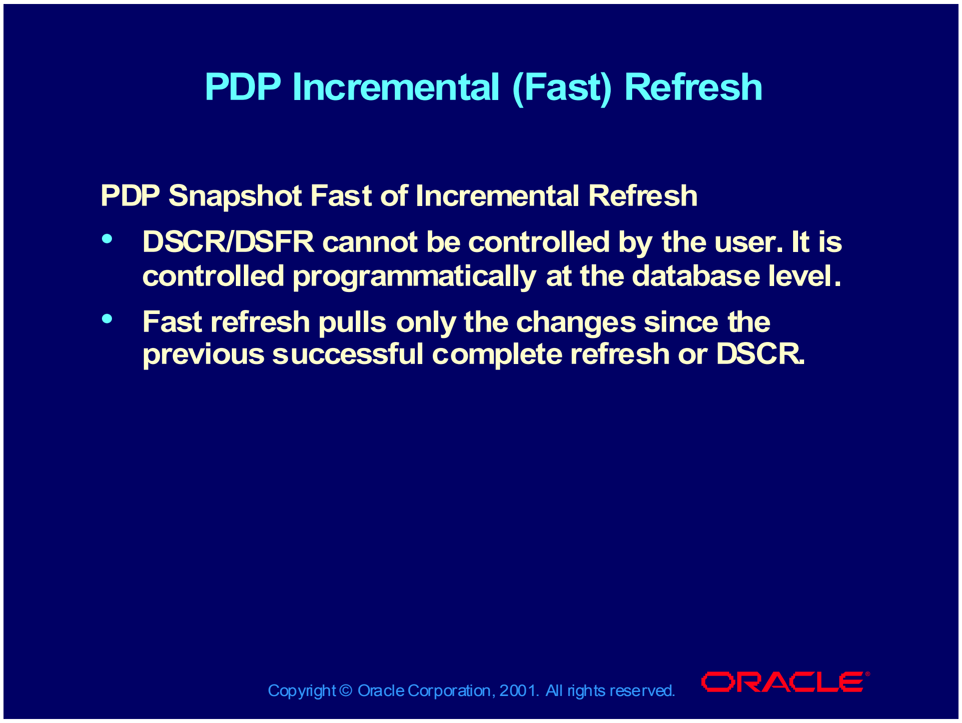
PDP Complete Refresh (continued)

Database snapshot tables maintain the same integrity constraints as the master tables or views during the Refresh Snapshot process.

PDP Complete Refresh

- Pulls data from `MRP_AD_xxx_V` and `MRP_AP_xxx_V` views into the source instance
- Moves extracted data to the APS staging tables
- Ignores the status of the most recent refresh

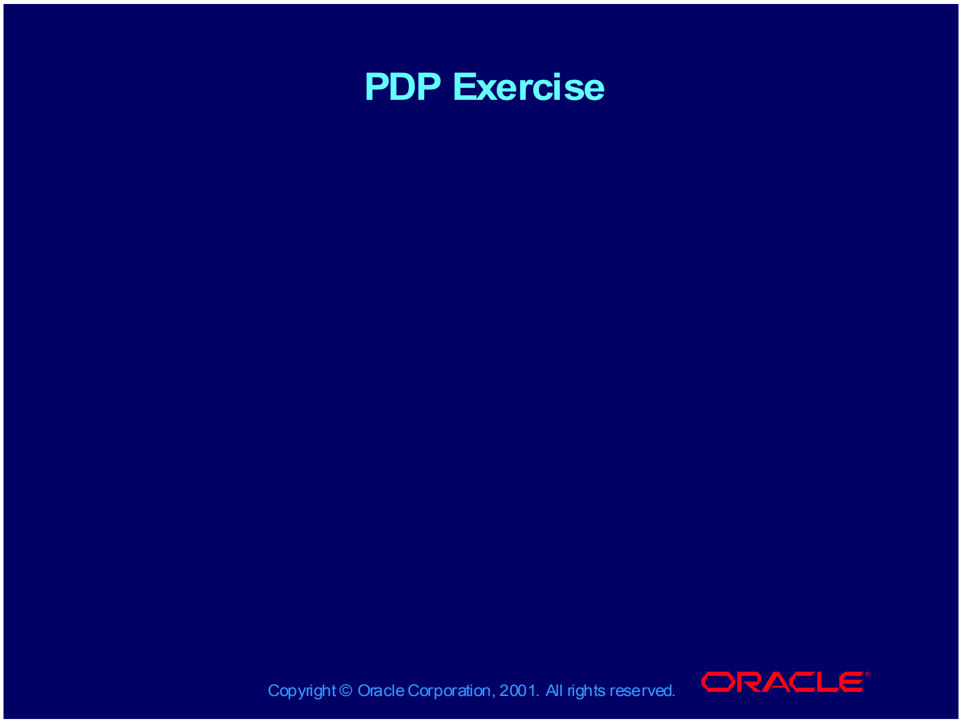
ORACLE



Refresh Snapshot

The Refresh Snapshot process triggered by PDP (launched based on concurrent program input parameters for Planning/ATP Data Collection report set) performs a Database Snapshot Fast Refresh (DSFR).

PDP Exercise



PDP Exercise

1. Log in to SQL*Plus on the source instance.
2. View the columns of the controlling table at the source instance;

   ```
   Desc mrp_ap_apps_instances <return>
   Select *
   from mrp_ap_apps_instances;
   ```

3. Log in to SQL*Plus on the destination instance.
4. View the columns of the controlling table at the destination instance:

   ```
   Desc msc_apps_instances <return>
   Select * from msc_apps_instances;
   ```

Observe the differences between the data in these two tables.

Notice the `st_status` column referred to in the data pull process.

Refer to the *APS Technical Reference Manual* for more detailed information.

5. Using a database link, perform the following query on the source and destination instances.

   ```
   Select owner,
   substr(object_name,1,50) obj_name
   object_type
   from   DBA_OBJECTS
   where  status != 'VALID'
   ```

```
and    object_name like 'MSC%';
OWNER  OBJ_NAME           OBJECT_TYPE
----------------------      -----------
MSC  MSC_ATP_PLAN_SN      UNDEFINED
```

Notice that a snapshot is undefined. This happens because Global Available to Promise is not set up.

If you have invalid snapshots , run the following command at the SQL*Plus prompt: `alter snapshot <snapshot name> compile;`

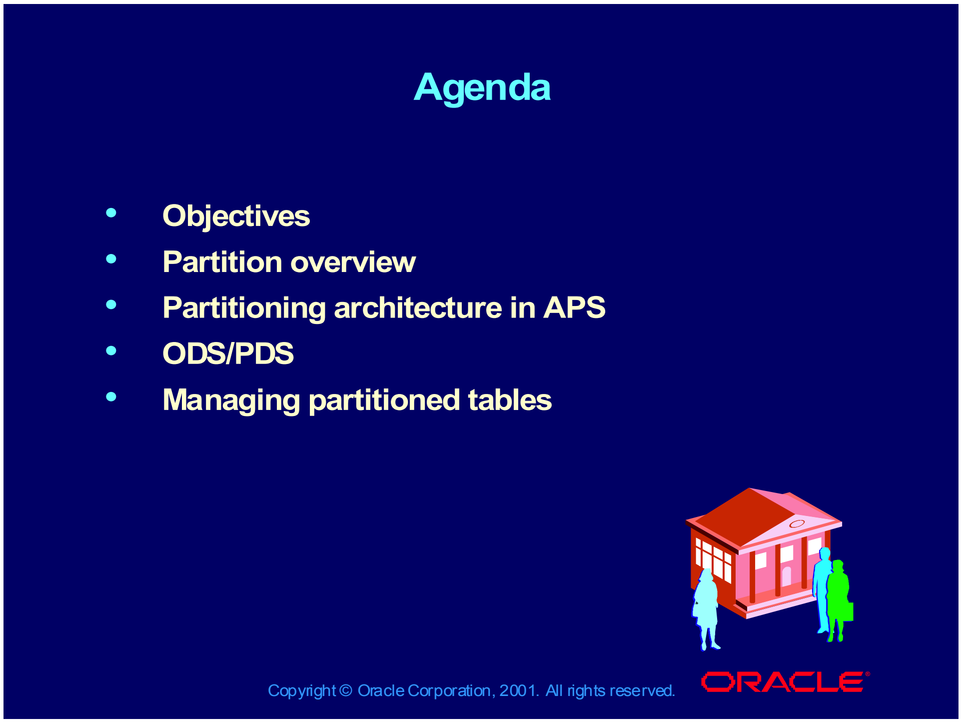
If you have invalid database objects, use `adadmin` to remedy the situation.

Plan Partitions

Chapter 5

Plan Partitions

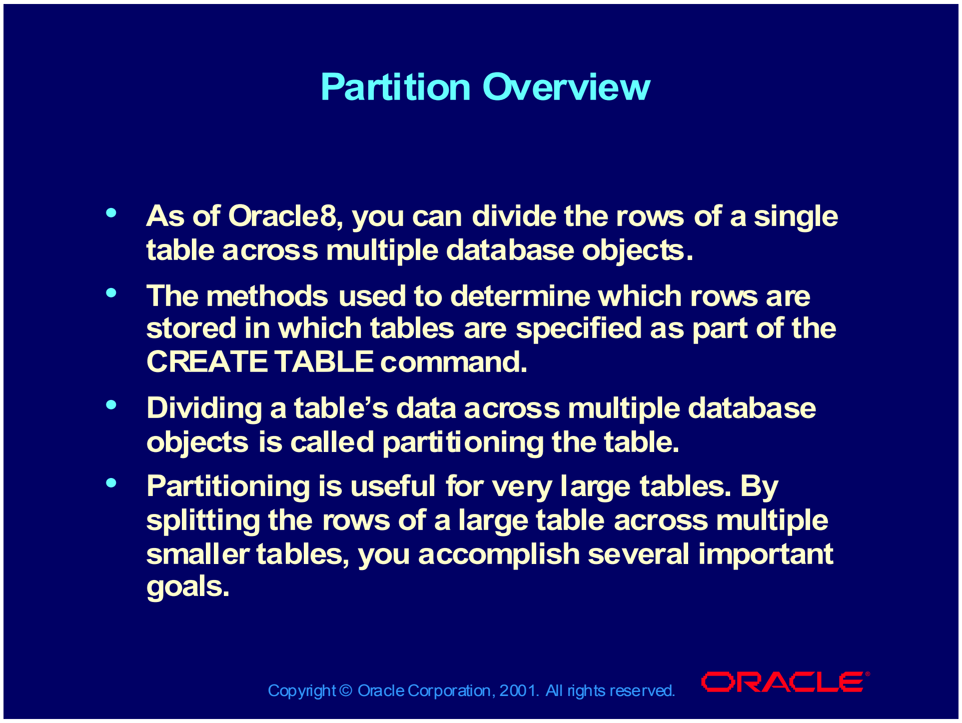
Agenda


Agenda
• Objectives
• Partition overview
• Partitioning architecture in APS
• ODS/PDS
• Managing partitioned tables
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Objectives

After completing this chapter, you should be able to do the following:

- Describe how Oracle APS uses partitioning for data segregation
- Recognize partitioning as implemented in ODS/PDS
- Describe how to manage partitions generated through plan creation



Partition Overview

You may want to create an index, reorganize the database, or perform other operations, only to discover that the resulting downtime may be too long for very large or mission-critical tables.

One way to avoid significant downtime during operations is to create and use partitions. You can use partitions by dividing very large tables into multiple, small pieces (partitions), which offer significant improvements in availability, administration, and table scan performance.

Partitions are scanned in parallel when all partitions are either skipped or accessed in parallel.

Partition constraints are for skipping only, not for allocating work to query server processes.

The number of partitions is unrelated to the degree of parallelism. Full parallelism is used even if a single partition is not skipped.

Partition Overview

Partition views offer increased manageability and flexibility during queries. Individual partitions can be:

- Added and dropped independently and efficiently
- Reorganized, backed up, and restored independently
- Split, merged and loaded incrementally while maintaining local indexes
- Loaded in parallel

ORACLE

Partition Overview (continued)

You create a partition by dividing a large table into multiple physical tables using partitioning criteria. Then, for future queries, Oracle brings the table together as a whole.

Partition Overview

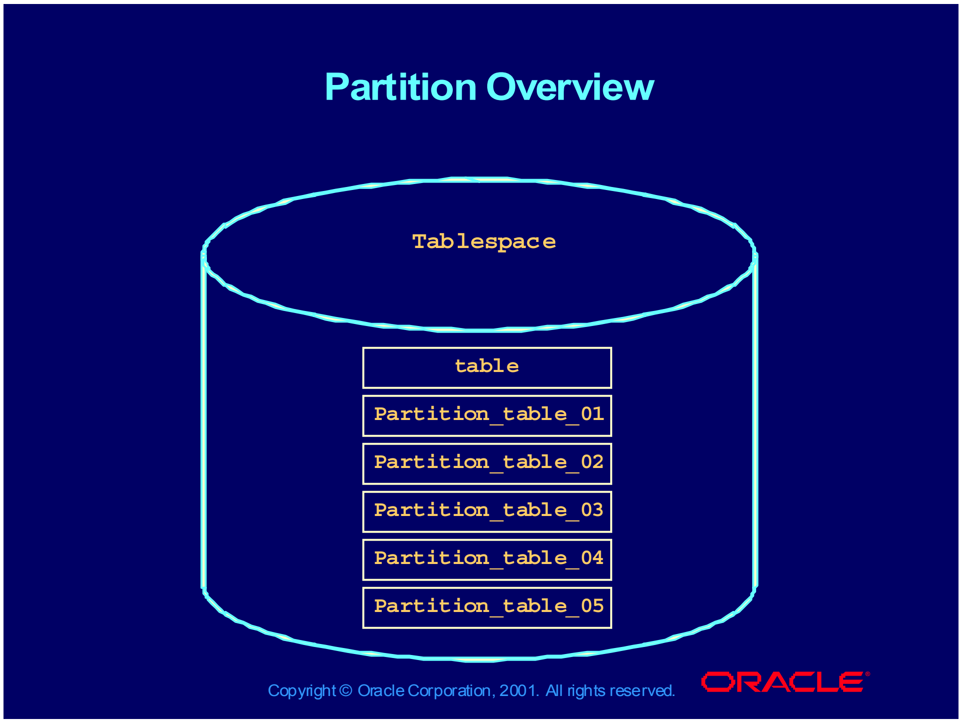
Partition Overview

- The performance of a query may improve because the database may have to search only one part of a table for the desired data (called a partition).
- It may be easier to load or delete data in the table partitions.
- Deleting plan-specific information is isolated to perhaps one partition.
- Backup and recovery may perform better because the database looks at only a part of a larger table at a time.

Partition Overview (continued)

You now have the ability to limit the impact of data corruption to one segment. Also, you can use a key range to select from a partition view only the partitions that fall within that range.

Partitions are scanned in parallel when all partitions are either skipped or accessed in parallel. Partition constraints are for skipping only, not for allocating work to query server processes. The number of partitions is unrelated to the degree of parallelism. Full parallelism is used even if a single partition is not skipped.

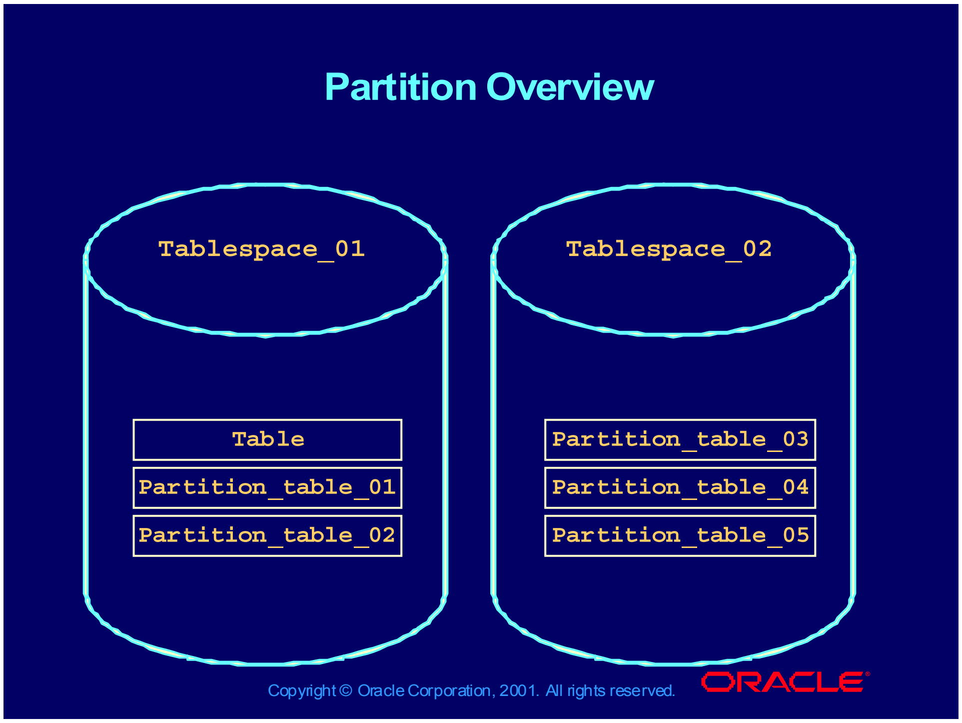


Partition Overview (continued)

When a table is partitioned, Oracle RDBMS creates additional database objects corresponding to the desired number of partitions.

Each partition consumes disk resources as demonstrated later in the module exercise.

Partition Overview

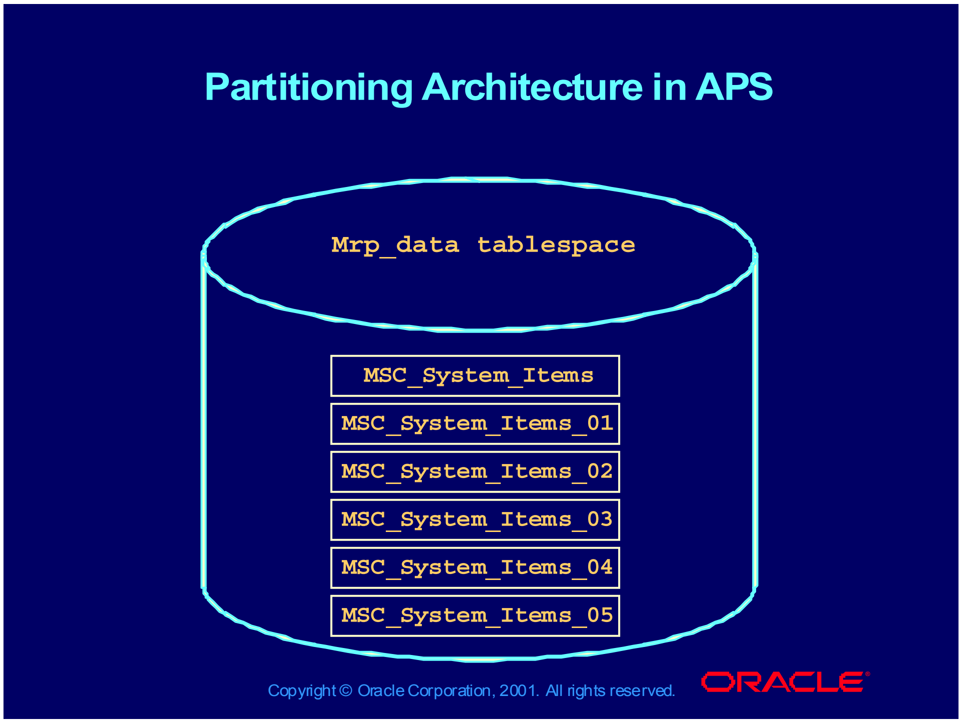


Partition Overview (continued)

Oracle RDBMS enables the Oracle Applications DBA to manage partitions. By separating large, heavily accessed data within different tablespaces, performance may improve.

As indicated on the slide, the Oracle RDBMS created the base table and two table partitions within one tablespace and three table partitions in a separate tablespace.

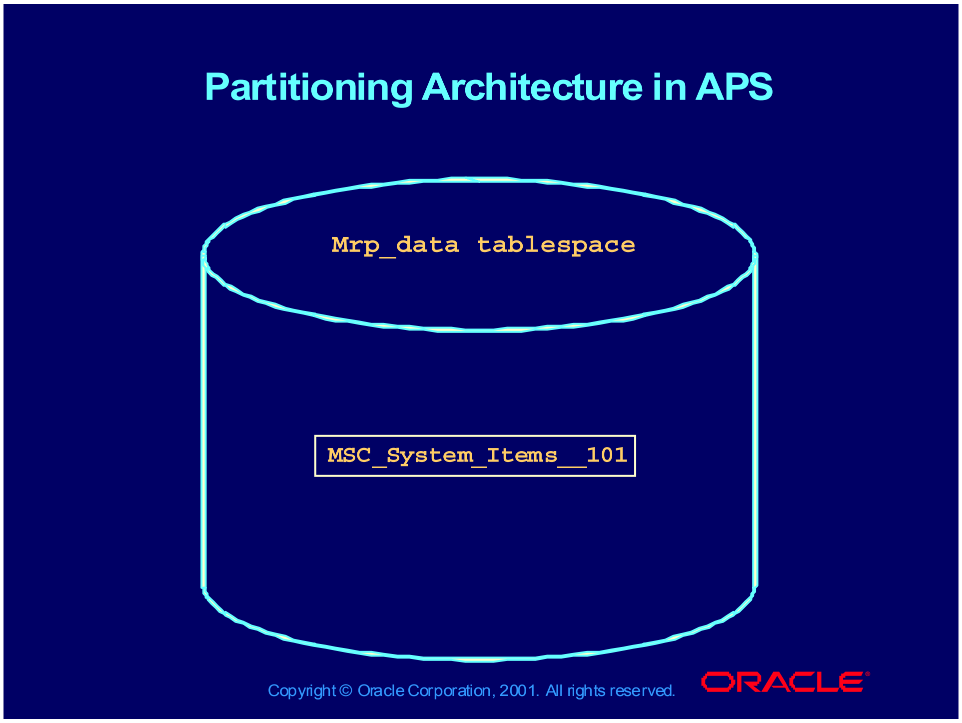
Partitioning Architecture in APS



Partitioning Architecture in APS

By default, Oracle APS Rapid Install creates five planning partitions for each partitioned table in the MSC schema.

On the slide, you see the result of partitioning on MSC System items for plan partitions.



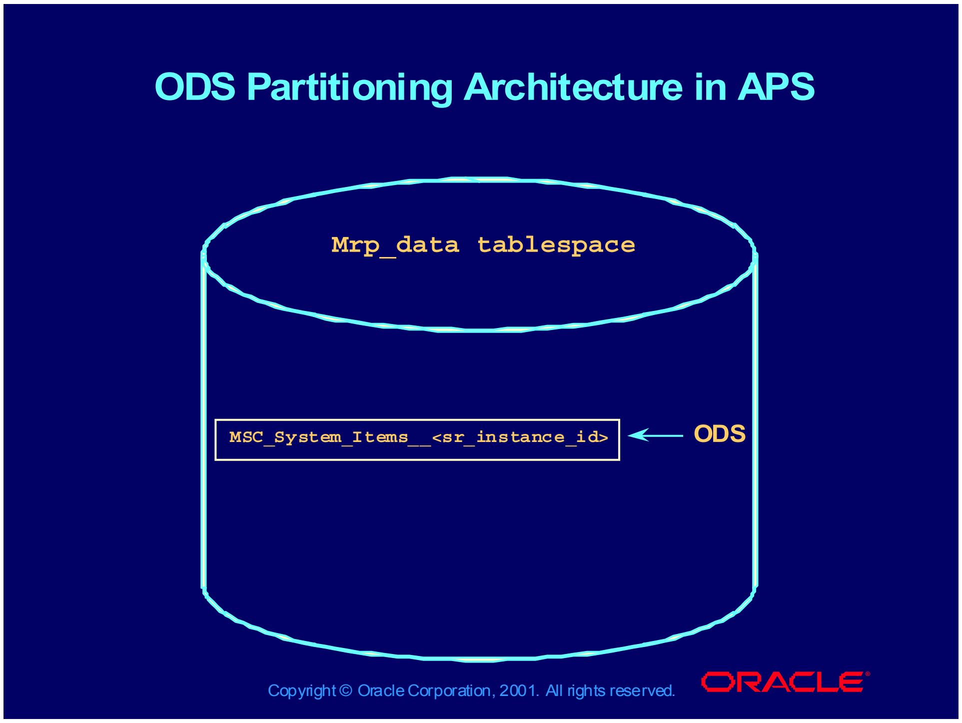
Partitioning Architecture in APS (continued)

By default, Oracle APS Rapid Install creates one instance partition.

Notice that the table is identifiable by the source instance ID that is appended to the database object name.

Two underscores separate the database object name from sr_instance_id.

The slide shows the result of partitioning on MSC System items.



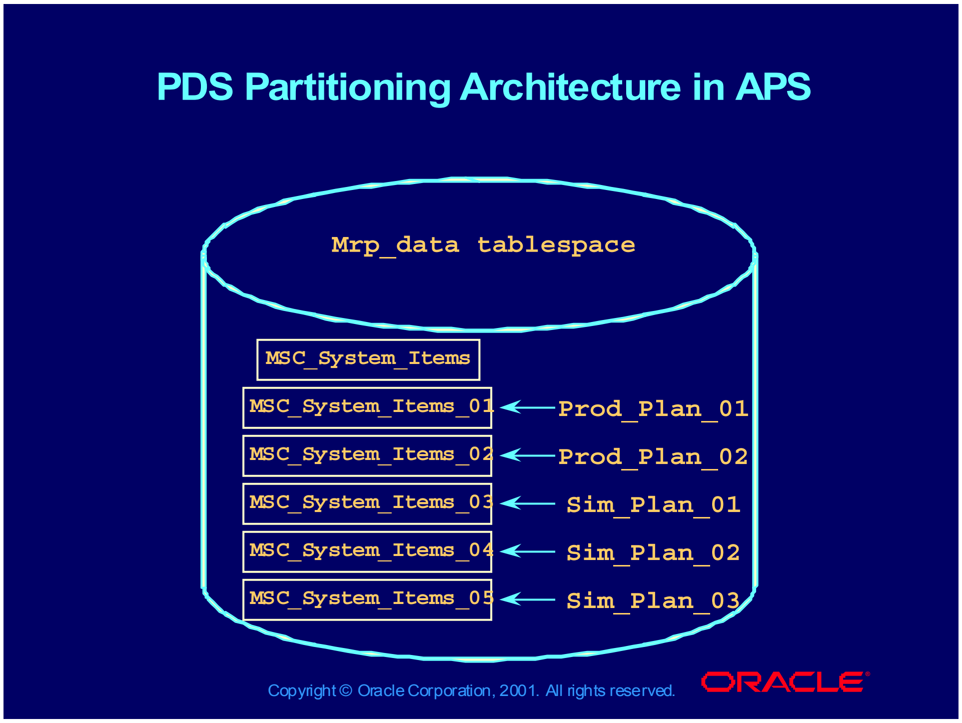
ODS

Data in the ODS is identified by sr_instance_id and plan_id. Rows with a plan_id = -1 belong to the ODS.

ODS can be used by Global Available to Promise. The profile option INV:Capable To Promise controls this functionality. If this profile option is set to 5—Enable PL/SQL Based ATP without Planning Output, Global Available to Promise will use ODS data for order commitments.

The planning snapshot process snapshots data chosen by each plan from the ODS and writes ASCII flat files corresponding to each of the entities chosen by the snapshot.

PDS Partitioning Architecture in APS



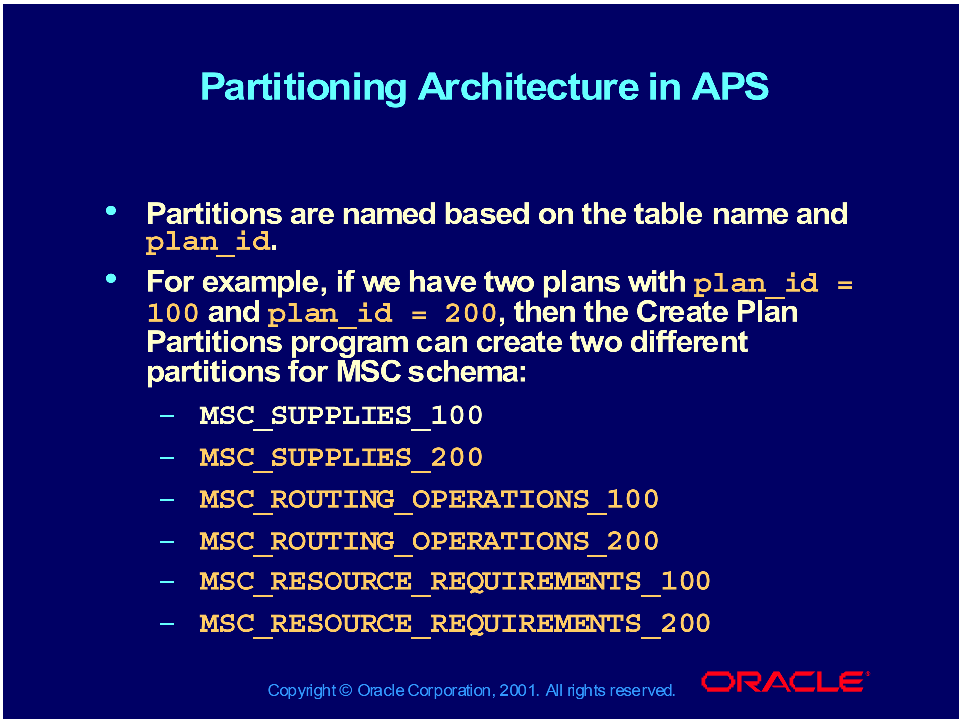
PDS

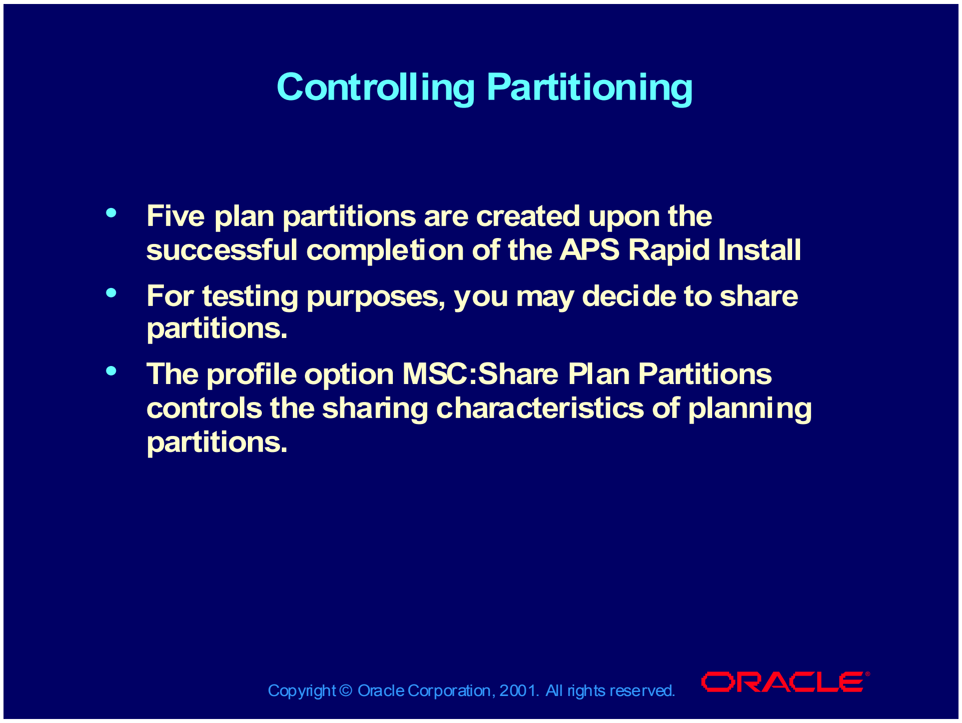
Data in the PDS is identified by instance_id and plan_id. Rows with a plan_id that equal the internally assigned sequence belong to the PDS. The following lesson demonstrates this association.

PDS can be used by Global Available to Promise. The profile option INV:Capable To Promise controls this functionality. If this profile option is set to 4—Enable PL/SQL Based ATP with Planning Output, Global Available to Promise will use PDS data for order commitments.

The basic concurrent processing architecture resembles the current snapshot architecture. Partitioned PDS tables provide the following significant advantages:

- Copy plan
  - Delete tasks
  - Purge plan
  - Partitioned local indexes
- Can be manipulated separately by the Oracle RDBMS


Partitioning Architecture in APS
• Partitions are named based on the table name and plan_id.
• For example, if we have two plans with plan_id = 100 and plan_id = 200, then the Create Plan Partitions program can create two different partitions for MSC schema:
– MSC_SUPPLIES_100
– MSC_SUPPLIES_200
– MSC_ROUTING_OPERATIONS_100
– MSC_ROUTING_OPERATIONS_200
– MSC_RESOURCE_REQUIREMENTS_100
– MSC_RESOURCE_REQUIREMENTS_200
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE



Controlling Partitioning

Always check for evidence of successful completion by searching through the log files.

When the profile option MSC:Share Plan Partitions is set to Share Plan Partitions, Oracle will use only one partition to store all plan data.

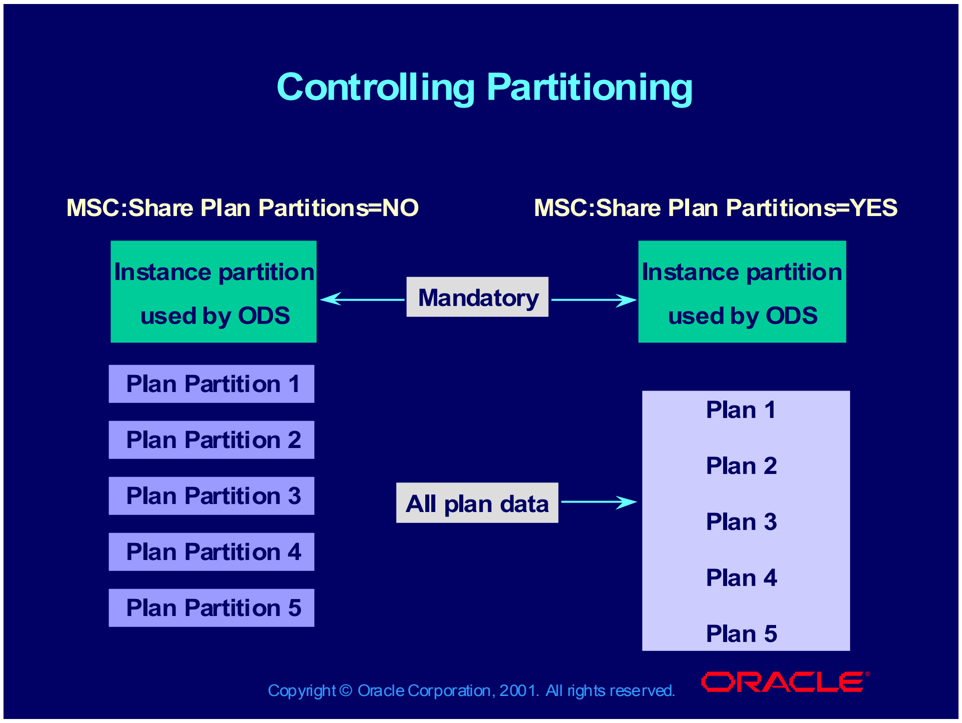
Do not create plan partitions above those needed for your plans. Unused plan partitions consume space in the Oracle RDBMS SGA, possibly affecting performance.

Each table must be partitioned. There are 18 tables in the MSC schema in which partitioning is used. If you create 10 partitions, this results in 180 database objects, not including indexes.

If you have implemented Share Plan Partitions and now want to implement the standard APS partitions, you must perform database maintenance tasks.

Drop the instance partition with a plan_id = -99999, which was originally created to hold all of the plans in a shared partition. The database will not allow you to create additional partitions with an existing partition labeled as -99999.

Controlling Partitioning



Controlling Partitioning (continued)

By default, five plan partitions are created.

Notice that the instance partition is created in both implementations. This portion of the APS data store is known as the ODS. This is striped by instance_id and plan_id = -1.

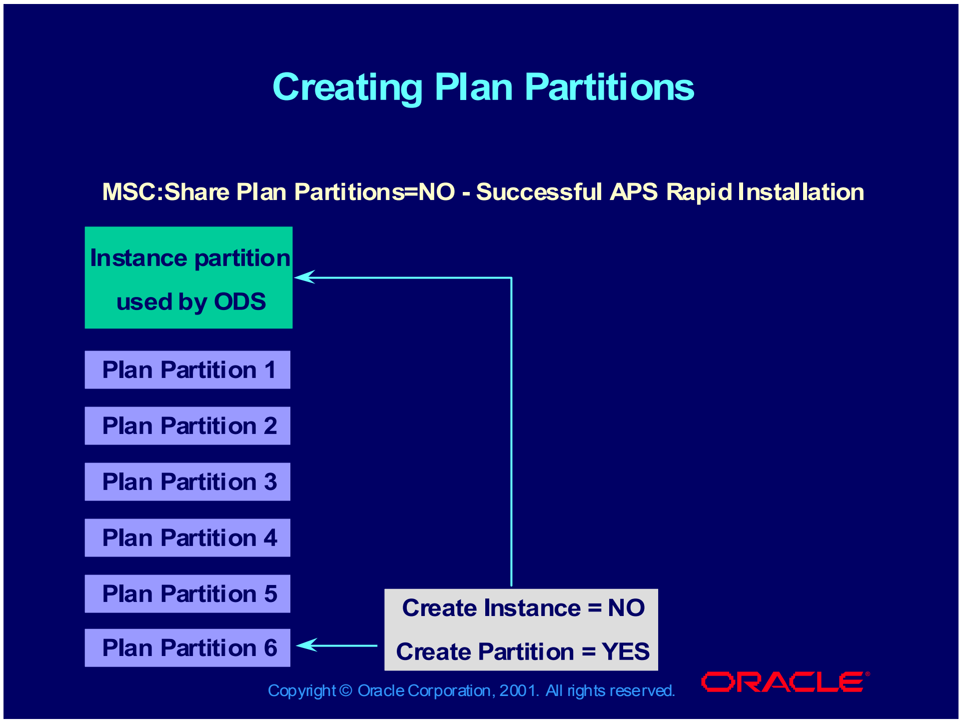
When the profile option MSC:Share Plan Partitions is set to Share Plan Partitions, Oracle will use only one partition to store all plan data.

Although setting this profile option to YES may be beneficial when testing small amounts of data, using this setting in production may reduce the performance gains possibly realized when using default partition implementation.

Set MSC:Share Plan Partitions to NO when using APS in production.

Use the Create APS Partitions concurrent program, to create a new instance and additional planning partitions. Initially, you will have to run this from the System Administrator responsibility.

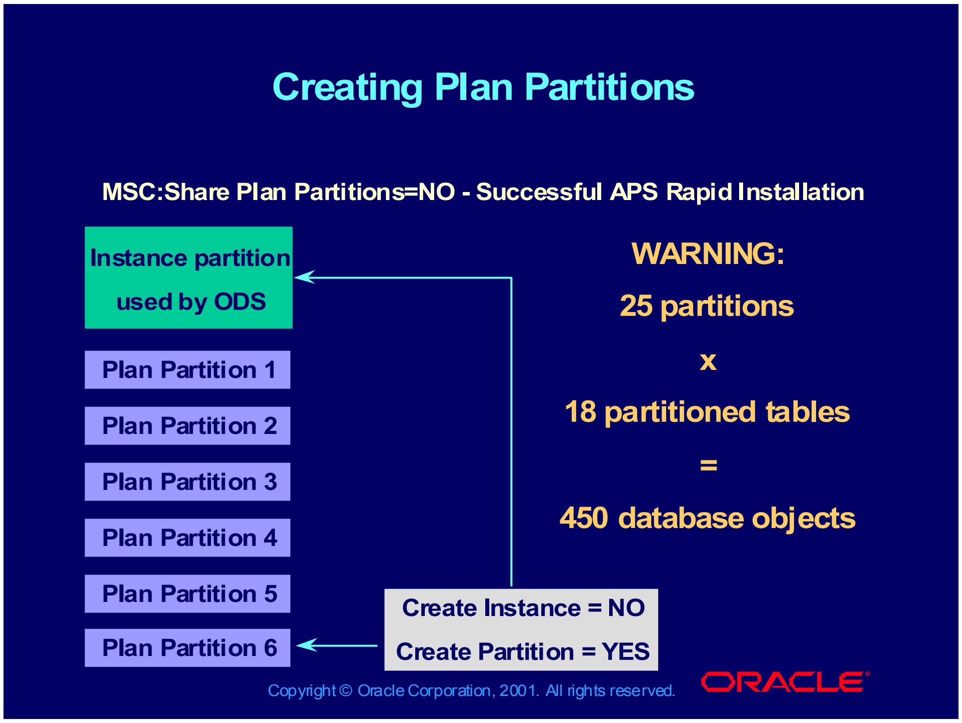
Creating Plan Partitions



Creating Plan Partitions

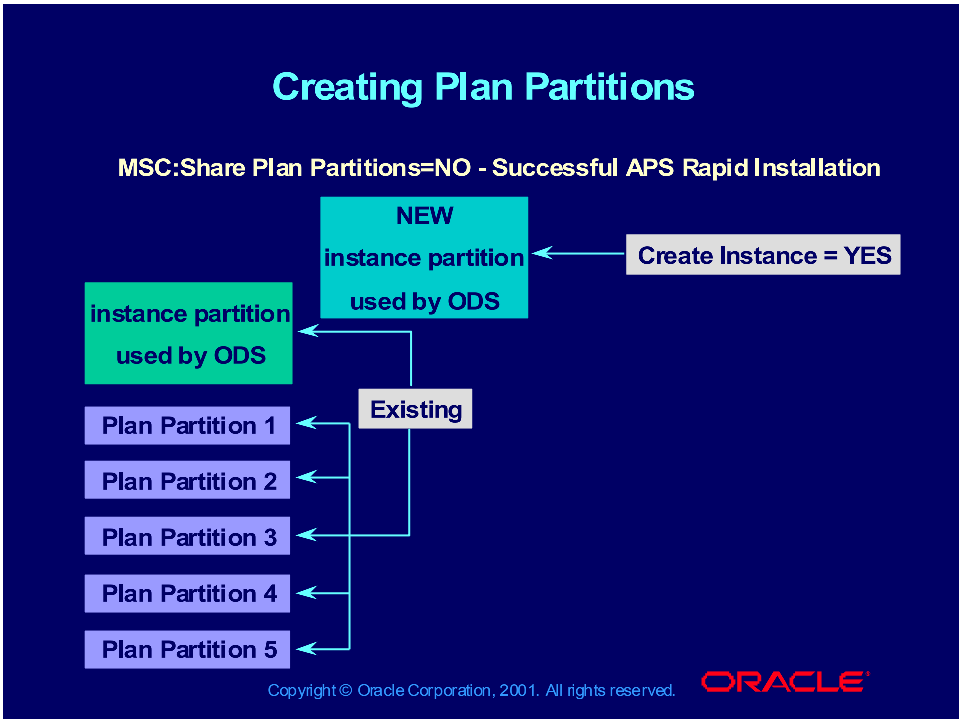
If you need to add another partition, ensure that you do not create another instance partition. Upon successful completion of APS Rapid Install, an instance partition is created. Simply add another partition.

Do not add any partitions beyond that required for your implementation. Try to keep the number of partitions low. Do not create the partitions if you do not intend to use them immediately.



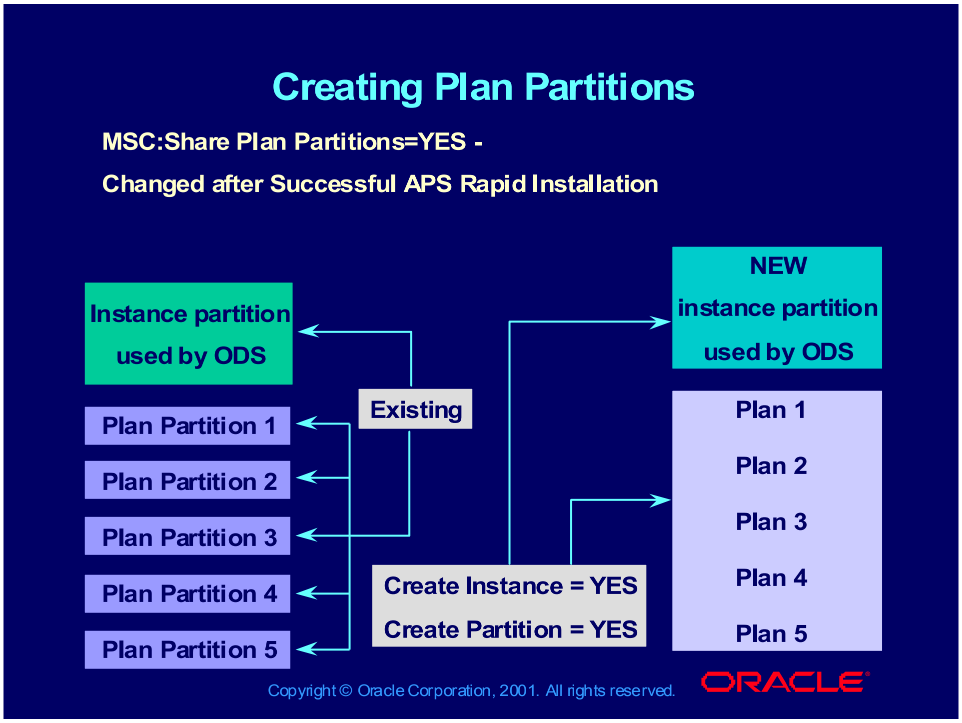
Creating Plan Partitions (continued)

If you need to add another planning partition, ensure that you do not create another instance partition. Upon successful completion of APS Rapid Install, five instance partition are created. Simply add another planning partition.



Creating Plan Partitions (continued)

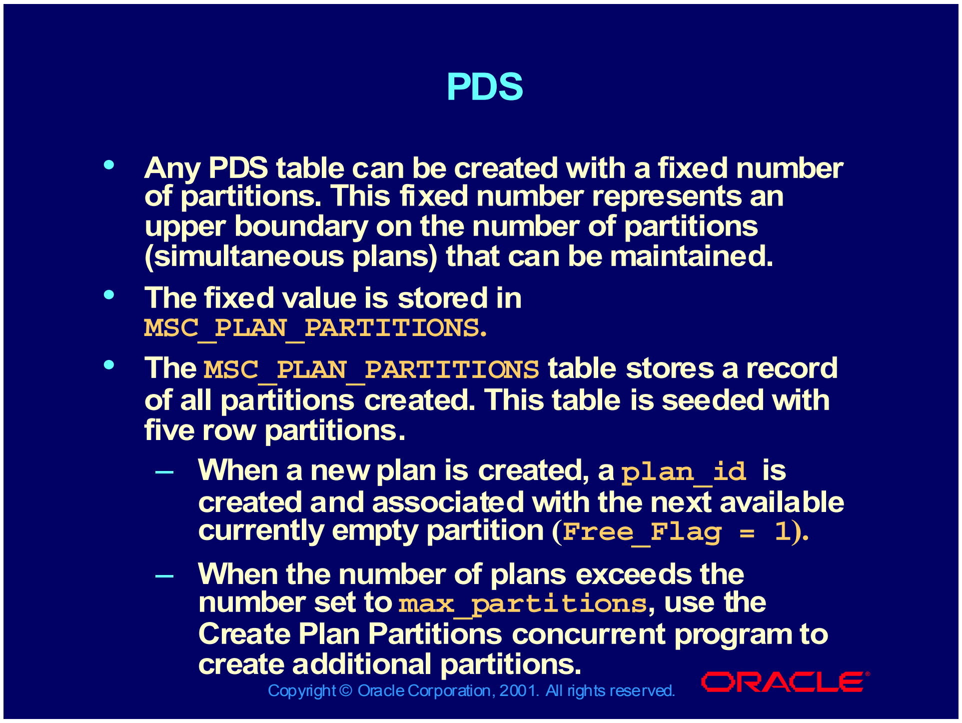
If you need to add another instance partition, ensure that you do not create another planning partition. Upon successful completion of APS Rapid Install, an instance partition is created. Simply add another instance partition.



Creating Plan Partitions (continued)

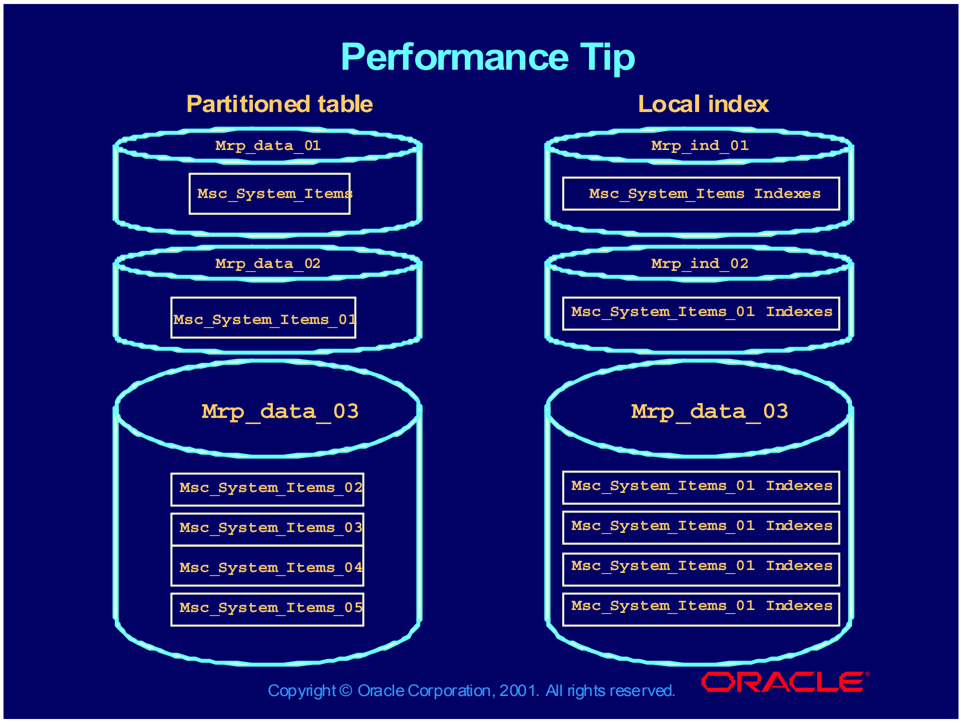
By setting MSC:Share Plan Partitions to YES and running the Create Plan Partitions concurrent program, the Oracle RDBMS will create a new instance partition as well as a primary partition for ASCP plans.

Partitions ending with 99999 belong to the shared partition implementation.



Column	Type	Nullable	Comments
Plan_id	NUMBER	Not Null	All the possible plan_ids. Values from 1 to max_partitions.
Plan_name	Varchar2	Null	Plan name of the plan that is assigned to this partition.
Free_flag	NUMBER	Null	(Yes/No) Flag indicates if the partition corresponding to this plan_id is free.

Performance Tip



Performance Tip

If you are not using disk stripping, consider splitting partitions into separate tablespaces.

You now have the ability to limit the impact of controller and disk access throughput by:

- Storing partitions in unique tablespaces
- Placing these tablespaces in different data files
- Initializing the data files on different disks

Ensuring that the disks are controlled by separate controllers maximum throughput, limited by your server throughput, is more readily achieved.

The local keyword instructs Oracle to create a separate index for each partition, with greater protection against row access problems.

Below you see how the create table DDL instructs the Oracle RDBMS to create 100 partitions on table T:

```
CREATE TABLE T
    ( COL1 NUMBER,
      COL2  NUMBER,
      COL3  Varchar2 ,….
       Plan_id  Number,
      Plan_name   Varchar2  NOT NULL )
PARTITION BY RANGE (plan_id)
```

```
     ( PARTITION T_1 VALUES LESS THAN (2)
         TABLESPACE tsa,
       PARTITION T_2 VALUES LESS THAN (3)
         TABLESPACE tsb,
       PARTITION T_3 VALUES LESS THAN (4)
         TABLESPACE tsc,
       PARTITION T_4 VALUES LESS THAN (5)  ….
PARTITION T_100 VALUES LESS THAN (100)
         TABLESPACE tsd );
```

Below you see how the create index DDL instructs the Oracle RDBMS to create the indexes on table T locally:

```
CREATE INDEX T_IND_1
ON T(plan_id)
LOCAL
(PARTITION PART1
 TABLESPACE APS_IND,
 PARTITION PART2
 TABLESPACE APS_IND,
 PARTITION PART3
 TABLESPACE APS_IND);
```

The simplest strategy is to create two tablespaces, one to hold the partitioned tables and one to hold the partitioned indexes.

Consider spreading the disk access load across as many disks as required. By creating a tablespace for ODS, one for PDS and yet another for partitioned indexes, you can spread the load across several disks.

It is also possible to create a separate tablespace for each partition and spread the tablespaces across multiple disks.

It is recommended that you separate partition tables from nonpartitioned tables because storage requirements often change for partitioned tables. These tables are involved in pulled and planning data storage.

Refer to Metalink, *Oracle8i Concepts*, "Partitioned Tables, and Indexes" and *Oracle8i Administrator's Guide*, "Managing Partitioned Tables and Indexes" before making any changes.

Managing Partitioned Tables

- You can use the ALTER TABLE command to add, drop, exchange, move, modify, rename, split, and truncate partitions.
- Place large table partitions in different tablespaces. Place these tablespaces in data files that reside on different disks.

ORACLE

Managing Partitioned Tables

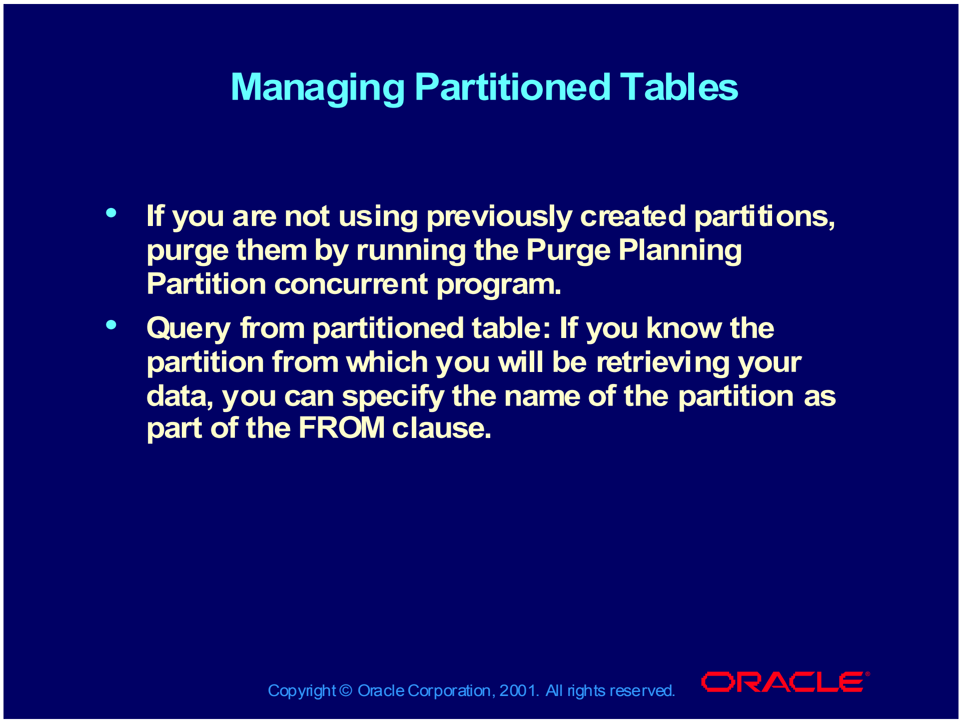
By default, Oracle APS Rapid Install will create local indexes.

Tablespace management is a Database Administrator function.

You may have to use the ALTER TABLE command to perform routine maintenance on frequently used partitioned tables. For example, the partition range column may have changed significantly, causing data fragmentation.

It is advised that you do not create and maintain empty partitions. Currently, the Cost-Based Optimizer (CBO) divides the total number of rows found in a table and its partitions and then divides the total number of rows by the count of existing partitions. This result is used by the CBO to plan for the execution of queries. This could skew the CBO calculations, thus causing possible performance problems. This will be repaired in the future.

When a table is partitioned, its data is stored in separate tables. When an index is created on the partitioned table, the index can be partitioned so that each of the table partitions has a matching index partition. The matching index partitions are called local indexes.



Managing Partitioned Tables (continued)

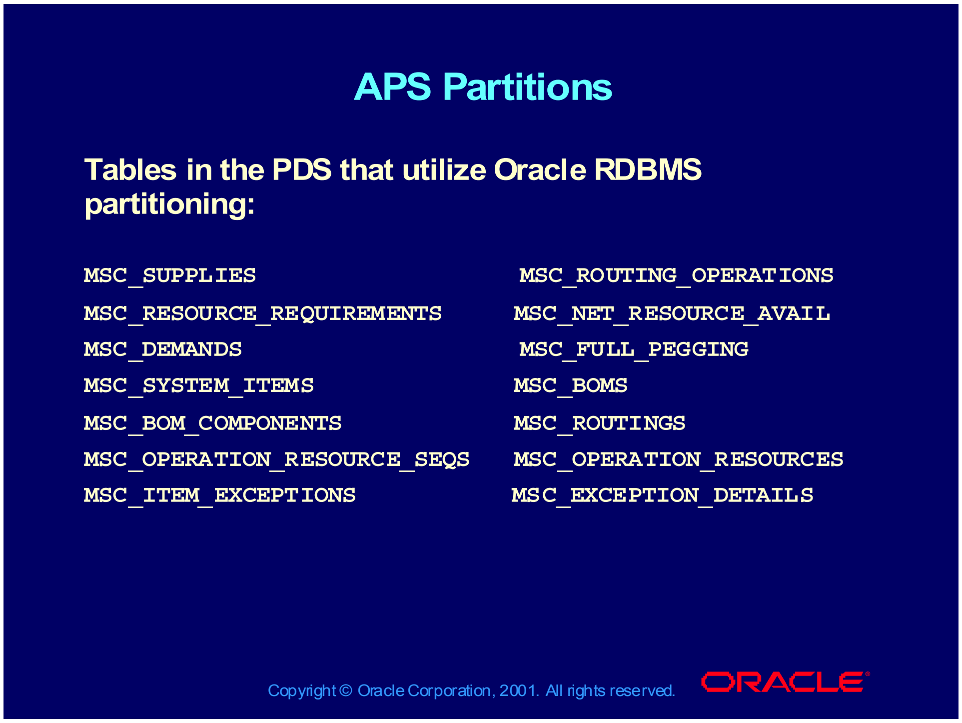
Select *

from msc_system_items (PART4)

where segment1 = 'MY-PART';

In this example, the partition to be searched for information regarding MY-PART is explicitly named.

Generally, you do not identify your partition. You let Oracle manage this activity. Partitions may be dynamic, causing problems with your query in the future.

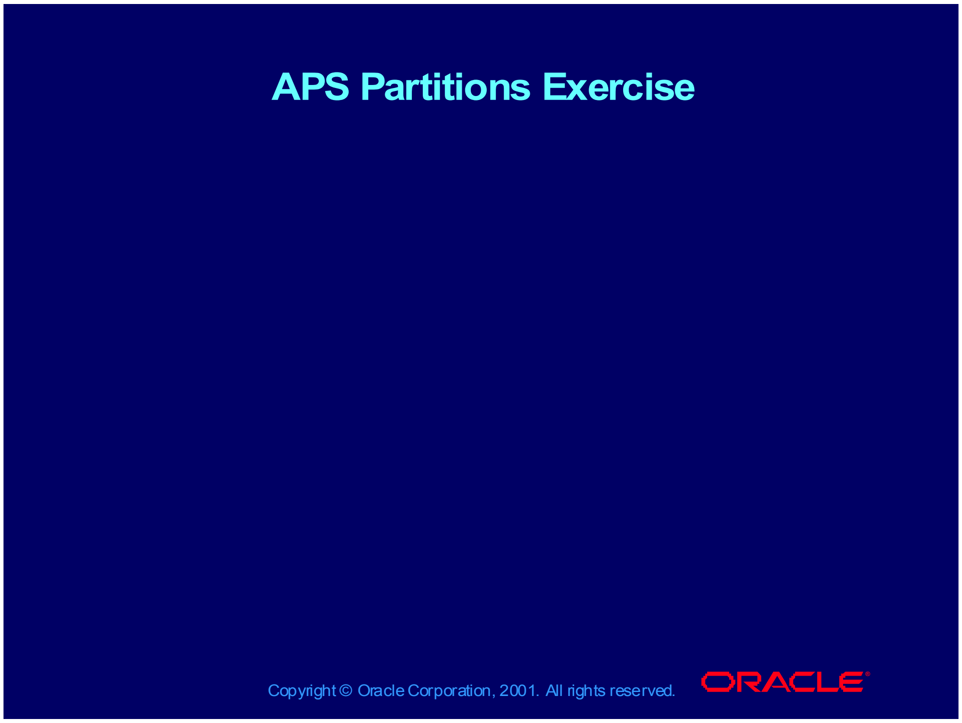


APS Partitions

Range partitioning maps rows to partitions based on the range of a column. All partitions in the planning tables share the same column and constraint definitions as the actual planning tables.

The tables involved in the planning instance are listed on the slide.

The Create Plan Partitions concurrent program is the only supported means of creating instance or planning partitions.



APS Partitions Exercise

1. Log in to SQL*Plus on the destination instance.
2. Verify that the partitioning option is by running the following query:

```
select value
 from v$option
 where parameter = 'Partitioning';
 VALUE
 -------------------
 TRUE
```

3. Look at the tables that control the partitioning in APS:

```
desc msc_plan_partitions
 desc msc_share_plan_partitions
```

4. Look at the contents of msc_plan_partitions:

```
select  b.partition_number,
            b.plan_id,
            b.plan_name,
            a.compile_designator,
            b.free_flag
 from    msc_plans a, msc_plan_partitions b
```

```
where   a.plan_id = b.plan_id
order by b.partition_number;
```

5. Using the all_tables tables, determine the tables that are partitioned:

```
Select  table_name,
            partitioned
 from    all_tables
where   table_name like 'MSC%'
and partitioned = 'YES';
TABLE_NAME                   PAR
--------------------------   ------
MSC_BOMS                      YES
MSC_BOM_COMPONENTS            YES
MSC_DEMANDS                   YES
MSC_EXCEPTION_DETAILS         YES
MSC_FULL_PEGGING              YES
MSC_ITEM_CATEGORIES           YES
MSC_ITEM_EXCEPTIONS           YES
MSC_NET_RESOURCE_AVAIL        YES
MSC_OPERATION_RESOURCES       YES
MSC_OPERATION_RESOURCE_SEQS   YES
MSC_RESOURCE_REQUIREMENTS     YES
MSC_ROUTINGS                  YES
MSC_ROUTING_OPERATIONS        YES
MSC_SALES_ORDERS              YES
MSC_SUPPLIES                  YES
MSC_SYSTEM_ITEMS              YES

16 rows selected.
```

6. Use the following query to return the partitions used by the instance as well as the Operational Data Store and the Planning Data Store. These objects are part of the MSC schema.

```
select substr(object_name,1,15) obj_name,
           substr(subobject_name,1,15) subobj_name,
           object_type,
           object_type,
           status
 from dba_objects
 where subobject_name like '%__%'
```

```
  and object_type not in ('JAVA CLASS','INDEX
PARTITION')
  and owner = 'MSC'
  and object_name = 'MSC_SYSTEM_ITEMS';
```

Key Transformations

Chapter 6

Key Transformations

Agenda

Agenda

- Objectives
- Key transformation overview
- Key transformation during PDP
- Key transformation: item
- Key transformation: business entities

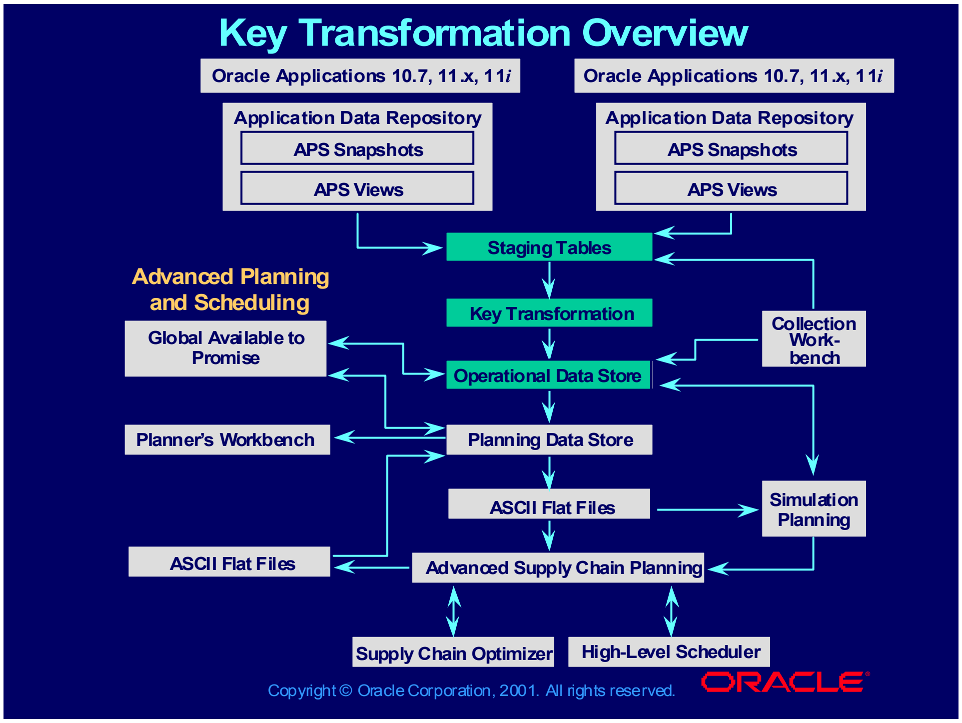
Objectives

Objectives

After completing this chapter, you should be able to do the following:

- Describe key transformation into APS
- Map key transformation using SQL*Plus

ORACLE



Key Transformation Overview

As data is collected from source instances, each distinct entity is combined within APS.

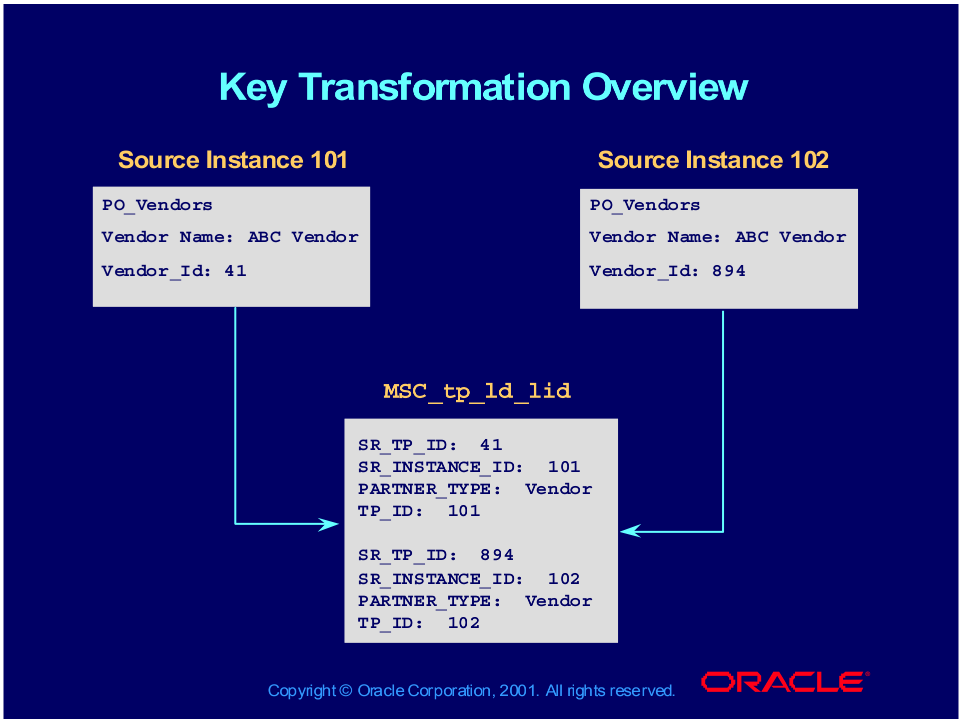
Key transformation is provided for the internal item number, inventory_Item_Id, as well as for the business entities:

- Category Sets
- Suppliers
- Supplier Sites
- Customers
- Customer Sites

The first step is to generate the new APS local ID for the new source key that does not exist in the mapping tables.

The second step is to transform the source ID to the local ID.

Key Transformation Overview



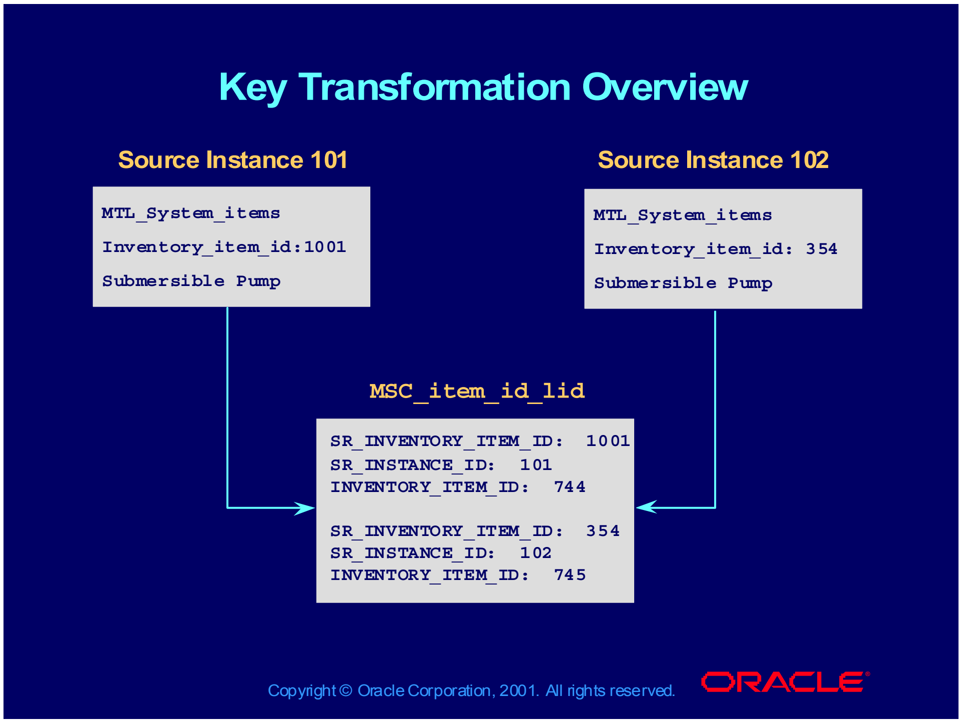
Key Transformation Overview (continued)

To track data entities back to the source instance, APS transforms each unique business entity into a unique ID within APS.

The slide shows ABC Vendor from two different organizations. As the key transformation process executes, APS creates a row in MSC_TP_ID_LID for each vendor. In this example, two rows are created in the key transformation table. Each row contains the source instance ID, partner type, source vendor ID, and the newly created APS local ID. TP_ID contains the unique primary key for APS processing.

Each table, within APS, that ends with LID is a local ID mapping table.

Key Transformation Overview



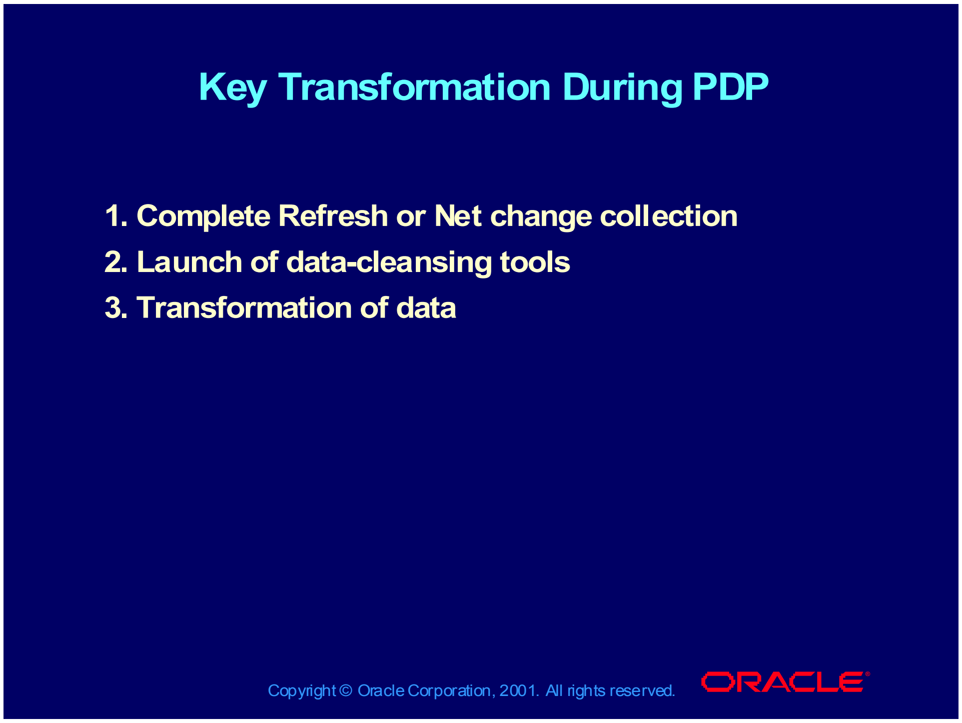
Key Transformation Overview (continued)

To track data entities back to the source instance, APS transforms each unique business entity into a unique ID within APS.

The slide shows a Submersible Pump in both organizations. As the key transformation process executes, APS creates a row in MSC_ITEM_ID_LID for each distinct Inventory_Item_Id. In the example, two rows are created in the key transformation table. Each row contains the source instance ID, the source Inventory_Item_Id, and the newly created APS local ID. Inventory_item_ID contains the unique primary key for APS processing.

Each table, within APS, that ends with LID is a local ID mapping table.

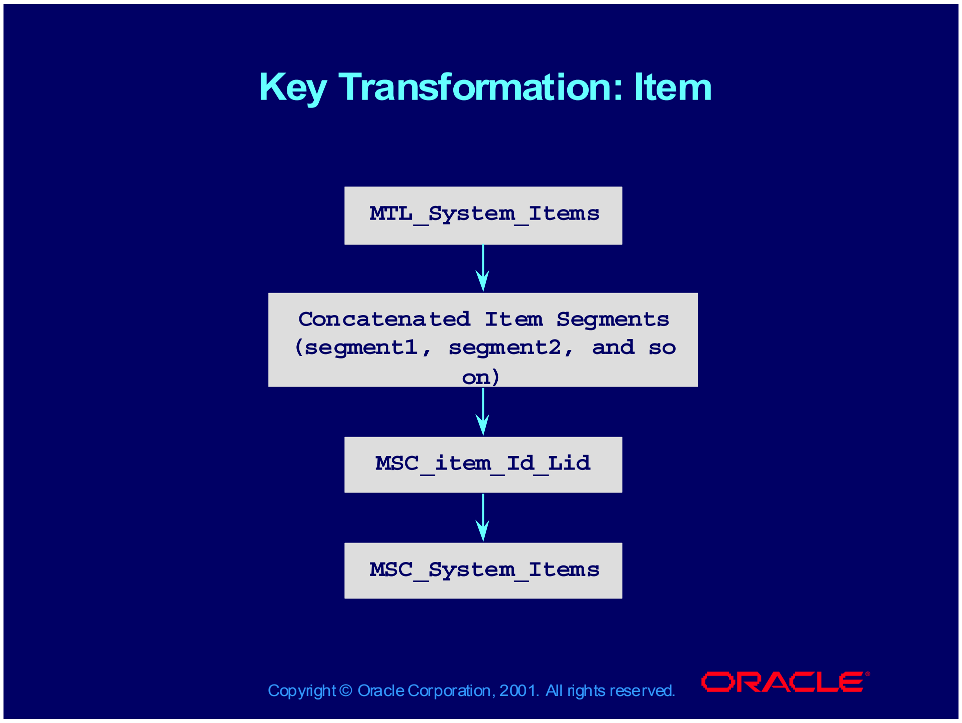
Key Transformation During PDP



Key Transformation During PDP

- Data residing within the staging tables is still identified by the unique source key. This data has not been transformed.
- Data is cleansed, by user-provided tools, before the key transformation takes place.
- After the source data is transformed, the LID tables stay intact even during a complete refresh.

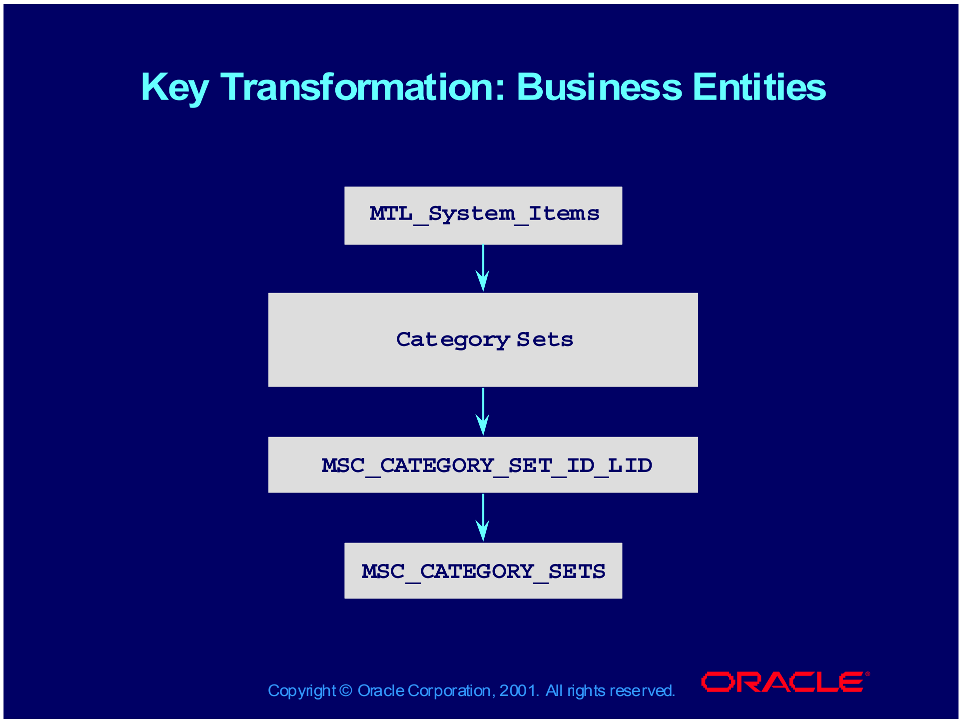
Key Transformation: Item



Key Transformation: Item

In this example of Inventory_Item_Id transformation, the data finally resides within the MSC_System_Items table.

Note that the MSC_System_Items contains the organization_id of the item from the source instance as well.



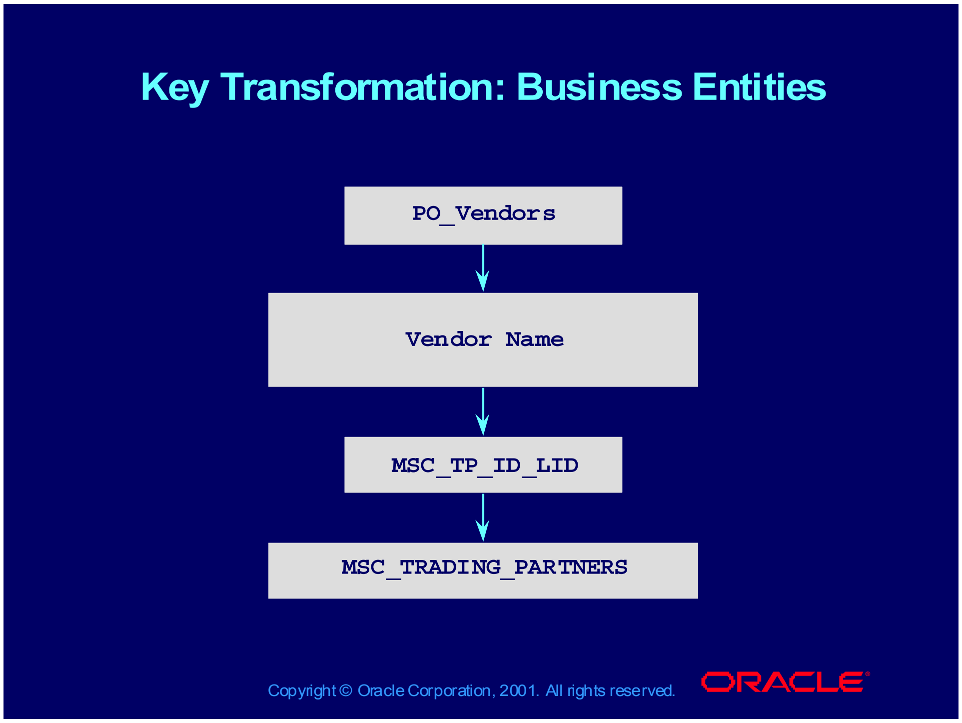
Key Transformation: Business Entities

If it is in the Complete Refresh mode, all the ODS data of this instance is flagged as deleted first. Then the staging table data will be treated as an insert or an update based on its existence in the ODS.

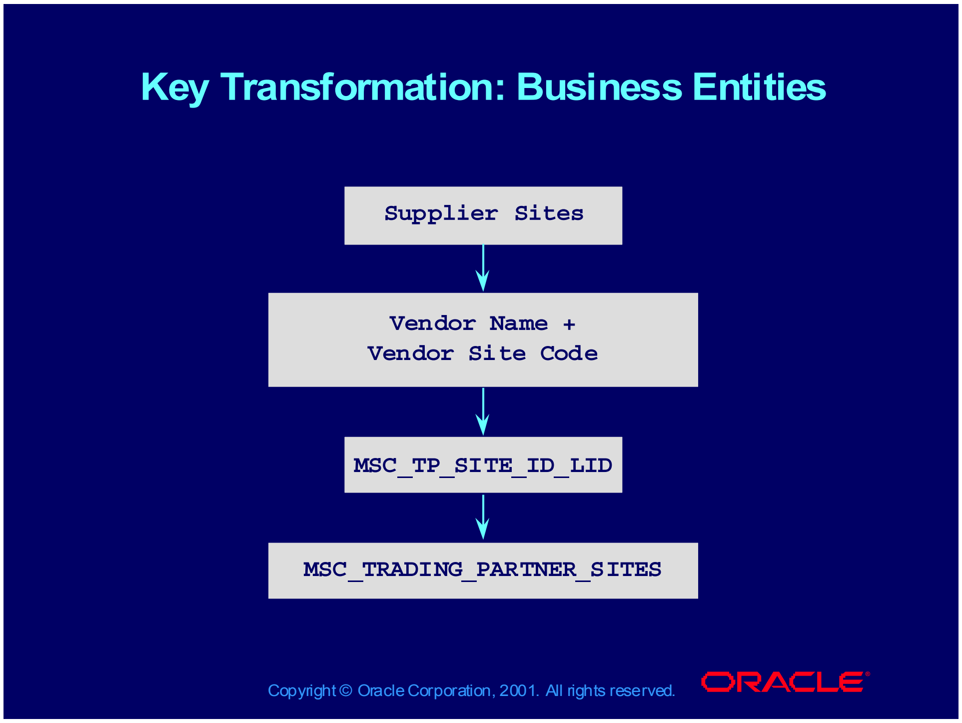
If the ODS data is in the Incremental Refresh mode, and if the source data is for deletion, an update or a delete will be performed according to the relationship between the source data and the ODS record. If the source data is for insertion or update, an update or an insert will be performed according to the relationship between the source data and the ODS record.

The chart describes the relationships within APS.

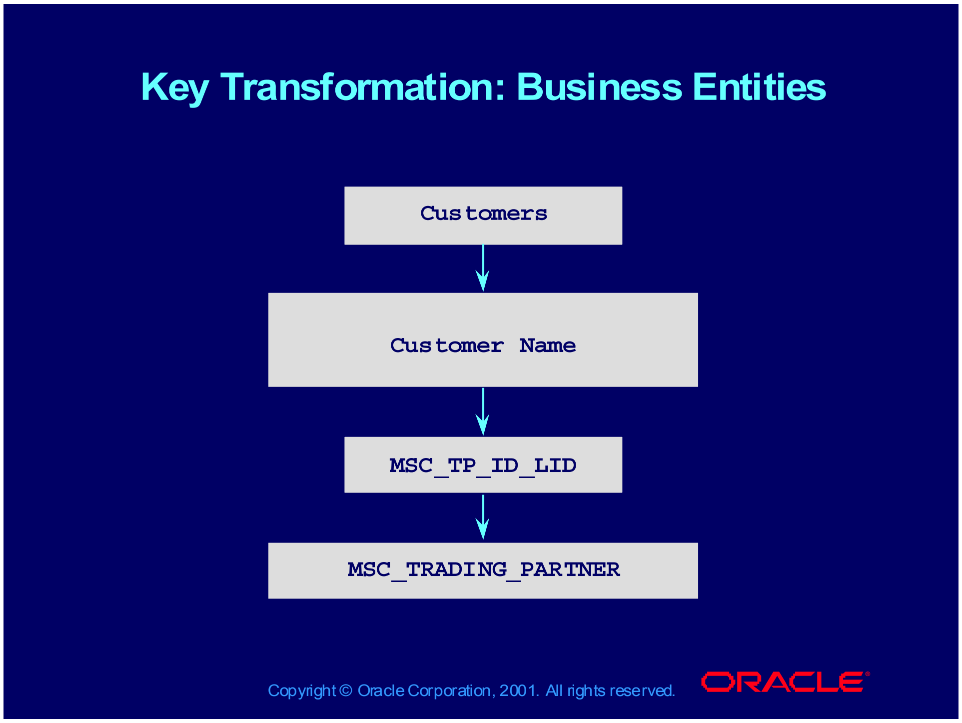
Key Transformation: Business Entities

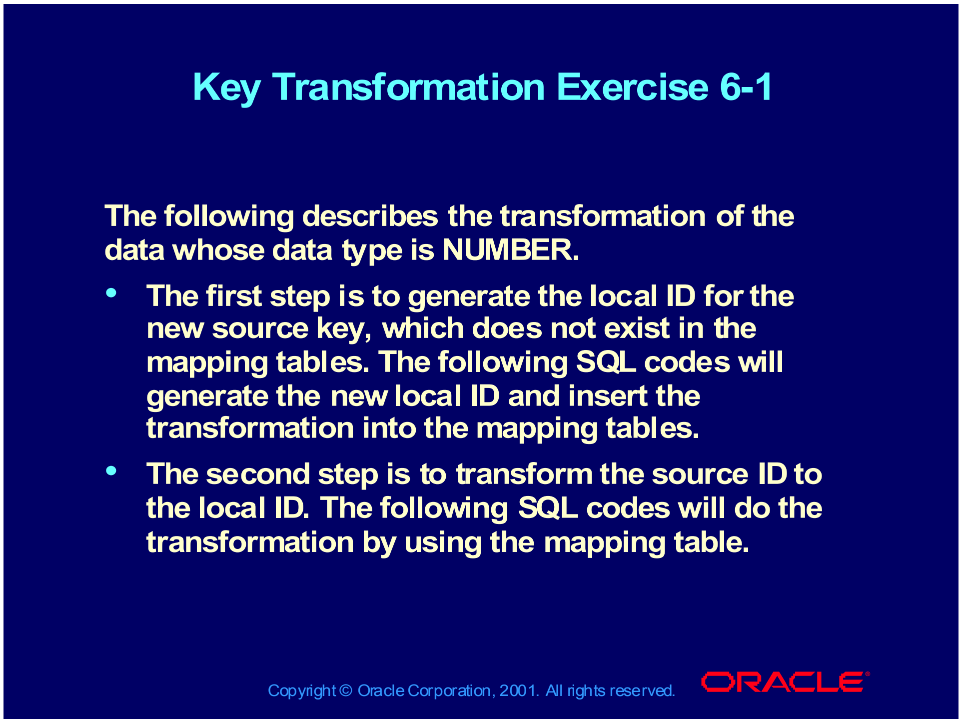


Key Transformation: Business Entities



Key Transformation: Business Entities





Key Transformation Exercise

1. Log in to SQL*Plus on the destination instance.

```
desc msc_instance_orgs
    Name                 Null?             Type
------------       ---------------    ---------------
SR_INSTANCE_ID         NOT NULL           NUMBER
ORGANIZATION_ID        NOT NULL           NUMBER
LAST_UPDATE_DATE       NOT NULL           DATE
LAST_UPDATED_BY        NOT NULL           NUMBER
CREATION_DATE          NOT NULL           DATE
desc msc_apps_instances
Name                 Null?                Type
----------       ---------------    ------------------
INSTANCE_CODE    NOT NULL           VARCHAR2(3)
APPS_VER         NOT NULL           NUMBER
INSTANCE_TYPE    NOT NULL           NUMBER
DBS_VER          NOT NULL           NUMBER
A2M_DBLINK
VARCHAR2(128)
M2A_DBLINK
```

```
VARCHAR2(128)
ENABLE_FLAG     NOT NULL         NUMBER
LRID                             NUMBER
LRTYPE
VARCHAR2(1)
APPS_LRN        NOT NULL          NUMBER
INSTANCE_ID     NOT NULL          NUMBER

  desc MSC_ITEM_ID_LID
  Name                          Null?          Type
  -------------------    -------------    ---------
  SR_INVENTORY_ITEM_ID    NOT NULL          NUMBER
  SR_INSTANCE_ID          NOT NULL          NUMBER
  INVENTORY_ITEM_ID       NOT NULL          NUMBER

  desc msc_system_items
  Name                        Null?              Type
--------------------   ---------------   ----------
  PLAN_ID                 NOT NULL          NUMBER
  ORGANIZATION_ID         NOT NULL          NUMBER
  INVENTORY_ITEM_ID       NOT NULL          NUMBER
  SR_INSTANCE_ID          NOT NULL          NUMBER
  SR_INVENTORY_ITEM_ID                      NUMBER
  ITEM_NAME
  VARCHAR2(40)
  LOTS_EXPIRATION                           NUMBER
  LOT_CONTROL_CODE        NOT NULL          NUMBER
  SHRINKAGE_RATE                            NUMBER
```

Refer to the *APS Technical Reference Manual* for more detailed information.

2. Transform the items:

```
 select *
  from msc_item_id_lid
  where inventory_item_id <200;

SR_INVENTORY_ITEM_ID  SR_INSTANCE_ID  INVENTORY_ITEM_ID
-------------------   -------------  -------------------
                209              21                185
                398              21                399
                484              21                195
                920              21                188
```

```
922                 21                  189
924                 21                  98
926                 21                  99

11 rows selected.
```

3. Verify that APS SR_INVENTORY_ITEM_ID is the actual inventory_item_id in the source instance:

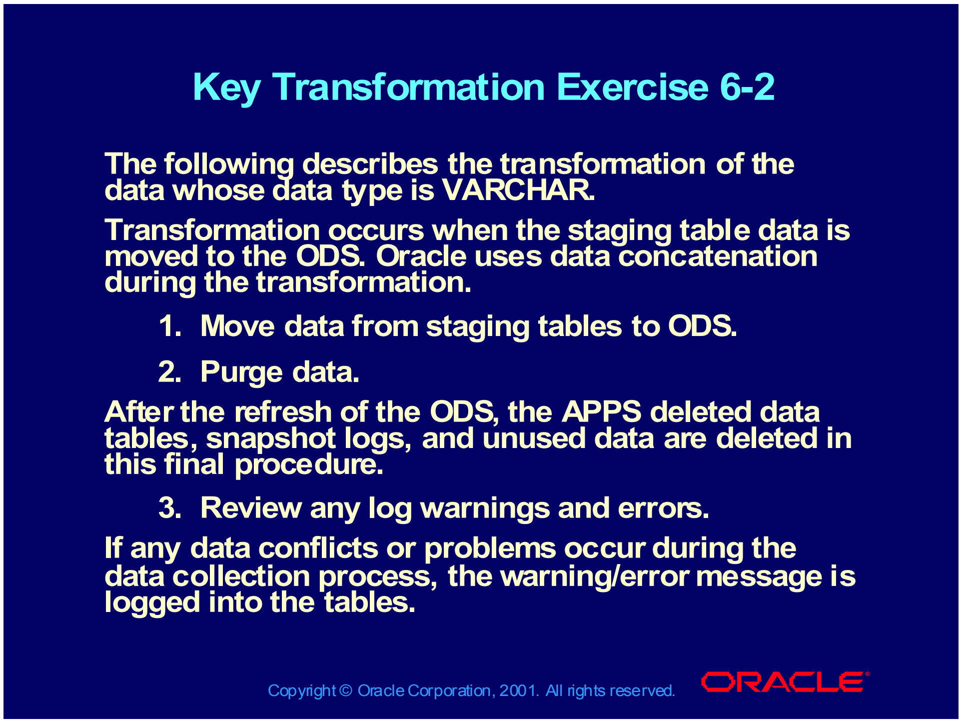
```
select distinct substr(segment1,1,15) part_no
from mtl_system_items@MSC_TO_SOURCE db link name here
where inventory_item_id in (185,399,195,188,189,98,99);

PART_NO
---------------
CM23591
CM23592
CM23595
CM42581
CM86324
CM93501

7 rows selected.
```

4. Tie the source and planning instances together:

```
select substr(a.segment1,1,20) "source item name",
        substr(c.item_name,1,20) "APS item name"
  from  mtl_system_items@MSC_TO_SOURCE db link name
here a,
        msc_item_id_lid b,
        msc_system_items c,
        msc_instance_orgs d
  where c.inventory_item_id in (185,399)
  and b.inventory_item_id = c.inventory_item_id
  and a.inventory_item_id = b.sr_inventory_item_id
  and d.organization_id = c.organization_id
  and a.organization_id = d.organization_id;
```

```
source item name        APS item name
-----------------       --------------------
CM93501                   CM93501
CM93503                   CM93503

2 rows selected.
```

Key Transformation Exercise 6-2



Key Transformation Exercise

1. Log in to the destination instance.

```
desc msc_tp_id_lid
```

2.

```
 Name               Null?          Type
 --------           ------------   ---------
 SR_TP_ID           NOT NULL       NUMBER
 SR_INSTANCE_ID     NOT NULL       NUMBER
 PARTNER_TYPE       NOT NULL       NUMBER
 TP_ID              NOT NULL       NUMBER
```

3.

```
select vendor_id
 from   po_vendors@MSC_TO_SOURCE db link name here
 where  vendor_name like 'Vision%';
 VENDOR_ID
 -----------------
               41
```

4.

```
select *
 from   msc_tp_id_lid
```

```
  where sr_tp_id = 41;

SR_TP_ID          SR_INSTANCE_ID   PARTNER_TYPE      TP_ID
----------      ------------     ----------------  --------
      41                21                     1        184
```

5.

```
 select substr(partner_name,1,30)"APS Partner Name"
  from   msc_trading_partners
  where  partner_id = 184;

  APS Partner Name
  -----------------------
  Vision
```

6. Combine source and destination:

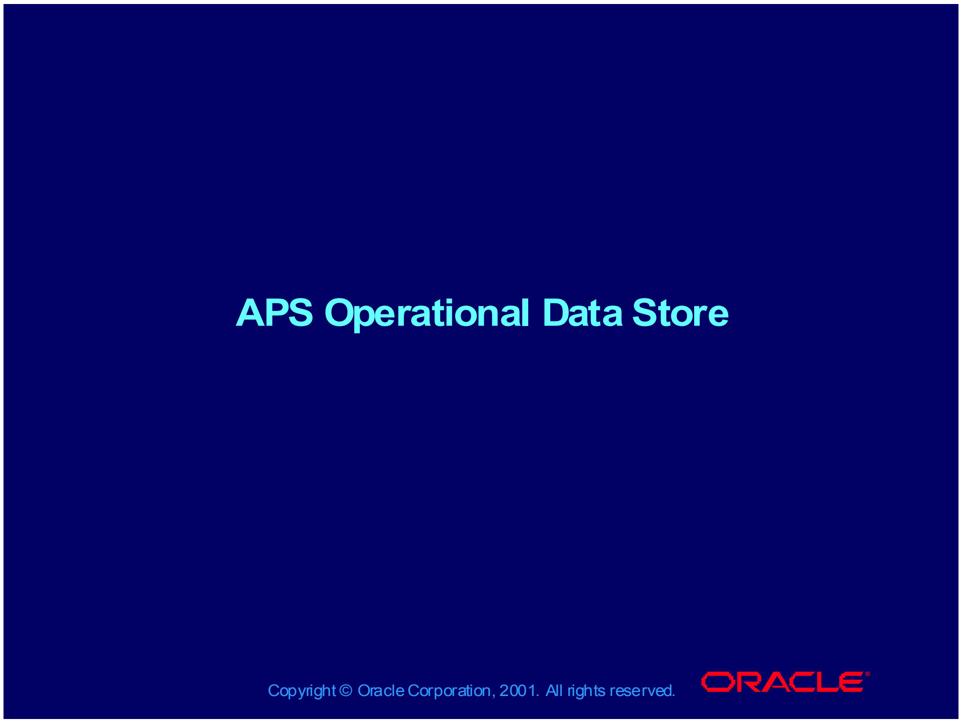
```
 select substr(vendor_name,1,25) "Source Vendor Name"
  from   po_vendors@MSC_TO_SOURCE db link name here
  where  vendor_id in
          (select sr_tp_id
           from msc_tp_id_lid)
  and    vendor_name like 'Vision%';

  Source Vendor Name
  --------------------------
  Vision
  1 rows selected.
```

7.

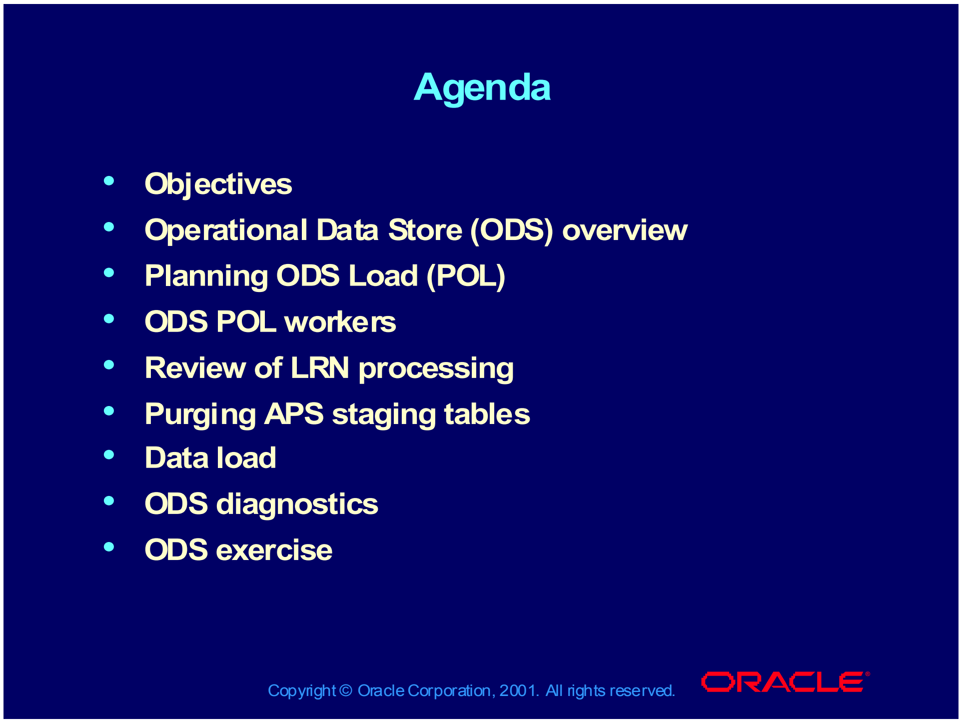
```
 select substr(a.vendor_name,1,30) "Source Name",
         substr(b.partner_name,1,30) "APS Name"
  from   po_vendors@MSC_TO_SOURCE db link name here
a, msc_trading_partners b
  where  a.vendor_id = b.sr_tp_id
  and    a.vendor_name like 'Vision%';

  Source Name           APS Name
  ----------------      --------------
  Vision                 Vison
  1 rows selected.
```

APS Operational Data Store

Chapter 7


APS Operational Data Store
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Agenda



Agenda
• Objectives
• Operational Data Store (ODS) overview
• Planning ODS Load (POL)
• ODS POL workers
• Review of LRN processing
• Purging APS staging tables
• Data load
• ODS diagnostics
• ODS exercise
Copyright © Oracle Corporation, 2001. All rights reserved.

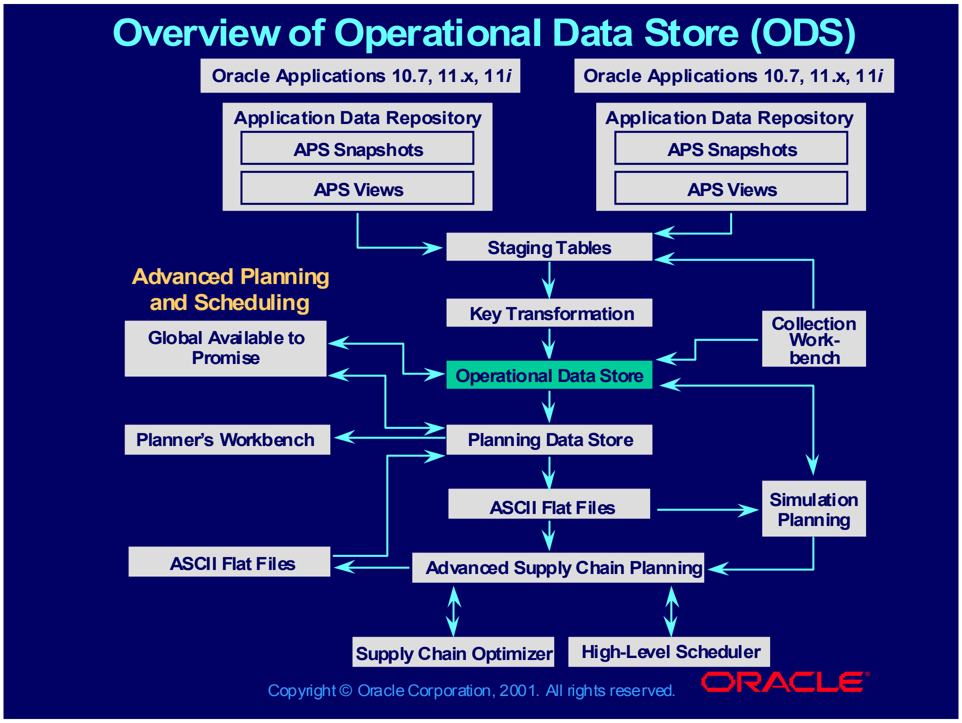
Objectives

Objectives

After completing this chapter, you should be able to do the following:

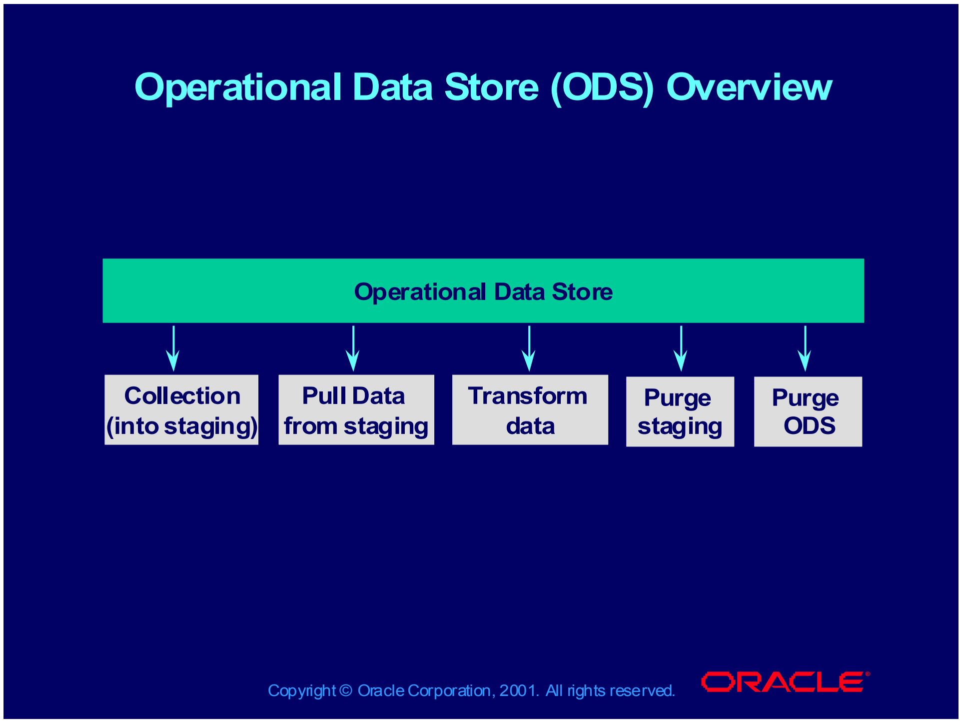
- Describe the role of the ODS in the data collection process
- Determine the processes running during an ODS load
- Explain the use of LRN
- Describe purging staging tables during ODS load
- Describe actions that cause ODS load failure

Overview of Operational Data Store (ODS)

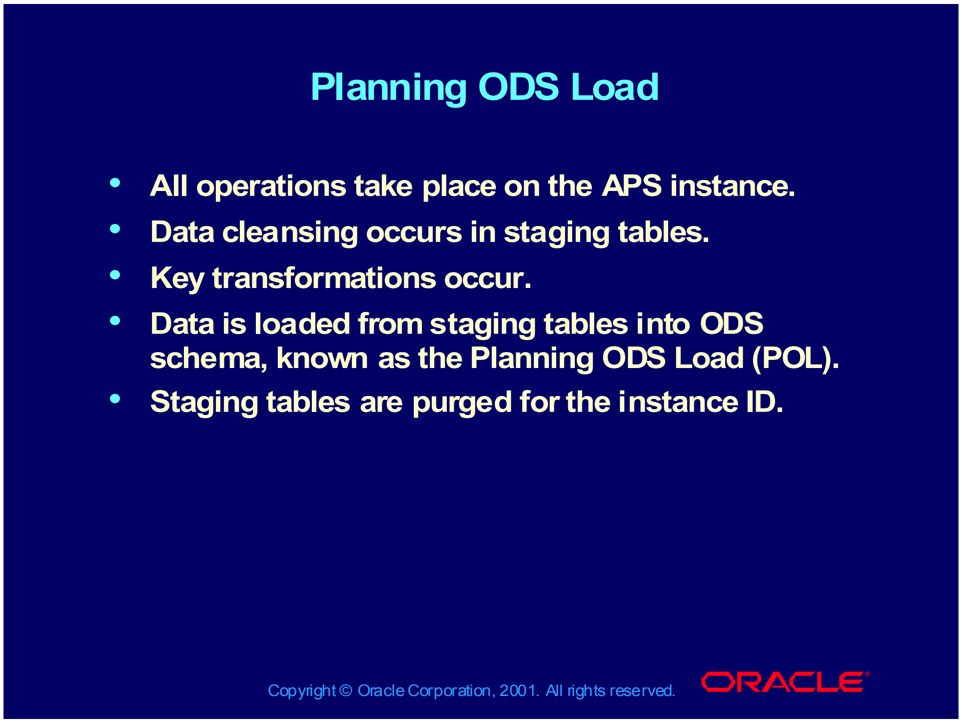


Overview of Operational Data Store (ODS) Overview

- All the data that is required for planning is first collected from the source instances into the Planning Server staging tables eventually residing in the ODS.
- Each source instance can have its own refresh interval.
- Failure in the data collection process of one source instance will not affect the data collection of other instances unless the failure occurs on the destination or APS instance.
- The data collection process is required both in the decentralized and centralized modes of operation.
- The data collection process polls the source database snapshot tables instead of the application tables.
- Database snapshots used by APS are created in the schemas for the corresponding applications on the source instance.
- Plan_id = -1 for all ODS data. This is plan-neutral data that has not been associated with a plan.
- ODS data can also be used for Global Available to Promise.


Operational Data Store (ODS) Overview
Operational Data Store
Collection (into staging)
Pull Data from staging
Transform data
Purge staging
Purge ODS
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Planning ODS Load



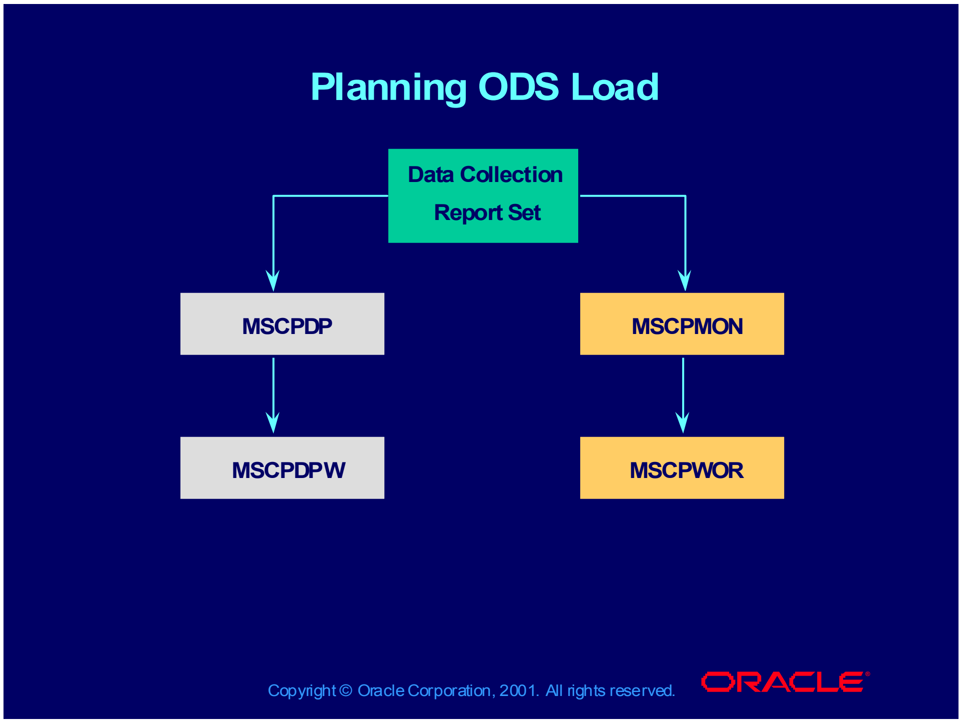
Planning ODS Load

If launched in Complete Refresh mode, all of the ODS data associated with the collected instance is flagged as deleted first.

The staging table data will be treated as an insert or an update based on its existence in the ODS.

If the ODS data is in the Incremental Refresh mode, and if the source data is for deletion, an update or a delete will be performed according to the relationship between the source data and the current state of the row in the ODS.

If the source data for insertion or update, an update or insertion will be performed according to the relationship between the source data and the ODS record.



Planning ODS Load (continued)

- POL (Planning ODS Load) is the second request in the Data Collection Report Set that runs sequentially after MSCPDP on the APS instance.
- The POL concurrent request launches the POL monitor. This process first checks the status of data collection for the corresponding instance (INSTANCE_ID is passed as parameter when the report set is launched).
- The POL monitor updates the value of ST_STATUS in MSC_APPS_INSTANCES to 3 to indicate that Planning ODS Load is in process.
- The POL monitor performs the key transformation as well. See the “Key Transformations” module.

ODS POL Workers

- The POL monitor launches multiple workers based on the No. of Workers parameter specified in the input parameters for the POL request during the data collection launch.
- Multiple POL workers perform different tasks and communicate to each other using database pipes. POL workers transform the data according to its status (insert, update, or delete) in the staging tables for the different entities and refresh the data in the ODS.
- For each refresh process, the data is updated with the Last_Collection_Refresh_Number (LCRN).
- If the status of the transformed data is “delete,” the corresponding data in the ODS is flagged as deleted.

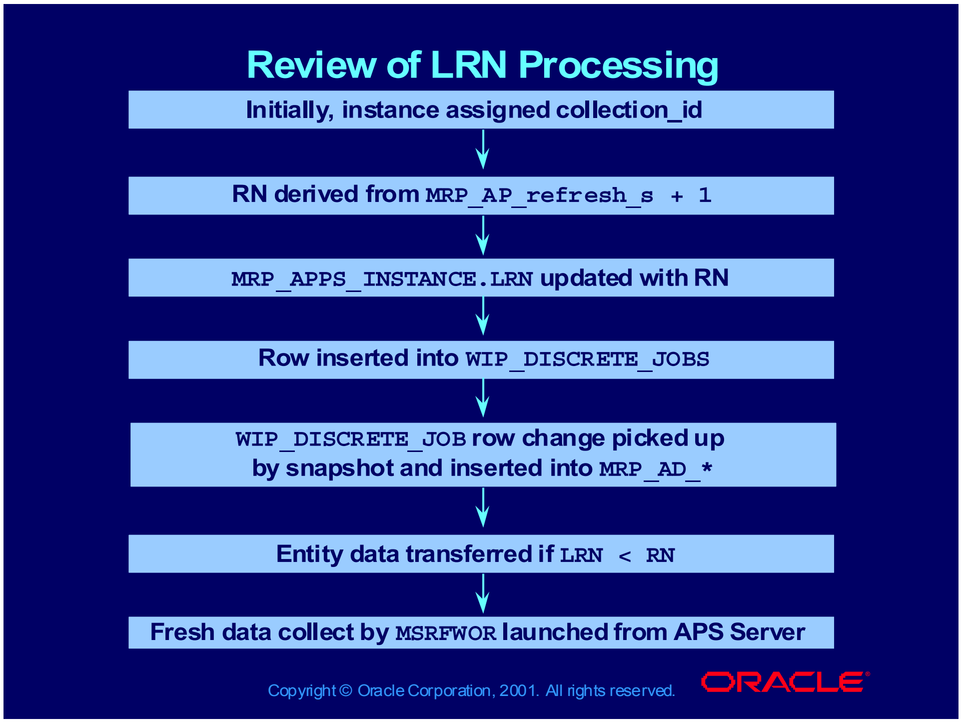
POL Workers

Launch the workers.

The pull program improves performance by identifying and distributing data transformation and collection tasks that could be done in parallel.

The POL monitor waits for all workers to complete processing. After all the workers complete successfully, the Last refresh type is updated to Null in MSC_APPS_INSTANCES for the source instance code.

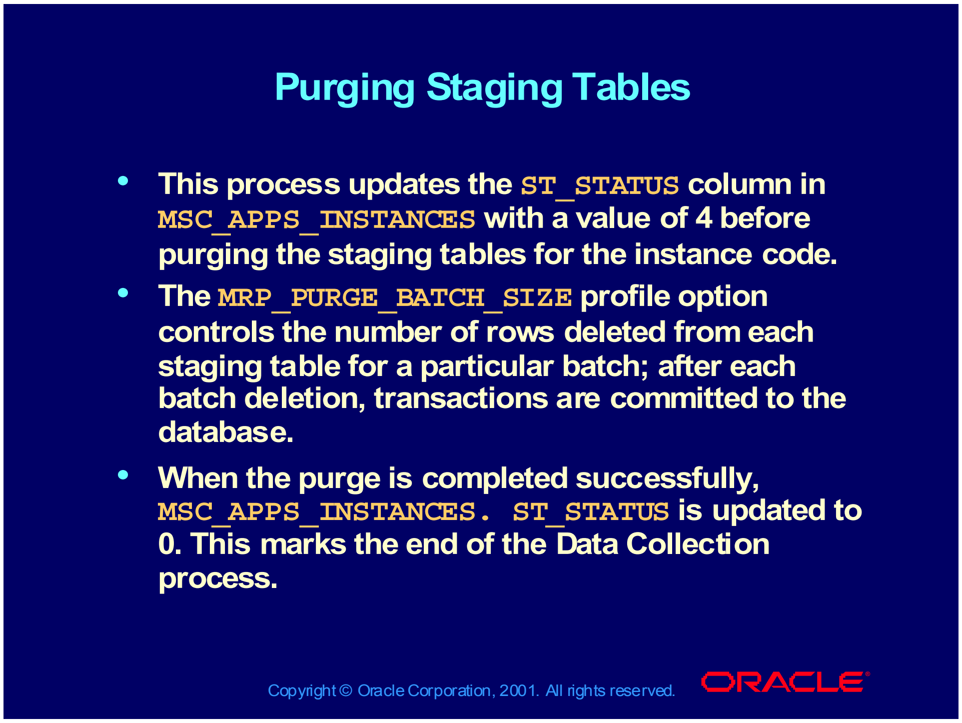
The monitor then launches the Purge Staging Tables concurrent request (MSCPDCP).



Review of LRN Processing

A new collection ID is fetched from column NEXTVAL in sequence MSC_COLLECTION_S.

The NEXTVAL is inserted into the Last_Collection_Id (LCID) column in the MSC_APPS_INSTANCES table for the corresponding instance code.



Purging Staging Tables

RBS usage is controlled by the setting of the MRP_PURGE_BATCH_SIZE profile option . A higher number forces increased usage of RBS space. See the “Application, Server, and RDBMS Performance and Maintenance” module.

Data Load

Data Load

Perform the ID transformation, if it is required, and load the data into the ODS. The executable concurrent programs to perform this operation are:

- `MSCLMON` (monitor):

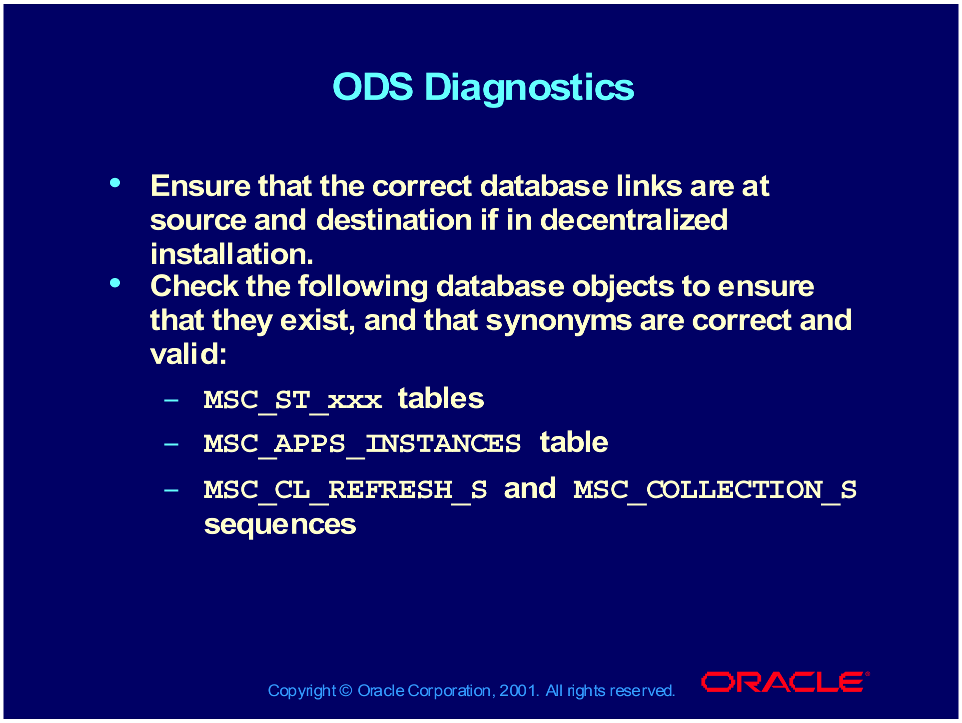
  `PL_SQL` procedure `MSC_CLBA_P.monitor`

- `MSCLWOR` (worker):

  `PL_SQL` procedure `MSC_CLBA_P.worker`

ORACLE

ODS Diagnostics



ODS Diagnostics

```
select table_name
from all_tables
where table_name like 'MSC_ST_%'

TABLE_NAME
------------------------------
MSC_ST_ASSIGNMENT_SETS
MSC_ST_BILL_OF_RESOURCES
MSC_ST_BIS_BUSINESS_PLANS
MSC_ST_BIS_PFMC_MEASURES
MSC_ST_BIS_TARGETS
MSC_ST_BIS_TARGET_LEVELS
MSC_ST_BOMS
MSC_ST_BOM_COMPONENTS
MSC_ST_BOR_REQUIREMENTS
MSC_ST_CALENDAR_DATES
MSC_ST_CALENDAR_SHIFTS  (partial list - BUT - schema
is present)
select   INSTANCE_CODE,
         APPS_VER,
```

```
          DBS_VER,
          substr(A2M_DBLINK,1,20) link,
          substr(M2A_DBLINK,1,20) link2,
          ENABLE_FLAG
from      msc_apps_instances;
INS     APPS_VER    DBS_VER     LINK         LINK2      ENABLE_FLAG
-----   --------    ------     ----------    -------   ------------
crm    3                  0   to_vis51   to_crm51        1
vis    3                  0                              1
select LRID,
       APPS_LRN,
       LCID,
       ST_STATUS,
       CLEANSED_FLAG,
       GMT_DIFFERENCE
from msc_apps_instances;
LRID   APPS_LRN   LCID  ST_STATUS CLEANSED_FLAG  GMT_DIFFERENCE
-----  --------   ----- --------- ------------- ---------------
46        90         46        0             1               8
61        121         2         2                            8
select     owner,
     status
from all_objects
where object_name = 'MSC_CL_REFRESH_S';
OWNER                           STATUS
------------------------------   -------
APPS                              VALID
MSC                               VALID
APPS_MRC                          VALID
select     owner,
     object_name,
     status
from all_objects
where object_name like 'MSC%'
and status != 'VALID';
OWNER           OBJECT_NAME               STATUS
------------- --------------------    ------------
MSC             MSC_ATP_PLAN_SN          INVALID
```

ODS Diagnostics

Planning ODS Load (POL) will fail if any of the following are true:

- The POL process is currently running for the instance that you are attempting to load.
- ODS staging tables are empty:

  `MSC_APPS_INSTANCES. ST_STATUS = 0`
- Planning Data Pull is running:

  `MSC_APPS_INSTANCES. ST_STATUS = 4`
- Data is being purged from the Staging tables for the instance:

  `MSC_APPS_INSTANCES. ST_STATUS = 3`

ODS Diagnostics (continued)

This includes Oracle Applications Data Store (ADS) as well as legacy data. This acts as the input for the Memory-Based Planner snapshot (MSCNSP: MB Snapshot). ODS and PDS (Planning Data Store) share the same physical database objects.

A special plan identifier (-1) is used for the distinction between the ODS and PDS.

A special plan identifier (-99999) is used for Global Available to Promise.

ODS Diagnostics Exercise

Investigating LRN

ODS Diagnostics Exercise

1. Log in to the destination instance.
2.

```
Select   last_number
  from     all_sequences
  where    sequence_name = 'MSC_COLLECTION_S';
  LAST_NUMBER
  ----------------------
                       21
```

3. Identify collection data entity:

```
select   instance_id,
             lrn,
             lrd,
             sn_status
  from     mrp_ap_apps_instances;
INSTANCE_ID      LRN          LRD        SN_STATUS
------------    --------    ----------  ------------
          21        153      10-AUG-00              -1
```

3.

```
  select last_number
```

```
from   all_sequences
where  sequence_name = 'MRP_AP_REFRESH_S';
LAST_NUMBER
---------------------
                     164
```

4.

```
desc MRP_AD_DSCR_JOBS
Name                 Null?              Type
------------------   --------------   -----------
WJS_NET_QTY_FLAG                         NUMBER
WJS_MPS_NET_QTY_FLAG                     NUMBER
JOB_TYPE                                 NUMBER
WIP_ENTITY_ID                            NUMBER
LAST_UPDATE_DATE     NOT NULL            DATE
LAST_UPDATED_BY      NOT NULL            NUMBER
CREATION_DATE        NOT NULL            DATE
CREATED_BY           NOT NULL            NUMBER
LAST_UPDATE_LOGIN                        NUMBER
REQUEST_ID                               NUMBER
PROGRAM_APPLICATION_ID                   NUMBER
PROGRAM_ID                               NUMBER
PROGRAM_UPDATE_DATE                      DATE
RN                                       NUMBER
```

5. Check the rows that were trapped when source data was deleted:

```
 Select wjs_net_qty_flag,
        wip_entity_id,
        rn
from   MRP_AD_DSCR_JOBS;

WJS_NET_QTY_FLAG   WIP_ENTITY_ID            RN
----------------  ---------------------     -----
             1                   20095     81
             1                   21702     81
             1                   21703     81
```

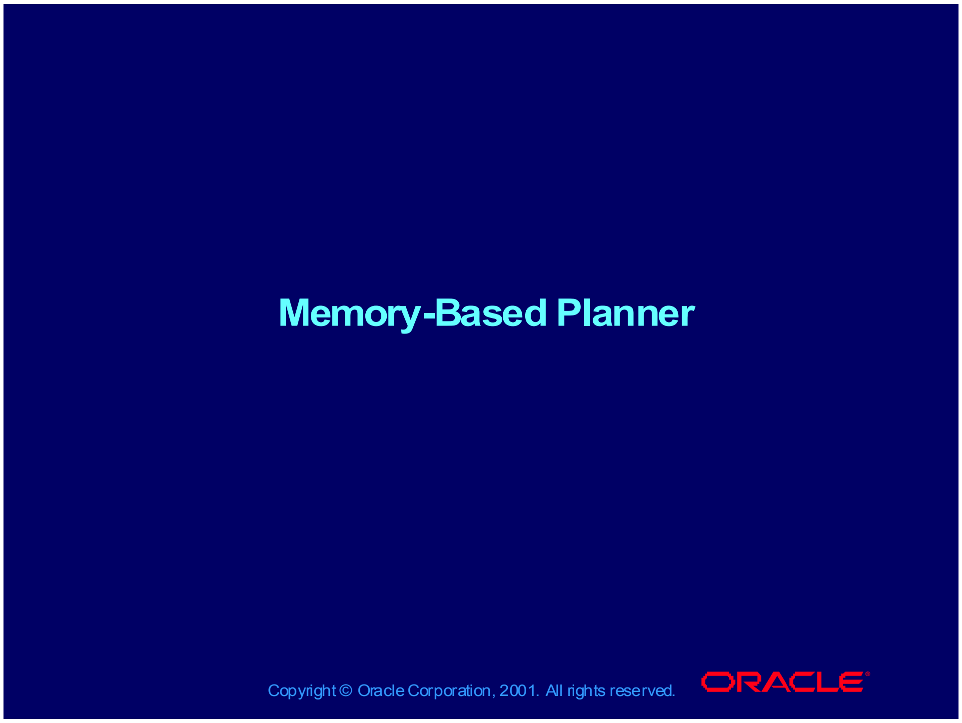
6.

```
select query
from   all_snapshots
```

```
where name = 'WIP_DSCR_JOBS_SN';
SELECT wdj.ALTERNATE_BOM_DESIGNATOR,
       wdj.ALTERNATE_ROUTING_DESIGNATOR,
       wdj.BOM_REVISION,
       wdj.BUILD_SEQUENCE,
       wdj.DEMAND_CLASS,
       wdj.END_ITEM_UNIT_NUMBER,
       wdj.FIRM_PLANNED_FLAG,
       wdj.JOB_TYPE,
       wdj.LINE_ID,
       wdj.MPS_NET_QUANTITY,
       wdj.MPS_SCHEDULED_COMPLETION_DATE,
       wdj.NET_QUANTITY,
       wdj.ORGANIZATION_ID,
       wdj.PRIMARY_ITEM_ID,
       wdj.PROJECT_ID,
       wdj.QUANTITY_COMPLETED,
       wdj.QUANTITY_SCRAPPED,
       wdj.SCHEDULE_GROUP_ID,
       wdj.SCHEDULED_COMPLETION_DATE,
       wdj.SCHEDULED_START_DATE,
       wdj.STATUS_TYPE,
       wdj.TASK_ID,
       wdj.WIP_ENTITY_ID,
       0 RN
FROM WIP.WIP_DISCRETE_JOBS wdj
WHERE wdj.status_type IN (1, 3, 4, 6);
```

Memory-Based Planner

Chapter 8


Memory-Based Planner
Copyright © Oracle Corporation, 2001. All rights reserved.

Agenda

- Objectives
- Overview of Memory-Based Planner
- ILOG technologies
- Supply Chain Optimizer
- High-Level Scheduler
- Memory-Based Planner (MBP)
- MBP database maintenance
- Simulation planning

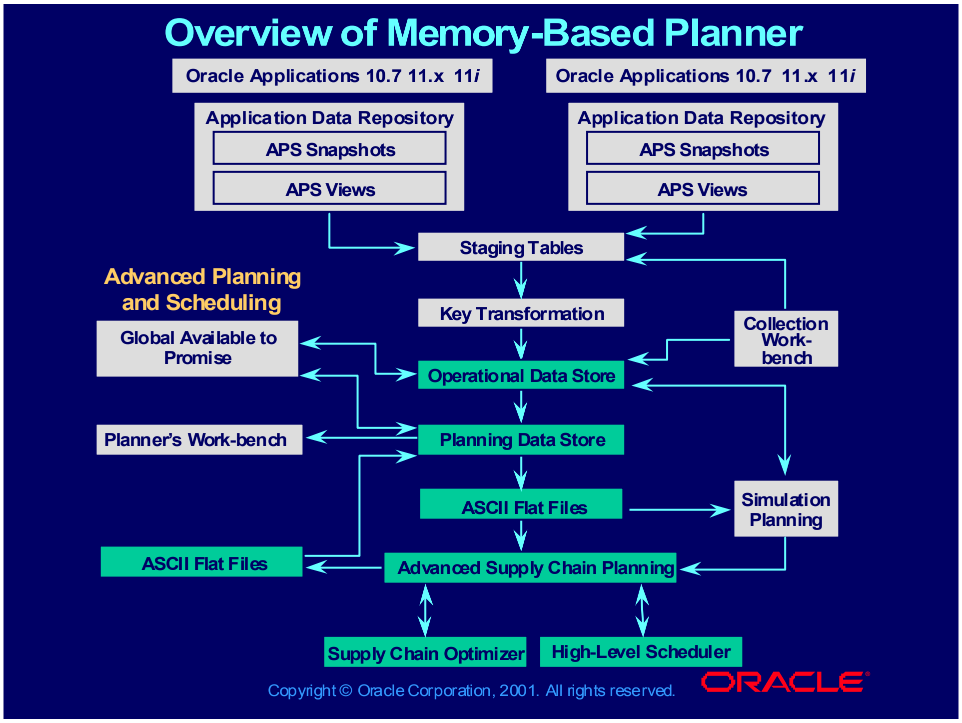
ORACLE

Objectives

Objectives

After completing this chapter, you should be able to do the following:

- Describe MBP functions with APS
- Describe MBP trees
- Describe MBP inputs and outputs



Overview of Memory-Based Planner (MBP)

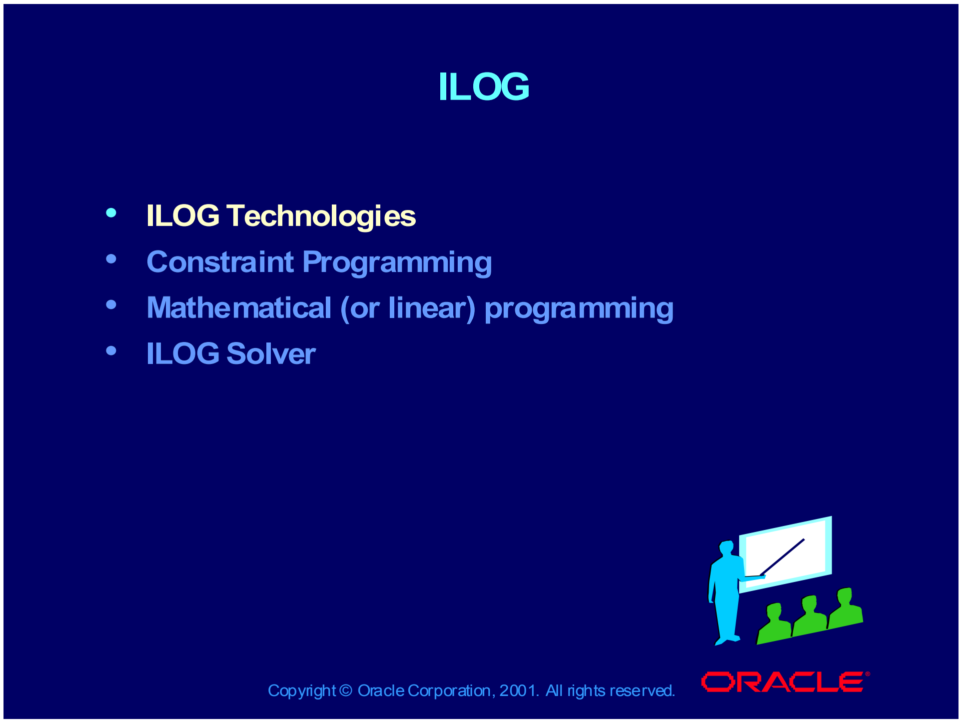
Depending on the mode of operation, the MBP process will require data from the ODS, PDS, or ASCII flat files.

ASCII files are produced before the planning data is inserted into the MBP model.

Interaction between the MBP, Supply Chain Optimizer, and High-Level Scheduler produces a custom plan according to user settings.

Data is written to ASCII flat files before being loaded into the PDS.

Data in the PDS that has a plan_id > -1 is extracted from the ODS.



ILOG Technologies

Only ILOG supports multiple, complementary optimization technologies covering the entire breadth of industry problems, from high-level planning to tactical operations.

ILOG

Constraint Programming

This more recent innovation is a highly effective technology using domain reduction and constraint propagation to efficiently solve problems that are highly combinatorial with highly logical content.

These problems are usually difficult or impossible to represent with linear expressions.

Constraint programming uses information contained in the problem to prune the search space, in order to more rapidly identify feasible solutions. It is ideally suited for operational problems, which require fast, feasible answers.

Users can also guide the search process with their own knowledge of the problem. The constraint programming methodology allows the natural expression of very complex relationships, including logical expressions.

Constraint programming approaches are ideally suited for such operational problems as scheduling, sequencing, configuration, and routing, which are highly combinatorial, more intuitively represented with logical expressions, and require fast, feasible solution.

Mathematical (or Linear) Programming

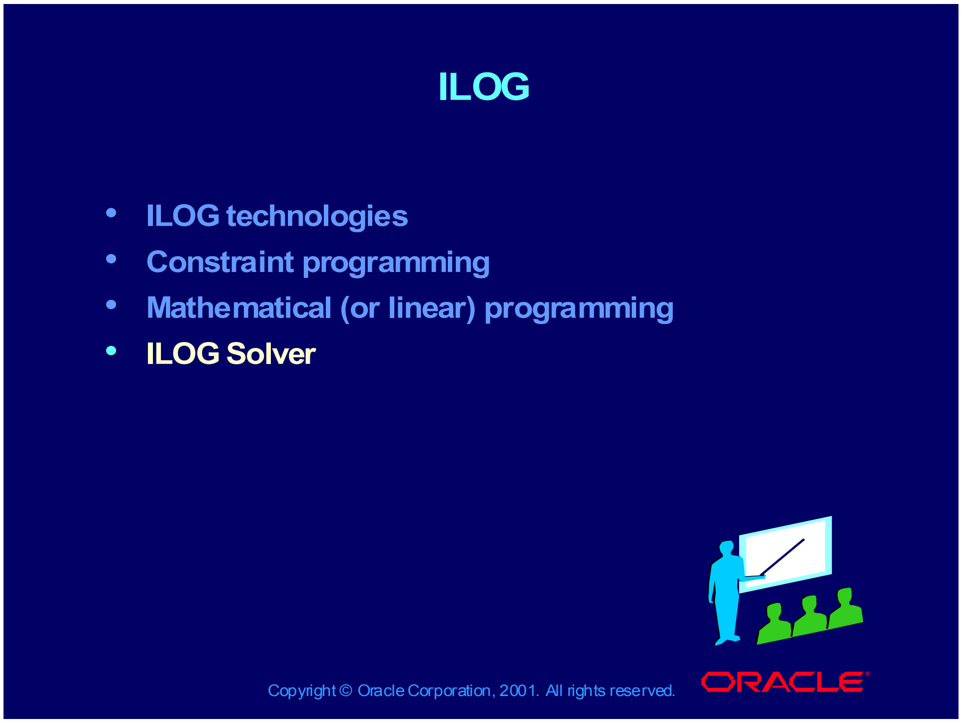
Linear and mixed-integer programming technologies have been employed with tremendous success in the manufacturing, transportation, telecommunications, natural resource, and utility industries since the 1960s.

Mathematical programming approaches for solving hundreds of problem types are well documented, understood, and deployed successfully in most industries. Algorithmic innovations pioneered by ILOG in the past 10 years have improved solution times by more than a factor of 100.

Together with advances in computers, this dramatic improvement means that problems considered unapproachable only a few years ago are now readily solved, many in real time.

Mathematical programming approaches are especially well suited for planning applications. They are unmatched at finding optimal solutions, computing them rapidly with powerful algorithms.

ILOG



ILOG Solver

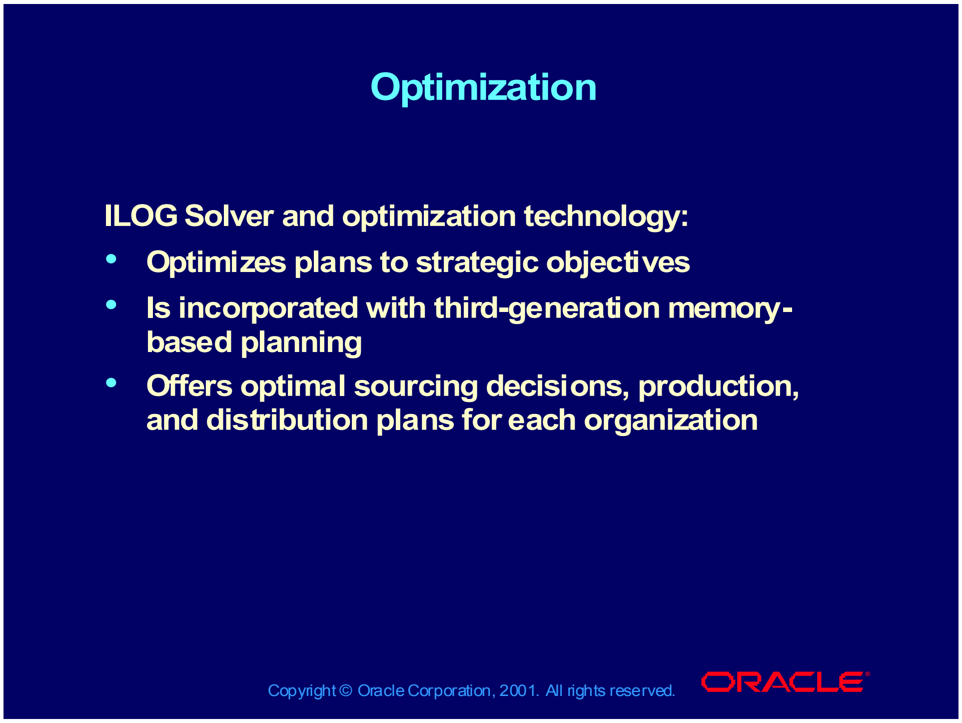
ILOG Solver is a core engine of the ILOG Optimization Suite. It provides cutting-edge optimization technology for powerful scheduling, sequencing, timetabling, configuration, dispatching, and resource allocation applications with logical constraints.

ILOG Solver can be enhanced with add-ons for tackling a wide range of problems. It is a C++ class library that implements the three aspects of constraint programming.

ILOG Scheduler is a library based on ILOG Solver that includes additional components dedicated to finite capacity scheduling problems.

Both are C++ codes that reads data from a data source and creates the variables and constraints of the problem.

Optimization



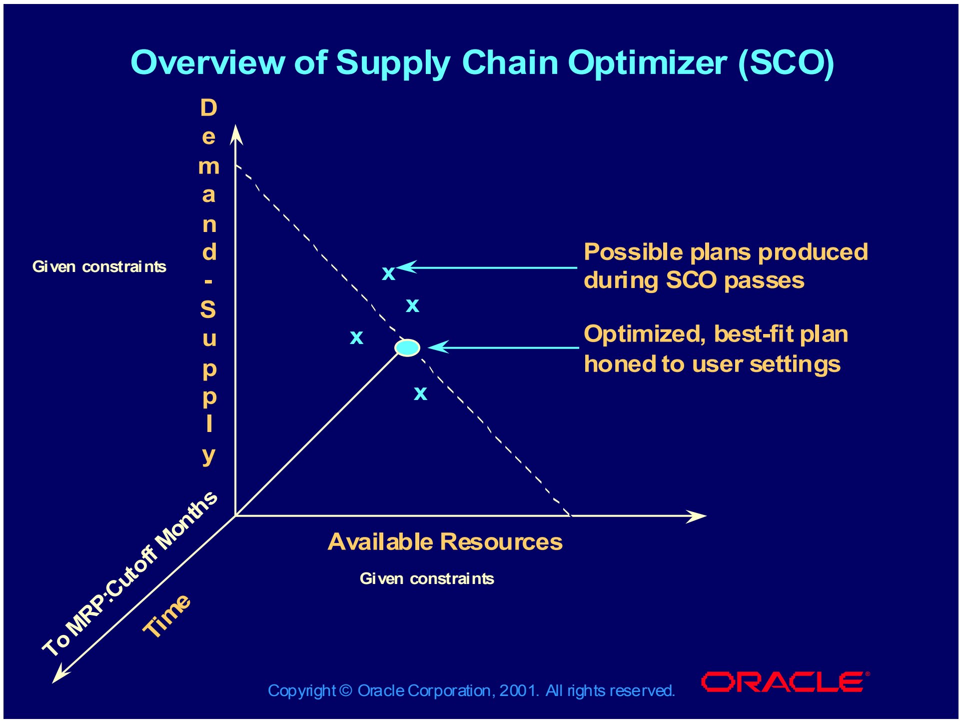
Optimization

The Oracle Advanced Planning System employs advanced ILOG Solver and optimization techniques that allows you to:

- Optimize your plans to financial and other enterprise-strategic objectives
- Create coordinated production and distribution plans for each organization by using memory-based planning
- Ensure that the plan is feasible and respects all of your constraints by using constraint-based scheduling engine

You can simultaneously plan material and capacity while considering the constraints of:

- Minimizing inventory costs
- Maximizing on-time delivery
- Maximizing overall plan profit



Overview of Supply Chain Optimizer (SCO)

- Given an amount of TIME, as indicated by the setting of the MRP:Cutoff Months profile option.
- Allocated with available RESOURCES as made available from the source ADS instances.
- Assigned a number of TASKS contained within demand, supply, netting, and so on.

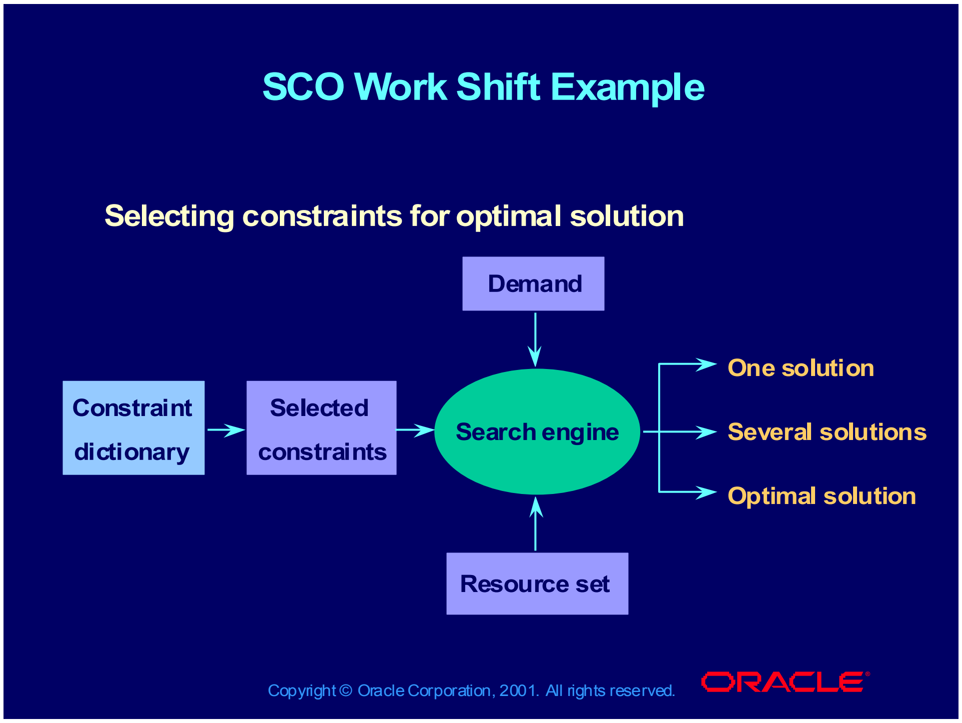
The SCO makes several passes at the planning problem eventually making the determination that further analysis would provide diminished results. As indicated in the diagram above, x represents attempts at problem resolution by the SCO.

It is important to note that as you adjust Penalty Factors or Plan Objectives, the volume of exception message creation will fluctuate.

Resources include:

- Transportation
- Supplier Capacity
- Material On Hand
- Available Resources
- CPLEX is a C library that efficiently implements Simplex (and other linear programming algorithms) to solve a problem that can be expressed as a conjunction of linear equations that state that weighted sums of numeric variables should have a given value

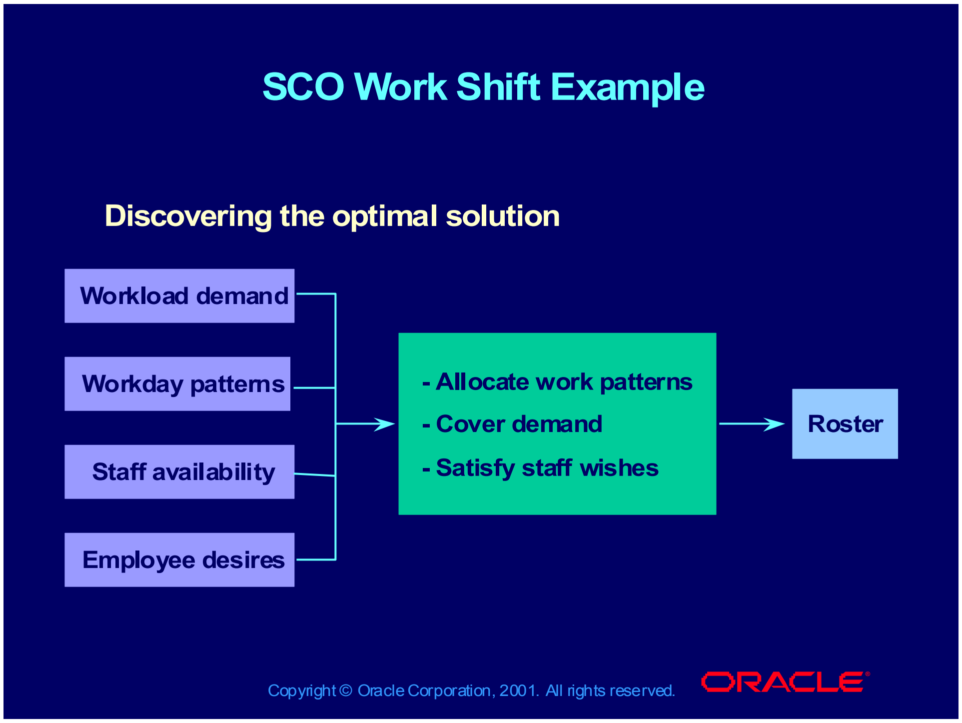
- Oracle APS solves the problem of planning the production of ordered finite products on finite capacity production taking into account practical industrial constraints.



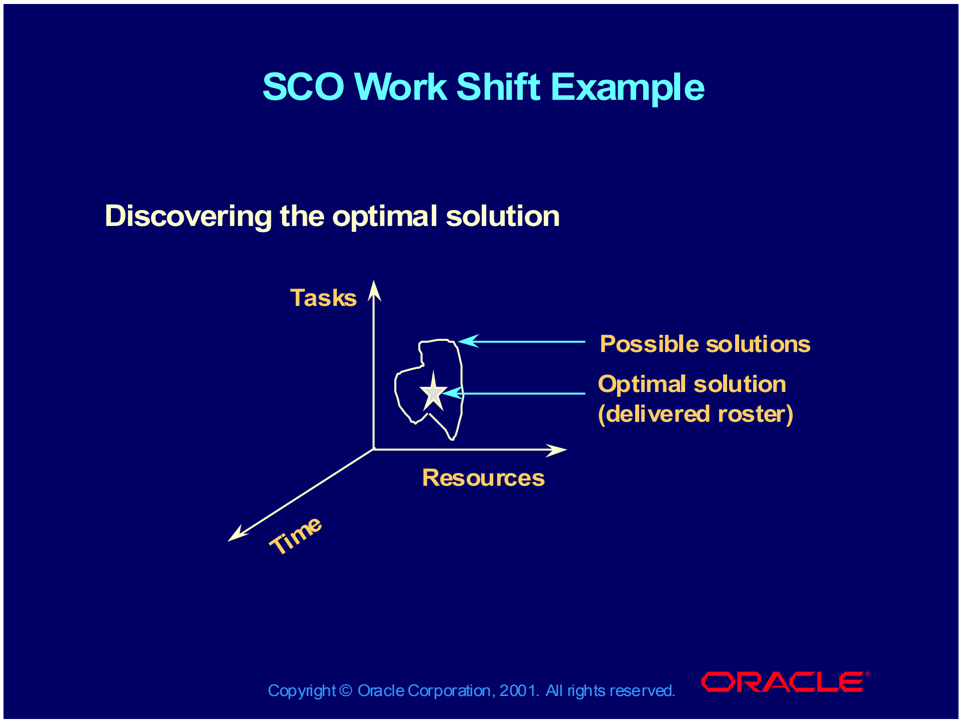
SCO Work Shift Example

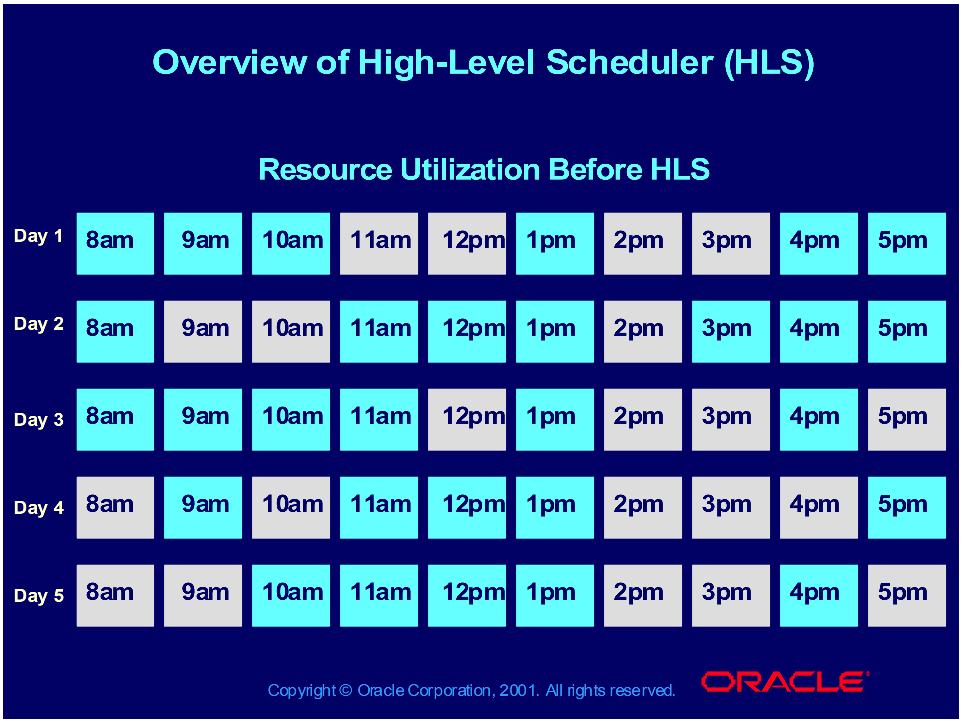
- To produce a duty roster, a system based on the ILOG Optimization Suite takes job requirements in the form of regular duties, activities, or workload curve.
- Incorporating global constraints, regulations, and union agreements, it then produces daily shifts and weekly patterns, and covers the demand to build the roster.
- To generate the roster, the system then considers individual constraints, employee skills and availability.
- You can easily modify such constraints and data at any time, address unforeseen problems such as sick leave or personal leave, and dynamically reschedule a new solution that can be fully adapted to any last-minute changes.
- You can also rapidly simulate the impact of any new regulations or any changes to your corporate organization.

SCO Work Shift Example



SCO Work Shift Example

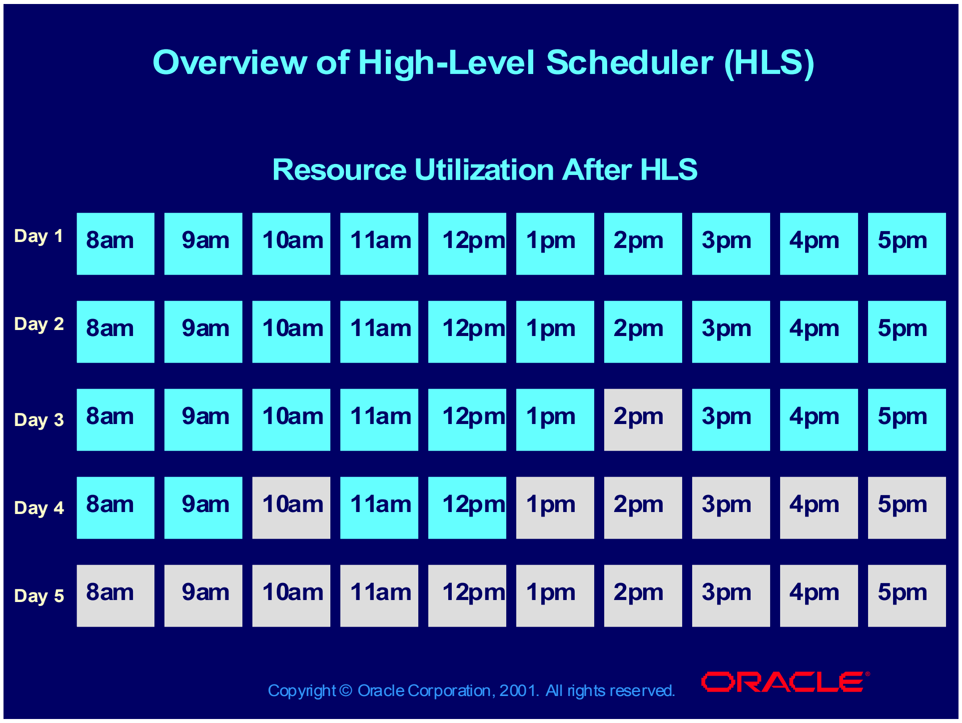




Resource Utilization Before HLS

MBP produces planned orders with a start date that is indifferent to resource utilization concerns. Planned order start dates can leave a resource schedule similar to the schedule above.

Overview of High-Level Scheduler (HLS)



Resource Utilization After HLS

After the HLS finishes smoothing resource utilization, there may be resource downtime due to supply chain lags, resource maintenance, and user defined plan objectives and penalties

ILOG

ILOG

ILOG Product	Module within Planning Engine	Description
ILOG Solver	High-Level Scheduler	C++ library for solving problems in planning, allocation, and optimization and exploiting constraint programming and object-oriented programming.
ILOG Scheduler	High-Level Scheduler	C++ library based on Solver that offer features adapted to solving problems in scheduling and resource allocation.
ILOG CPLEX	Supply Chain Optimizer	C library for solving large-scale optimization problems in the area of planning and resource allocation using techniques from linear programming and mixed-integer programming.

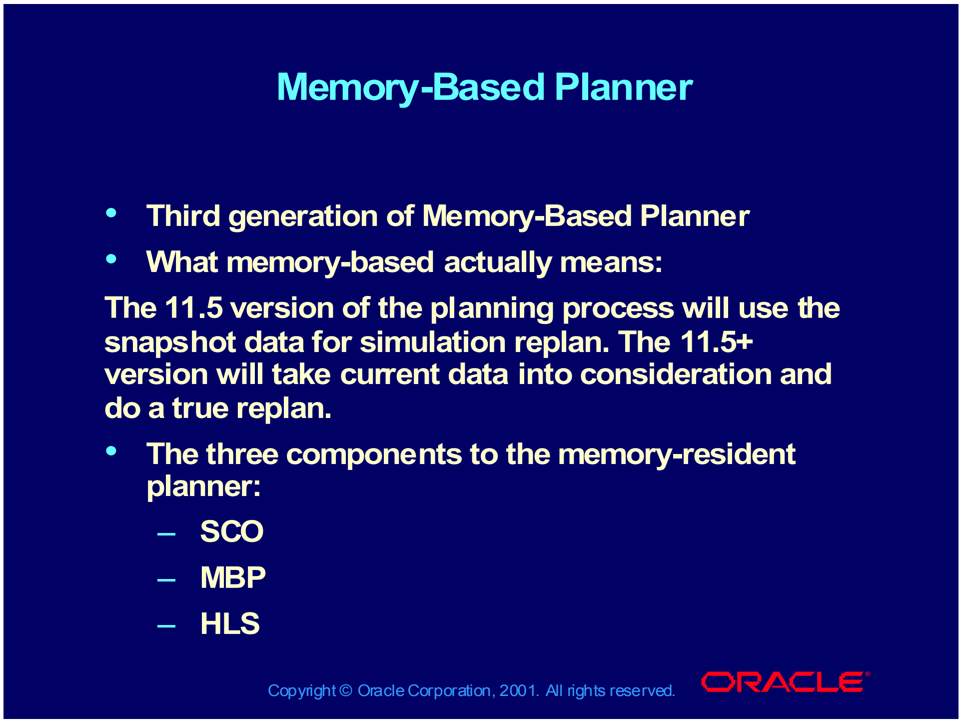
ILOG

- ILOG is an international software company founded in France in 1987. ILOG is a French research institute concentrating in Computer Science
- ILOG sells software components that are semi-finite software bricks in the form of C++/Java libraries
- I LOG has 3 lines of products : Optimization, Visualization, and Control
- Oracle-APS uses products from the Optimization suite which is based on constraint programming

Constraint programming is a software engineering paradigm that consists in separating the resolution of problems in 3 pieces:

- Modeling: expressing equations/relations among variables of the problem
- Propagation: attach a polynomial partial algorithm to each equation or sets of equations—reduces/fixes the values of some variables
- Search: explore possible values for the variables in a search tree

Memory-Based Planner



Memory-Based Planner

Third Generation of Material Requirements Planning in MEMORY

- Pure Internet Client
- Nothing need be installed on the client above Netscape
- Single Common Data Model

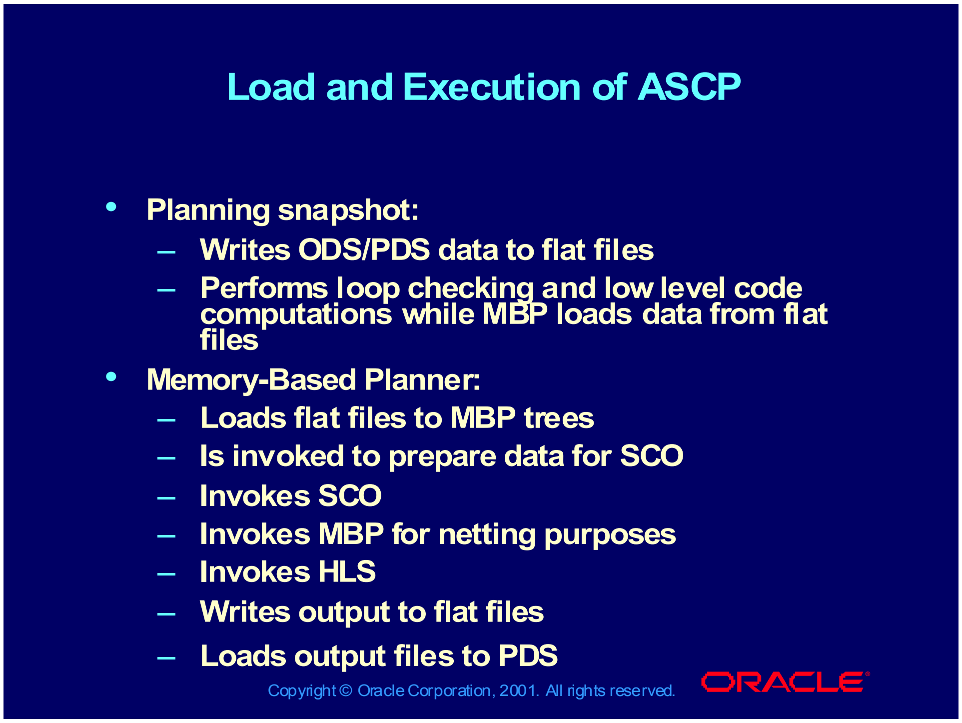
What does Memory Based actually mean?

The three executables, SCO, MBP, HLS make a unique executable, sharing the same memory.

The advantages are:

- Everything is in memory, so we avoid critical I/O bottlenecks
- You use the existing MBP data model

Load and Execution of ASCP



Load and Execution of ASCP

- Data is extracted from the ODS/PDS data stores and written to ASCII flat files. The process uses $APPLCSF, if set. If $APPLCSF is not set, $MSC_TOP/out is used.
- Loop checking is done in the snapshot as well as low level computations. Loop checking is done for WIP, Sourcing and BOM. Raise your stack size, in unix, if looping becomes and issue.
- If your are using Direct Load, all SQL*Loader issues apply.

HLS Data Files

If you are experiencing problems with the HLS, you may need the HLS data files. As data is unloaded from the MBP, data files are prepared that are loaded into HLS. These data files can be captured by executing the following command from the unix command prompt:

setenv FDWHOAMI 0,704,20454

mkdir data (under $MSC_TOP, make sure you have space and privileges)

$MSO_TOP/bin/MSONEW <dbuser/passwd@db> 0 Y -d <plan_id>

After the MBP process completes, tar the files in the $MRP_TOP/data directory.

Tar -cuf data.tar data

* NT instructions will be available via Metalink.

Reading from Input Flat Files

Reading from Input Flat Files

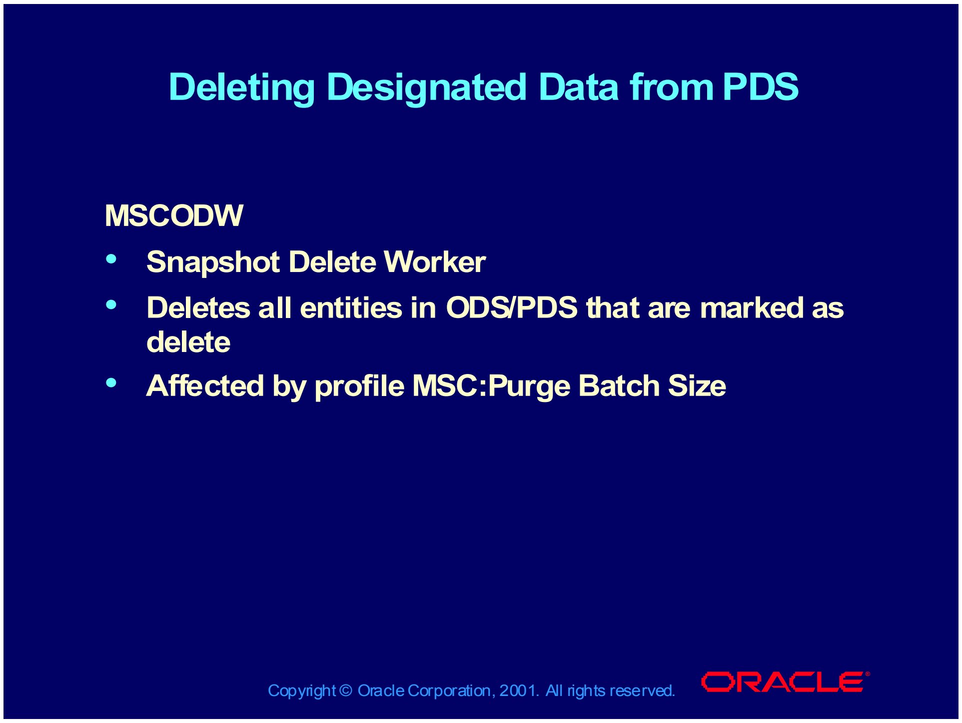
- MBP creates flat files of planning snapshot from ODS/PDS.
- MBP loads data trees and invokes Cplex SCO (only where plans options are set to run optimization and scheduling).
- High-Level Scheduling performs scheduling using ILOG Scheduler/Solver.
- MBP moves data from memory to ASCII flat files.
- MBP moves data into PDS.

Reading from Input Flat Files

The practice deleting these ASCII flat files is rather common. Please note that if these files are not present for a specific plan, simulation replans are not possible. The MBP would have to be launched again and new ASCII flat files would have to be created.

These files should be located in the directory $APPLCSF/out or $MSC_TOP/out. They can be identified by their extensions of .ctl, .dis, .dat and .bad.

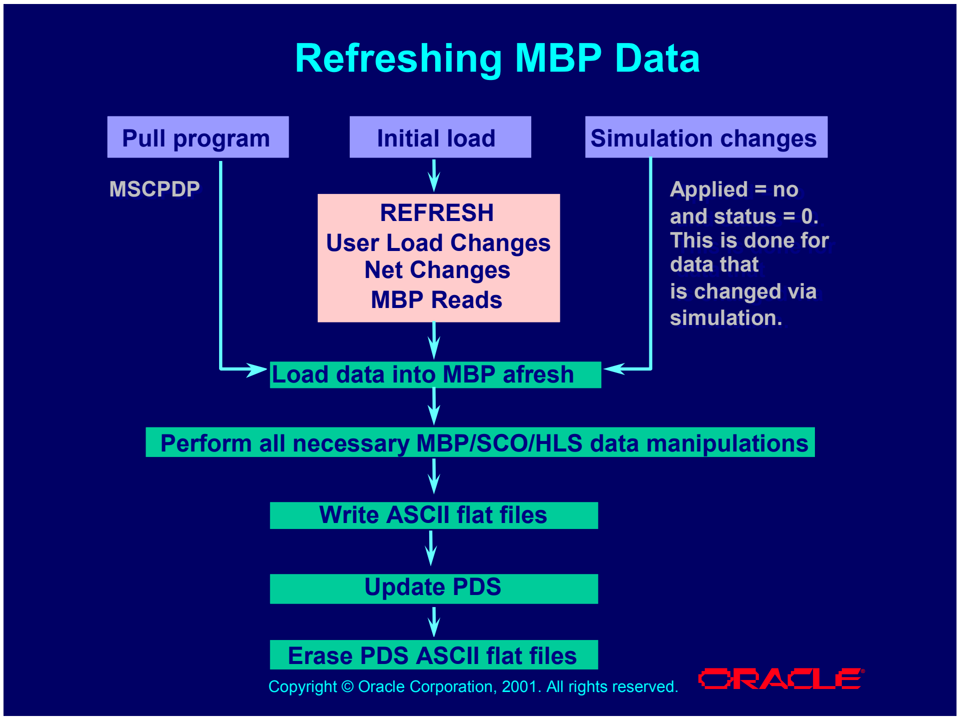
Deleting Designated Data from PDS



Deleting workers Designated Data from PDS

- This worker will commit changes to the database after rows are inserted/deleted/update >= MRP:Purge Batch Size.
- This setting will have an effect on Rollback Segment usage. A large setting, for this profile option, will cause greater usage of RBS.
- It is best to experiment with this setting and understand its impact on system resources.

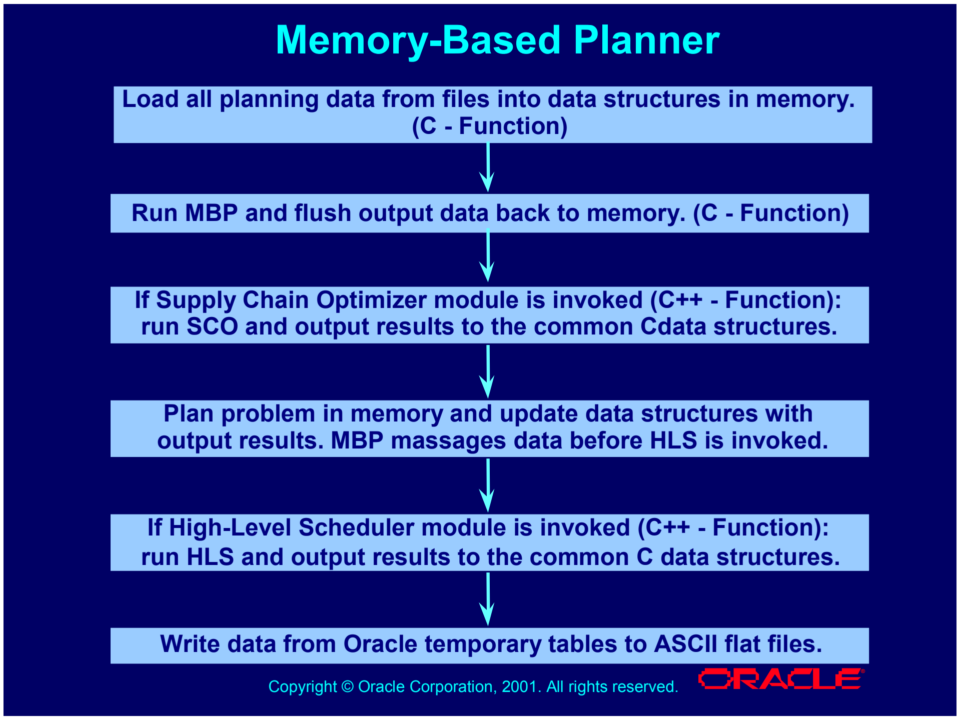
Refreshing MBP Data



Refreshing MBP Data

There are three methods in which data is made available to the planning engine.

Memory-Based Planner



Inside the Memory-Based Planner

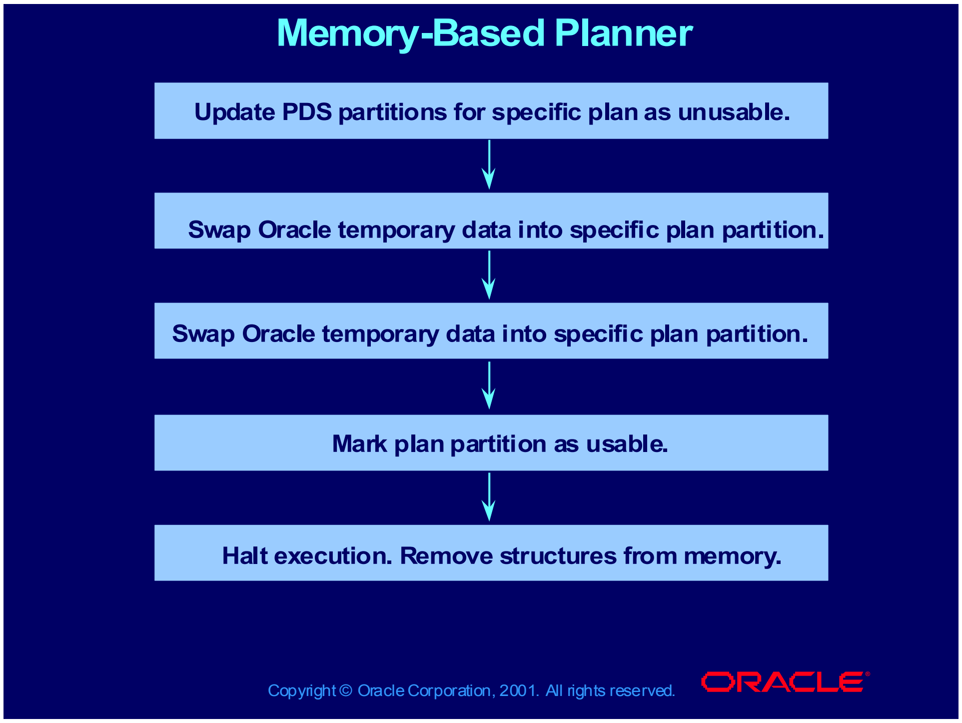
The processing flow of MSONEW is new to APS. With the introduction of ILOG libraries, additional data manipulation must take place. As we pass supply/demand information to ILOG and receive possible plans from ILOG, additional adjustments need to be made to the planned orders, time fences and dates.

The three sequential modules, discussed above, make ASCP. These modules, combined together, make a unique executable, sharing the same memory. This, in memory data model, is the existing C MBP data model currently used in 10.7 and 11.x. The difference is that C++ data wrappers are added to the data to make it presentable to the ILOG modules.

There are several advantages to the above implementation:

- Everything is in memory, so we avoid critical I/O bottlenecks.
- We use our existing data model, thus our existing MBP C code.

Memory-Based Planner



Memory-Based Planner

- While the data is swapped from the Oracle temporary tables into the PDS data store, the active partition is marked as unusable while the swap occurs.
- This leaves all other data, from other active plans, usable, enabling the execution of multiple plans simultaneously.
- Once the data is swapped from memory into the correct plan partition, the partition is marked as active, execution occurs as before MBP ran.

MBP

MBP output processing logic by day

Net against existing supply (on-hand and scheduled receipts) which has an available date earlier or equal to the demand date. The supply available dates are determined by SCO.

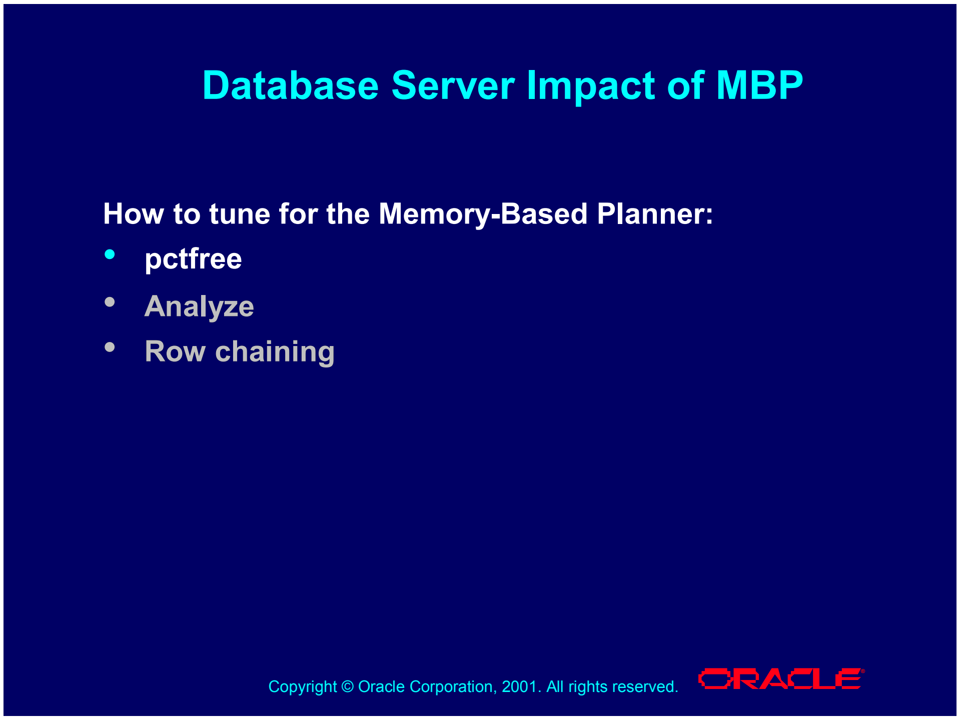
Net against demand (possibly created by SCO), starting with the earliest available supply date in order of sourcing rank, and continuing beyond the demand date, if necessary. Late supplies will be re-scheduled in. If SCO has created supplies for substitutes instead of the primary item, MBP will include these supplies as well.

For each supply considered by MBP, apply order modifiers and decrement other supplies in that time bucket and future SCO supplies (if necessary) by an equivalent quantity. The choice of supplies to decrement will be in increasing order of sourcing rank.

If the BOM, routing and source used by SCO to create the supply is valid, MBP will pass down demand based on that BOM, routing and source. If not, MBP selects the primary BOM, routing and source to pass down demand for the components.

When MBP explodes the BOM, the due dates for the components are calculated by offsetting the quantity dependant (and order modified) lead time from the adjusted supply date.

Because MBP makes changes to the BOM, routing and sources used by SCO, it may create additional supplies (apart from the ones to adjust order sizes) if necessary. In such cases, primary BOM, routing and source from the sourcing rules will be used.



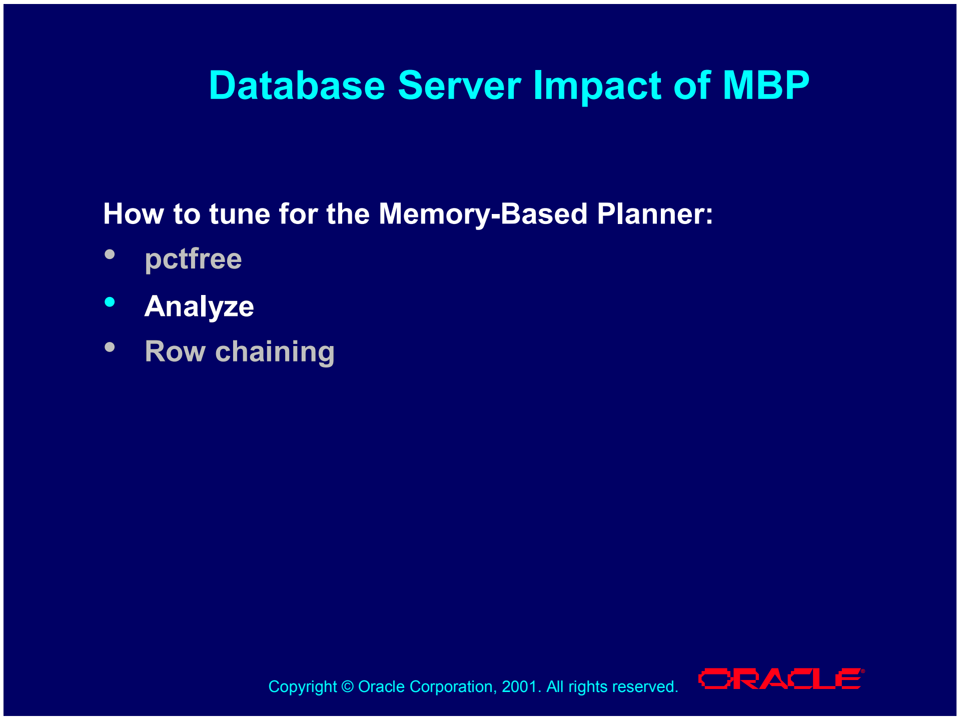
PCTFREE

Make sure your database objects in the MSC schema have a PCTFREE of zero.

Pctused and Pctfree determine the usage of Oracle blocks for row storage. Storage clauses control the spread of data within Oracle blocks. At a number of sites the major tuning act we have accomplished has been to improve the blocking factor on an otherwise fixed I/O bound application.

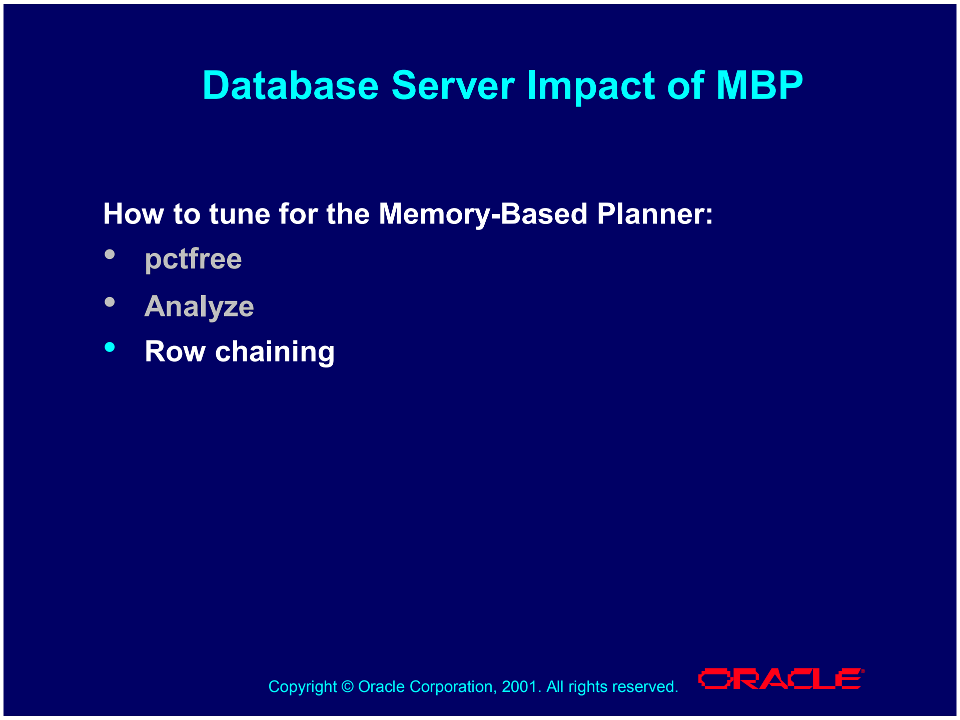
When building your indexes consider setting the pctfree to zero. If the table(s) indexed are not subject to growth, setting the pctfree may improve disk access. Set the pctfree to a lower number if the table is expected to grow.

DatabaseServer Impact of MBP



Analyze

- 11*i* uses the Cost Based Optimizer (CBO). Your init.ora should reflect this change. It is recommended that you set this to CHOOSE.
- Analyze using compute statistics, not estimate.
- Analyze should be performed on a regular basis according to your resource usage. Obviously a shop where MBP is ran more often would require more frequent execution of analyze.
- Analyze each partition separately.
- It is strongly recommended that you use the Oracle supplied utility for analyzing. $MSC_TOP/sql/MSCSTAT.sql will analyze the MSC schema.



ROW CHAINING

- You should rebuild tables when excessive row chaining occurs. This occurs when a row expands and can no longer fit into one single physical Oracle block. This can also occur if your pctfree has been set to low for a row that needs to expand due to user activity. Please keep in mind that indexes are also subject to chaining if the indexed columns are subject to growth.
- You can detect chained rows by running the following commands in SQL*Plus. Running the script, utlchain.sql, builds an empty table called Chained_Rows. After this script is completed, run the following command line once for each of the tables in the schema being investigated:

ANALYZE TABLE table_name LIST CHAINED ROW INTO Chained_Rows;

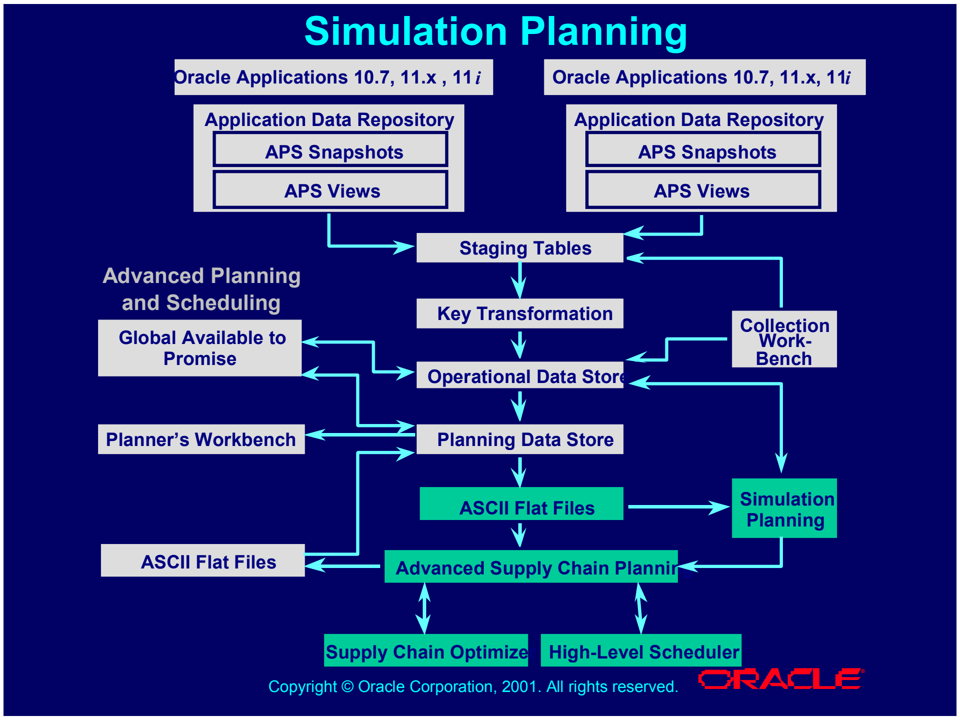
Once all tables have been analyzed, the Chained_Rows table will contain the names of the tables containing chained rows.

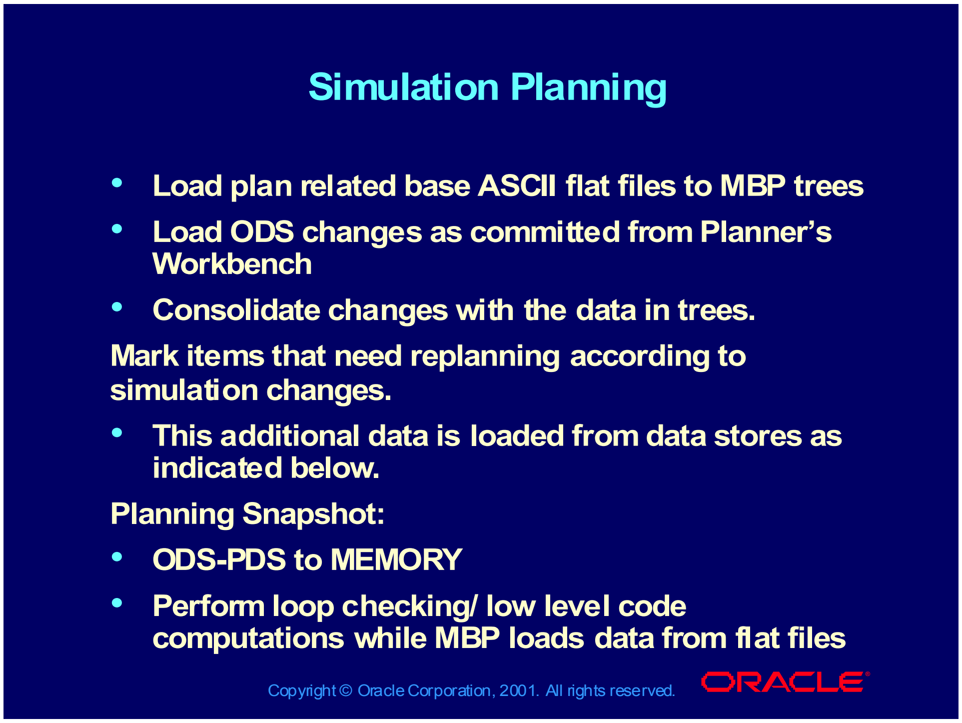
a. Run $ORACLE_HOME/rdbms/admin/utilchain.sql

This should be done before running ANALYZE.

b. ANALYZE TABLE MTL_DEMAND LIST CHAINED ROWS

(or the table of your choice)


Simulation Planning
Oracle Applications 10.7, 11.x , 11 i
Oracle Applications 10.7, 11.x, 11i
Application Data Repository
APS Snapshots
APS Views
Application Data Repository
APS Snapshots
APS Views
Staging Tables
Advanced Planning
and Scheduling
Key Transformation
Collection
Work-
Bench
Global Available to
Promise
Operational Data Store
Planner's Workbench
Planning Data Store
ASCII Flat Files
Simulation
Planning
ASCII Flat Files
Advanced Supply Chain Plannir
Supply Chain Optimize
High-Level Scheduler
Copyright © Oracle Corporation, 2001. All rights reserved.



Simulation Planning

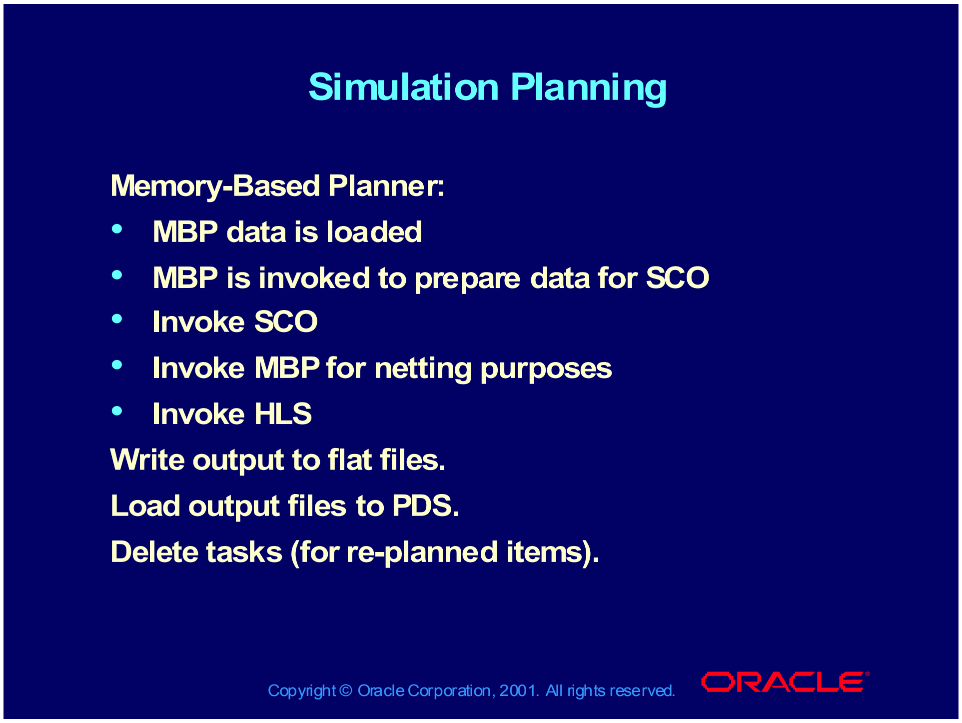
Simulation changes.

The five data groups that can be manipulated for simulation replan purposes are:

- Supply
- Demand
- Supplier Capacity
- Resource Requirements
- Resource Availability

Simulation Change Detection

Simulation changes are changes affected by the user using the PWB. These changes are saved by the PWB with applied = 'N' and status =0. These attributes are used to identify and load these changes back into MBP. Data changed by the workbench is written into the ODS.

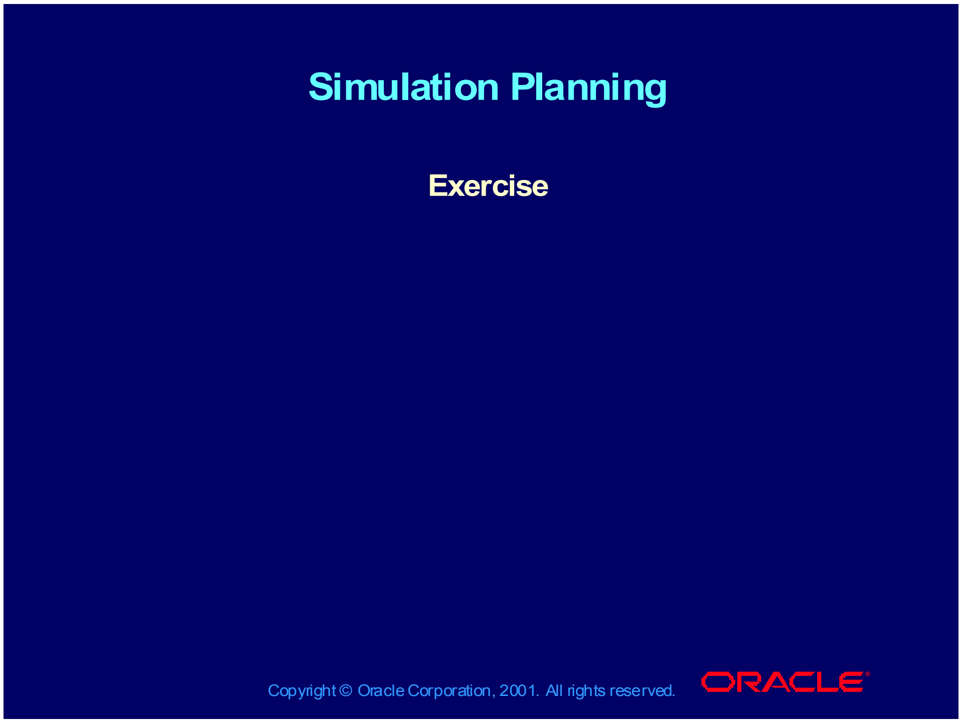


Simulation Planning

Write output to flat files:

This contains the base + net changes used by the plan run.

SCO and HLS may decide to plan for assemblies and/or components that have been affected by user changes introduced via the Planner's Workbench. For example, adding a new resource would trigger additional replanning activities.



Simulation Planning Exercise

1. Login to the destination instance.
2. Now we will select possible changes introduced into the msc_demands tables through the use of the Planner's Workbench.

```
select SCHEDULE_DESIGNATOR_ID sch_desg,
           PLAN_ID,
           INVENTORY_ITEM_ID,
           SR_INSTANCE_ID,
           APPLIED
 from   msc_demands
 where  applied = 2;
```

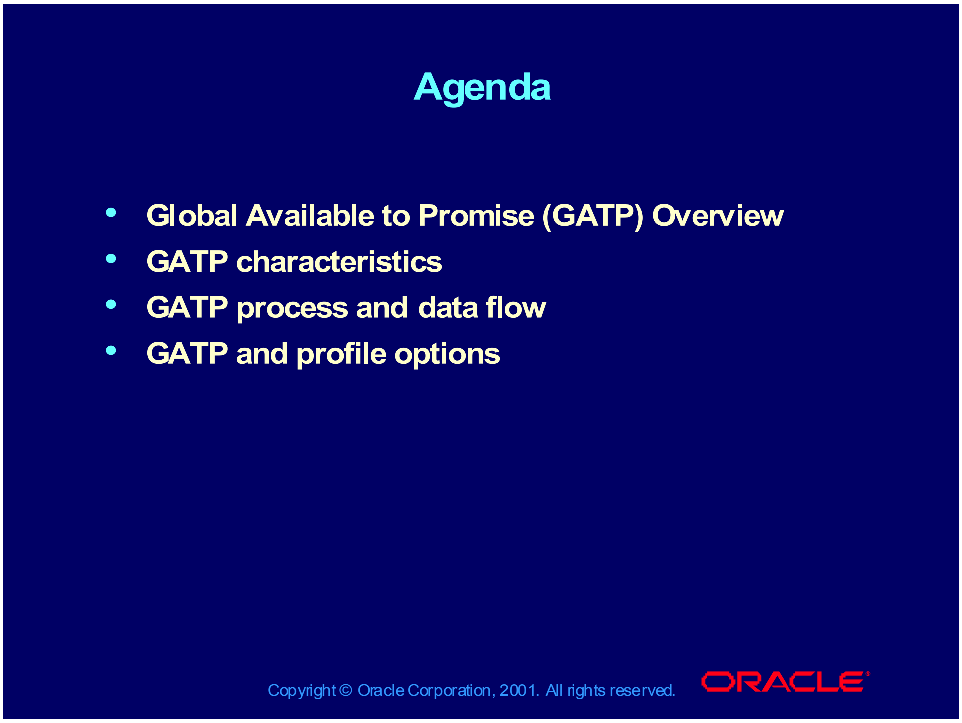
- What planning store do the above rows belong to?
- Are any of these rows marked for simulation planning?

High-Level Overview of Global Available to Promise

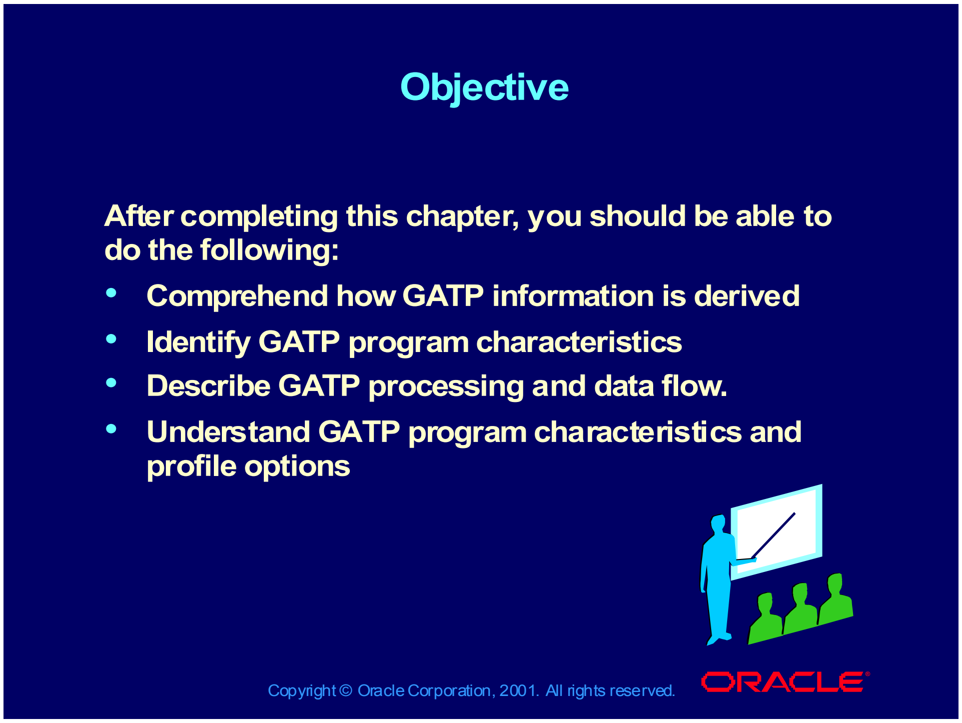
Chapter 9


High-Level Overview of
Global Available to Promise
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Agenda


Agenda
Global Available to Promise (GATP) Overview
GATP characteristics
GATP process and data flow
GATP and profile options
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Objective


Objective
After completing this chapter, you should be able to do the following:
• Comprehend how GATP information is derived
• Identify GATP program characteristics
• Describe GATP processing and data flow.
• Understand GATP program characteristics and profile options
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

GATP

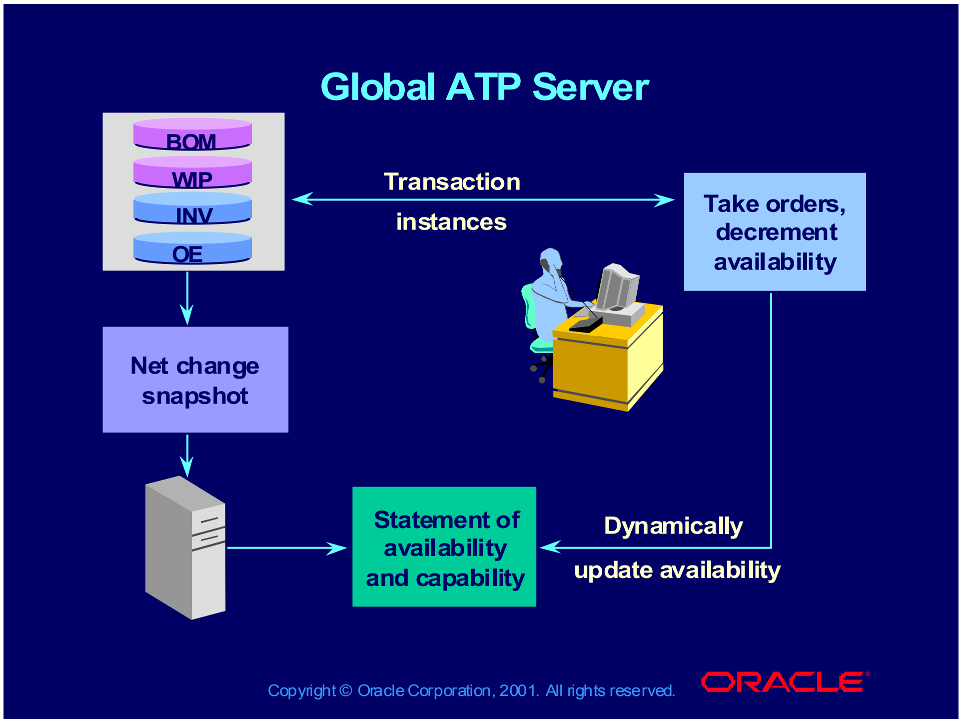
As order promising methodology has evolved, new terms in order promising have been coined to describe advanced order promising capabilities.

- Available to Promise refers to the ability to promise availability based on a pre-defined statement of current and planned supply.
- Capable to Promise refers to the additional ability to determine the availability of component materials and resources to meet unplanned demands.
- Capable to Deliver refers to considering transportation resources as well as considering the transit time to meet your customers delivery needs. Oracle Global ATP Server encompasses all these capabilities.

GATP

- Oracle Global ATP Server supports availability checks in a distributed environment.
- It can be deployed either as a component of a complete applications system, or by itself
- on a separate server.
- This flexibility allows you to support any combination of centralized and decentralized
- order promising.
- Transportation resources are not yet supported.

Global ATP Server



Global ATP Server

If you are implementing a Global Available to Promise a decentralized environment, consider setting the COMMIT_POINT_STRENGTH. By setting this value higher on the OLTP instance, compared to the APS server, the OLTP instance will control the distributed transactions.

GATP

ATP Information:

- **Planning Data Collection (PDP) and ATP Data Collection are two different report-sets with the same Concurrent Programs included in both sets.**
- **Oracle ATP uses the same data collection architecture as APS PDP if the Capable to Promise and database link profile options are set accordingly.**

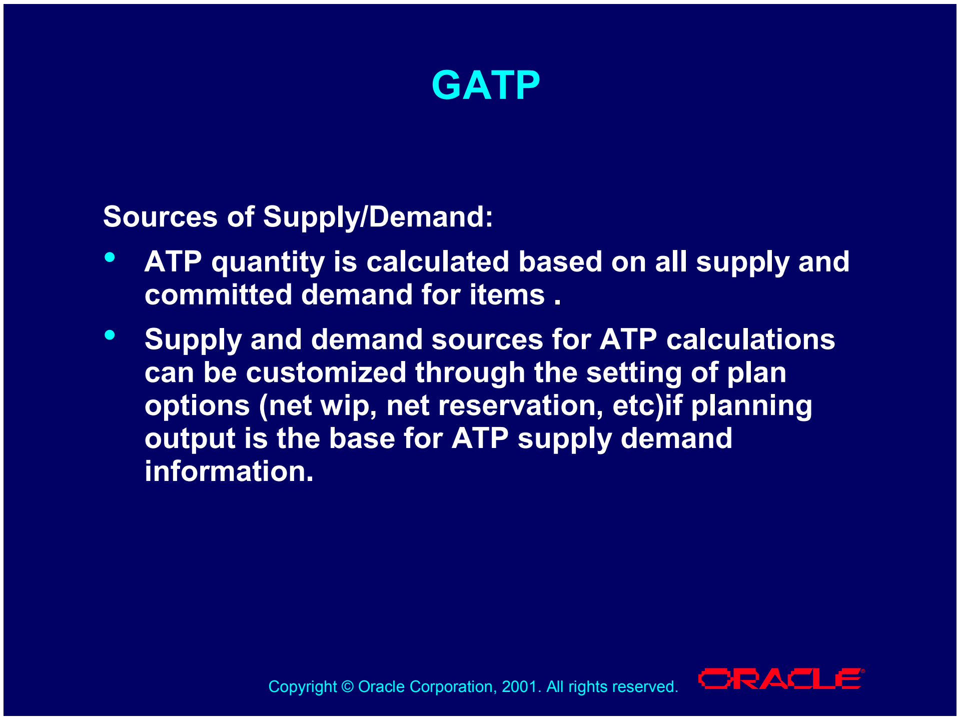
GATP

The planning data pull program, MSCPDP, controls the PDP worker, MSCPDPDW.

GATP

You can apply rules globally, to entire organizations, to categories of products, or to individual stock keeping units. More specific rules override more general rules, allowing you to apply default rules and maintain them on an exception basis.

GATP



GATP (continued)

Configurable Entities:

- Sources of supply and demand
- Infinite supply time fence
- Supply chain ATP
- ATP over-ride

ATP Configurable Parameters:

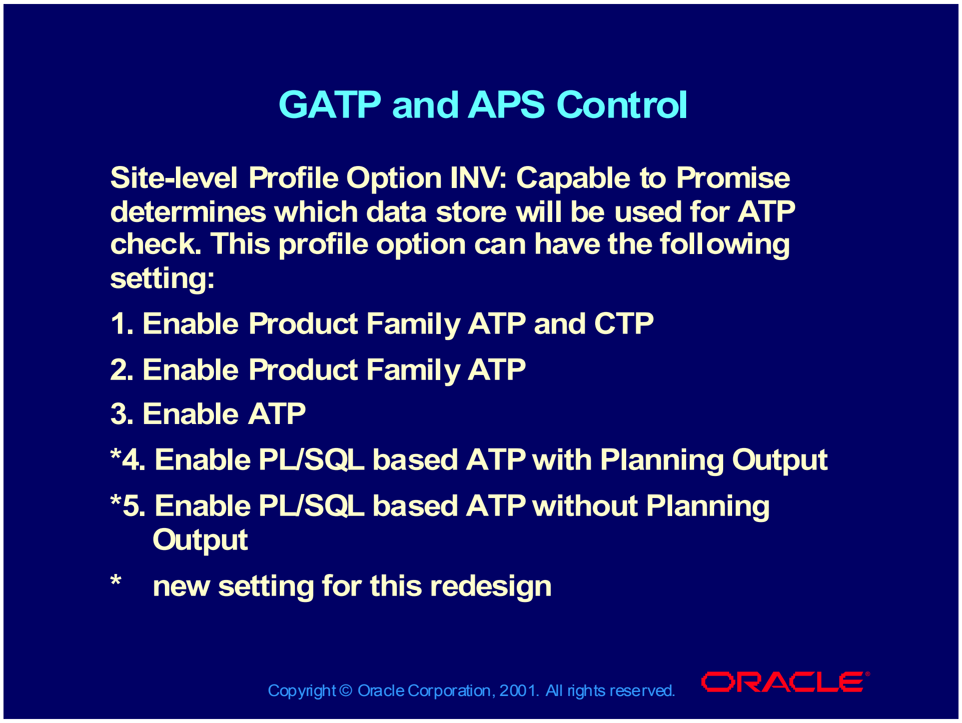
- ATP options
- ATP rules
- ATP flags

Module Integration:

- Order Management
- Inventory
- Bills of Material
- Advanced Planning and Scheduling
- Global Order Promising features:
- Global Statement of Availability
- Available to Promise
- Capable to Promise
- Capable to Deliver

- Multi-level supply-chain ATP
- Cumulative ATP
- Product Family ATP
- Ship Set/Delivery Set support
- Drop shipment support

GATP and APS Control



GATP and APS Control

This is set at the source instance.

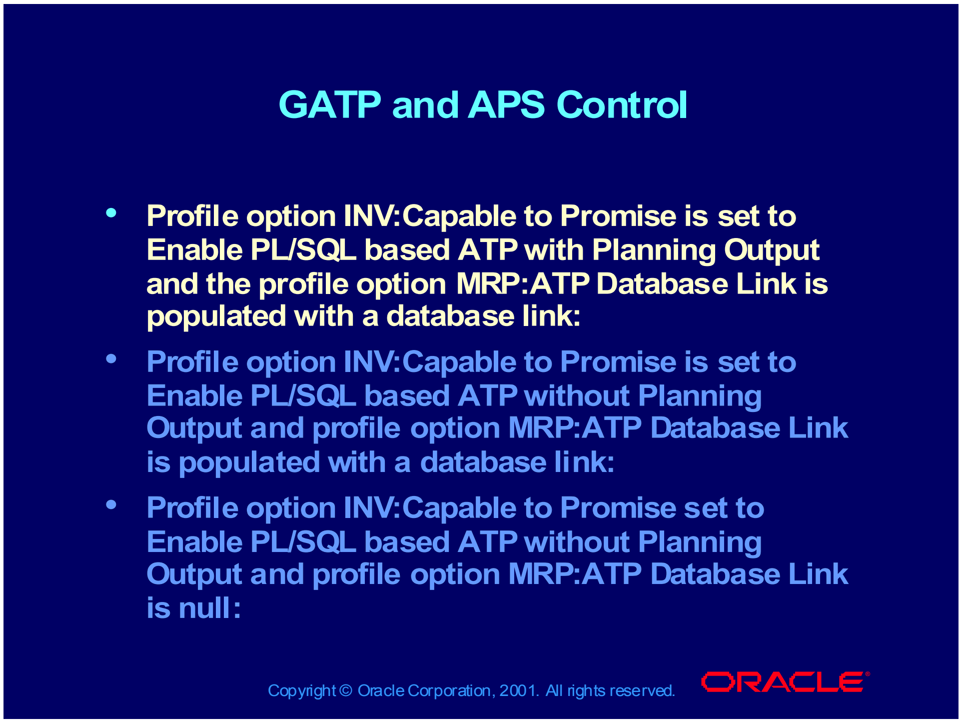
To perform GATP:

- Valid sourcing rule on the source defined and collected or entered into the APS server.
- Sourcing rule is part of the assignment set
- Insert the assignment set name into the profile option: MRP_AP Assignment Set
- Within the assignment set you need an item/org level or global level rule. Instead of filling the organization, supply the customer and customer site if desired.

For an item to participate in GATP, we expect the following two columns in MTL_Systems_Items to be populated:

- ATP Flag, atp_flag
- ATP Components Flag, atp_components_flag

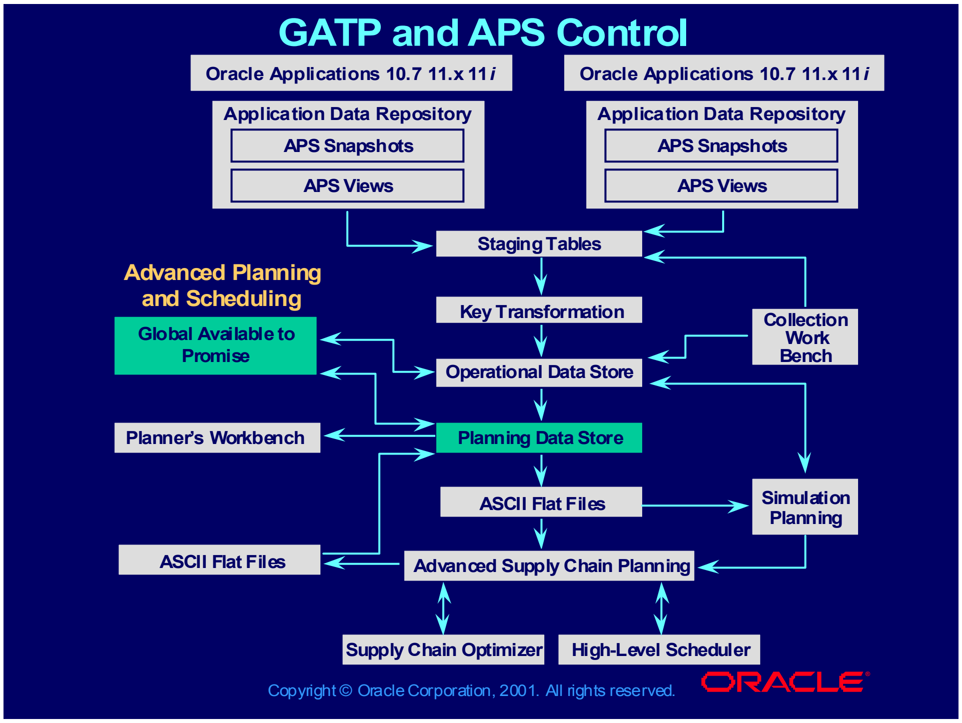
GATP and APS Control



GATP and APS Control (continued)

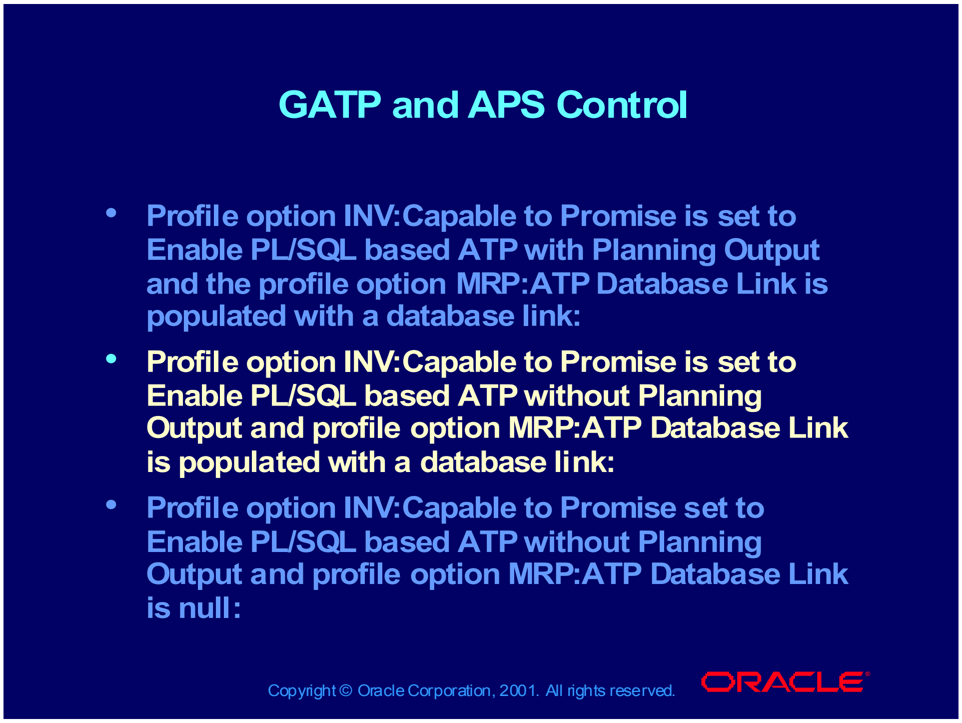
The PDS rows in the following tables are used:

```
MSC_SUPPLIES
MSC_DEMANDS
MSC_RESOURCE_REQUIREMENTS
MSC_NET_RESOURCE_AVAIL
MSC_INTERORG_SHIP_METHODS
MSC_SUPPLIER_CAPACITIES
MSC_SUPPLIER_FLEX_FENCES
```



Global Available To Promise (GATP)

- GATP information is derived from:
- PDS or Planning Data Store
- The GATP program:
- Reads from the previously collected item supply and item demand records that are found on the APS server. The supply and demand records were originally read from the execution system or Applications Data Store (ADS).
- Utilizes the multi-threaded capabilities of the Oracle database to process concurrent ATP request efficiently.
- Uses the same technology as APS data pull



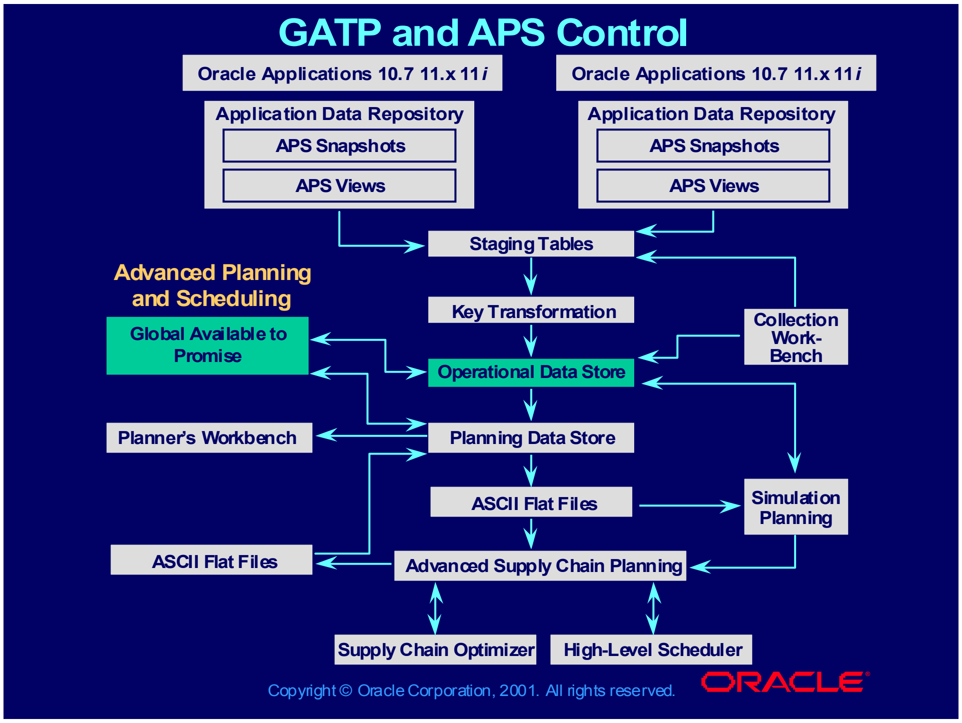
GATP and APS Control

- The rows in MSC_SUPPLIES, MSC_DEMANDS and MSC_SALES_ORDERS fromODS (plan_id = -1) are used for ATP information.
- Essentially, with the profile option set to 5, you have the same ATP functionality present in previous releases of Oracle Applications.
- This is not the same as in previous versions of Oracle Applications.
- MRPGATB.PLS

This SQL presents:

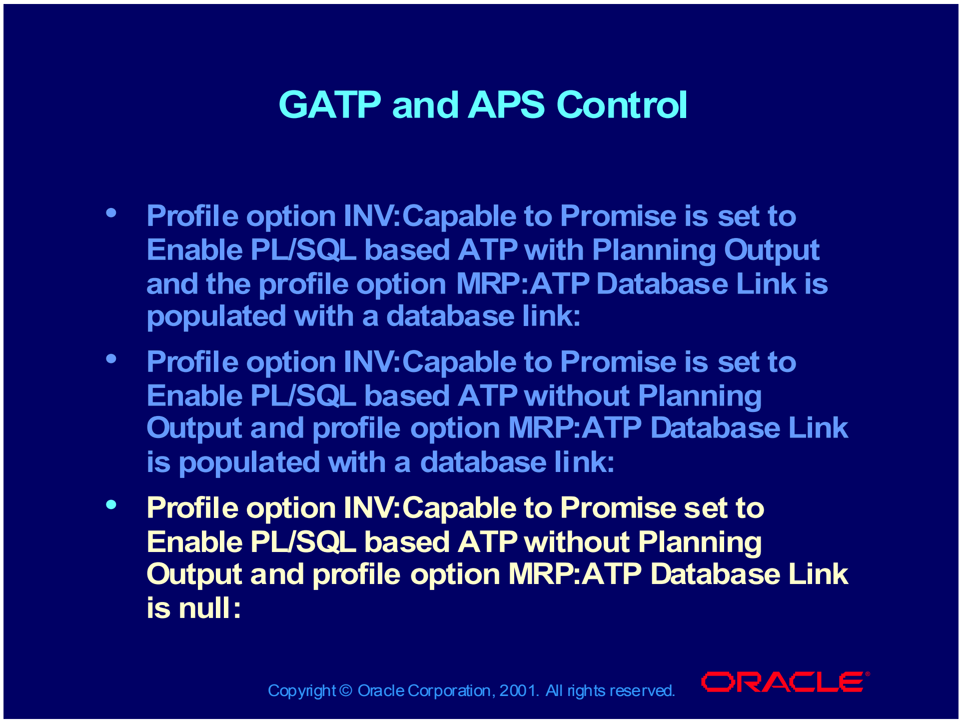
- Demand
- Supply
- Items
- msc_system_items
- msc_supplies
- msc_sales_orders
- msc_demands
- mrp_atp_details_temp
- mrp_atp_schedules_temp

GATP and APS Control



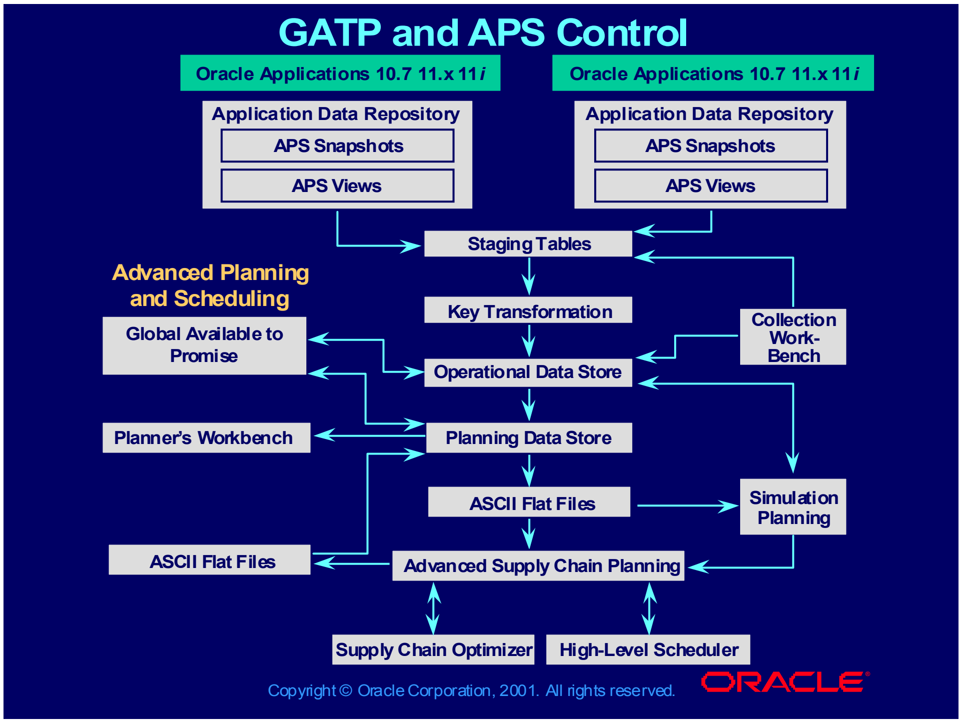
Global Available To Promise (GATP)

GATP information is derived from ODS or Operational Data Store



GATP and APS Control

Current supply/demand records in the execution system are used.

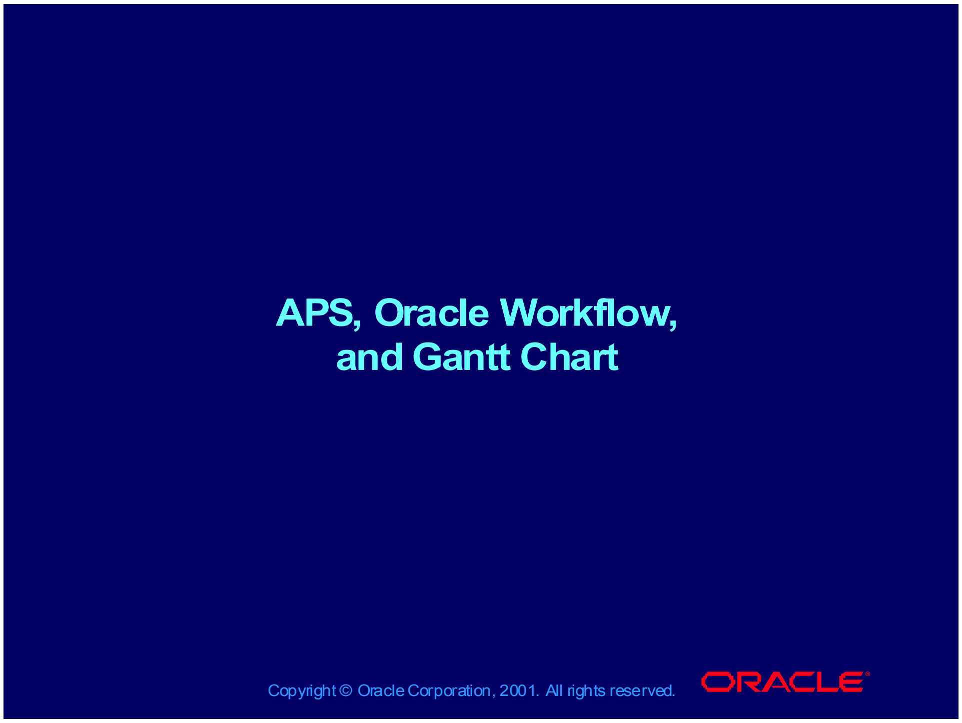


Global Available To Promise (GATP)

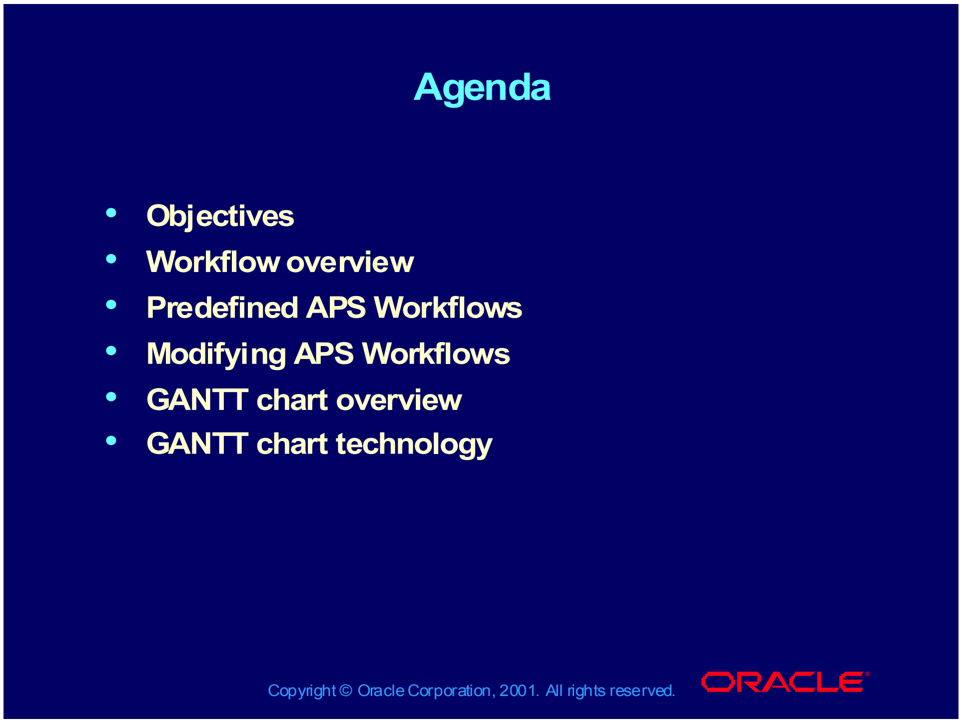
GATP information is derived from the application schema.

APS, Oracle Workflow, and Gantt Chart

Chapter 10


APS, Oracle Workflow,
and Gantt Chart
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Agenda


Agenda
• Objectives
• Workflow overview
• Predefined APS Workflows
• Modifying APS Workflows
• GANTT chart overview
• GANTT chart technology
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Objective

After completing this chapter, you should be able to do the following:

- Comprehend the usage of Workflow with APS
- Gain working knowledge of APS seeded workflows
- Understand possible workflow exceptions generated as a result of the planning process
- Become familiar with Oracles usage of GANTT charts

Workflow

- Materials and resource plans generate enormous amounts of data concerning your resources. Exception messages, generated by these plans, identify business conditions and statuses that require attention.
- In order to bring process and control over this potential flood of detail, we facilitate the acceptance, recognition, distribution and ultimately the resolution of the exception action item.

Workflow

- **Suppliers can access notifications regarding reschedules and cancels, consult/negotiate with planners/buyers.**
- **Customers can receive notifications regarding possible late orders, request partial deliveries to satisfy their immediate needs, acknowledge acceptance or denials of new promise dates.**
- **Oracle APS generates over 20 possible planning exceptions. Each of these are candidates for workflow.**

Predefined Oracle APS Workflows

- Item Forecast Workflow
- Sales Order Workflow
- Rescheduling Workflow
- Project Workflow
- Material and Resource Capacity Workflow

Predefined Oracle APS Workflows

Item Forecast Workflow

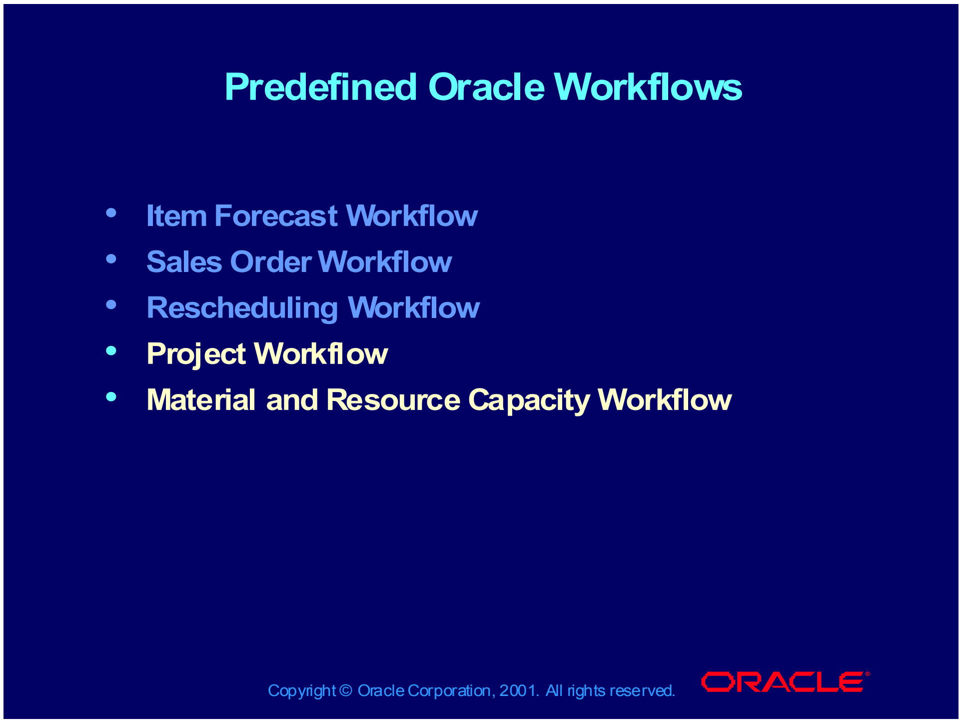
- Item is over committed
- Item has a shortage
- Item has excess inventory
- Items with expired lots
- Past due forecast
- Late supply pegged to a forecast
- Items below safety stock

Sales Order Workflow

Past due sales orders, Late supply pegged to a sales order

Rescheduling Workflow

- Item has orders to be rescheduled in\out
- Item has orders to be canceled
- Item has past due orders
- Item has orders with compression days



Predefined Oracle APS Workflows

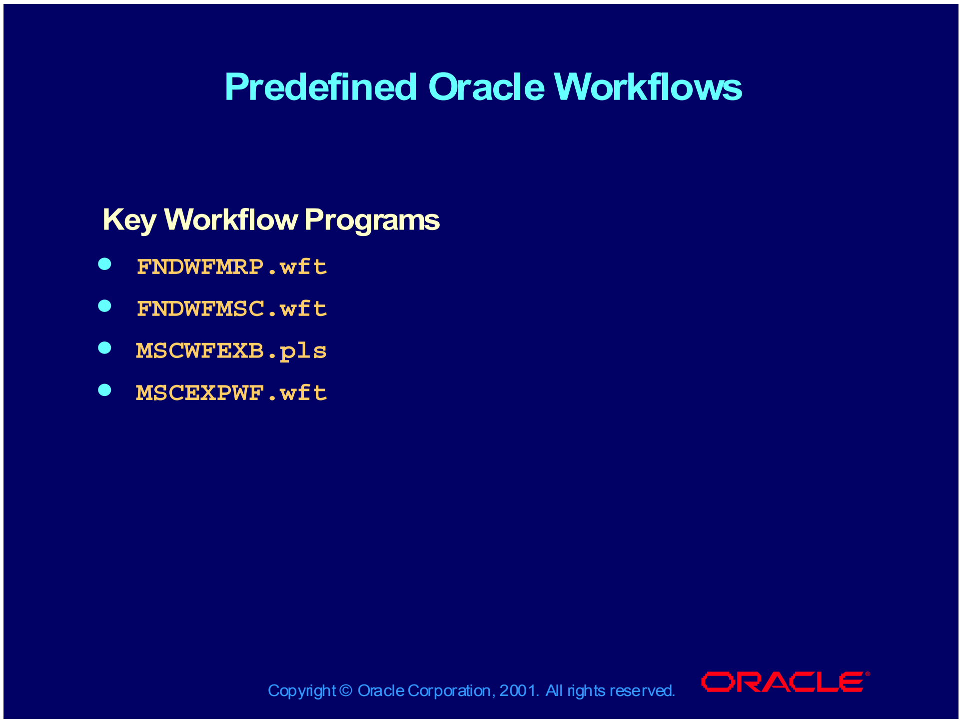
Project Workflow

- Items with shortage in a project
- Items with excess in a project
- Items allocated across projects

Material and Resource Capacity Workflow

Material constraint

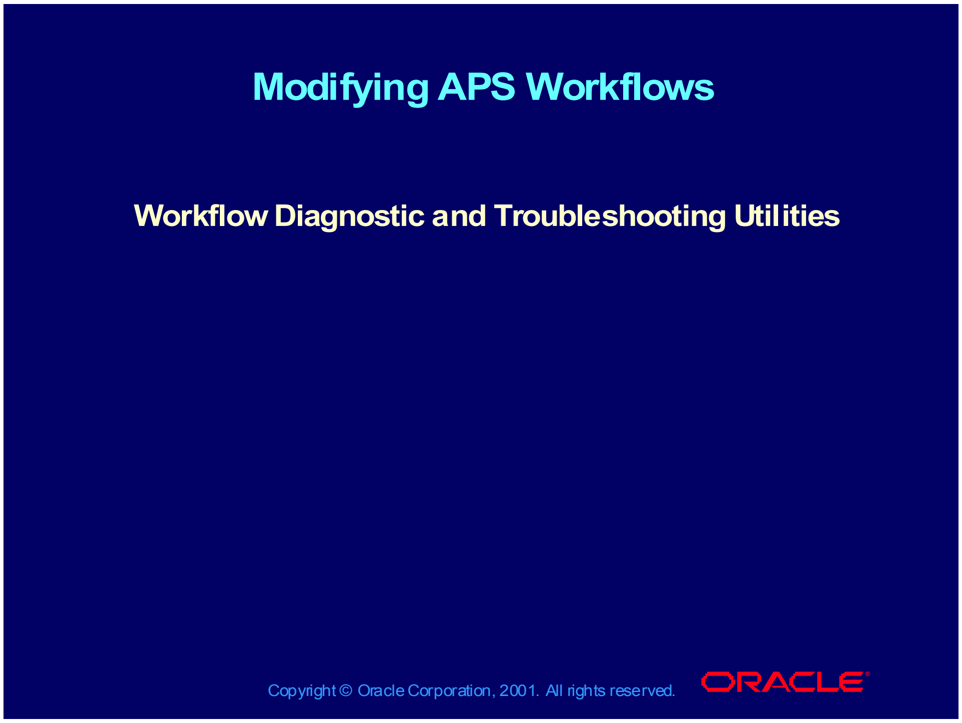
Predefined Oracle Workflows



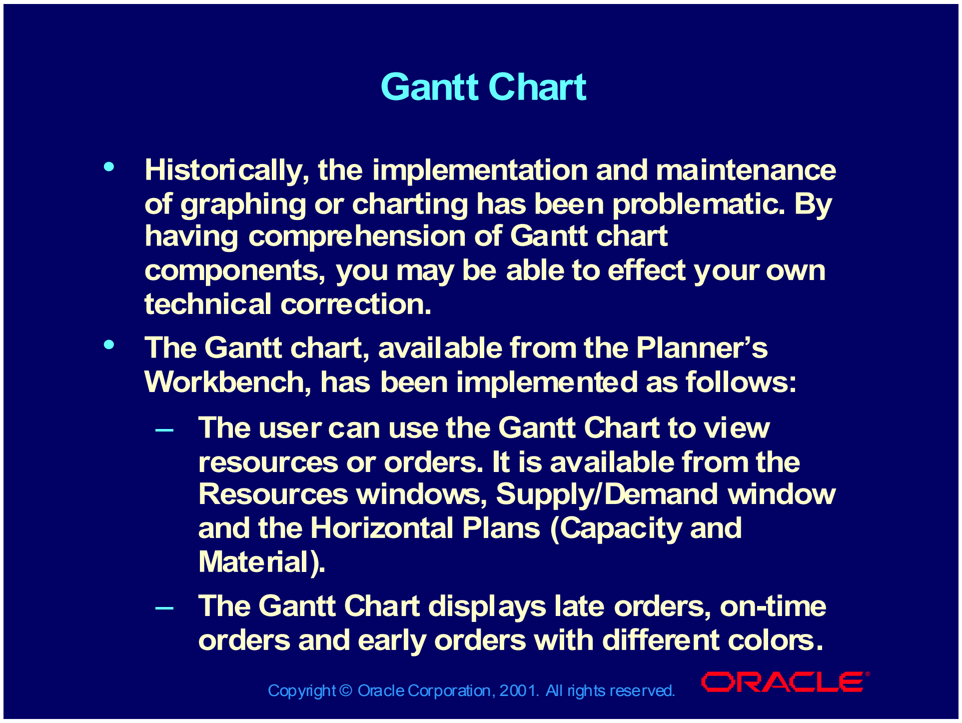
Predefined Oracle Workflows

To edit or re-configure predefined Oracle APS workflows, additional licensing is involved.

If a user has a need create additional custom workflows, then the user must purchase a license for Oracle Workflow Builder.


Modifying APS Workflows
Workflow Diagnostic and Troubleshooting Utilities
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Gantt Chart

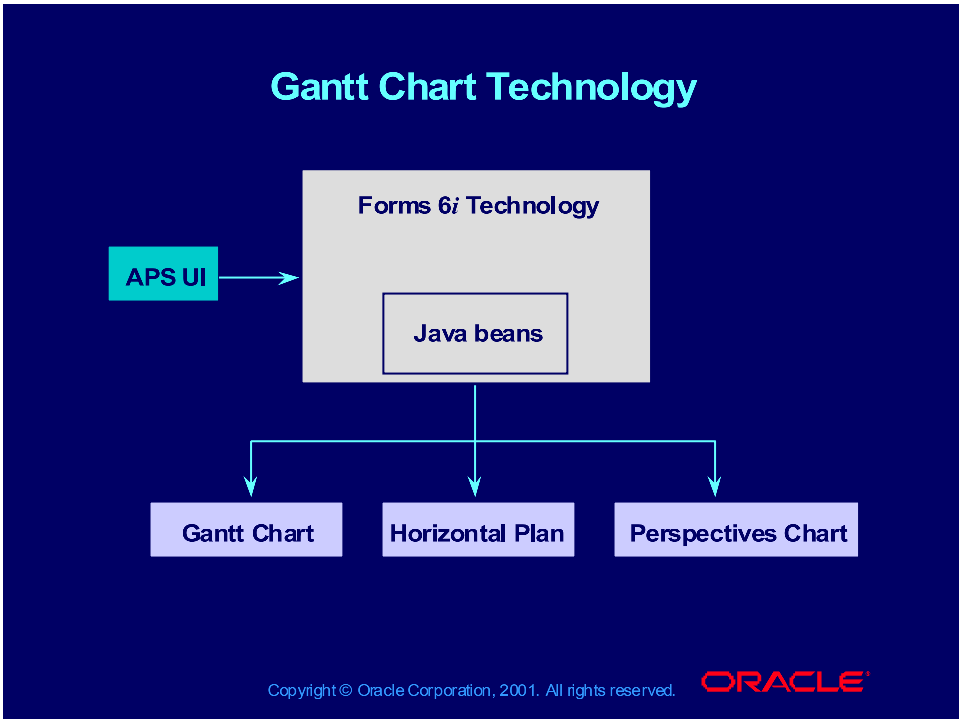


Gantt Chart

The Gantt chart can also be viewed from the items window.

After the user makes changes in the Gantt Chart view, we issue a replan. The following five actions are recorded:

- Supply changes
- Demand changes
- Supplier Capacity changes
- Resource availability changes
- Resource requirements



Gantt Chart Technology

- The diagram displays different Java modules within APS.
- Workflow and Java bean technology within APS

BeanArea is a new element in Forms6.0:

- It is intended to replace the 'userarea' element that has been used in Forms4.5.
- In addition to the existing support of java applets with userarea in Forms4.5, this new element also makes it possible to add Java beans into WebForms.
- A beanarea as a visible UI element inside a form, designed to have an applet running inside it.
- A beanarea that is designed to manage and manipulate multiple applets that run in their own separate frames.

Please also note that:

- BeanArea is only available and functional in Forms6.0.
- BeanArea is only functioning in WebForms.

Gantt Chart Technology

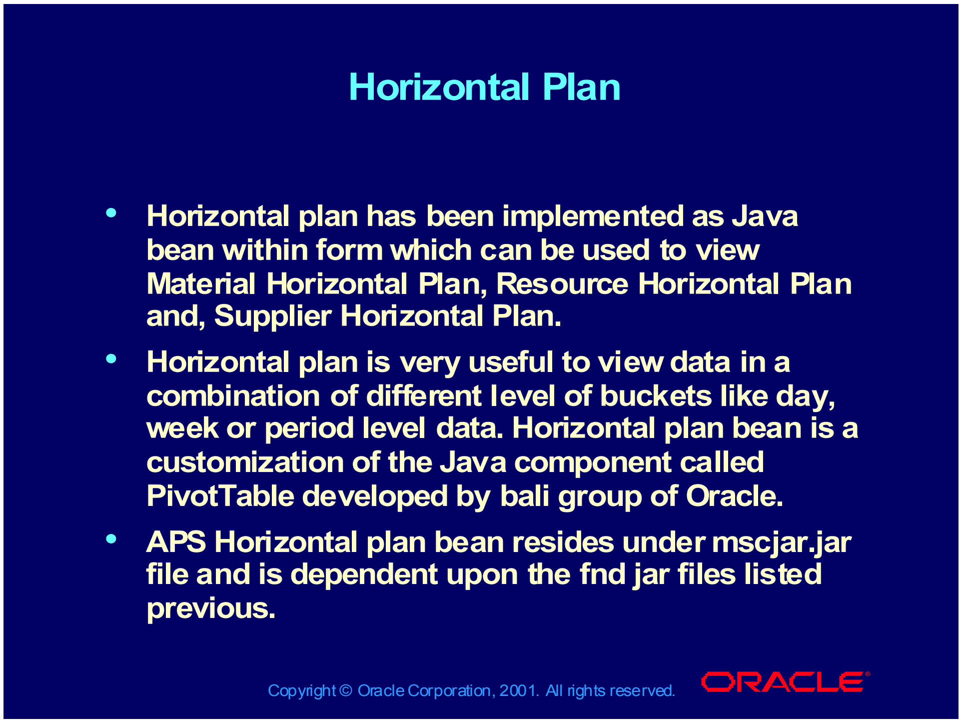
Gantt Chart technology is based on the Java component provided by AOL, called FNDGANTT. Hence, APS Gantt chart depends on the following FND jar files:

- fndgantt.jar
- fndpromise.jar
- fndjle.jar
- fndswing.jar
- fndewt.jar
- fndewtpv.jar
- fndtdg.jar

ORACLE

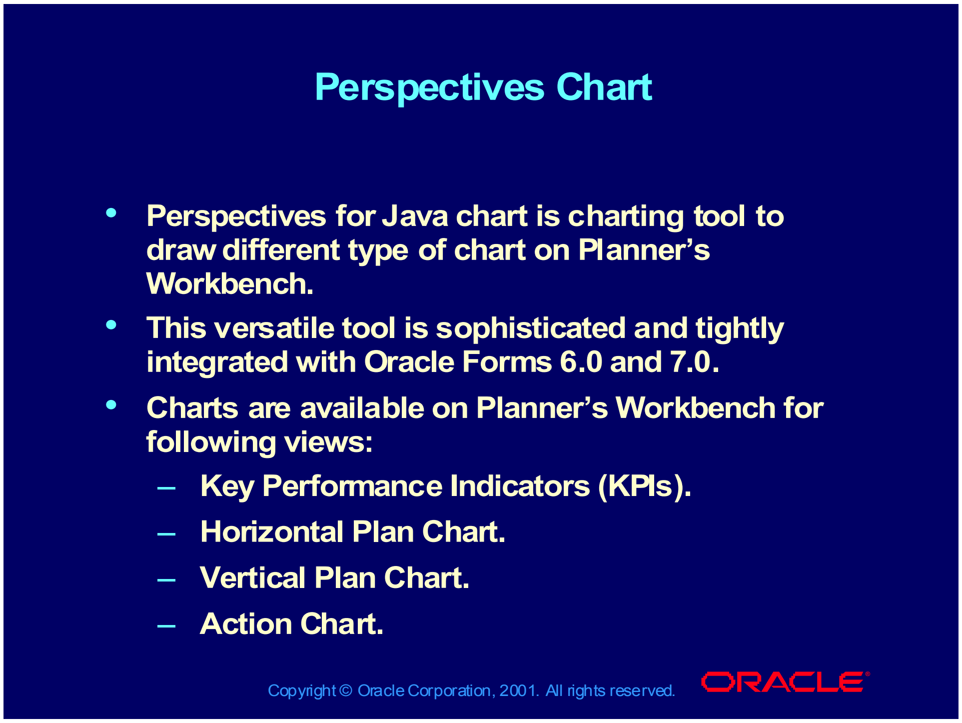
Gantt Chart Technology

The Gantt chart does not utilize the FND TCF socket server to fetch data directly from database.



Horizontal Plan

Horizontal plan is available on Planner’s Workbench.



Perspectives Chart

- The FND jar file, fndtdg.jar is responsible for the APS chart.
- APS charts reside in mscjar.jar

Deployment

- All WebForms (Forms6*i*) have a layout by Java layers, and are deployed as jar files on the client.
- Jar file source location:
  - All fnd jars are located under `$FND_TOP/java/jar/`
  - All APS jars are located under `$MSC_TOP/java/jar/`
- Runtime location of jar files:
  - All fnd jars are located under `/OA_JAVA/oracle/apps/fnd/jar/`
  - All APS jars are located under `/OA_JAVA/oracle/apps/msc/jar/`

Deployment

Jar files loading is very fast compared to loading of Java class itself.

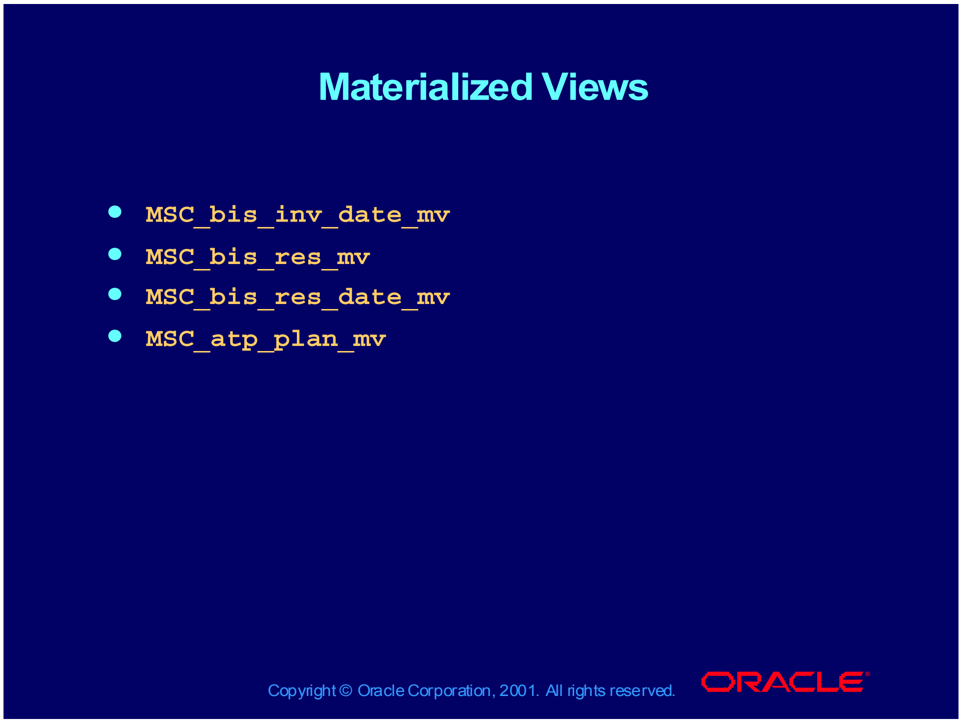
Materialized Views

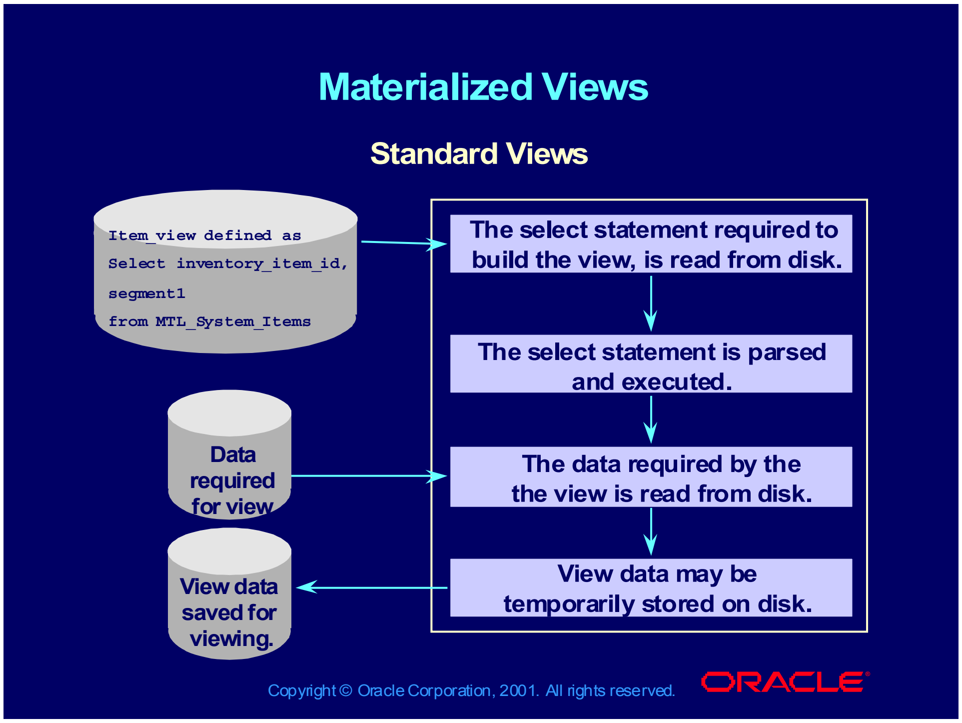
- In Oracle 8.1.5, and later, an atomic complete refresh of a single Materialize View (MV) truncates the MV.
- This is the only case in which atomic refresh truncates an MV instead of using delete to remove its rows.
- Unfortunately, truncate is not reversible, so if a failure occurs work on the MV cannot be rolled back the MV is marked invalid (i. e., unusable).
- In 8.1.6, this is changed so that atomic refresh will delete the MV's rows in this case too. This will make it possible for atomic refresh to always support transactional semantics.

Materialized Views

- To correct this problem, refresh the view by hand. See Oracle RDBMS Administration manual for your version of the Oracle RDBMS.
- The user can obtain the performance gain of using truncate when using complete refresh on a single MV by specifying that the MV be refreshed non-atomically, which always uses truncate.

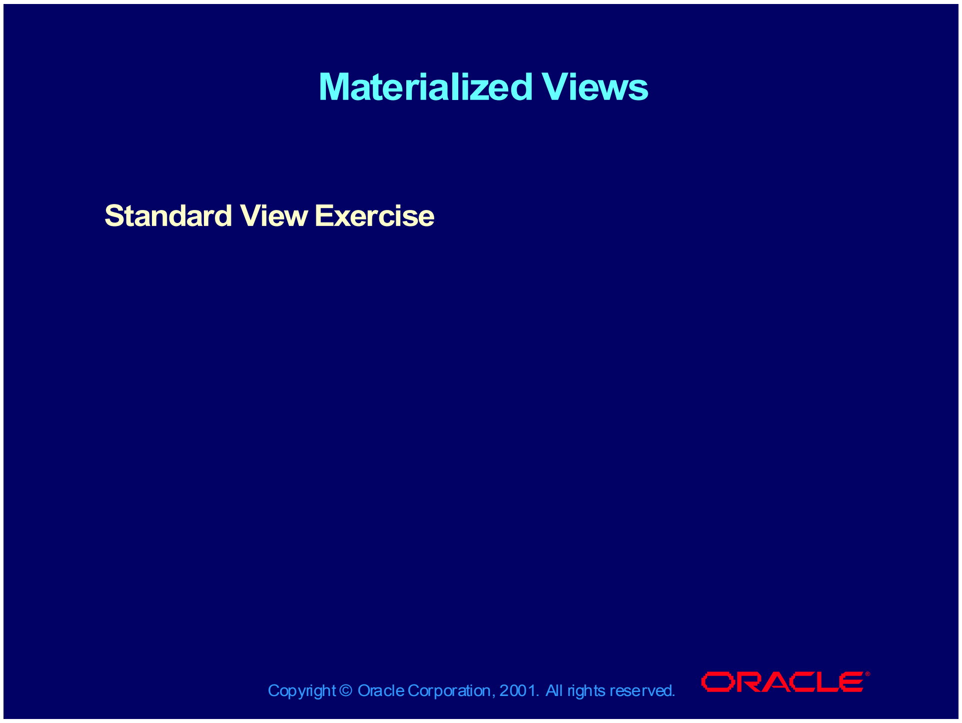
Materialized Views





Standard Views

- Executes SQL when data set is requested.
- Consumes CPU, disk, services bandwidth and memory while constructing the data set.
- Prepares the data set each time the user requests data.
- A normal does not consume disk space.



Standard View Exercise

1. A normal does not consume disk space. By executing the query below, you will see that a view is not inserted into the database:

```
select   pct_free,
             initial_extent,
             num_rows
 from     all_tables
 where    table_name = 'MSC_SOURCES_V';
 no rows selected
```

2. The SQL statement that is used to construct the view, is stored on disk and read into the SGA. To retrieve the SQL statement that constructs the view:

```
select   text
 from     all_views
 where    view_name = 'MSC_SOURCES_V';
```

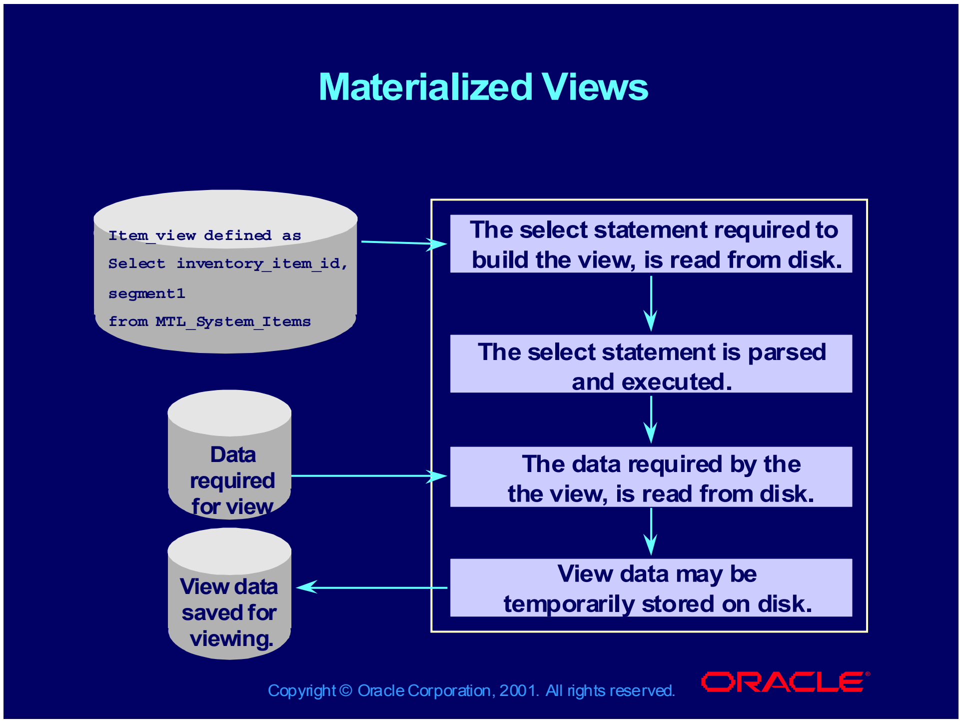
`SELECT MISLV. INVENTORY_ITEM_ID , MISLV.ORGANIZATION_ID , MISLV.SR_INSTANCE` and so on.

3. Views can become invalid. To discover if the view is active:

```
select   owner,
```

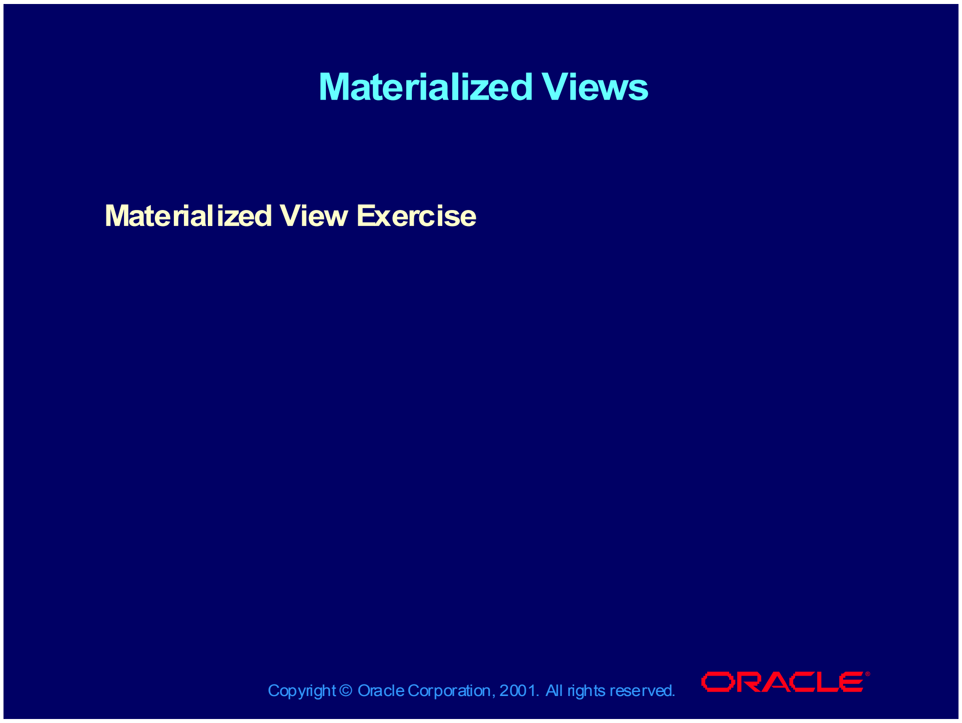
```
           status
from    dba_objects
where   object_name = 'MSC_SOURCES_V';

OWNER                STATUS
----------------     --------------
APPS                 VALID
APPS_MRC             VALID
```

Materialized Views



Materialized Views

- Refreshes data on a regularly scheduled basis.
- Consumes disk space for storage purposes.
- Data set is always ready for user access.



Materialized Standard View Exercise

1. Materialized views are database objects that contain snapshot data. Materialzed views refresh data on a regularly scheduled basis:

```
select  name,
             last_refresh
 from   all_snapshots
 where name like '%_MV';

 NAME                          LAST_REFRESH
 -------------------------  ----------------------
 MSC_BIS_INV_DATE_MV                 01-JAN-50
 MSC_BIS_RES_MV                      01-JAN-50
 MSC_BIS_RES_DATE_MV                 01-JAN-50
```

2. Materialized views are built on SQL statements. To retrieve the query that constructs the materialized view:

```
select  query
 from    all_snapshots
 where name = 'MSC_BIS_RES_MV';
```

```
  select avg(nvl(res.utilization,0)) utilization,
  count(nvl(res.utilization,0)) util_count,
  sum(nvl(res.utilization,0)) util_sum,
  res.plan_id,
  res.organization_id,
  res.sr_instance_id,
  mdr.department_class,
  mdr.resource_group_name,
  res.department_id,
  res.resource_id
  from msc_department_resources mdr,
       msc_bis_res_summary res
  where mdr.department_id = res.department_id
  AND mdr.resource_id = res.resource_id
  AND mdr.plan_id = res.plan_id
  AND mdr.sr_instance_id = res.sr_instance_id
  AND mdr.organization_id = res.organization_id
  group by res.plan_id,  res.organization_id,
res.sr_instance_id,  mdr.department_class,
           mdr.resource_group_name,
res.department_id,  res.resource_id;
```

3. Unlike a standard view, materialized views consume disk space:

```
select  pct_free,
            initial_extent,
            num_rows
 from   all_tables
 where table_name = 'MSC_BIS_RES_MV';


PCT_FREE     INITIAL_EXTENT      NUM_ROWS
----------   ----------------   ----------
        10              57344            0
```

4. To discover if the materialized view is active:

```
select owner,
```

```
          status,
          object_type
 from   dba_objects
 where object_name = 'MSC_BIS_RES_MV';

 OWNER            STATUS         OBJECT_TYPE
 ---------------- ------------- ------------------
 APPS                    VALID      SYNONYM
 MSC                     VALID      TABLE
 MSC                     INVALID    UNDEFINED
 APPS_MRC                VALID      SYNONYM
```

5. Additionally, run the following SQL to check the status of the snapshot upon which the materialized view is built:

```
select status
 from   all_snapshots
 where name like 'MSC_BIS_RES_MV';

 STATUS
 -----------
 VALID
```

6. Is the synonym present?

```
select owner,
        synonym_name,
        table_name
 from   all_synonyms
 where table_name = 'MSC_BIS_RES_MV';


 OWNER        SYNONYM_NAME      TABLE_NAME
 ---------    ----------------   ----------------
 APPS             MSC_BIS_RES_MV    MSC_BIS_RES_MV
 APPS_MRC         MSC_BIS_RES_MV    MSC_BIS_RES_MV
```

Charting FAQ

Problem:

After selecting the node on left panel of the workbench, right panel of the workbench is not showing anything or showing blank screen. Or, Workbench is performing very slow.

Explanation:

If you are launching the workbench for the first time after applying any patch or after bringing up the environment, application will load the new jar files which may cause little delay in popping up the workbench. This happens only at the first launch of workbench.

- Make sure MSC tables are analyzed.
- Make sure jar files are placed in correct directory for the run time environment as mentioned previous.

Problem:

Java exception occurred

Explanation:

Security exception when you log in. Make sure your Jinitiator's Applet access is set to unrestricted. In   such cases, close all of your browser window, open Jinitiator control panel, set access to unrestricted, apply this setting and open application again. Navigation, at the Windows screen, START/Programs/Jinitiator.

Problem:

Class verify error

Explanation:

Ask FND to provide correct version of jar files.

Problem:

Null pointer exceptions

Explanation:

For horizontal plan, make sure that you have selected at least one rowtype to be displayed for horizontal plan on Preference window.

Make sure all of your database objects are valid.

Problem:

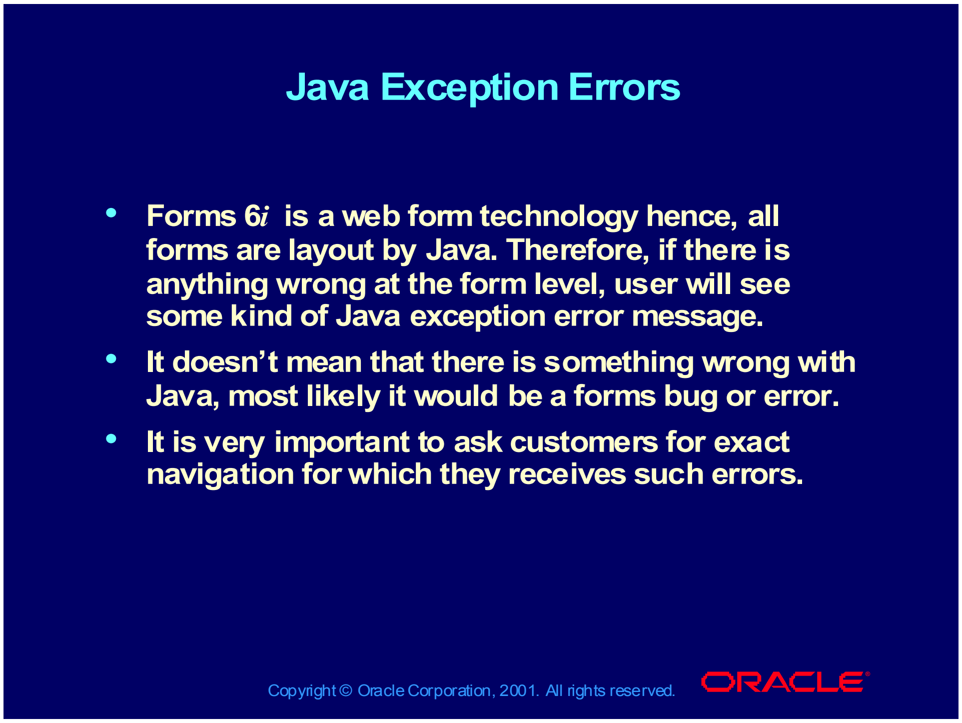
TCF connection fails

Explanation:

- Make sure the profile value for tcf, TCF:HOST and TCF:PORT. Are set correctly.

- To investigate the TCF Socket Server, CD to $OAH_TOP/common/admin/scripts and investigate the ADTCFCTL.sh script. This script contains the port and host that were set for the TCF Socket server. Use these values to set the profile options TCF:PORT and TCF:HOST.
- Investigate the TCF log file for further debugging directions.

Java Exception Errors



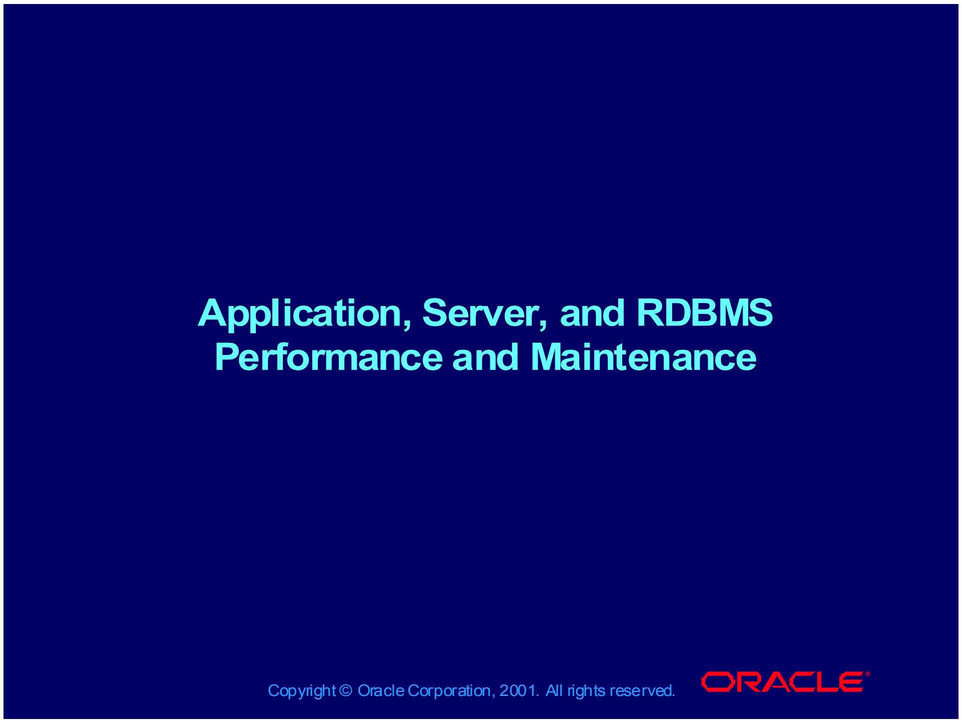
Java Exception Errors

In most of the Java exceptions, it is very helpful to have information from Jinitiator's Java console as well as information from the log file of the failed process.

For example, if there is something wrong with the TCF Socket Server, it is good idea to investigate the TCF Socket Server log file as well as Java console.

Application, Server, and RDBMS Performance and Maintenance

Chapter 11


Application, Server, and RDBMS
Performance and Maintenance
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE

Agenda

Agenda

- Objectives
- APS Database configuration considerations
- Application maintenance
- Server maintenance
- RDBMS maintenance

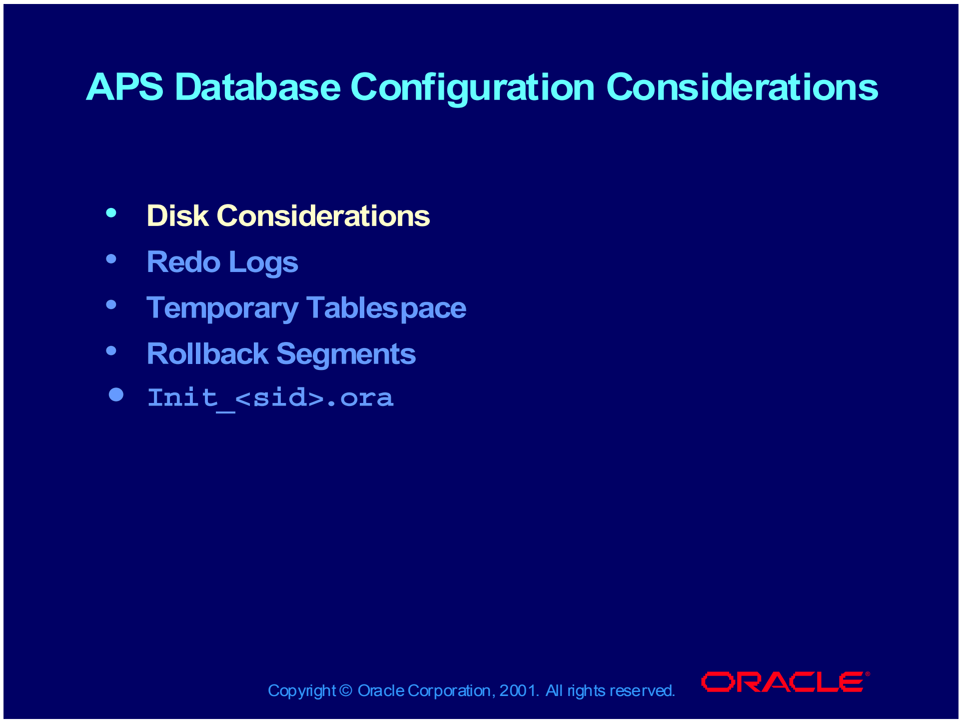
ORACLE

Objectives

Objectives

After completing this chapter, you should be able to do the following:

- Identify database configuration activities important to APS
- Describe common maintenance that can be performed for:
  - Application
  - Server
  - RDBMS



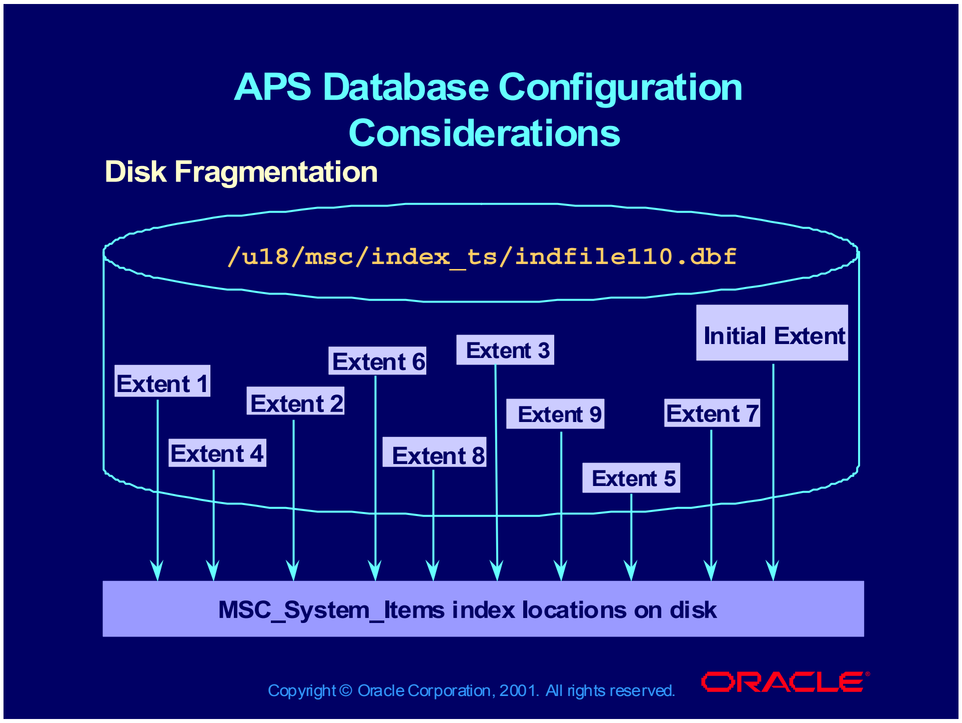
APS Database Configuration Considerations

Disk Striping:

- If your operating system has the ability to stripe logical disk volumes, it is strongly recommended that you utilize this capability. This will help eliminate I/O bottlenecks to data files with hot blocks, and it will enable Oracle Parallel Query (OPQ) to spread I/O across physical drives.
- Logical volumes should be striped across (between 4 and 8) physical devices. The tradeoff here is that while gaining more I/O paths with more physical drives, you are increasing the likelihood that a media failure will occur within your logical volume. Additionally, when there is a failure, you will need to restore more data, since the physical volume is participating in more and larger logical volumes. Mirroring and RAID architecture will reduce your risks here.
- If you do not or cannot use striping, then care should be taken to segregate your database's I/O by distributing the data and redo log files among several physical volumes. Redo logs (two members per group) should be isolated from any high I/O volume data files. Index and data tablespace's data files should be separated, as with temporary tablespaces and rollback segment's tablespaces.
- Creating data files of a small set of consistent sizes has many benefits particularly if you are using raw devices. Having empty raw devices of the same fixed sizes available, allows you to redistribute I/O easier since there will be the right size devices to which to move them. Also have empty raw

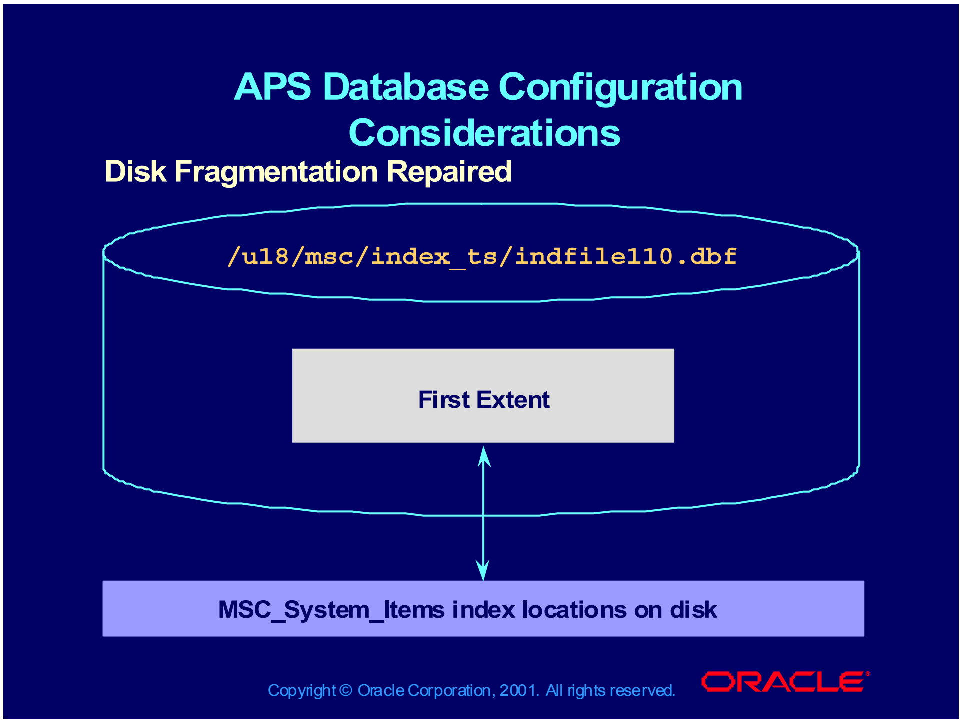
devices of the same size available is critical when having to move a data file from a failing device, or restoring from a backup to a different device.

- Choose your standard sizes based upon the size of your physical drives, and any constraints your volume management software may impose. Try for sizes of approximately 200M, 800M and 2G. If you avoid the larger 2G sizes, your database backups could finish faster, assuming you have parallel threads running, you can avoid having the last thread backing up a 2G file.



APS Database Configuration Considerations

- Fragmentation is real. It is not limited to Oracle. Every disk write operation potentially contributes to disk fragmentation.
- My example above indicates that I am using the operating system file called /u18/msc/index_ts/indfile110.dbf. This file is to be used for my index segments. After becoming familiar with my user business patterns, I believed that the index segment would grow at an nominal rate. Thus I sized the initial extent according to business needs for 6 months of growth.
- Some months had passed when I decided to investigate the tablespace usage within my index tablespace. Unbeknownst to me, my users had decided that they would increase the forecasting horizon an additional 12 months. This explained the choppy and frequent increase in extents. After investigating the primary index on the MTL_System_Items table, I had discovered 9 additional extents spread throughout the index tablespace.
- While 9 extents are not important, many installations that have database objects with over 500 extents, row level row chaining, poorly tuned operating system disk buffers, undersized SGA and overloaded disk controllers. These types of performance problems combined together, result in noticeable performance problems.



APS Database Configuration Considerations Exercise One

If your data is inserted into one extent, disk I/O is minimized.

Oracle Applications that have objects with maxextents < 500 and pctincrease > 0 could have performance problems now or into the future. The following scripts will assist you with maxextents and pctincrease.

1. Script for maxextents:

```
select substr(segment_name,1,25) obj_name,
        max_extents
 from dba_segments
 where max_extents < 500
 and (segment_type = 'TABLE'
 or segment_type = 'INDEX')
 and owner in
     (select oracle_username
      from applsys.fnd_oracle_userid)
```

Script for pctincrease:

```
select substr(segment_name,1,25) obj_name,
   pct_increase
from dba_segments
where pct_increase > 0
and (segment_type  = ‘TABLE’
```

```
or segment_type = 'INDEX')
and owner in (select oracle_username
from applsys.fnd_oracle_userid);
```

APS Database Configuration Considerations

APS Database Configuration Considerations

- Disk Considerations
- Redo Logs
- Temporary Tablespace
- Rollback Segments
- `Init_<sid>.ora`

ORACLE

APS Database Configuration Considerations

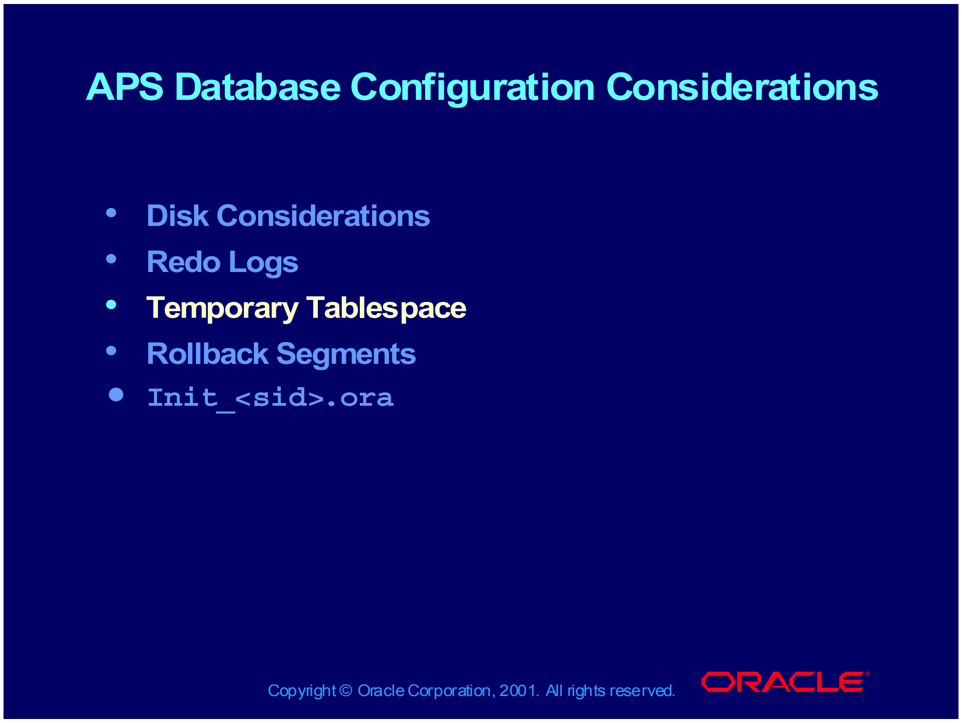
When configuring redo logs, there are several goals to bear in mind:

Redundancy: There needs to be at least two members of each group and they need to be stored on physically separate drives. This is critical is assuring database recovery after media failure. Two groups should be enough for a small implementation. If you find that processes are waiting on the archiver to copy the redo log, the increase the number to three or four groups. Four should be enough for most large implementations.

Log Buffer: The redo log buffer needs to be the right size. Too small and LGWR (log writer) becomes a bottleneck. Too large and you will waste memory without a performance benefit.

Size: Redo logs need to be large enough to avoid frequent file switches. The redo logs should be sized large enough to prevent log file switches from occurring more frequently than 10 minute intervals. Even during peak database load. This is to reduce the overhead of database checkpointing. The minimum size, for your redo logs should be 100m. Larger implementation could use 400m or 800m.

APS Database Configuration Considerations



APS Database Configuration Considerations Exercise Two

For the temporary tablespace, our goal is to reduce extent management and make the tablespace large enough to handle all the sorting and index building. One of the best ways to reduce extent management is by using the 'EXTENT MANAGEMENT LOCAL' clause.

```
create temporary tablespace temp
tempfile '/myfile'  size 400m
extent management local uniform size 4m;
```

- Instead of 400m, use one of your standard file sizes. If you are not using disk striping, you can place the data files on different physical drives.
- One note of caution here is that the files for this type of tablespace are not shown in the DBA_DATA_FILES view. You must query the DBA_TEMP_FILES view to track your temporary tablespace. You may need to adjust you diagnostic scripts accordingly.

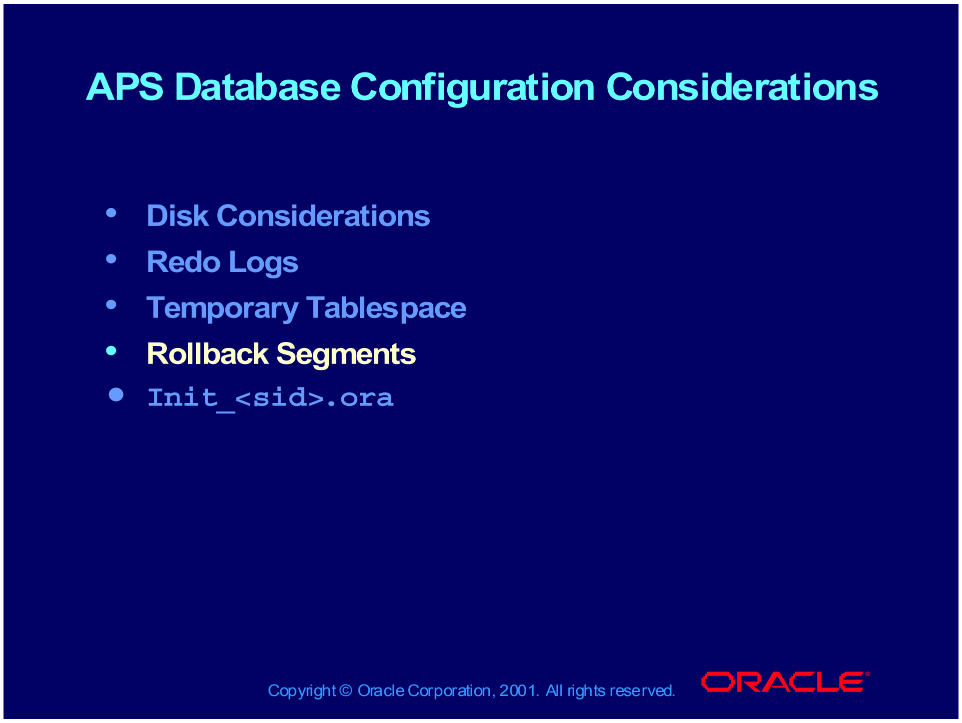
One note of caution here is that the files for this type of tablespace are not shown in the DBA_DATA_FILES view. You must query the DBA_TEMP_FILES view to track your temporary tablespace.

You may need to adjust you diagnostic scripts accordingly.

```
 select substr(file_name,1,30) file_name,
            file_id,
            tablespace_name
  from   dba_temp_files;
```

```
FILE_NAME                       FILE_ID         TABLESPACE_NAME
--------------                  -----------     -----------------
/u02/oracle/11i/apsdata/tmp1vi            1      TEMP
```

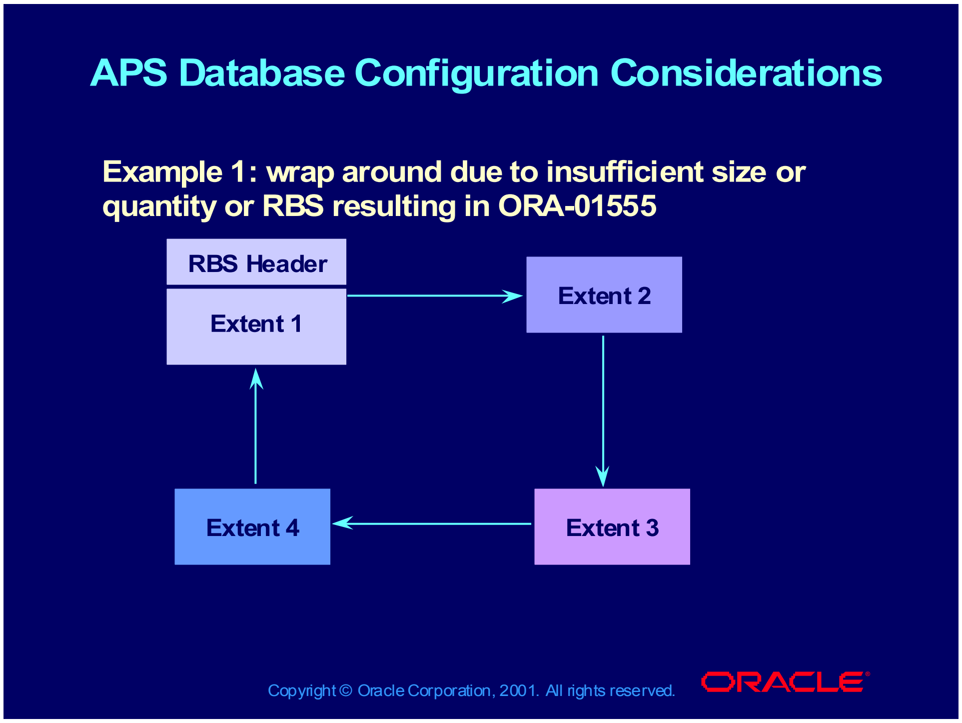
APS Database Configuration Considerations



APS Database Configuration Considerations

- Rollback segments tend to zap too much administrator energy. Whether the rollbacks are too small or too few in number, the method you choose to correctly size your RBS is within your grasp. When RBS are too small in size, ORA-01555 may result. This is a snapshot too old.
- Rollback segments should be sized large enough so that long running queries do not get 'ORA-01555' Snapshot too old. This occurs when your rollback segments are too small and a segment wraps around and begins to use previously used rollback space for your transaction. The tablespace should be sized large enough to handle the expansion of the rollback segment in order to accommodate a long running update.
- Use the OPTIMAL clause when creating the rollback segments. This will allow for the shrinking of any expanded rollback segments, making room for another rollback segment to expand the next time a long running update occurs. Bear in mind that some users of Oracle do not use the OPTIMAL clause in order to save the CPU/Disk resources.
- All rollback segments can be the same size when using large segments, since any segment can expand to accommodate the large transactions.
- If your RBS are too big and the optimal too low, you'll suffer performance hits relating to activity necessary to maintain the indicated size.
- A rollback segment maintains the snapshot of the changed data as long as the transaction is still active. This means that a commit or rollback has not

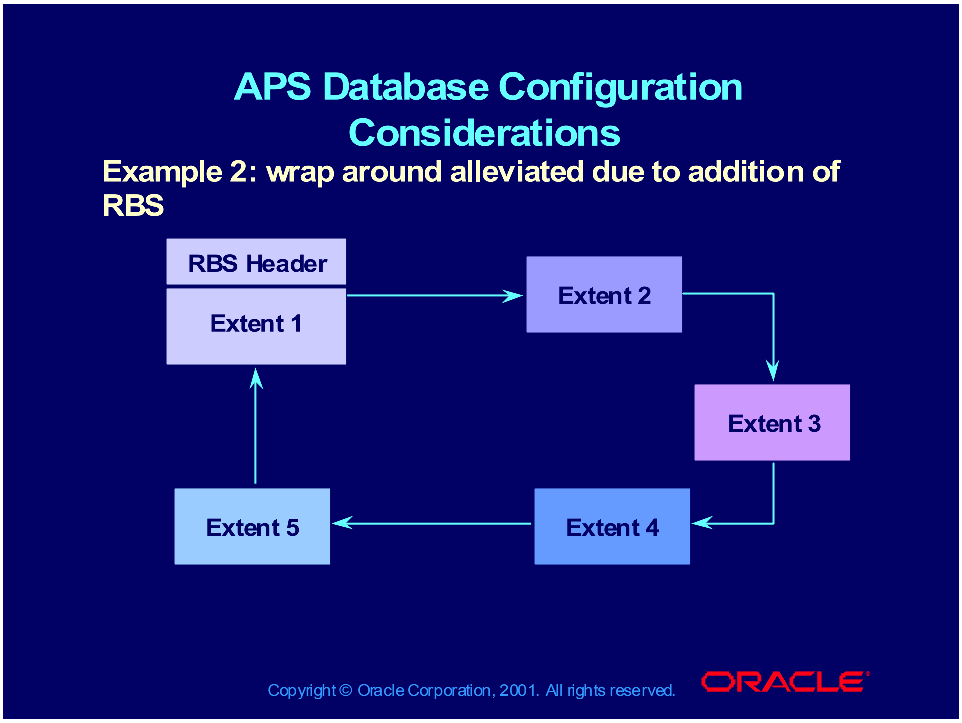
been issued. Once the transaction is complete, the data in the RBS is erased from the RBS thus freeing the RBS.



APS Database Configuration Considerations

Rollback Segments

Example 1 demonstrates a wrap-around. In this instance there were not enough RBS to handle the user/processing load. When the RBS started to reuse RBS number 1, an ORA-01555 occurred. A long running query may not be able to reconstruct the snapshot due to the wrap around concept and an overwrite of rollback segments could occur. Larger rollback segments will reduce the chance of reusing the committed transaction slots, but so large that they use needed disk space or waste CPU cycles in Oracle kernel maintenance.



APS Database Configuration Considerations Exercise Three

Example 2 demonstrates the wrap-around condition alleviated due to the addition of RBS. While this is a simple example, the point is proper RBS translates into increased system availability.

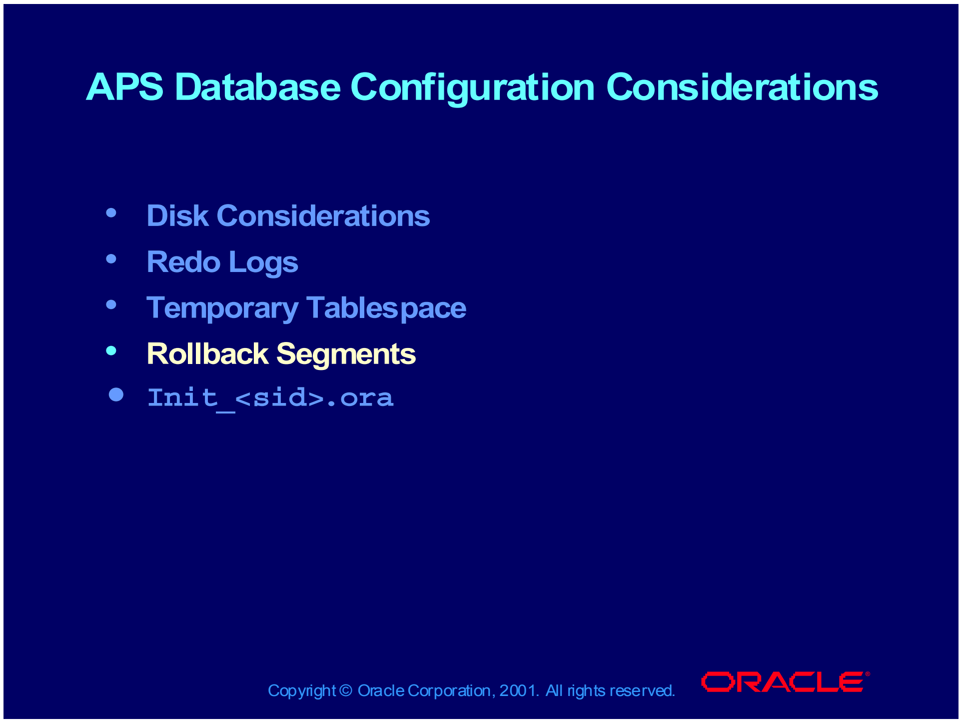
To determine an acceptable rollback segment configuration, analyze the results from the following query:

```
select *
  from    V*WAITSTAT
  where   OPERATION = 'buffer busy waits'
  and     CLASS = 'undo segment header';
```

If you have buffer busy waits, you have a problem. The seriousness of the RBS problem may be small, but by understanding the processing needs of your user community you can determine the immediate maintenance and RBS need. It is recommended that you have 1 RBS for every 10 online users and that you monitor the highwater mark for size, quantity and the setting of optimal. Longer running queries or batch type jobs may need their own RBS configuration. You may have to turn batch RBS on a night and OLTP RBS on during the day.

For additional information relating to RBS, see Note:100964.1 on Metalink.

APS Database Configuration Considerations



APS Database Configuration Considerations Exercise Four

All rollback segments can be the same size when using large segments, since any segment can expand to accommodate the large transaction.

For a small implementation, create segments such as:

- Create rollback segment rollback_segment_name
- Tablespace rollback_tablespace_name
- Storage (initial 10m next 10m minextents 20 optimal 200m);

Create at least two of these segments. Make your tablespace large enough to hold the optimal size of the segments plus enough for one of the segments to double in size. In this case, the tablespace should be at least 600m.

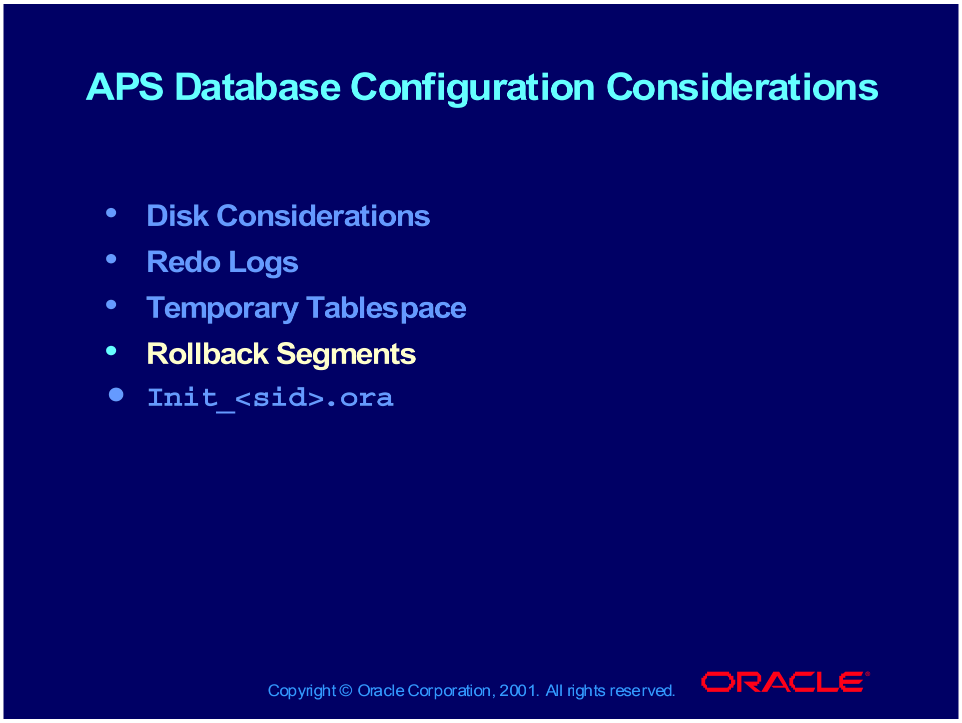
Larger implementations should use 4 or more segments of 20m extents with minextents = 20 and optimal 400m.

Monitor you rollback segments by querying the V$ROLLSTAT view:

```
select s.usn,
           s.status,
           r.name,
           s.extents,
           s.waits
 from  v$rollname r, v$rollstat s
 where s.usn = r.usn;
```

A high number of waits indicates that additional rollback segments are needed.

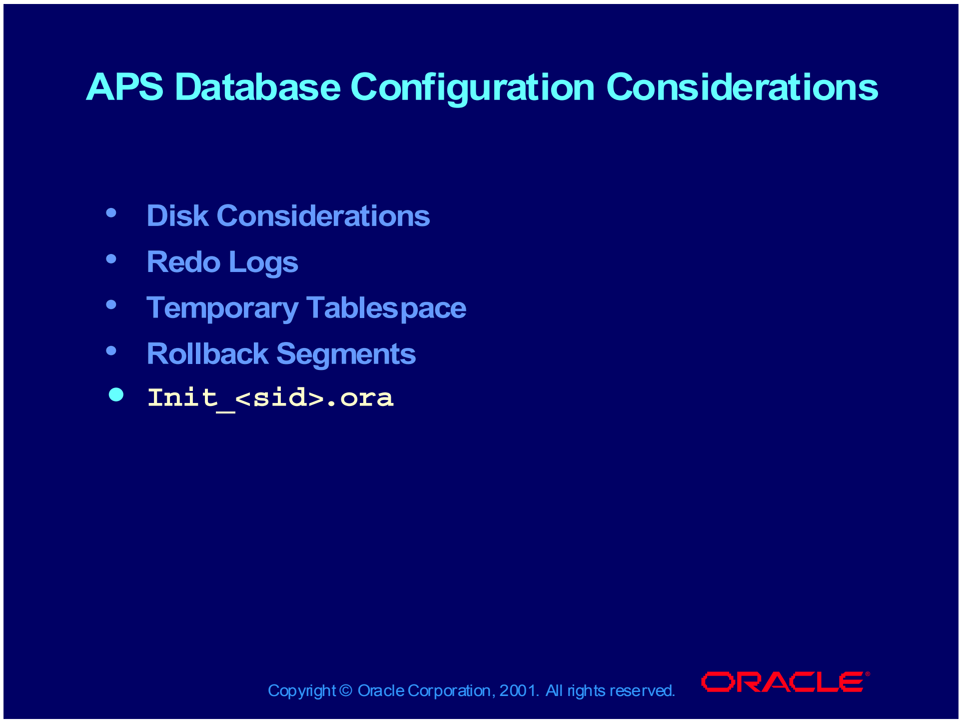
APS Database Configuration Considerations



APS Database Configuration Considerations

You may want to consider creating your tablespace for rollback segments with the ‘EXTENT MANAGEMENT LOCAL UNIFORM SIZE extent_size’ clause. This will greatly reduce extent management overhead when extending and shrinking rollback segments. Make the uniform size the same as your rollback segment extents.

If you choose to use the larger rollback segment sizes, the increase the transactions_per_rollback_segment parameter to 10 from the default of 5.



APS Database Configuration Considerations

- The parameters to note here are:
- job_queue_processes = specifies the number of concurrent processes to use to process jobs.
- job_queue_interval = time in seconds which controls how often the job processes wake up to process pending jobs.
- If you set job_queue_processes to 1 and one job fails, then it may keep getting re-submitted, and prevent everything else in the queue from being completed. So minimum convenient value is 2.
- It is recommended that you set job_queue_interval too. Default value is 60. It means that every minute job processes will wake up to process pending jobs. It could be too often for your needs and will overload your system. You can set it to 600 (10 minutes) or any other value you consider more adequate. You can set this parameter by adding the following line to the same init<sid>.ora file:

  job_queue_interval = 600 or the value you choose.
- The job_queue_processes must be >= 1 in order to execute the jobs automatically.

Application Maintenance

Application Maintenance

- Purge staging tables
- `.MGR/ .LOG/ .OUT` output file locations
- Clean up the concurrent manager tables.
- Item planning
- Manufacturing and scheduling planning horizons

ORACLE

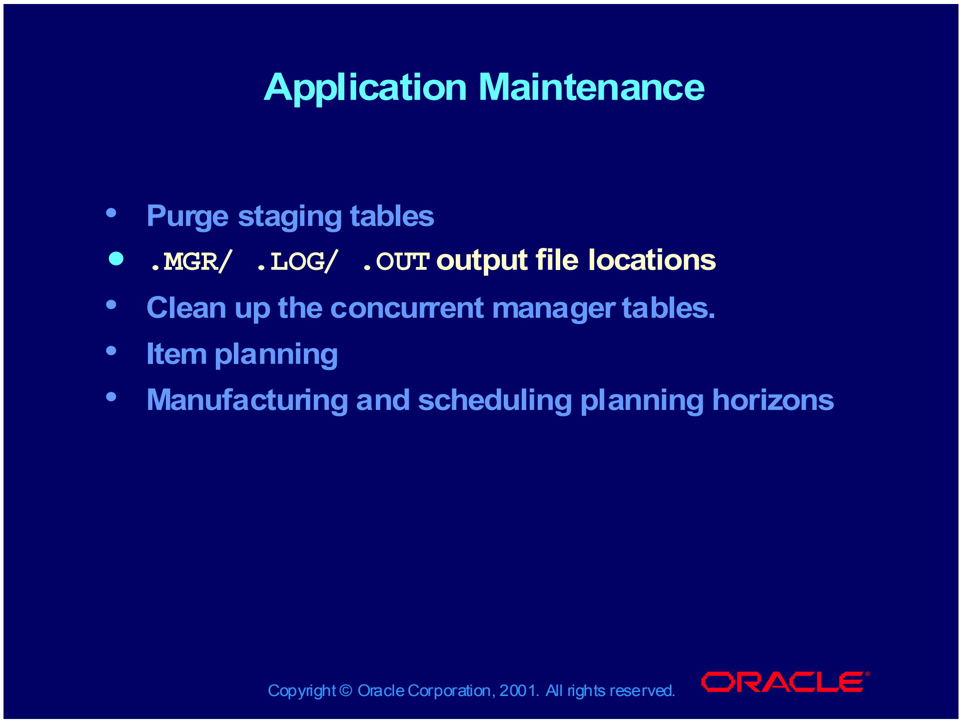
Purging

This process updates column ST_STATUS in MSC_APPS_INSTANCES to 4 (to indicate Staging table purge is in process) before purging the Staging tables for the Instance code. Profile option : MRP_PURGE_BATCH_SIZE controls the number of rows deleted from each Staging table in a particular batch. After every batch deletion, the transactions are committed to the database. When the Purge is completed successfully, the ST_STATUS in MSC_APPS_INSTANCES is updated to 0. This marks the end of the Data Collection process.

The profile option, MRP_PURGE_BATCH_SIZE will affect the usage of your Roll Back Segments, RBS. This represents the number of rows processed before a commit occurs. If you are having problems with rollbacks, ORA-00060, or ORA-00054 set this to a lower number.

If you are not using previously created partitions, purge them by running the "Purge Planning Partition" concurrent program.

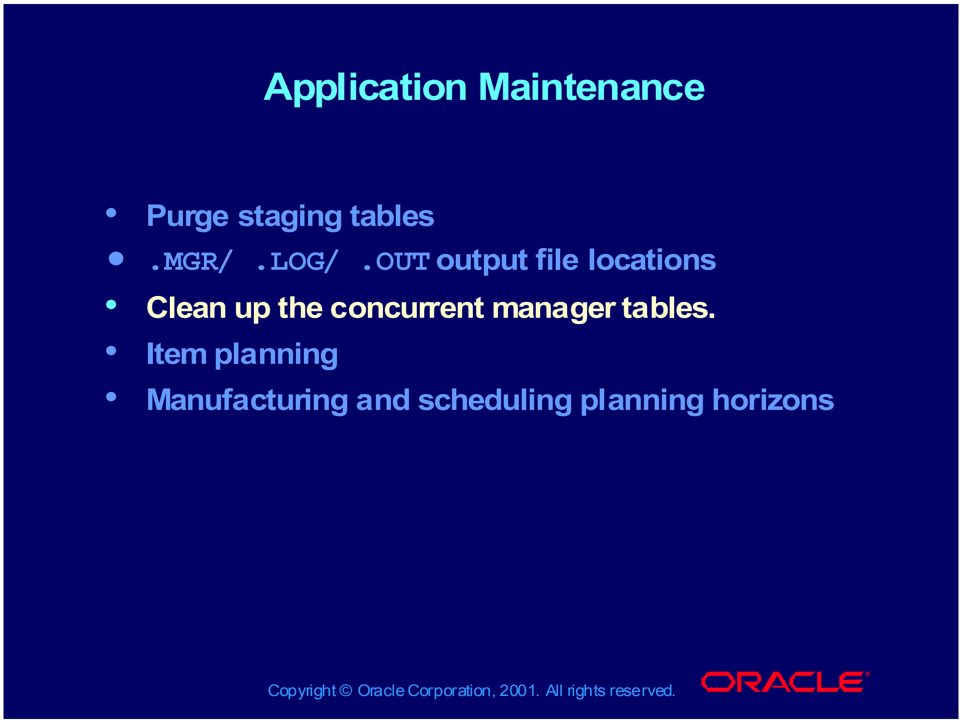
If you have data residing within the staging tables, run the concurrent request "Purge Staging Tables."



.MGR/ .LOG/ .OUT output file locations

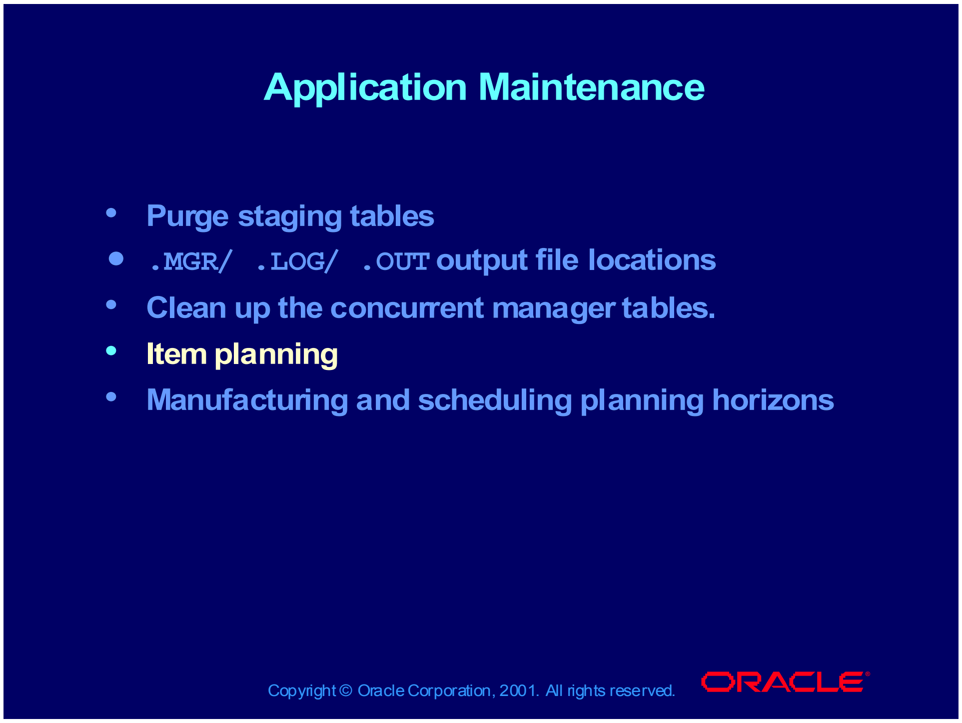
- Keep these files away from disks that are being used for Rollback segments and redo logs.
- The concurrent manager will place log files in application home directory under the log directory and output files in the application home directory under the out directory.
- Note that $APPLCSF must be a full, absolute path. If $APPLCSF is not set, it places the files under the product top of the application associated with the request. So for example, a PO report would place files under $PO_TOP/$APPLLOG and $PO_TOP/$APPLOUT.
- Of course, all these directories must exist, have enough space, be purged regularly and have the correct permissions.
- Note that all concurrent requests produce a log file, but not necessarily an output file.

Application Maintenance



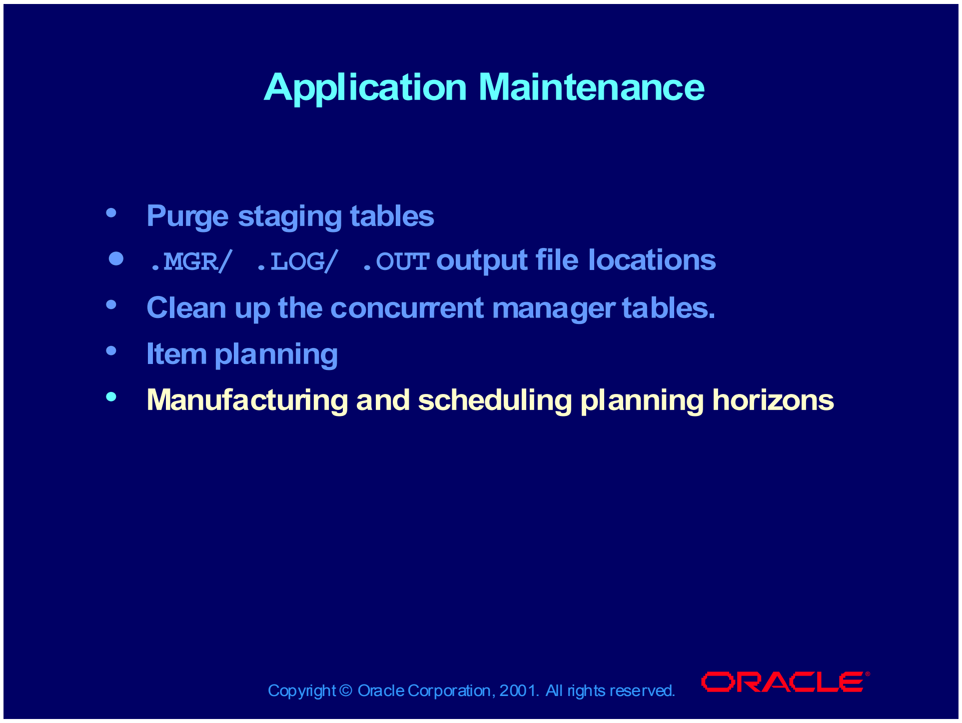
Concurrent Process Cleanup

- Run the process that cleans up the concurrent manager tables, FNDCPPUR
- Specify the necessary parameters to purge both table entries and $APPLCSF/log and /out files. Consider eliminating “dead” processes in fnd_concurrent_requests as part of your weekly maintenance.



Item Planning

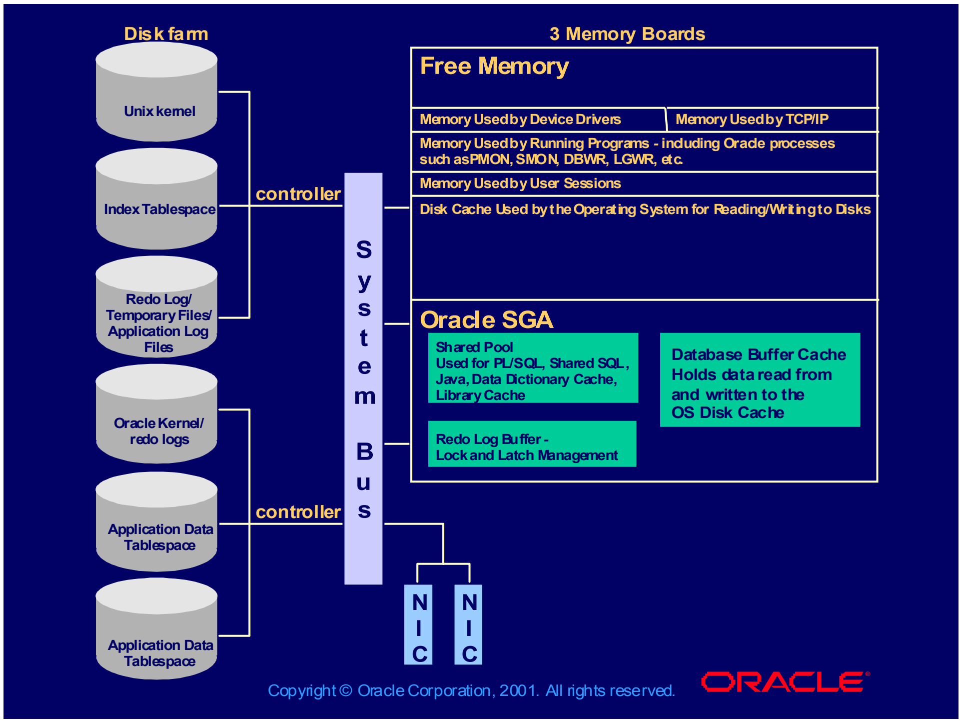
You may wish to try planning for all items. This will prevent the exploder process from running. Consider stress testing demand schedule items vs all items for your specific implementation of Oracle applications.



Manufacturing and Scheduling Planning Horizons

- What is your forecast bucket size?
- What is your planning bucket size?

Best performance can be achieved by using larger bucket sizes as time progresses into your planning horizon.

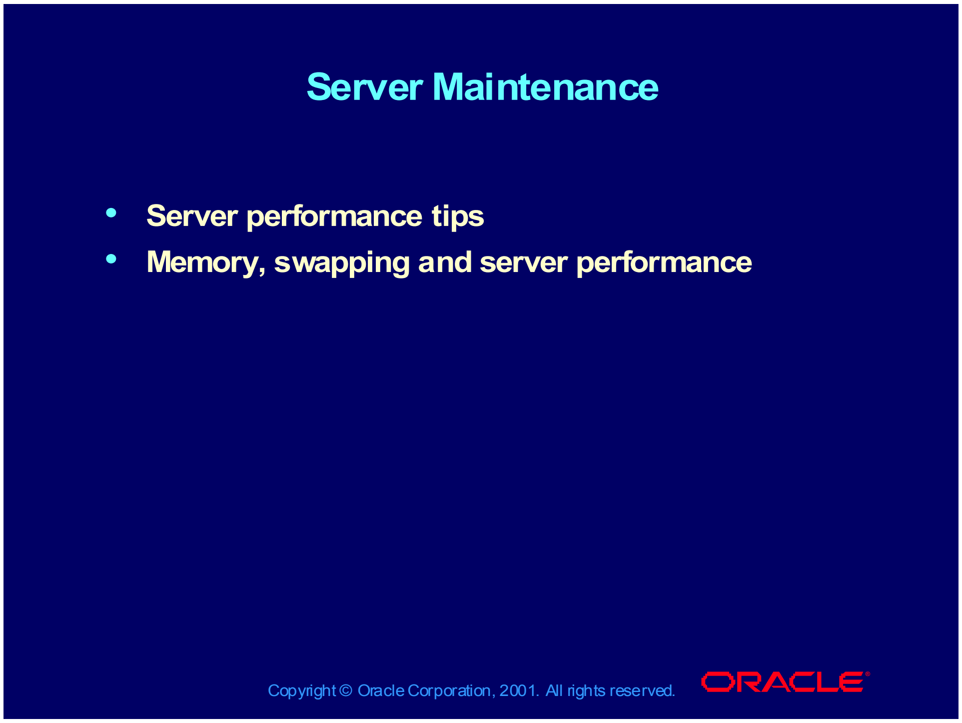


Server Maintenance

- Managing your unix or NT server(s) is an important topic in your maintenance methodology. If your servers resources are limited, extra attention may be needed to achieve optimal performance. Consider the following potential bottlenecks:
  - NIC card software and setup,
  - SQL*Net software version or setup,
  - Disk controller saturation,
  - Disk fragmentation or disk shortages,
  - Unix configured memory availability or memory paging and swapping,
  - Oracle database setup and performance,
  - User processing requirements,
  - CPU limitations and saturation,
  - Operating system software versions,
  - Resource loads such as backups.
- Available on Metalink, Note:100991.1 discuses unix server performance and note: discusses NT performance. By drilling into these documents you will learn how sar can be used to alert a system administrator of coming problems. By working through note: you will learn how the NT Performance Monitor can beused to alert the system administrator of highwater mark usage. By translating these tools into a methodology, you

will have everything you need to monitor your enterprise servers and plan for maintenance.

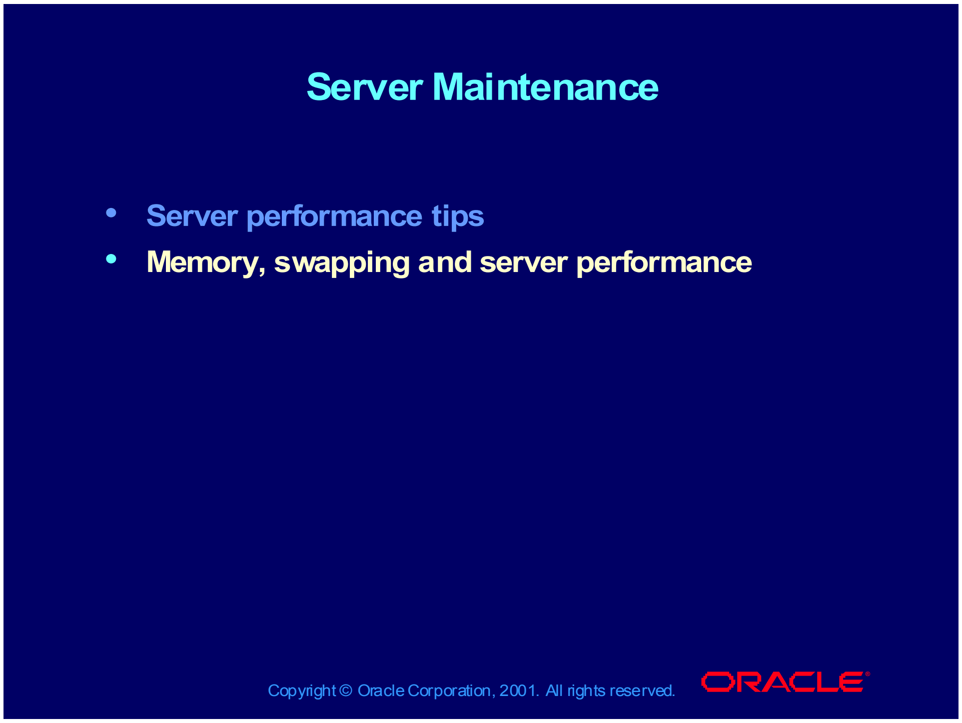
- By utilizing hardware vendor tools or by automating the use of unix commands such as sar, netstat, ipcs and vmstat (Performance Monitor in NT—for an in-depth NT diagnostic guide, see note), you can continuously maintain optimal performance. Establish benchmarks and monitor the demand on your hardware. By charting the usage trends, you'll be far ahead of corporate demand. Set your monitoring systems to alert the system administrator that the system load is outside of your performance target area. And that routine maintenance needs to be scheduled.
- Determine bottlenecks quickly by developing a checklist and automating as much of it as possible. Use this checklist every time a change is made to any server within your enterprise. Additionally, provide a checklist to be used when reporting problems. For example, if your performance has degraded suddenly, have any software changes been made? This type of approach will ensure maximum performance and availability.
- Always keep a log of changes made and who made them. Consider an electronic quality checklist, enforcing quality compliance. When changes are made to systems, permission/privilege problems or failing to recompile executables can lead to a production down scenario. When migrating databases, do you have a checklist of quality points?
- Do database department policies conflict with server department policies? For example, do backups run smoothly, within the allotted time?



Performance Tips

- Monitor Operating System Cache. It is important to note that Operating System cache must be tuned according to your needs.
- Monitor CPU Utilization during MBP execution. If your CPU usage is frequently 60%–70%, watch for spikes. Spikes at 100% represent loss of process service or processing time.
- Disk I/O check: APS uses disk efficiently. There may be performance enhancements that can be made to Oracle SQL, but poor utilization of disk space is a leading contributor to performance problems.

Server Maintenance



Memory, Swapping, and Server performance

Your database server keeps running out of memory and my server performance monitor indicates swapping. What can you do?

a. Using Note:100991.1, available on Metalink, determine your free memory and memory usage.

b. Examine the output of the ulimit -a command.

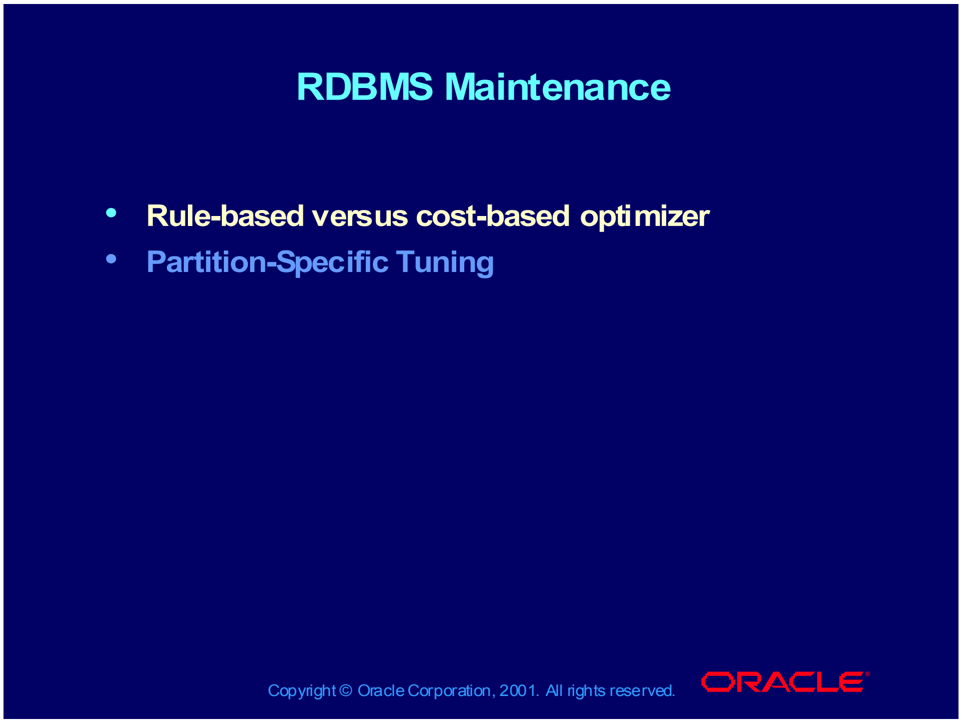
c. Is your largest shared memory segment as large as your SGA+3%?

d. What is the CPU usage? When your CPUs are utilized 100%, and you have evidence of swapping, thrashing is occurring.

e. Examine how you are using your platform memory.

f. Perhaps your SGA is too big? Can you reduce the size of the shared_pool? See Note:100964.1, available on Metalink.

RDBMS Maintenance



Rule-based versus cost-based optimizer

Prior to 11*i*, Oracle Applications have used the rule based optimizer.

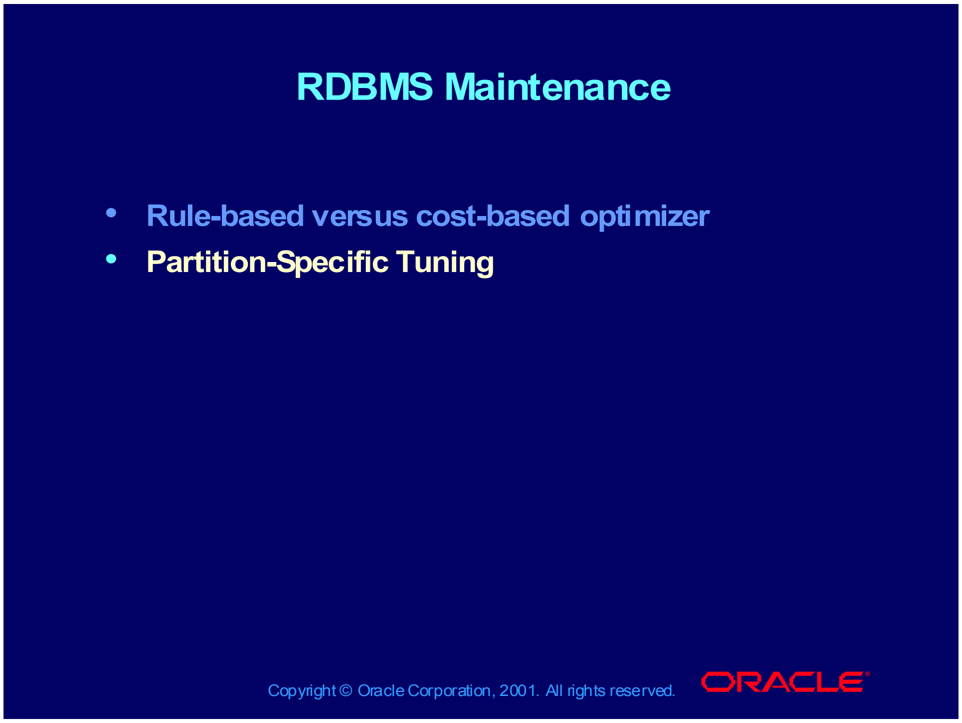
Applications development has reviewed all SQL statements and tuned them to be used by the cost based optimizer. As an Oracle Applications DBA, you need to be aware of the FND_STATS utility. CBO uses statistics to decide the execution of SQL statements and FND_STATS is a vital part of collecting this data.

To analyze the MSC Schema run $MSC_TOP/bin/MSCSTAT MSC 10 4

a. Analyze all of your tables regularly. If you analyze only your busiest or largest tables, the CBO may make wrong decisions. Remember that timeliness is important. A table that is filled during a later batch run needs to be analyzed as well.

b. If you are hunting for poorly written code, the CBO will most likely use the leftmost table as the driving table and join order will also be opposite of the rules based optimizer as the CBO will join left to right in the from clause.

c. When comparing performance and execution of the optimizer, it is a good idea to replicate the production environment in test as closely as possible. Otherwise, your results could vary.

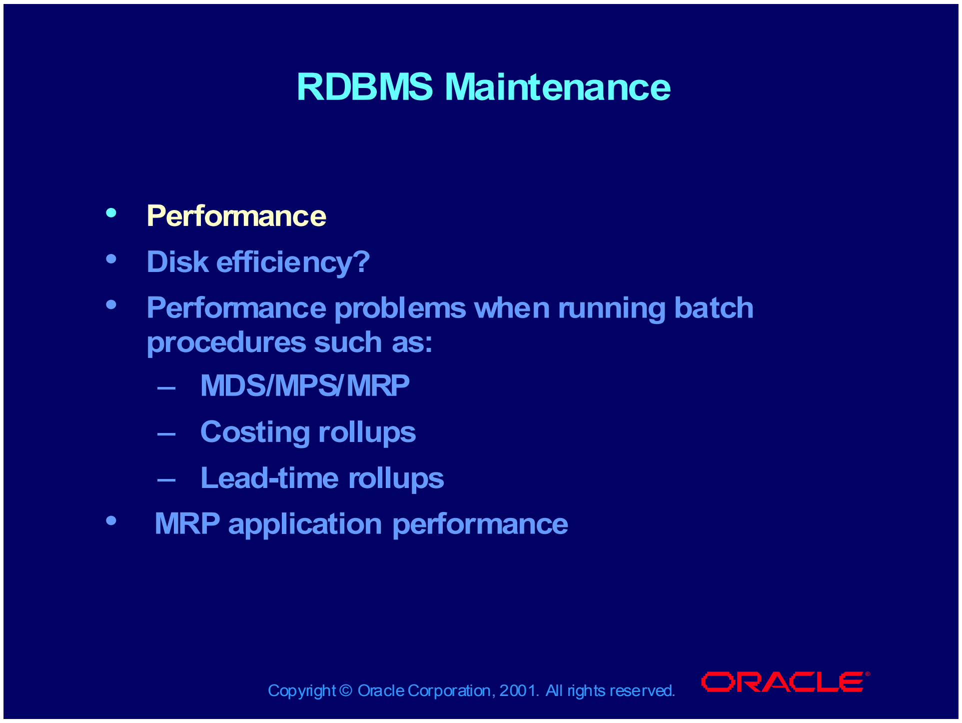


Partition Specific Tuning

You can use the alter table command to add, drop, exchange, move, modify, rename, split, and truncate partitions. You may have to use the alter table command to perform routine maintenance on heavily used partitioned tables.

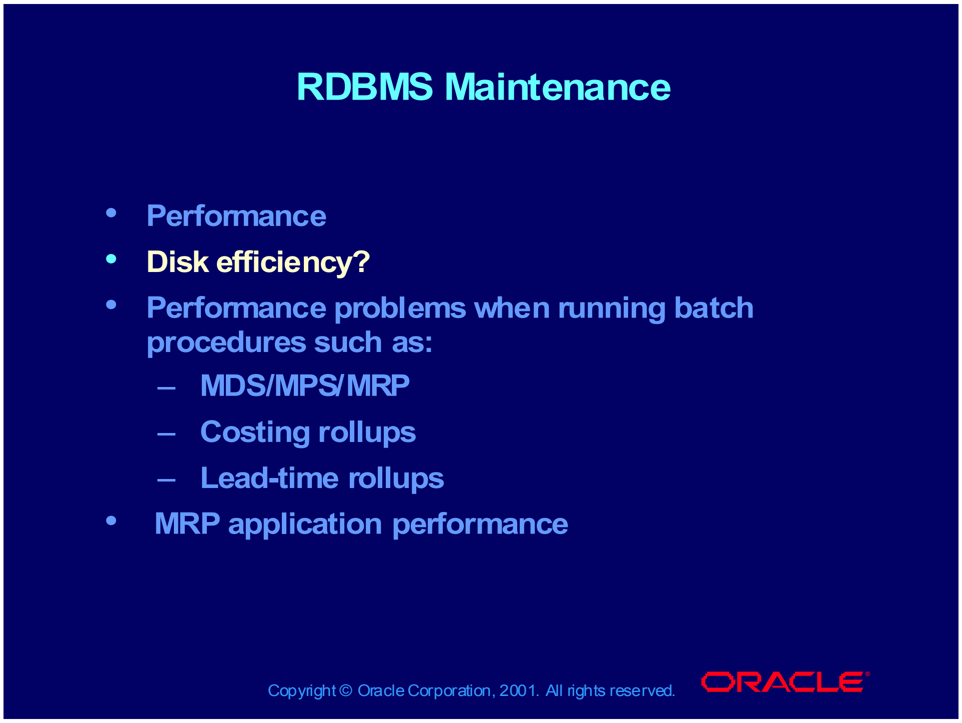
For example, the partition range column may have significantly changed causing fragmentation.

Place large table partitions in different tablespaces. Place these tablespaces within datafiles that reside on different disks.



Performance

If the performance has gradually declined, database maintenance may be necessary. If the performance issue has suddenly appeared, enterprise or site data processing changes may be the culprit. If this performance issue has been evident since the implementation of Oracle Applications, consider reviewing the human tasks that surround the automated process. Can changes be made to the human task?



Disk Efficiency

(All notes listed below are available on Metalink.)

a. Separate Redo Logs from data. See Note:100964.1

b. Split tables and indexes. See Note:100960.1

c. Do you have a schedule for reorganizing your data?

d. Do you regularly analyze your database objects? See Note:101015.1

e. Have you studied your disk hit ratio?

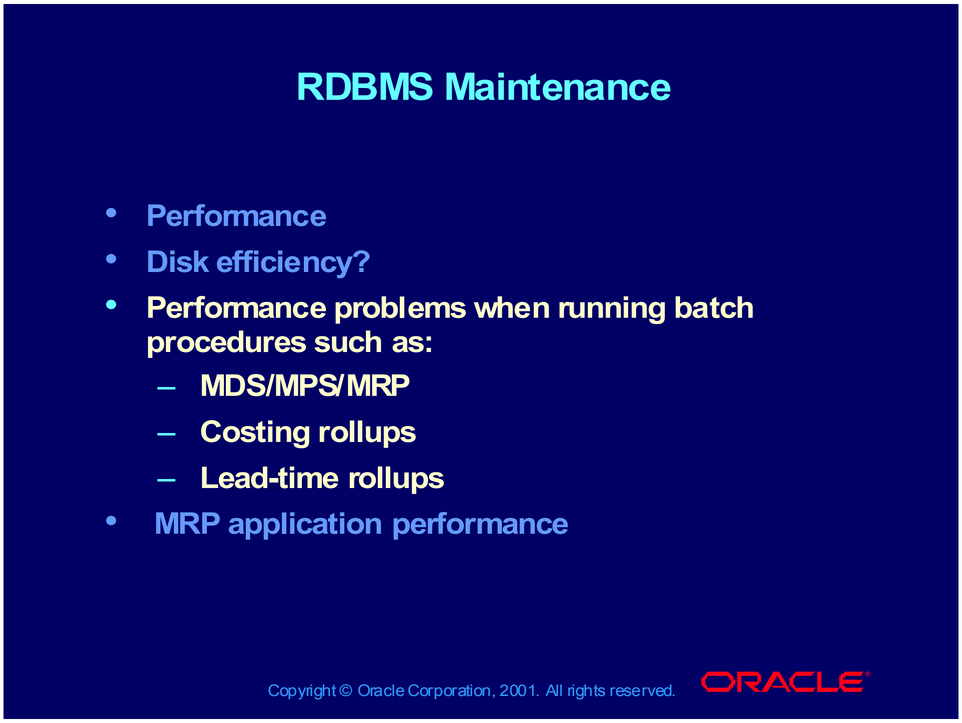
f. See Note:100991.1 concerning SAR.

g. Have you investigated your sort_area_size efficiency? See Note: 100964.1

h. Run the balanced disk query found in Note:100960.1

i. Are your partitions spread correctly throughout the disk farm?

RDBMS Maintenance

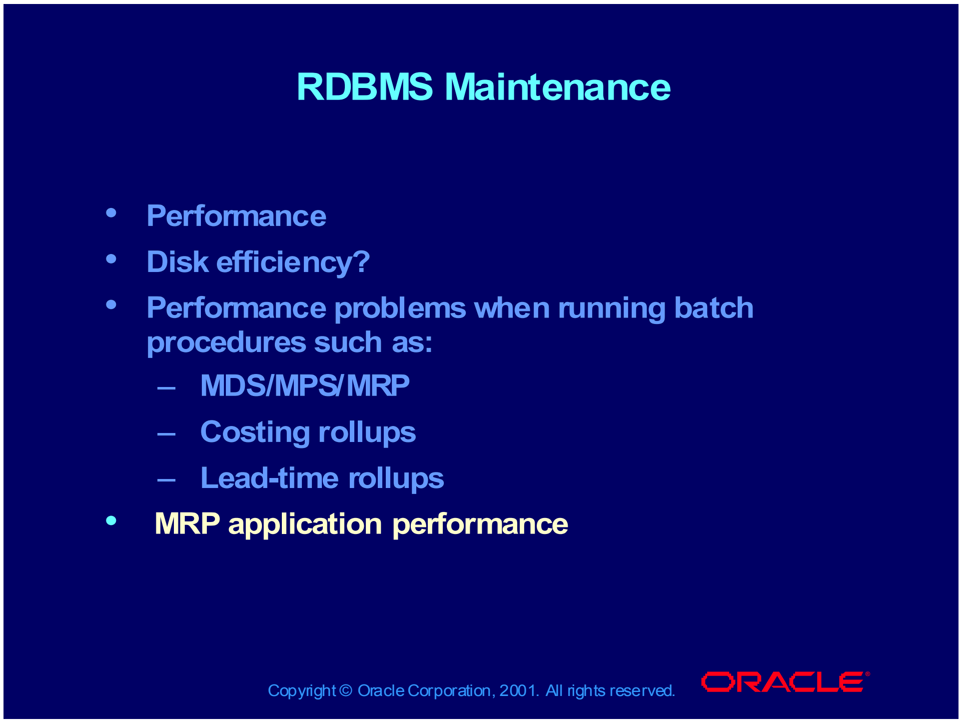


Performance problems in batch

(All notes listed below are available on Metalink.)

Likely causes:

- Database objects the require maintenance. See Note:100960.1
- Resources that are overly taxed. See Note:100991.1
- Application use. See Note:101015.1
- Undersized Oracle kernel. See Note:100964.1
- Undersized rollback segments. See Note:100960.1



MRP Application Performance

(All notes listed below are available on Metalink.)

Investigate the planning methodology used when implementing and using the Oracle manufacturing applications. You may be able to make functional adjustments that will improve performance. See Note:101015.1

Appendix A: Staging Tables

Chapter 12

Staging Tables

The following is a list of the intermediate staging tables used to capture different datasets from the source instance.

Bill of Materials

MSC_ST_BOM_COMPONENTS

MSC_ST_BOMS

MSC_ST_COMPONENT_SUBSTITUTES

MSC_ST_ROUTINGS

MSC_ST_ROUTING_OPERATIONS

MSC_ST_OPERATION_RESOURCES

MSC_ST_OPERATION_RESOURCE_SEQS

MSC_ST_PROCESS_EFFECTIVITY

MSC_ST_OPERATION_COMPONENTS

Bill of Resources

MSC_ST_BILL_OF_RESOURCES

MSC_ST_BOR_REQUIREMENTS

Calendar

MSC_ST_CALENDAR_DATES

MSC_ST_PERIOD_START_DATES

MSC_ST_CAL_YEAR_START_DATES

MSC_ST_CAL_WEEK_START_DATES
MSC_ST_RESOURCE_SHIFTS
MSC_ST_CALENDAR_SHIFTS
MSC_ST_SHIFT_DATES
MSC_ST_RESOURCE_CHANGES
MSC_ST_SHIFT_TIMES
MSC_ST_SHIFT_EXCEPTIONS
MSC_ST_NET_RESOURCE_AVAIL

Demands

MSC_ST_DEMANDS

Demand Classes

MSC_ST_DEMAND_CLASSES

Hard Reservations

MSC_ST_RESERVATIONS

Items

MSC_ST_SYSTEM_ITEMS

Supply

MSC_ST_SUPPLIES

Sales Orders

MSC_ST_SALES_ORDERS

Resources

MSC_ST_SIMULATION_SETS
MSC_ST_RESOURCE_GROUPS
MSC_ST_DEPARTMENT_RESOURCES

Categories

MSC_ST_ITEM_CATEGORIES
MSC_ST_CATEGORY_SETS

Safety Stock

MSC_ST_SAFETY_STOCKS

Schedule Designator

MSC_ST_DESIGNATORS

Sourcing

MSC_ST_ASSIGNMENT_SETS
MSC_ST_SOURCING_RULES
MSC_ST_SR_ASSIGNMENTS
MSC_ST_SR_RECEIPT_ORG
MSC_ST_SR_SOURCE_ORG
MSC_ST_INTERORG_SHIP_METHODS

Subinventory

MSC_ST_SUB_INVENTORIES

Resource Requirements

MSC_ST_RESOURCE_REQUIREMENTS

Supplier Capacity

MSC_ST_ITEM_SUPPLIERS

MSC_ST_SUPPLIER_CAPACITIES

MSC_ST_SUPPLIER_FLEX_FENCES

Trading Partners

MSC_ST_LOCATION_ASSOCIATIONS

MSC_ST_TRADING_PARTNER_SITES

MSC_ST_TRADING_PARTNERS

Unit Numbers

MSC_ST_UNIT_NUMBERS

Parameters

MSC_ST_PARAMETERS

Projects

MSC_ST_PROJECTS

MSC_ST_PROJECT_TASKS

ATP Rules

MSC_ST_ATP_RULES

Business Intelligence System

MSC_ST_BIS_PFMC_MEASURES

MSC_ST_BIS_TARGET_LEVELS

MSC_ST_BIS_TARGETS

MSC_ST_BIS_BUSINESS_PLANS

MSC_ST_BIS_PERIODS

Planners

MSC_ST_PLANNERS

Unit Of Measures

MSC_ST_UNITS_OF_MEASURE

MSC_ST_UOM_CLASS_CONVERSIONS

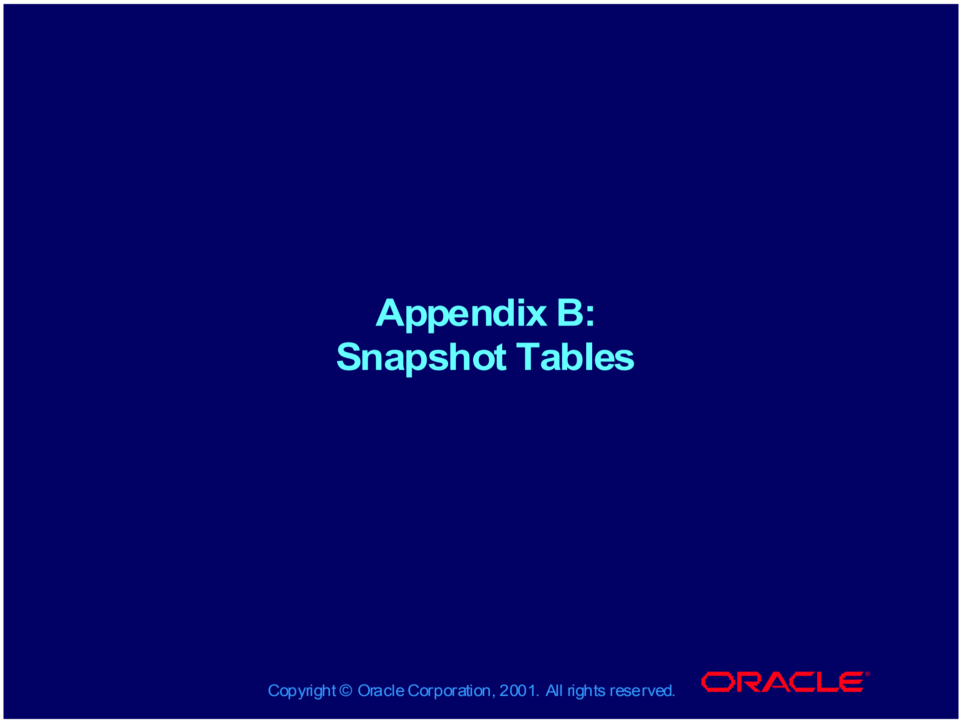
MSC_ST_UOM_CONVERSIONS

Partner Contacts

MSC_ST_PARTNER_CONTACTS

Appendix B: Snapshot Tables

Chapter 13



Snapshot Tables

The following is a list of database snapshot tables used to capture data from an Oracle Application source instance.

Inventory

MTL_SUPPLY_SN

MTL_U_SUPPLY_SN

MTL_U_DEMAND_SN

MTL_SYS_ITEMS_SN

MTL_OH_QTYS_SN

MTL_MTRX_TMP_SN

MTL_DEMAND_SN

Bills of Material

BOM_BOMS_SN

BOM_INV_COMPS_SN

BOM_OPR_RTNS_SN

BOM_OPR_SEQS_SN

BOM_OPR_RESS_SN

BOM_RES_CHNGS_SN

Material Requirements Planning

MRP_SCHD_DATES_SN

Order Management

OE_ODR_LINES_SN

Work in Process

WIP_DSCR_JOBS_SN

WIP_WREQ_OPRS_SN

WIP_FLOW_SCHDS_SN

WIP_WOPRS_SN

WIP_REPT_ITEMS_SN

WIP_REPT_SCHDS_SN

WIP_WLINES_SN

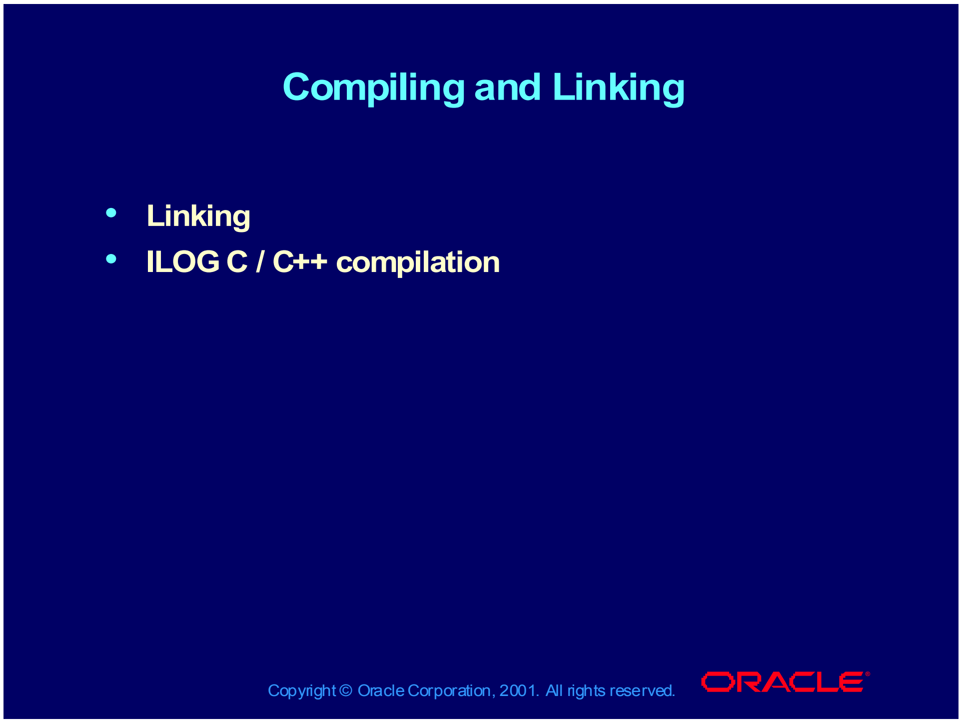
Purchasing

PO_SI_CAPA_SN

Appendix C: Compiling and Linking

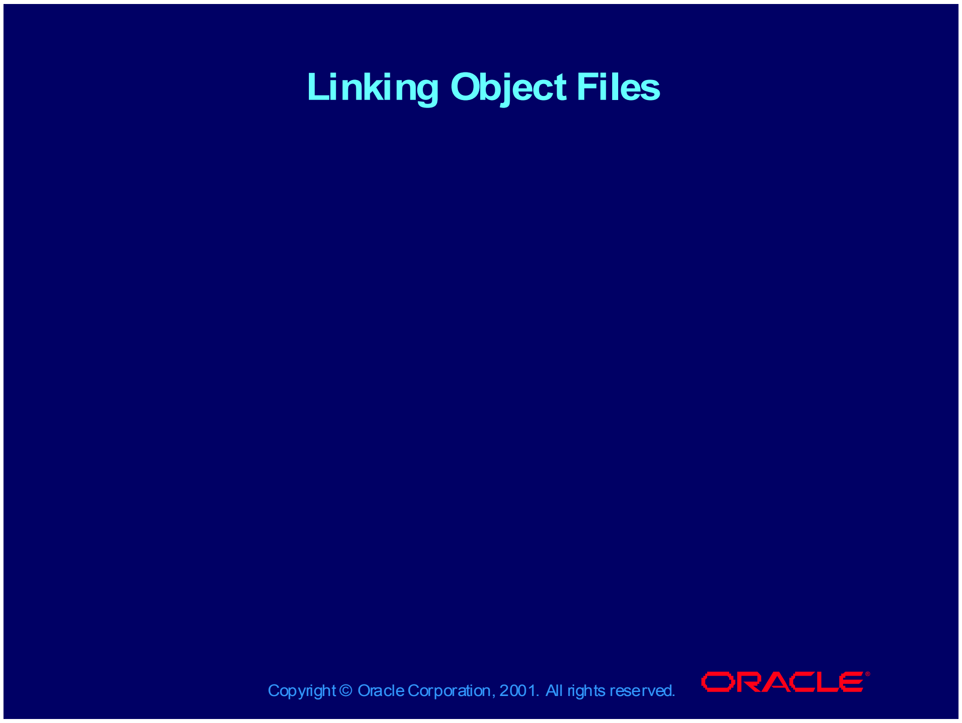
Chapter 14


Appendix C:
Compiling and Linking
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE



Compiling and Linking

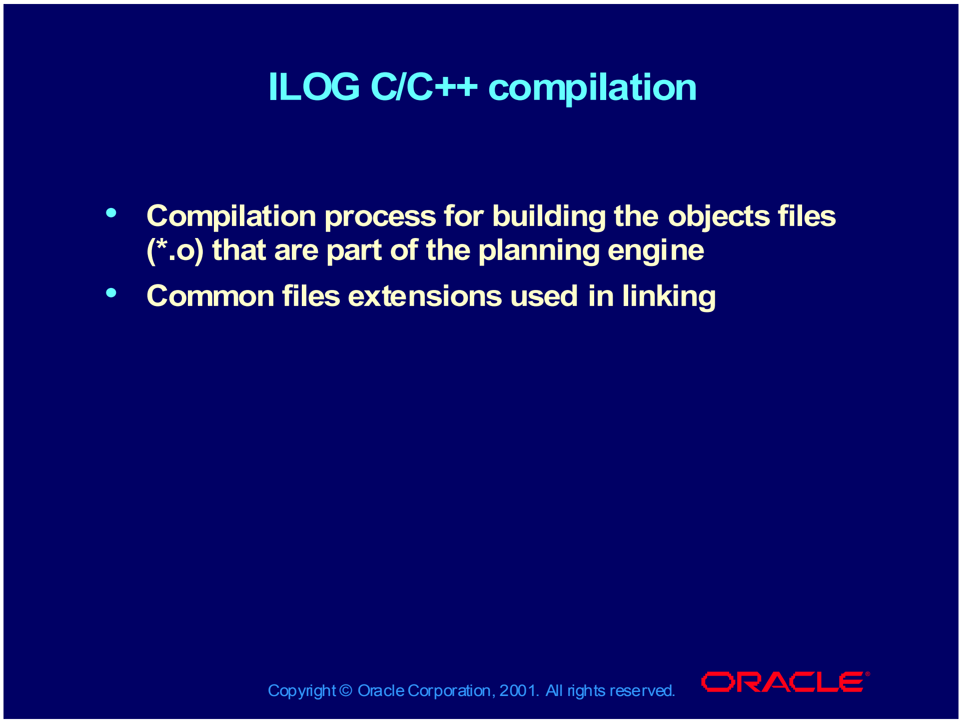
- All the C source files that are reused from MBP are compiled using standard *genmake* (C-compiler) to generate the C object files. All the C++ source files are compiled with the C++ compiler to generate the C++ objects. Lastly all the C and C++ object files are linked using a C++ compiler.
- Compiler directives are used around the calls to the C++ modules to permit conditional compilation. The conditions that govern inclusion of the C++ code are customer licensing and availability of the ILOG products. The Make file is also modified to include conditional linking based on the product licensing.
- Scalability and performance is achieved by ensuring all three modules share the in-memory data structures that have been optimized to hold all the planning data.
- **Problem:**
- The executable file /opt/APS/applmgr/11i/mso/11.5.0/bin/MSONEW for this concurrent program has a problem and cannot be executed.
- **Solution:**
- To recompile this executable they ran the following command: adrelink.sh force=y ranlib=y "mso MSONEW"



Linking Object Files

- All the object files (those compiled with C and C++ compiler) will be linked on site using a C++ linker.
- The optimization option for MSC will be provided only on those ports that ILOG supports and also based on the customer licensing option.
- ILOG objects would always be linked in to make the executable MRSMSC.
- $SH_TOP contains all third party executables.

ILOG C/C++ compilation

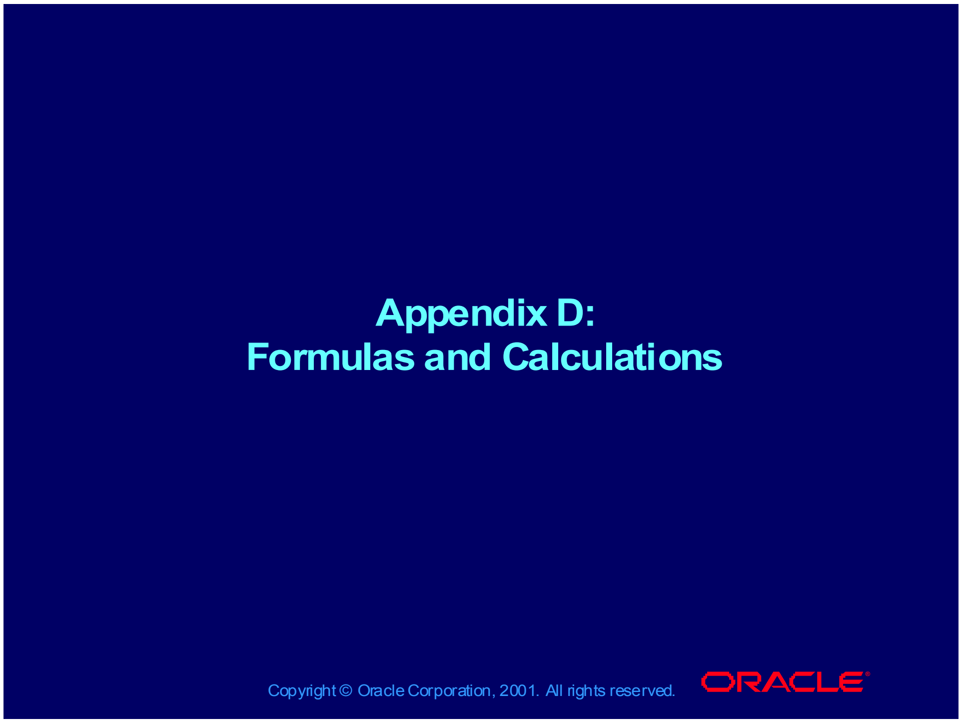


ILOG C / C++ compilation

- The source files for MSO have the form msc*.ppc, msc*.opp, msc*.cpp.
- The files with extensions *.ppc and *.opp are standard Oracle "c" pre-compiler files which are compiled using the *genmake* shell script to create the appropriate object files. These files have code to manage database connections and also include standard headers and other product specific headers.
- The files with extension *.cpp are "c++" source files that includes all the ILOG product headers and also headers that are shared with the *.ppc files. These shared headers contain type definitions (typedefs) for the data structures shared between C and C++ code. The c++ source files are compiled with a standard C++ compiler to generate a set of object files *.o.
- The file mscmain.cpp contains the *main()* function that makes calls to the c code in *.ppc and c++ code in *.cpp. The calls to functions in *.cpp are enclosed within compiler directives to permit conditional compilation based on existence of ILOG support and license. This file is compiled with a C++ compiler to build mscmain.o.
- The linking phase refers to mscmain.o and other MSC c++ objects only in the case when the ILOG license and support exists, otherwise only the c objects are linked to build the executable. The linking changes are explained in the following section.

Appendix D: Formulas and Calculations

Chapter 15



Objectives Formulas

Inventory Turns Objective

The Inventory Turns objective is calculated as follows:

Maximize inventory turns = Minimize [(*inventory carrying cost*)]

Inventory Carrying Cost =

{(average inventory per bucket)*(carrying cost pct)*(item cost)}

Inventory turns are maximized by minimizing inventory carrying cost. Inventory carrying cost is summed up for all items in all time buckets. Set the carrying cost percent at two different levels: use the item attribute “Planning: Carrying Cost Percent” or use the system profile option “MSO: Inventory Carrying Costs Percent.” SCO will use the item attribute value if NOT NULL, otherwise it will use the system profile option value.

Inventory Turns are calculated based upon the summary table MRP_BIS_INV_DETAIL. For each item, rows are entered. Each row has columns for both inventory quantity and cost. These include:

Beginning inventory (nettable inventory + nonnettable inventory _ issued WIP);

MDS Demand;

PO Supply;

WIP Supply.

To calculate the cost, the quantity is multiplied by item cost except in the case of WIP Supply where the quantity is multiplied by the resource cost. Repetitive items are denormalized to show the quantity each day of the schedule.

Warning: Using the Inventory Turns Objective will have *no effect* if the users fails to set the Carrying Cost Percent at either the item attribute level or the system profile level. A NULL value for Carrying Cost Percent is interpreted as 0 and that means your inventory carrying costs are already minimized!

Plan Profit Objective

The Plan Profit objective is calculated as follows:

Maximize Plan Profit => Maximize [(*plan revenue*) - (*plan cost*)]

plan revenue =

{(sales order line price)*(sales order quantity)} + {(item list price)*(item discount)*(forecast quantity)}

Plan revenue is calculated and summed up for all items with independent demand in all time buckets. For forecasted items, the Item List Price is from the Price List specified by the System Profile Option "MSO: Plan Revenue List Price." For forecast late demand penalty cost calculation, the value of the item discount is obtained from the column AVERAGE_DISCOUNT in MSC_SYSTEM_ITEMS. This value is obtained from a profile option and is also used in calculating plan KPI's.

plan cost = (*item cost*) + (*transportation cost*) + (*inventory carrying cost*)

Plan cost is calculated and summed up for all items, resources, and ship methods in all time buckets.

item cost = {(resource cost)*(resource quantity used)} + {(buy item cost)*(buy item quantity)} +{(process cost)*quantity using process)}

transportation cost = {(transfer quantity)*(item weight)*(shipping cost per unit weight)} + {(buy quantity)*(item weight)*(shipping cost per unit weight)}

inventory carrying cost = {(average inventory per bucket)*(carrying cost pct)*(item cost)}

Note: The item cost for buy items will be the current standard cost in the receiving organization. The item cost for make items is rolled up by SCO depending on whether the primary BOM and Routing or alternate BOM and Routing is selected.

Plan Profit is the most aggregate of objectives in the sense that it combines multiple costs.

On-time Delivery Objective

The On-time Delivery objective is calculated as follows:

Maximize On-time Delivery => Minimize [(*penalty cost for late demand*)]

Penalty cost for late demands is calculated and summed up for all items with independent demand in all time buckets.

penalty cost for late demand = {(penalty cost percentage for late demand)*(days late)* (forecast quantity of late demand)* (item list price per unit)*(item discount)}

+

{(penalty cost percentage for late demand)*(days late)*(sales order line quantity of late demand)*(sales order line price per unit)}
+
{(penalty cost percentage for unmet demand)*(days late)* (forecast quantity of unmet demand)* (item list price per unit)*(item discount)}
+
{(penalty cost percentage for unmet demand)*(days late)* (sales order line quantity of unmet demand)*(sales order line price per unit)}

Penalty cost for late demands sums two types of costs: late demand cost and unmet demand cost. An unmet demand is simply a very late demand. Specifically, it is a demand for which the plan generates supply that exceeds the demand due date by more than allowable days early/late. Allowable days early/late is a user-set profile option.

Penalty cost percentage for late demand is a user-specified plan option.

Penalty cost percentage for unmet demand is a system-supplied plan option, obtained by multiplying the penalty cost percentage for late demand by a constant that is greater than 1. This makes unmet (very late) demands cost more than late demands.

Implicit Objectives

In addition to the objectives defined above, which can be selected/weighted or deselected by the user, Oracle ASCP maintains a set of implicit (hidden) objectives that it takes into consideration no matter what you select. These objectives are defined to be the negative of various penalty costs. this is the same thing as saying the objective is to minimize the sum of the various penalty costs, as follows:

Maximize Implicit Objective => Minimize {(penalty cost for late demand)
+(penalty cost for unmet demand)
+(penalty cost for supplier capacity violation)
+(penalty cost for resource capacity violation)
+(penalty cost for transport apacity violation)
+(penalty cost for safety stock violation)
+(penalty cost for any unused supply)
+(penalty cost for using alternate sources)
+(penalty cost for using alternate resources)
+(penalty cost for using substitute items)
+(penalty cost for using alternate BOM's)
+(penalty cost for using alternate routings)}

Combining Objectives

Oracle ASCP combines the above objectives into the following objective function:

overall objective => maximize

{w1*(plan profit) + w2*(on-time delivery) + w3*(inventory turns) + 1.0*(implicit objective)}

The user enters the Objective weights w1–w3 on the Optimization Tab of Plan Options and the values are restricted to the range 0 to 1. Setting an objective's weight to 0 directs Oracle ASCP not to consider that particular objective. Setting an objective's weight to 1 places the maximum possible emphasis on that objective.

Objective weights w1–w3 may be set independently. Beware interdependent objectives. Some costs are contained in more than one objective. For example, inventory carrying cost is a part of both the Plan Profit and Inventory Turns objectives. Therefore, only use these two objectives together if it is desired to artificially weight inventory carrying cost higher than the other costs (item cost, transportation cost) contained within Plan Profit.

A more subtle case of interdependent objectives is penalty cost for late demand, which appears both in the On-time Delivery objective and in the implicit objectives not seen by the user. Thus, no matter what the weight on on-time delivery, Oracle ASCP considers late demand cost in its planning decision-making.

Objective weights w1–w3 in general do not precisely show the relative importance of each objective in planning decisions. As can be seen from the above definition of the overall objective, the percentage of the overall objective value occupied by a particular objective depends also on the dollar magnitude of the objective, and it is the product of the weight and the dollar magnitude of the objective which reflects the relative importance of each objective in planning decisions.

Key indicators are calculated in MRPBISUS.pls and MRPBISUB.pls.

These are PL/SQL packages. To find the version of these packages, follow the procedures as detailed below.

Gross Margin information is calculated in a materialized view that is based on MRP_BIS_INV_DEATIL. The view uses the schedule quantities and costs to calculate and display the plan revenue, cost and margin.

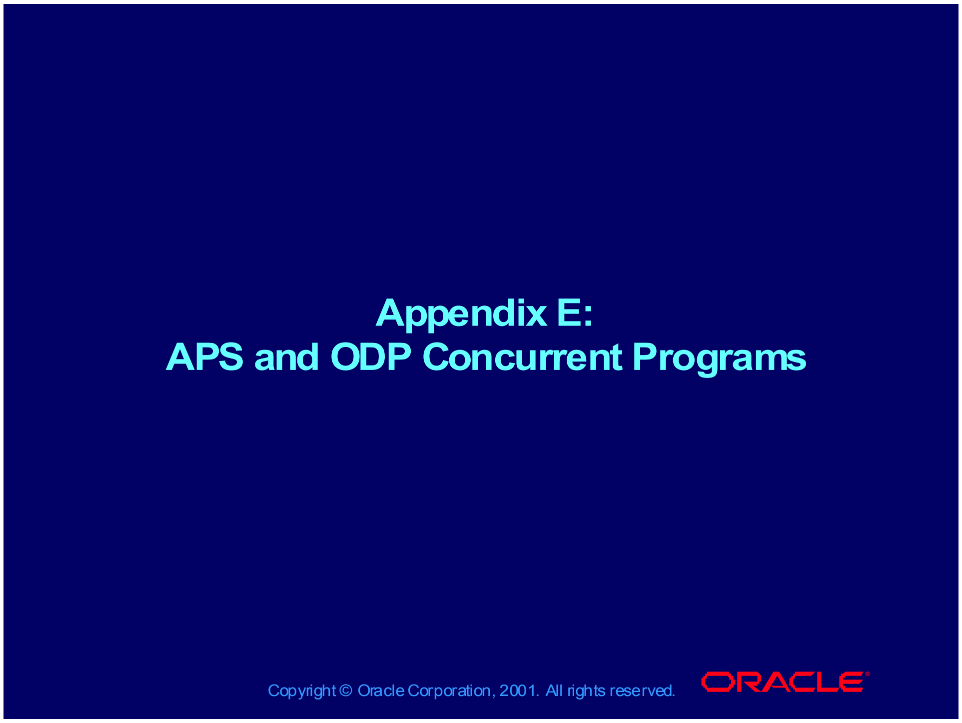
Planned Utilization is calculated based upon the view MRP_BIS_RES_SUMMARY. This view contains a row for each statement of availability or requirement.

On-time Delivery is calculated for all sales orders and forecasts based on their due dates. It is based on the table MRP_EXCEPTION_DETAILS.

Appendix E: APS and ODP Concurrent Programs

Chapter 16

Appendix E: APS and ODP Concurrent Programs



APS Concurrent Programs as of 11.5.1

Descriptive Program Name	Program Short Name	Execution method
Auto-Release Planned Orders	MSCAPO	PL/SQL
Calculate End Assemblies	MSCEAP	SPAWNED
Copy Distribution Plan	MSCCPP3	SPAWNED
Copy Manufacturing Plan	MSCCPP2	SPAWNED
Copy Production Plan	MSCCPP1	SPAWNED
Delete Planning Data Collection	MSCDPDC	PL/SQL
Delete Planning Data Pull	MSCDPDP	PL/SQL
Launch Distribution Planning Process	MSCSLPPR3	PL/SQL
Launch Manufacturing Planning Process	MSCSLPPR2	PL/SQL
Launch Production Planning Process	MSCSLPPR1	PL/SQL
Loader Worker	MSCSLD	PAWNED
Snapshot Monitor	MSCNEW	SPAWNED
Memory-Based Snapshot	MSCNSP	SPAWNED
Memory-Based Snapshot Worker	MSCNSW	SPAWNED
Net Change Simulation Planner	MSCNCP	SPAWNED
Online Planner	MSCOLP	SPAWNED
Online Planner Delete Worker	MSCODW	SPAWNED
Planner Delete Worker	MSCPDW	SPAWNED
Planning Data Pull	MSCPDP	PL/SQL
Planning Data Pull Worker	MSCPDPW	PL/SQL

Planning Operational Data Store Load	MSCPDC	PL/SQL
Purge Designator	MSCPRG	SPAWNED
Snapshot Delete Worker	MSCSDW	SPAWNED
Snapshot Monitor	MSCMON	SPAWNED
Delete collection program	MSCPDEL	PL/SQL
Msc planner delete worker	MSCPDW	Spawned
Msc snapshot delete worker	MSCSDW	Spawned
Msc snapshot delete worker	MSCSDW	Spawned
Create Existing Plan Partitions	MSCCFPAE	PL/SQL
Create Plan Partition	MSCCFPAR	PL/SQL
Drop Existing Plan Partitions	MSCDFPAE	PL/SQL
Drop Plan Partition	MSCDFPAR	PL/SQL
Collection program	MSCPMON	PL/SQL
Collection program worker	MSCPWOR	PL/SQL
Create plan partitions	MSCCRPAR1	PL/SQL
Launch planner	MSCLPLAN	PL/SQL
Memory Based Planner	MSONEW	SPAWNED by msonew
Net Change Simulation Planner	MSONCP	SPAWNED by msonew
Online Planner	MSOOLP	SPAWNED by msonew
Create Component Architecture Flexfields	MSCCONF	PL/SQL
Planning Data Collection - Purge Staging Table	MSCPDCP	PL/SQL
Planning Operational Data Store Load Worker	MSCPDCW	PL/SQL

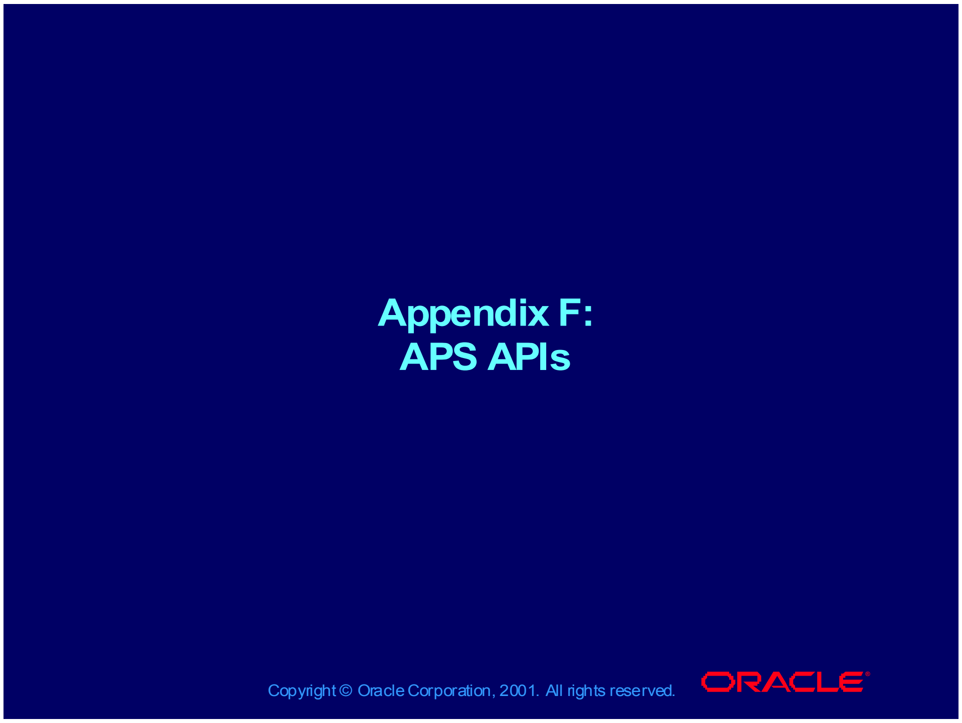
ODP Concurrent Programs

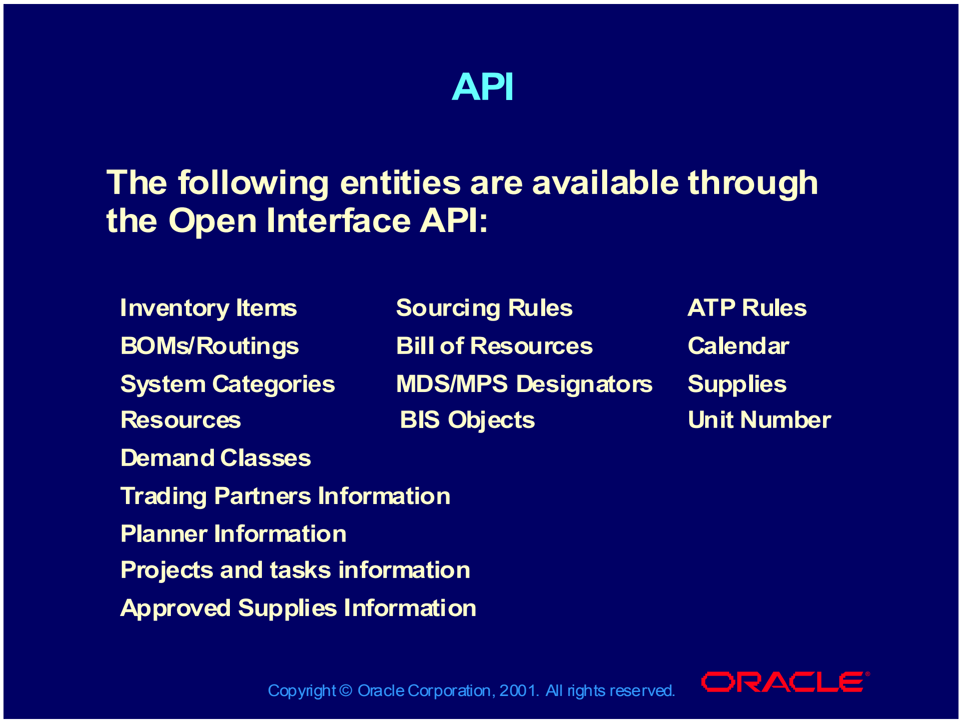
Descriptive Program Name	**Program Short Name**	**Execution method**
Collection program for booking history	MSDCPBO	PL/SQL
Collection program for currency conversion	MSDCPCC	PL/SQL
Collection program for sales forecast	MSDCPSF	PL/SQL
Collection program for sales opportunities	MSDCPOP	PL/SQL
Collection program for shipment history	MSDCPSH	PL/SQL
Collection program for uom conversion	MSDCPUC	PL/SQL
Collect booking data	MSDCBD	PL/SQL
Collect currency conversion data	MSDCCC	PL/SQL
Collect all fact data	MSDCFD	PL/SQL
Collect level data	MSDCLV	PL/SQL
Collect manufacturing forecast	MSDCMF	PL/SQL
Collect shipment data	MSDCSD	PL/SQL
Collect sales forecast	MSDCSF	PL/SQL
Collect sales opportunity data	MSDCSO	PL/SQL

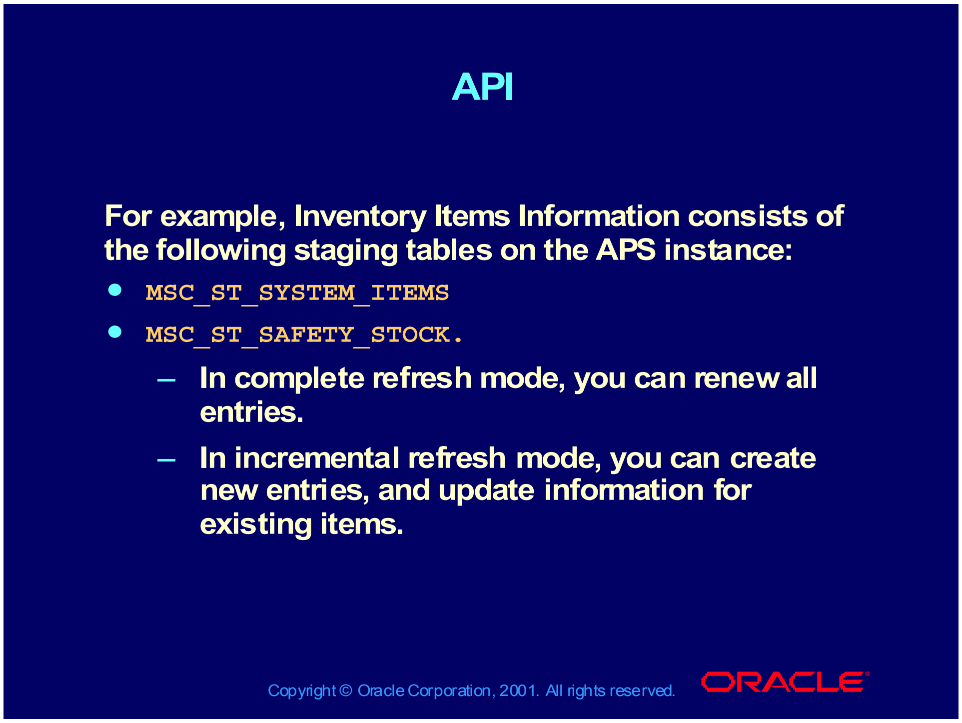
Collect time data	MSDCTD	PL/SQL
Collect uom conversion data	MSDCUC	PL/SQL
Pull booking data	MSDPBD	PL/SQL
Pull currency conversion data	MSDPCC	PL/SQL
Pull all fact data	MSDPFD	PL/SQL
Pull level data	MSDPLV	PL/SQL
Pull manufacturing forecasts	MSDPMF	PL/SQL
Pull shipment data	MSDPSD	PL/SQL
Pull sales forecasts	MSDPSF	PL/SQL
Pull sales opportunity	MSDPSO	PL/SQL
Pull time data	MSDPTD	PL/SQL
Pull UOM conversions	MSDPUC	PL/SQL
Translation program for booking history	MSDTSBO	PL/SQL
Translation program for currency conversion	MSDTSCC	PL/SQL
Translation program for sales forecast	MSDTSSF	PL/SQL
Translation program for sales opportunities	MSDTSOP	PL/SQL
Translation program for shipment history	MSDTSSH	PL/SQL
Translation program for uom conversion	MSDTSUC	PL/SQL
Translation program for manufacturing forecast	MSDTSMF	PL/SQL
Collection program for manufacturing forecast	MSDCPMF	PL/SQL

Appendix F: APS APIs

Chapter 17


Appendix F:
APS APIs
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE


API
The following entities are available through the Open Interface API:
Inventory Items
BOMs/Routings
System Categories
Resources
Demand Classes
Trading Partners Information
Planner Information
Projects and tasks information
Approved Supplies Information
Sourcing Rules
Bill of Resources
MDS/MPS Designators
BIS Objects
ATP Rules
Calendar
Supplies
Unit Number
Copyright © Oracle Corporation, 2001. All rights reserved.
ORACLE



API

APS open interface consists of loading the MSC_ST_* schema and running the data pull program to make the data available to ASCP.

Appendix G: Profile Options and Descriptive Flexfield Information

Chapter 18

Appendix G: Profile Options and Descriptive Flexfield Information

ORACLE

Profile Options

Profile options and associated flexfields

ORACLE

Profile Options

MSC: Aggregate Resource Name Flexfield Attribute

MSC: Alternate Resource for an Operation Flexfield Attribute

MSC: Cost of Using a BOM/ Routing Flexfield Attribute

MSC: Default Workbench Height

MSC: Default Workbench Width

MSC: Map Server Host

MSC: Map Server Port

MSC: Planning Currency US dollar

MSC: Priority for Substitute Items Flexfield Attribute

MSC: Priority of Alternate Resources for an Operation Flexfield Attribute

MSC: Resource Group for a Line Flexfield Attribute

MSC: Resource Type for an Operation Flexfield Attribute

MSC: Simultaneous Resource Sequence Flexfield Attribute

MSC: ATP Assignment Set

MSC: Sourcing Rule Category Set

MSD_CONVERSION_TYPE

MSD_CURRENCY_CODE

MSD_ONE_STEP_COLLECTION

MSO_GLOBAL_DMD_PENALTY

MSO_GLOBAL_ITEM_PENALTY
MSO_GLOBAL_RES_PENALTY
MSO_CARRYING_COST_PCT
MSO_MAX_DAYS_LATE
MSO_DEMANDS_PER_GROUP
MSO_MAX_BACKTRACKS
MSO_FLOATING_POINT_PRECISION
MSO_MAX_EXCESS_RESOURCE_UNITS
MSO_MAX_LEAD_TIME_FACTOR
MSO_LP_ALGORITHM
INV: Capable to Promise

Descriptive flexfields and associated Profile Options

The following descriptive flexfield will be defined via the Forecast Items form (for forecasts) or in the Scheduling region of the Sales Orders form (for sales orders). This will be stored in the table SO_LINES_ALL (CORRECT?) and MRP_FORECAST_DATES.

Penalty Cost Factor for Late Demands (at the demand level)

Profile Option: MSO_FCST_PENALTY

The following descriptive flexfields will be defined via the Master\Organization Items form. This will be stored in the table MTL_SYSTEM_ITEMS.

Penalty Cost Factor for Late Demands (at the item level)

Profile Option: MSO_ITEM_DMD_PENALTY

Penalty Cost Factor for Exceeding Material Capacity (at the item level)

Profile Option: MSO_ITEM_CAP_PENALTY

The following descriptive flexfields will be defined via the Organizations Parameters form. This will be stored in the table MTL_PARAMETERS.

Penalty Cost Factor for Late Demands (at the org level)

Profile Option: MSO_ORG_DMD_PENALTY

Penalty Cost Factor for Exceeding Material Capacity (at the org level)

Profile Option: MSO_ORG_ITEM_PENALTY

Penalty Cost Factor for Exceeding Resource Capacity (at the org level)

Profile Option: MSO_ORG_RES_PENALTY

Penalty Cost Factor for Exceeding Transportation Capacity (at the org level)

Profile Option: MSO_ORG_TRSP_PENALTY

The following descriptive flexfield will be defined via the Supplier-Item Attributes form (in the header region). It will be stored in PO_ASL_ATTRIBUTES.

Penalty Cost Factor for Exceeding Material Capacity (at the item/vendor level)

Profile Option: MSO_SUP_CAP_PENALTY

The following descriptive flexfield will be defined via the Sales Order form (in the Scheduling region of the Sales Orders form). It will be stored in SO_LINES_ALL.

Penalty Cost Factor for Late Sales Orders

Profile Option: MSO_SO_PENALTY

The following descriptive flexfield will be defined via the Department Resources form. This will be stored in the table BOM_DEPARTMENT_RESOURCES.

Penalty Cost Factor for Exceeding Resource Capacity (at the resource level)

Profile Option: MSO_RES_PENALTY

Aggregate Resource for a Resource (based on the existing flexfield "Aggregate Resource Id")

Profile Option: MSC_AGGREG_RES_NAME

The following descriptive flexfield will be defined via the Department Resources form. This will be stored in the table BOM_OPERATIONS_RESOURCES.

Priority of Alternate Resources for an Operation

Profile Option: MSC_ALT_RES_PRIORITY

Simultaneous Resource Sequence (similar to the existing flexfield "Factory Planner Resource Type.")

Profile Option: MSC_SIMUL_RES_SEQ

Alternate Resources for an Operation (similar to the existing flexfield "Factory Planner Resource Type.")

Profile Option: MSC_ALT_OP_RES

The following descriptive flexfield will be defined via the Inter-location Transit Times form. It will be stored in MTL_INTERORG_SHIP_METHODS.

Penalty Cost Factor for Exceeding Transportation Capacity (at the ship method level)

Profile Option: MSO_TRSP_PENALTY

The following descriptive flexfield will be defined via the Substitute Components form. This will be stored in the table BOM_SUBSTITUTE_COMPONENTS.

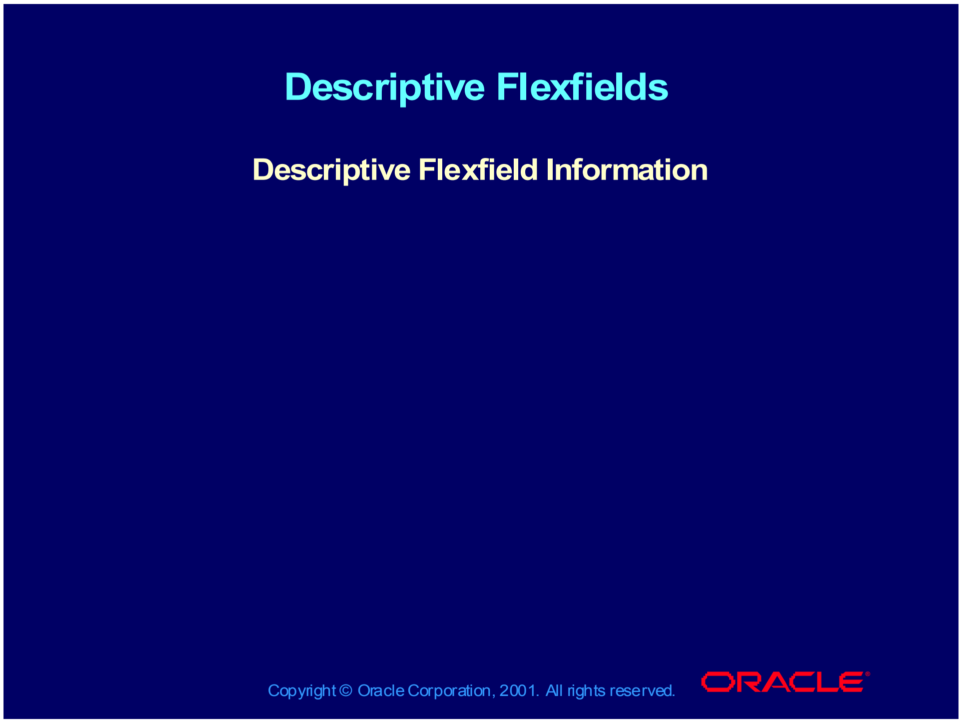
Priority for Substitute Items

Profile Option: MSC_BOM_SUBST_PENALTY

The following descriptive flexfield will be defined via the Bills of Material form. It will be stored in the table BOM_BILLS_OF_MATERIALS.

Cost of Using Alternate BOM/Routing

Profile Option: MSC_ALT_BOM_COST



Descriptive Flexfield Information

Oracle provides a patch (releases (10.7, 11) to seed new descriptive flexfields.

In Release 11*i*, customers do not need to apply a patch, they simply run a concurrent program that will contain profiles, value sets, and a concurrent program. Apply the patch and then run the “Create Planning Flexfields” (MRP/Reports) concurrent program to define which attributes will be used for which flexfield. This is similar to how I2 works now. The concurrent program will setup and compile the descriptive flexfields and then set the profiles to reflect which attribute is being used (used by our code to find their data).

APS Descriptive Flexfield structures are updateable, but if a user wants to remove a flexfield column, they will have to keep the profile in sync with the flexfield definition. For example, if they change from ATTRIBUTE1 to ATTRIBUTE2 they need to also change the corresponding profile from 1 to 2.

Table	**Attributes required**	**Form**
mtl_system_items	2	Define items
mtl_parameters	4	Define Org
bom_department_resources	2	Dept Resources
po_asl_attributes	1	Supplier Item attributes
bom_substitute_components	1	Substitute components
mtl_interorg_ship_methods	1	Inter-location transit times
bom_bills_of_materials	1	Define Bills of Material

mrp_forecast_dates	1	Forecast Details
so_lines_all	1	Sales Order lines
bom_operation_resources	3	Operation Resources

Profile options and how they work with descriptive flexfield information:

You have a descriptive flexfield on the Items window for Late Demands Penalty. The value you enter from the window will be inserted into ATTRIBUTE4 (this is defined by users with a concurrent program that is a one-time setup step). This descriptive flexfield has a corresponding profile option whose value is 4. The user does not need to be concerned with this profile -- it so the collections program knows which field to look in. In addition, these is another profile option for Late Demands Penalty that lets you define a global value for Late Demands Penalty.

Example:

MSO_GLOBAL_DMD_PENALTY

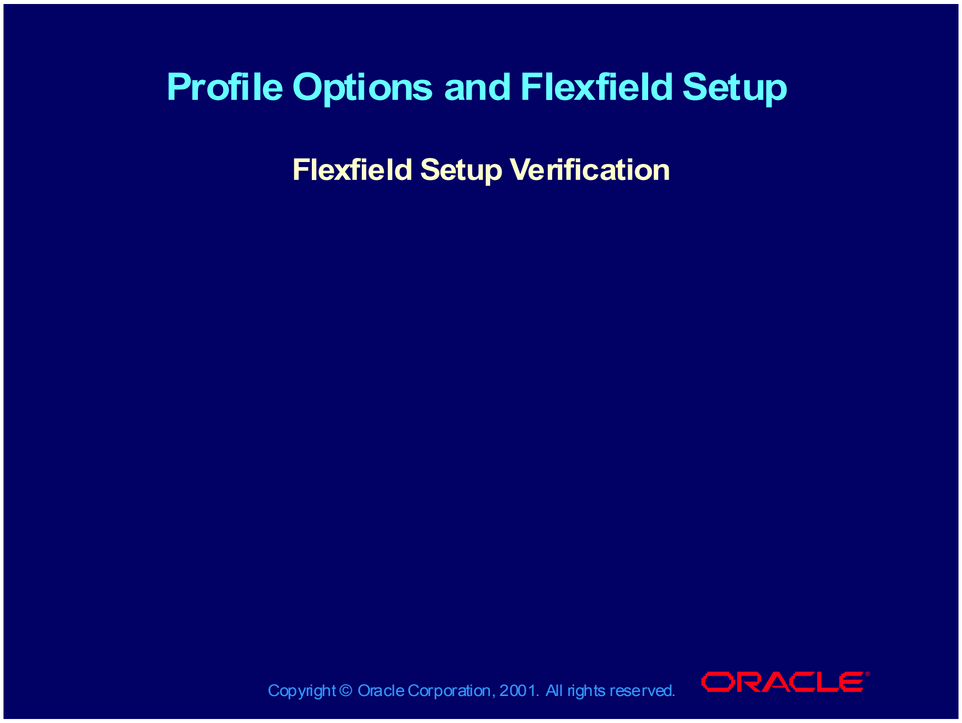
MSO_ITEM_DMD_PENALTY

MSO_ORG_DMD_PENALTY

The 3 profile options mentioned store different types of values. For instance, MSO_GLOBAL_DMD_PENALTY stores the penalty factor for late demands at the profile option level

MSO_ITEM_DMD_PENALTY stores the column name in the table MTL_SYSTEM_ITEMS that contains the item level demand penalty factor. For example, if you enter the value “4” for this profile option, the item level demand penalty factor will be stored in the column ATTRIBUTE4 in MTL_SYSTEM_ITEMS.

MSO_ORG_DMD_PENALTY stores the column name in the table MTL_PARAMETERS that contains the org level demand penalty factor.



Profile Options and Flexfield Setup Exercise

Run the query in SQL*Plus.

```
Set pages 67
Set lines 80
Spool flex_setup.rep
col descriptive_flexfield_name noprint
col Name format a64 trunc
col Val format a15
col seq format 999
col xx format a3
Prompt “     * indicates APS flex field”
Break on descriptive_flexfield_name skip 1
select COLUMN_SEQ_NUM seq,
       END_USER_COLUMN_NAME descr,
       APPLICATION_COLUMN_NAME,
       descriptive_flexfield_name,
       decode (END_USER_COLUMN_NAME,
       'Late Demands Penalty', '*',
       'Material Over-Capacity Penalty', '*',
       'Late Demands Penalty', '*',
```

```
        'Material Over-Capacity Penalty', '*',
        'Resource Over-Capacity Penalty', '*',
        'Transport Over-Cap Penalty', '*',
        'Aggregate Resource','*',
        'Resource Over-Capacity Penalty', '*',
        'Material Over-Capacity Penalty', '*',
        'Substitute Priority', '*',
        'Transport Over-Cap Penalty', '*',
        'Cost of Using a BOM/Routing', '*',
        'Late Forecasts Penalty', '*',
        'Late Order Penalty', '*',
        'Principal Resource', '*',
        'Resource Priority', '*',
        'Resource Step Number', '*',
        'Resource Group','*',
        'Demand Priority','*',' ') xx
  from fnd_descr_flex_column_usages
       /* Flex filed tables */
  where descriptive_flexfield_name
in('MTL_SYSTEM_ITEMS' ,

'MTL_PARAMETERS','BOM_DEPARTMENT_RESOURCES','PO_ASL_AT
TRIBUTES',

'BOM_SUBSTITUTE_COMPONENTS','MTL_INTERORG_SHIP_METHODS
','BOM_BILL_OF_MATERIALS',

'MRP_FORECAST_DATES','SO_LINES','OPERATION_RESOURCES',
'WIP_LINES',
        'MRP_SCHEDULE_DATES')
  and   DESCRIPTIVE_FLEX_CONTEXT_CODE = 'Global Data
Elements'
  order by
  decode
(descriptive_flexfield_name,'MTL_SYSTEM_ITEMS' , 1,

'MTL_PARAMETERS',2,'BOM_DEPARTMENT_RESOURCES',3,'PO_AS
L_ATTRIBUTES',4,
'BOM_SUBSTITUTE_COMPONENTS',5,'MTL_INTERORG_SHIP_METHO
DS',6,'BOM_BILL_OF_MATERIALS',7,
```

```
'MRP_FORECAST_DATES',8,'SO_LINES',9,'OPERATION_RESOURC
ES',10,'WIP_LINES',11,
  'MRP_SCHEDULE_DATES',12) , COLUMN_SEQ_NUM;
  select t.USER_PROFILE_OPTION_NAME Name,
         profile_option_value Val
  from   fnd_profile_options_tl t,
         fnd_profile_option_values v ,
         fnd_profile_options o
  where  v.profile_option_id = o.profile_option_id
  and    o.profile_option_name in (
  /* 10.7 and 11.0 */
  'MRP_ITEM_DMD_PENALTY', 'MRP_ITEM_CAP_PENALTY',
  'MRP_ORG_DMD_PENALTY', 'MRP_ORG_ITEM_PENALTY',
  'MRP_ORG_RES_PENALTY', 'MRP_ORG_TRSP_PENALTY',
  'MRP_AGGREG_RES_NAME', 'MRP_RES_PENALTY',
  'MRP_SUP_CAP_PENALTY', 'MRP_BOM_SUBST_PRIORITY',
  'MRP_TRSP_PENALTY', 'MRP_ALT_BOM_COST',
  'MRP_FCST_PENALTY', 'MRP_SO_PENALTY',
  'MRP_RESOURCE_TYPE', 'MRP_ALT_RES_PRIORITY',
  'MRP_SIMUL_RES_SEQ', 'MRP_LINE_RES_GROUP',
  'MRP_DMD_PRIORITY_FLEX_NUM',
  /* 11i */
  'MSO_ITEM_DMD_PENALTY', 'MSO_ITEM_CAP_PENALTY',
  'MSO_ORG_DMD_PENALTY', 'MSO_ORG_ITEM_PENALTY',
  'MSO_ORG_RES_PENALTY', 'MSO_ORG_TRSP_PENALTY',
  'MSC_AGGREG_RES_NAME', 'MSO_RES_PENALTY',
  'MSO_SUP_CAP_PENALTY', 'MSC_BOM_SUBST_PRIORITY',
  'MSO_TRSP_PENALTY', 'MSC_ALT_BOM_COST',
  'MSO_FCST_PENALTY', 'MSO_SO_PENALTY',
  'MSC_RESOURCE_TYPE', 'MSC_ALT_RES_PRIORITY',
  'MSC_SIMUL_RES_SEQ', 'MSC_LINE_RES_GROUP',
  'MRP_DMD_PRIORITY_FLEX_NUM')
  and  t.PROFILE_OPTION_NAME = o.profile_option_name
  order by decode(o.profile_option_name,
  'MRP_ITEM_DMD_PENALTY',1, 'MRP_ITEM_CAP_PENALTY',2,
  'MRP_ORG_DMD_PENALTY',3, 'MRP_ORG_ITEM_PENALTY',4,
  'MRP_ORG_RES_PENALTY',5, 'MRP_ORG_TRSP_PENALTY',6,
'MRP_AGGREG_RES_NAME',7, 'MRP_RES_PENALTY',8,
```

```
  'MRP_SUP_CAP_PENALTY',9,
'MRP_BOM_SUBST_PRIORITY',10,
  'MRP_TRSP_PENALTY',11, 'MRP_ALT_BOM_COST',12,
  'MRP_FCST_PENALTY',13, 'MRP_SO_PENALTY',14,
  'MRP_RESOURCE_TYPE',15, 'MRP_ALT_RES_PRIORITY',16,
  'MRP_SIMUL_RES_SEQ',17, 'MRP_LINE_RES_GROUP',18,
  'MRP_DMD_PRIORITY_FLEX_NUM',19,
  /* 11i */
  'MSO_ITEM_DMD_PENALTY',1, 'MSO_ITEM_CAP_PENALTY',2,
  'MSO_ORG_DMD_PENALTY',3, 'MSO_ORG_ITEM_PENALTY',4,
  'MSO_ORG_RES_PENALTY',5, 'MSO_ORG_TRSP_PENALTY',6,
  'MSC_AGGREG_RES_NAME',7, 'MSO_RES_PENALTY',8,
  'MSO_SUP_CAP_PENALTY',9,
'MSC_BOM_SUBST_PRIORITY',10,
  'MSO_TRSP_PENALTY',11, 'MSC_ALT_BOM_COST',12,
  'MSO_FCST_PENALTY',13, 'MSO_SO_PENALTY',14,
  'MSC_RESOURCE_TYPE',15, 'MSC_ALT_RES_PRIORITY',16,
  'MSC_SIMUL_RES_SEQ',17, 'MSC_LINE_RES_GROUP', 18,
  'MRP_DMD_PRIORITY_FLEX_NUM',19,1000);

spool off
exit  ;
```