CE12

Chapter 9:
Advanced SQL and PL/SQL Topics

Guide to Oracle 10*g*

Slide 1

CE12 Global to this PPT: Please note that most figure slides have figure caption as slide title, not heading - okay? (The individual slides are commented.)
CE, 8/1/2005

Lesson A Objectives

After completing this lesson, you should be able to:

- Create and use indexes
- Work with PL/SQL stored program units
- Create server-side stored program units in SQL*Plus
- Use Forms Builder to create stored program units

Database Indexes

- Database table index
  - Distinct database table
  - Contains data values with corresponding columns that specify physical locations of records that contain data values
- ROWID
  - Internal location of record in database
  - Encoded using internal data format

Database Indexes (continued)

- Index on specific table field
  - ROWID value
  - Sorted indexed field value
- Oracle 10*g* automatically creates index on table's primary key
  - Create indexes on columns that users often use in search conditions

Creating an Index

- Create after adding data
- Syntax:

```
CREATE INDEX index_name
ON tablename (index_fieldname);
```

Creating Composite Indexes

- Composite index
  - Multiple sorted columns
  - For queries that contain multiple search conditions primary search field
- Secondary search field
- Syntax:

```
CREATE INDEX index_name
ON tablename(index_fieldname1,
  index_fieldname2, …);
```

Viewing Index Information Using the Data Dictionary Views

- Query data dictionary views
  - Retrieve information about database objects
- Retrieve index information

CE1

Querying the USER_INDEXES Data Dictionary View (Partial Output Shown)

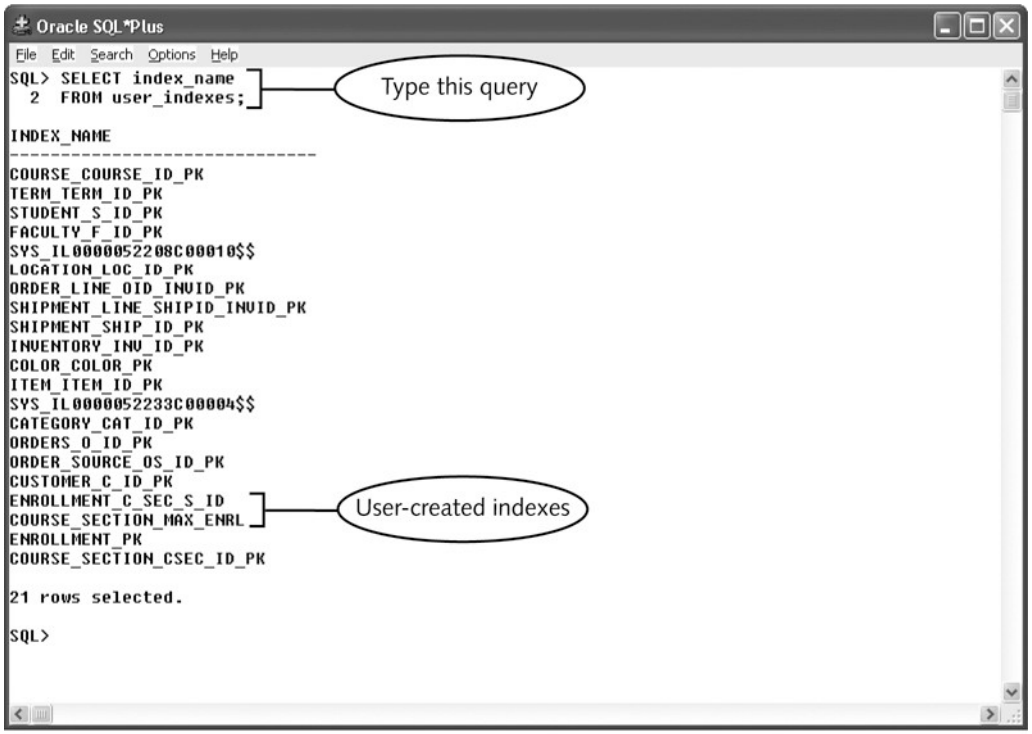

Figure 9-8 Querying the USER_INDEXES data dictionary view (partial output shown)

Slide 8

CE1 Slide title is figure caption, not heading - okay?
CE, 8/1/2005

Dropping an Index

- Drop index when:
  - Applications no longer use queries aided by index
  - Index does not improve query performance
    - Enough to justify overhead created on insert, update, and delete operations
- Syntax:
  - `DROP INDEX` *`index_name`*`;`

Determining When to Create an Index

- Create index when:
  - Table contains large number of records
  - Field contains wide range of values
  - Field contains large number of null values
  - Queries frequently use field in search condition or join condition
  - Most queries retrieve less than 2% to 4% of table rows

Determining When to Create an Index (continued)

- Do not create index when:
  - Table does not contain large number of records
  - Applications do not use proposed index field in query search condition
  - Most queries retrieve more than 2% to 4% of table records
  - Applications frequently insert or modify table data
- Decision based on judgment and experience

Overview of PL/SQL Stored Program Units

- Program unit
  - Self-contained group of program statements that can be used within larger program
- Anonymous PL/SQL programs
- Stored PL/SQL program units
- Server-side program units
- Client-side program units

CE2

Types of Oracle 10*g* Stored Program Units

Program Unit Type	Description	Where Stored	Where Executed
Procedure	Can accept multiple input parameters, and return multiple output values	Database	Server-side
Function	Can accept multiple input parameters, and can return a single output value	Database	Server-side
Library	Contains code for multiple related procedures or functions	Operating system file	Client-side
Package	Contains code for multiple related procedures, functions, and variables and can be made available to other database users	Database	Server-side
Database trigger	Contains code that executes when a user inserts, updates, or deletes records	Database	Server-side

Table 9-1 Types of Oracle10*g* stored program units

Slide 13

CE2 Slide title is figure caption, not heading - okay?
CE, 8/1/2005

Creating Stored Program Units

- Procedure
  - Receive multiple input parameters
  - Return multiple output values or return no output values
  - Perform action such as inserting, updating, or deleting database records
- Function
  - Receive multiple input parameters
  - Always returns single output value

Stored Program Unit Procedures

- CREATE PROCEDURE command
  - Header
  - Parameter declarations list
  - Program unit body
  - Exception section

CE3

Syntax to Create a Stored Program Unit Procedure

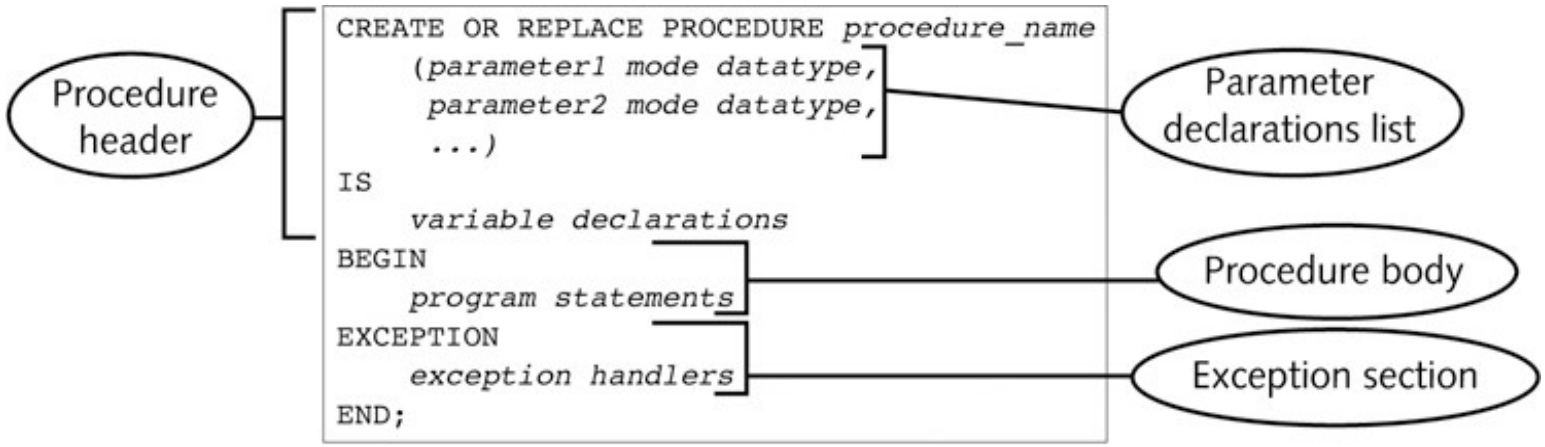

Figure 9-9 Syntax to create a stored program unit procedure

Slide 16

CE3 Slide title is figure caption, not heading - okay?
CE, 8/1/2005

Creating the Parameter Declarations List

- Defines parameters
- Declares associated data types
- Parameter mode
  - IN
  - OUT
  - IN OUT

CE5

Creating a Stored Procedure in SQL*Plus

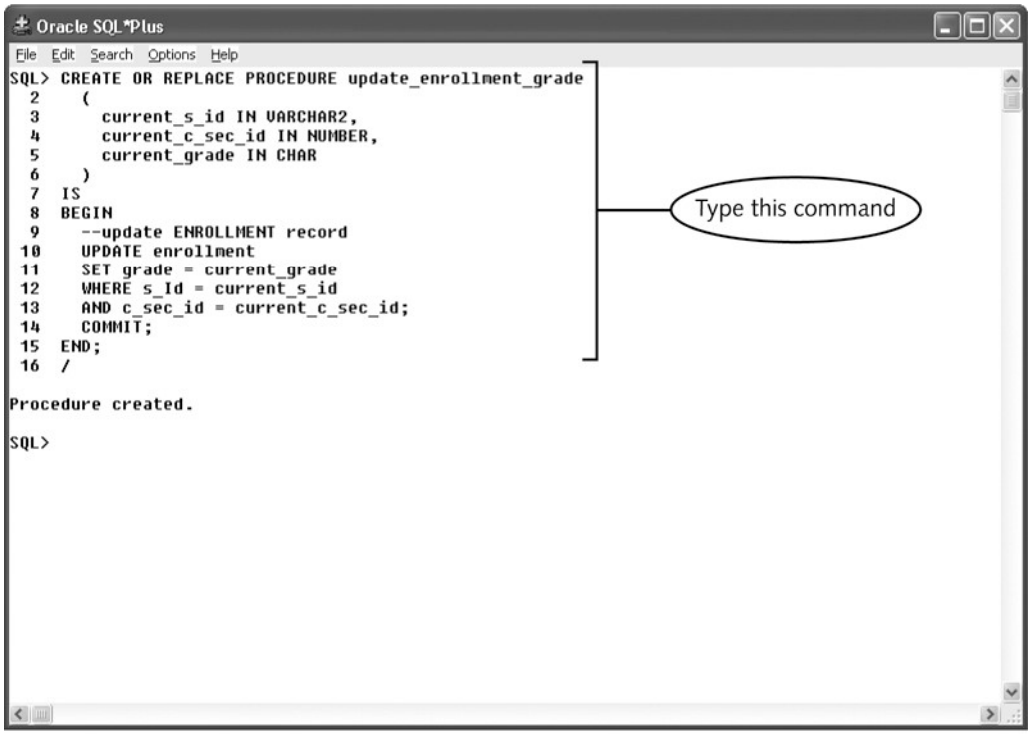

Figure 9-10 Creating a stored procedure in SQL*Plus

Slide 18

CE5 Please note that the text in this figure may be hard to read.
CE, 8/1/2005

Debugging Stored Program Units in SQL*Plus

- Similar to debugging any program
- Identify program line causing error
- SQL*Plus interpreter displays error warning message
  - Does not automatically display compile error messages and line locations
  - Writes all compile errors to system table
    - Access using USER_ERRORS data dictionary view
    - Execute SHOW ERRORS command

Calling a Stored Procedure

- Execute directly from SQL*Plus command line
- Create separate PL/SQL program that contains
  - Command to call stored procedure
  - Passes parameter values to procedure
- Calling stored procedure from SQL*Plus command line:

```
EXECUTE procedure_name
(parameter1_value,
 parameter2_value, ...);
```

CE6

Passing Parameters to a Procedure

```
Procedure Header
PROCEDURE update_enrollment_grade
       (current_s_idINVARCHAR2,current_c_sec_id IN NUMBER, current_grade INVARCHAR2)

Procedure Call: EXECUTE update_enrollment_grade(MA100,12,B);
```

Figure 9-13 Passing parameters to a procedure

Slide 21

CE6 Slide title is figure caption, not heading - okay?

Please note that this seems odd since it comes between two slides with the same title/heading (slides 20 and 22).
CE, 8/1/2005

Calling a Stored Procedure (continued)

- Variables passed for each parameter
  - Must be in same order as parameters appear in parameter declarations list
- Calling stored procedure from separate PL/SQL program
  - Similar to calling stored procedure from SQL*Plus command line
  - Omit EXECUTE command
  - `update_enrollment_grade(MA100, 12, B);`

Creating a Stored Program Unit Function

- Use `CREATE OR REPLACE FUNCTION` command
- *function_return_value_datatype*
  - Defines data type that function returns
- *return_value_variable*
  - Declares variable that represents function return value
- `RETURN` command

CE7

Commands to Create a Stored Program Unit Function

```
CREATE OR REPLACE FUNCTION function_name
     (parameter1 mode datatype,
      parameter2 mode datatype,
      ...)
RETURN function_return_value_datatype
IS
     return_value_variable datatype;
     other variable declarations
BEGIN
     program statement
     RETURN return_value_variable;
EXCEPTION
     exception handlers
     RETURN EXCEPTION_NOTICE;
END;
```

Defines function return value

Returns value to calling program

Displays exception message in calling program

Figure 9-16 Commands to create a stored program unit function

Slide 24

CE7 Slide title is figure caption, not heading - okay?
CE, 8/1/2005

Calling a Function

- Syntax:

```
variable_name :=
  function_name(parameter1,
  parameter2, ...);
```

- Variables passed for parameter values
  - Must be in same order as parameters appear in function declaration