

Oracle Database: Conceptos Fundamentales de SQL II

Volumen II • Guía del Alumno

D64260CS10

Edición 1.0

Enero de 2010

D73593

ORACLE

Autores

Chaitanya Koratamaddi
 Brian Pottle
 Tulika Srivastava

Colaboradores y Revisores Técnicos

Claire Bennett
 Ken Cooper
 Yanti Chang
 Laszlo Czinkoczki
 Burt Demchick
 Gerlinde Frenzen
 Joel Goodman
 Laura Garza
 Richard Green
 Nancy Greenberg
 Akira Kinutani
 Wendy Lo
 Isabelle Marchand
 Timothy Mcglue
 Alan Paulson
 Manish Pawar
 Srinivas Putrevu
 Bryan Roberts
 Clinton Shaffer
 Hilda Simson
 Abhishek Singh
 Jenny Tsai Smith
 James Spiller
 Lori Tritz
 Lex van der Werff
 Marcie Young

Redactores

Amitha Narayan
 Daniel Milne
 Raj Kumar

Diseñador Gráfico

Satish Bettegowda

Editores

Veena Narasimhan
 Pavithran Adka

Copyright © 2010, Oracle. Todos los derechos reservados.

Renuncia

En este curso se ofrece una visión general de las funciones y mejoras planificadas en la versión 11g. Únicamente pretende ayudarle a evaluar las ventajas de negocio de actualizar a 11g y planificar los proyectos de TI.

Este curso, en cualquiera de sus formatos (incluidos los ejercicios prácticos del curso y el material impreso), contiene información de propiedad en exclusiva de Oracle. Este curso y la información que contiene no se podrán revelar, copiar, reproducir o distribuir a ninguna persona ajena a Oracle sin el consentimiento previo por escrito de Oracle. Este curso y su contenido no forman parte del acuerdo de licencia ni se pueden incorporar a ningún acuerdo contractual con Oracle, sus subsidiarias o filiales.

Este curso es sólo para fines informativos y únicamente pretende ayudarle a planificar la implantación y la actualización de las funciones del producto descritas. No existe ningún compromiso de entregar ningún material, código o funcionalidad, y no se confiará en él al tomar decisiones de compra. El desarrollo, la publicación y la distribución de las funciones o funcionalidades descritas en este documento seguirán siendo a entera discreción de Oracle.

Este documento contiene información propiedad de Oracle Corporation y se encuentra protegido por las leyes de copyright, así como por otras leyes de propiedad intelectual. El usuario podrá realizar copias o imprimir este documento para su uso exclusivo en los cursos de formación de Oracle. Este documento no podrá ser modificado ni alterado en modo alguno. Salvo que la legislación de copyright lo considere un uso legítimo, no podrá utilizar, compartir, descargar, cargar, copiar, imprimir, mostrar, representar, reproducir, publicar, conceder licencias, enviar, transmitir ni distribuir este documento total ni parcialmente sin autorización expresa por parte de Oracle.

La información contenida en este documento está sujeta a cambio sin previo aviso. Si detecta cualquier problema en el documento, le agradeceremos que nos lo comunique por escrito a: Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 EE. UU. Oracle Corporation no garantiza que este documento esté exento de errores.

Aviso sobre Restricción de Derechos

Si esta documentación se entrega al Gobierno de los EE. UU. o a cualquier entidad que la utilice en nombre del Gobierno de los EE. UU., se aplicará la siguiente disposición:

U.S. GOVERNMENT RIGHTS

The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

Aviso de Marca Registrada

Oracle es una marca comercial registrada de Oracle Corporation y/o sus filiales. Todos los demás nombres pueden ser marcas comerciales de sus respectivos propietarios.

Contenido

I Introducción

- Objetivos I-2
- Agenda I-3
- Objetivos del Curso I-4
- Requisitos del Curso I-5
- Agenda I-6
- Tablas Utilizadas en este Curso I-8
- Apéndices Utilizados en este Curso I-9
- Entornos de Desarrollo I-10
- Agenda I-11
- Revisión de la Restricción de Datos I-12
- Revisión de Ordenación de Datos I-13
- Revisión de Funciones SQL I-14
- Revisión de Funciones de Una Sola Fila I-15
- Revisión de Tipos de Funciones de Grupo I-16
- Revisión de Subconsultas I-17
- Revisión de Manipulación de Datos I-18
- Agenda I-19
- Documentación sobre SQL de Oracle Database I-20
- Recursos Adicionales I-21
- Resumen I-22
- Práctica I: Visión General I-23

1 Control del Acceso de los Usuarios

- Objetivos 1-2
- Agenda 1-3
- Control del Acceso de los Usuarios 1-4
- Privilegios 1-5
- Privilegios del Sistema 1-6
- Creación de un Usuario 1-7
- Privilegios del Sistema de Usuario 1-8
- Asignación de Privilegios del Sistema 1-9
- Agenda 1-10
- ¿Qué es un Rol? 1-11
- Creación y Asignación de Privilegios a un Rol 1-12
- Cambio de Contraseña 1-13

Agenda 1-14
Privilegios de Objeto 1-15
Asignación de Privilegios de Objeto 1-17
Transferencia de Privilegios 1-18
Confirmación de Privilegios Otorgados 1-19
Agenda 1-20
Revocación de Privilegios de Objeto 1-21
Prueba 1-23
Resumen 1-24
Práctica 1: Visión General 1-25

2 Gestión de Objetos de Esquema

Objetivos 2-2
Agenda 2-3
Sentencia ALTER TABLE 2-4
Adición de Columnas 2-6
Modificación de Columnas 2-7
Borrado de Columnas 2-8
Opción SET UNUSED 2-9
Agenda 2-11
Adición de Sintaxis de Restricción 2-12
Adición de Restricciones 2-13
Cláusula ON DELETE 2-14
Restricciones Diferidas 2-15
Diferencia entre INITIALLY DEFERRED y INITIALLY IMMEDIATE 2-16
Borrado de Restricciones 2-18
Desactivación de Restricciones 2-19
Activación de Restricciones 2-20
Restricciones en Cascada 2-22
Cambio de Nombre de Columnas de Tabla y Restricciones 2-24
Agenda 2-25
Visión General de Índices 2-26
CREATE INDEX con la Sentencia CREATE TABLE 2-27
Índices Basados en Funciones 2-29
Eliminación de Índices 2-30
DROP TABLE ... PURGE 2-31
Agenda 2-32
Sentencia FLASHBACK TABLE 2-33
Uso de la Sentencia FLASHBACK TABLE 2-35
Agenda 2-36

Tablas Temporales	2-37
Creación de una Tabla Temporal	2-38
Agenda	2-39
Tablas Externas	2-40
Creación de un Directorio para la Tabla Externa	2-41
Creación de Tablas Externas	2-43
Creación de una Tabla Externa mediante ORACLE_LOADER	2-45
Consulta de Tablas Externas	2-47
Creación de una Tabla Externa mediante ORACLE_DATAPUMP: Ejemplo	2-48
Prueba	2-49
Resumen	2-51
Práctica 2: Visión General	2-52
3 Gestión de Objetos con Vistas de Diccionario de Datos	
Objetivos	3-2
Agenda	3-3
Diccionario de Datos	3-4
Estructura del Diccionario de Datos	3-5
Uso de las Vistas de Diccionario	3-7
Vistas USER_OBJECTS y ALL_OBJECTS	3-8
Vista USER_OBJECTS	3-9
Agenda	3-10
Información sobre Tablas	3-11
Información sobre Columnas	3-12
Información sobre Restricciones	3-14
USER_CONSTRAINTS: Ejemplo	3-15
Consulta de USER_CONS_COLUMNS	3-16
Agenda	3-17
Información sobre Vistas	3-18
Información sobre Secuencias	3-19
Confirmación de Secuencias	3-20
Información sobre Índices	3-21
USER_INDEXES: Ejemplos	3-22
Consulta USER_IND_COLUMNS	3-23
Información sobre Sinónimos	3-24
Agenda	3-25
Adición de Comentarios a una Tabla	3-26
Prueba	3-27
Resumen	3-28
Práctica 3: Visión General	3-29

4 Manipulación de Juegos de Datos Grandes

- Objetivos 4-2
Agenda 4-3
Uso de Subconsultas para Manipular Datos 4-4
Recuperación de Datos mediante una Subconsulta como Origen 4-5
Inserción mediante una Subconsulta como Destino 4-7
Uso de la Palabra Clave WITH CHECK OPTION en Sentencias DML 4-9
Agenda 4-11
Visión General de la Función por Defecto Explícita 4-12
Uso de Valores por Defecto Explícitos 4-13
Copia de Filas de Otra Tabla 4-14
Agenda 4-15
Visión General de Sentencias INSERT de Varias de Tablas 4-16
Tipos de Sentencias INSERT de Varias Tablas 4-18
Sentencias INSERT de Varias Tablas 4-19
INSERT ALL Incondicional 4-21
INSERT ALL Condicional: Ejemplo 4-23
INSERT ALL Condicional 4-24
INSERT FIRST Condicional: Ejemplo 4-26
INSERT FIRST Condicional 4-27
INSERT mediante giro 4-29
Agenda 4-32
Sentencia MERGE 4-33
Sintaxis de Sentencias MERGE 4-34
Fusión de Filas: Ejemplo 4-35
Agenda 4-38
Seguimiento de Cambios en Datos 4-39
Ejemplo de Consulta de Flashback de Versiones 4-40
Cláusula VERSIONS BETWEEN 4-42
Prueba 4-43
Resumen 4-44
Práctica 4: Visión General 4-45

5 Gestión de Datos Situados en Distintas Zonas Horarias

- Objetivos 5-2
Agenda 5-3
Zonas Horarias 5-4
Parámetro de Sesión TIME_ZONE 5-5
CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP 5-6

Comparación de Fecha y Hora en la Zona Horaria de una Sesión	5-7
DBTIMEZONE y SESSIONTIMEZONE	5-9
Tipos de Dato TIMESTAMP	5-10
Campos TIMESTAMP	5-11
Diferencia entre DATE y TIMESTAMP	5-12
Comparación de Tipos de Dato TIMESTAMP	5-13
Agenda	5-14
Tipos de Dato INTERVAL	5-15
Campos INTERVAL	5-17
INTERVAL YEAR TO MONTH: Ejemplo	5-18
Tipo de Dato INTERVAL DAY TO SECOND: Ejemplo	5-20
Agenda	5-21
EXTRACT	5-22
TZ_OFFSET	5-23
FROM_TZ	5-25
TO_TIMESTAMP	5-26
TO_YMINTERVAL	5-27
TO_DSINTERVAL	5-28
Horario de Verano	5-29
Prueba	5-31
Resumen	5-32
Práctica 5: Visión General	5-33

6 Recuperación de Datos mediante Subconsultas

Objetivos	6-2
Agenda	6-3
Subconsultas de Varias Columnas	6-4
Comparaciones de Columnas	6-5
Subconsulta de Comparación Pairwise	6-6
Subconsulta de comparación No Pairwise	6-8
Agenda	6-10
Expresiones de Subconsultas Escalares	6-11
Subconsultas Escalares: Ejemplos	6-12
Agenda	6-14
Subconsultas Correlacionadas	6-15
Uso de Subconsultas Correlacionadas	6-17
Agenda	6-19
Uso del Operador EXISTS	6-20
Buscar Todos los Departamentos que No Tienen Empleados	6-22
Subconsulta Correlacionada UPDATE	6-23

Uso de la Subconsulta Correlacionada UPDATE	6-24
Subconsulta Correlacionada DELETE	6-26
Uso de una Subconsulta Correlacionada DELETE	6-27
Agenda	6-28
Cláusula WITH	6-29
Cláusula WITH: Ejemplo	6-30
Cláusula WITH Recursiva	6-32
Cláusula WITH Recursiva: Ejemplo	6-33
Prueba	6-34
Resumen	6-35
Práctica 6: Visión General	6-37

7 Soporte para Expresiones Normales

Objetivos	7-2
Agenda	7-3
¿Qué Son Expresiones Regulares?	7-4
Ventajas de Utilizar Expresiones Regulares	7-5
Uso de Funciones y Condiciones de Expresiones Regulares en SQL y PL/SQL	7-6
Agenda	7-7
¿Que Son Metacaracteres?	7-8
Uso de Metacaracteres con Expresiones Normales	7-9
Agenda	7-11
Condiciones y Funciones de Expresiones Regulares: Sintaxis	7-12
Realización de una Búsqueda Básica mediante la Condición REGEXP_LIKE	7-13
Sustitución de Patrones mediante la Función REGEXP_REPLACE	7-14
Búsqueda de Patrones mediante la Función REGEXP_REPLACE	7-15
Búsqueda de Patrones mediante la Función REGEXP_SUBSTR	7-16
Agenda	7-17
Subexpresiones	7-18
Uso de Subexpresiones con el Soporte para Expresiones Regulares	7-19
¿Por qué Acceder a la n Subexpresión?	7-20
REGEXP_SUBSTR: Ejemplo	7-21
Agenda	7-22
Uso de la Función REGEXP_COUNT	7-23
Expresiones Regulares y Restricciones de Control: Ejemplos	7-24
Prueba	7-25
Resumen	7-26
Práctica 7: Visión General	7-27

Apéndice A: Prácticas y Soluciones

Apéndice B: Descripciones de las Tablas**Apéndice C: Uso de SQL Developer**

- Objetivos C-2
¿Qué es Oracle SQL Developer? C-3
Especificaciones de SQL Developer C-4
Interfaz de SQL Developer 1.5 C-5
Creación de una Conexión a la Base Datos C-7
Examen de Objetos de Bases de Datos C-10
Visualización de la Estructura de la Tabla C-11
Examen de Archivos C-12
Creación de un Objeto de Esquema C-13
Creación de una Nueva Tabla: Ejemplo C-14
Uso de la Hoja de Trabajo de SQL C-15
Ejecución de Sentencias SQL C-18
Guardado de Scripts SQL C-19
Ejecución de Archivos de Script Guardados: Método 1 C-20
Ejecución de Archivos de Script Guardados: Método 2 C-21
Formato del Código SQL C-22
Uso de Fragmentos C-23
Uso de Fragmentos: Ejemplo C-24
Depuración de Procedimientos y Funciones C-25
Informes de Bases de Datos C-26
Creación de un Informe Definido por el Usuario C-27
Motores de Búsqueda y Herramientas Externas C-28
Definición de Preferencias C-29
Restablecimiento del Diseño de SQL Developer C-30
Resumen C-31

Apéndice D: Uso de SQL*Plus

- Objetivos D-2
Interacción de SQL y SQL*Plus D-3
Sentencias SQL frente a Comandos SQL*Plus D-4
Visión General de SQL*Plus D-5
Conexión a SQL*Plus D-6
Visualización de la Estructura de la Tabla D-7
Comandos de Edición SQL*Plus D-9
Uso de LIST, n y APPEND D-11
Uso del Comando CHANGE D-12
Comandos de Archivos SQL*Plus D-13
Uso de los Comandos SAVE y START D-14

Comando SERVEROUTPUT	D-15
Uso del Comando SQL*Plus SPOOL	D-16
Uso del Comando AUTOTRACE	D-17
Resumen	D-18

Apéndice E: Uso de JDeveloper

Objetivos	E-2
Oracle JDeveloper	E-3
Database Navigator	E-4
Creación de una Conexión	E-5
Examen de Objetos de Bases de Datos	E-6
Ejecución de Sentencias SQL	E-7
Creación de Unidades de Programa	E-8
Compilación	E-9
Ejecución de una Unidad de Programa	E-10
Borrado de una Unidad de Programa	E-11
Ventana Structure	E-12
Ventana del Editor	E-13
Navegador de Aplicaciones	E-14
Despliegue de Procedimientos Java Almacenados	E-15
Publicación de Java en PL/SQL	E-16
¿Cómo Puedo Obtener más Información sobre JDeveloper 11g?	E-17
Resumen	E-18

Apéndice F: Generación de Informes Agrupando Datos Relacionados

Objetivos	F-2
Revisión de Funciones de Grupo	F-3
Revisión de la Cláusula GROUP BY	F-4
Revisión de la Cláusula HAVING	F-5
GROUP BY con los Operadores ROLLUP y CUBE	F-6
Operador ROLLUP	F-7
Operador ROLLUP: Ejemplo	F-8
Operador CUBE	F-9
Operador CUBE: Ejemplo	F-10
Función GROUPING	F-11
Función GROUPING: Ejemplo	F-12
GROUPING SETS	F-13
GROUPING SETS: Ejemplo	F-15
Columnas Compuestas	F-17

- Columnas Compuestas: Ejemplo F-19
- Agrupamientos Concatenados F-21
- Agrupamientos Concatenados: Ejemplo F-22
- Resumen F-23

Apéndice G: Recuperación Jerárquica

- Objetivos G-2
- Datos de Ejemplo de la Tabla EMPLOYEES G-3
- Estructura de Árbol Natural G-4
- Consultas Jerárquicas G-5
- Recorrido por el Árbol G-6
- Recorrido por el Árbol: De Abajo Arriba G-8
- Recorrido por el Árbol: De Arriba Abajo G-9
- Clasificación de Filas con la Pseudocolumna LEVEL G-10
- Aplicación de Formato a Informes Jerárquicos con LEVEL y LPAD G-11
- Eliminación de Ramas G-13
- Resumen G-14

Apéndice H: Escritura de Archivos de Comandos Avanzados

- Objetivos H-2
- Uso de SQL para Generar SQL H-3
- Creación de un Script Básico H-4
- Control del Entorno H-5
- Imagen Completa H-6
- Volcado del Contenido de una Tabla en un Archivo H-7
- Generación de un Predicado Dinámico H-9
- Resumen H-11

Apéndice I: Componentes Arquitectónicos de Oracle Database

- Objetivos I-2
- Arquitectura de Oracle Database: Visión General I-3
- Estructuras de Servidor de Oracle Database I-4
- Conexión a la Base de Datos I-5
- Interacción con Oracle Database I-6
- Arquitectura de Memoria de Oracle I-8
- Arquitectura de Proceso I-10
- Proceso de Escritores de Base de Datos I-12
- Proceso de Escritor de Log I-13
- Proceso de Punto de Control I-14
- Proceso de Supervisión del Sistema I-15

- Proceso de Supervisión de Procesos I-16
- Arquitectura de Almacenamiento de Oracle Database I-17
- Estructuras de Bases de Datos Físicas y Lógicas I-19
- Procesamiento de Sentencias SQL I-21
- Procesamiento de Consultas I-22
- Pool Compartido I-23
- Caché de buffers de la base de datos I-25
- Área Global de Programa (PGA) I-26
- Procesamiento de una Sentencia DML I-27
- Buffer de Redo Log I-29
- Segmento de Rollback I-30
- Procesamiento COMMIT I-31
- Resumen de Arquitectura de Oracle Database I-33

Prácticas y soluciones adicionales

Apéndice A

Prácticas y Soluciones

Tabla de Contenido

Prácticas y Soluciones de la Lección I.....	3
Práctica I-1: Acceso a los Recursos de SQL Developer	4
Práctica I-2: Uso de SQL Developer	5
Soluciones a la Práctica I-1: Acceso a los Recursos de SQL Developer.....	7
Soluciones a la Práctica I-2: Uso de SQL Developer	8
Prácticas y Soluciones de la Lección 1	17
Práctica 1-1: Control del Acceso de los Usuarios.....	17
Soluciones a la Práctica 1-1: Control del Acceso de los Usuarios	20
Prácticas y Soluciones de la Lección 2	25
Práctica 2-1: Gestión de Objetos de Esquema	25
Soluciones a la Práctica 2-1: Gestión de Objetos de Esquema	31
Prácticas y Soluciones de la Lección 3	39
Práctica 3-1: Gestión de Objetos con Vistas de Diccionario de Datos	39
Soluciones a la Práctica 3-1: Gestión de Objetos con Vistas de Diccionario de Datos	43
Prácticas y Soluciones de la Lección 4	47
Práctica 4-1: Manipulación de Juegos de Datos Grandes	47
Soluciones a la Práctica 4-1: Manipulación de Juegos de Datos Grandes.....	51
Prácticas y Soluciones de la Lección 5	56
Práctica 5-1: Gestión de Datos Situados en Distintas Zonas Horarias	56
Soluciones a la Práctica 5-1: Gestión de Datos Situados en Distintas Zonas Horarias	59
Prácticas y Soluciones de la Lección 6	62
Práctica 6-1: Recuperación de Datos mediante Subconsultas	62
Soluciones a la Práctica 6-1: Recuperación de Datos mediante Subconsultas	66
Prácticas y Soluciones de la Lección 7	70
Práctica 7-1: Soporte para Expresiones Regulares	70
Soluciones a la Práctica 7-1: Soporte para Expresiones Regulares	72

Prácticas y Soluciones de la Lección I

En esta práctica, revisará los recursos disponibles de SQL Developer. También aprenderá acerca de la cuenta de usuario que utilizará en este curso. A continuación, iniciará SQL Developer, creará una nueva conexión a la base de datos y examinará las tablas de HR. También definirá algunas preferencias de SQL Developer, ejecutará sentencias SQL y un bloque PL/SQL anónimo mediante una hoja de trabajo de SQL. Por último, accederá y marcará la documentación de Oracle Database 11g y de otros sitios web útiles que puede utilizar en este curso.

Práctica I-1: Acceso a los Recursos de SQL Developer

En esta práctica, realice las siguientes tareas:

1. Acceda a la página inicial de SQL Developer.
 - a. Acceda a la página inicial de SQL Developer en línea disponible en:
http://www.oracle.com/technology/products/database/sql_developer/index.html
 - b. Marque la página para acceder más fácilmente a ella en el futuro.
2. Acceda al tutorial de SQL Developer disponible en línea en:
<http://st-curriculum.oracle.com/tutorial/SQLDeveloper/index.htm>. A continuación, revise las siguientes secciones y demostraciones asociadas:
 - a. Pasos Iniciales
 - b. Trabajar con Objetos de la Base de Datos
 - c. Acceso a los Datos

Práctica I-2: Uso de SQL Developer

1. Inicie SQL Developer con el ícono del escritorio.
2. Cree una conexión de base de datos con la siguiente información:
 - a. Connection Name: myconnection
 - b. Nombre de usuario: oraxx, donde xx es el número del PC (pregunte al instructor para que le asigne una cuenta ora fuera del rango de cuentas ora21-ora40.)
 - c. Password: oraxx
 - d. Hostname: localhost
 - e. Port: 1521
 - f. SID: orcl (o el valor proporcionado por el instructor)
3. Pruebe la nueva conexión. Si el estado es Success, conéctese a la base de datos mediante esta nueva conexión.
 - a. Haga clic en el botón Test de la ventana New>Select Database Connection.
 - b. Si el estado es Success, haga clic en el botón Connect.
4. Examine la estructura de la tabla EMPLOYEES y muestre sus datos.
 - a. Amplíe la conexión myconnection haciendo clic en el signo más situado junto a la misma.
 - b. Amplíe el ícono Tables. Para ello, haga clic en el signo más situado junto a él.
 - c. Visualice la estructura de la tabla EMPLOYEES.
 - d. Visualice los datos de la tabla DEPARTMENTS.
5. Ejecute algunas sentencias SELECT básicas para consultar los datos en la tabla EMPLOYEES en el área de la hoja de trabajo de SQL. Utilice los iconos Execute Statement (o pulse F9) y Run Script (o pulse F5) para ejecutar las sentencias SELECT. Revise los resultados de ambos métodos de ejecución de sentencias SELECT en las páginas con separadores adecuadas.
 - a. Escriba una consulta para seleccionar el apellido y salario de los empleados cuyo salario sea inferior o igual a 3.000 dólares.
 - b. Escriba una consulta para mostrar el apellido, el ID de cargo y la comisión de todos los empleados que no tienen derecho a recibir una comisión.
6. Defina su preferencia de rutas de acceso del script en /home/oracle/labs/sql2.
 - a. Seleccione Tools > Preferences > Database > Worksheet Parameters.
 - b. Introduzca el valor en el campo Select default path to look for scripts.
7. Introduzca lo siguiente en el cuadro Enter SQL Statement

```
SELECT employee_id, first_name, last_name,  
       FROM employees;
```

Práctica I-2: Uso de SQL Developer (continuación)

8. Guarde la sentencia SQL en un script mediante la opción de menú File > Save As.
 - a. Seleccione File > Save As.
 - b. Asigne el nombre `intro_test.sql` al archivo.
 - c. Coloque el archivo en la carpeta `/home/oracle/labs/sql2/labs`.
9. Abra y ejecute `confidence.sql` de la carpeta `/home/oracle/labs/sql2/labs` y observe el resultado.

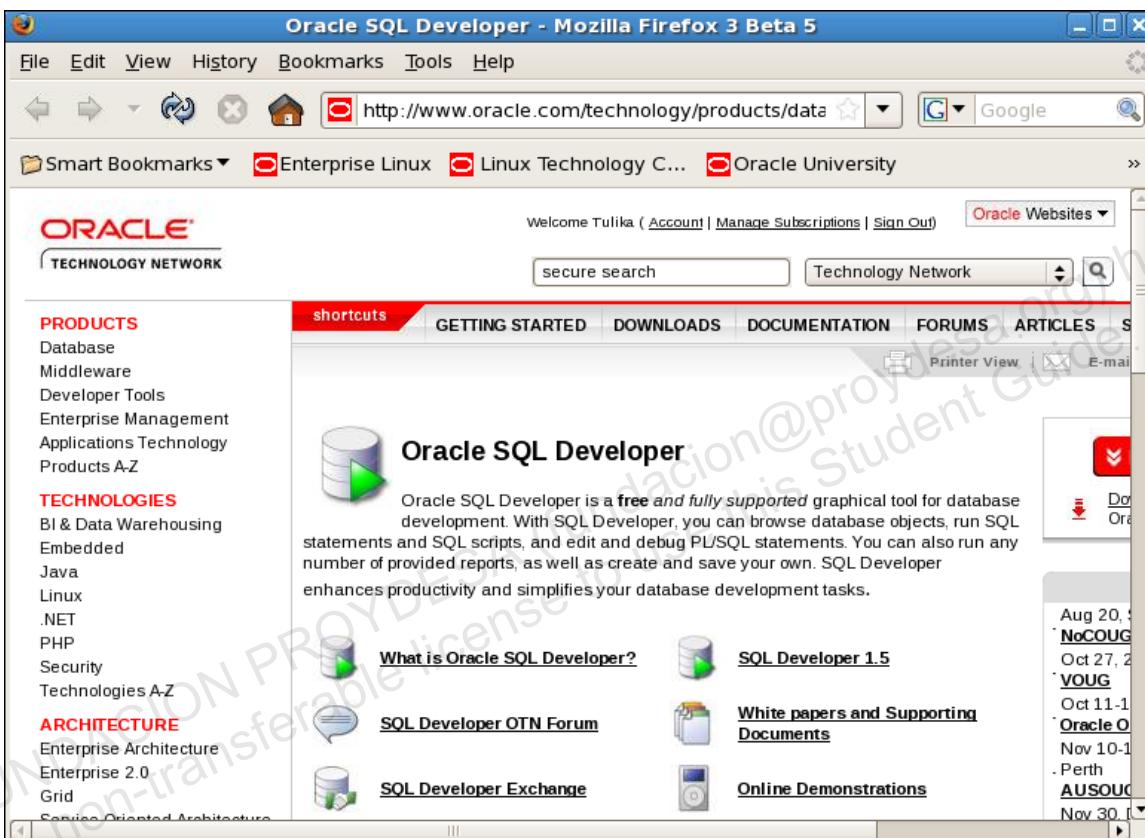
Soluciones a la Práctica I-1: Acceso a los Recursos de SQL Developer

1. Acceda a la página inicial de SQL Developer.

a. Acceda a la página inicial de SQL Developer en línea disponible en:

http://www.oracle.com/technology/products/database/sql_developer/index.html

La página inicial de SQL Developer se muestra de la siguiente forma:



- b. Marque la página para un acceso futuro más sencillo.
2. Acceda al tutorial de SQL Developer disponible en línea en:

<http://st-curriculum.oracle.com/tutorial/SQLDeveloper/index.htm>.

A continuación, revise las siguientes secciones y demostraciones asociadas:

- a. Qué Hacer Primero
- b. Trabajar con Objetos de Base de Datos
- c. Acceso a Datos

Soluciones a la Práctica I-2: Uso de SQL Developer

1. Inicie SQL Developer con el icono del escritorio.

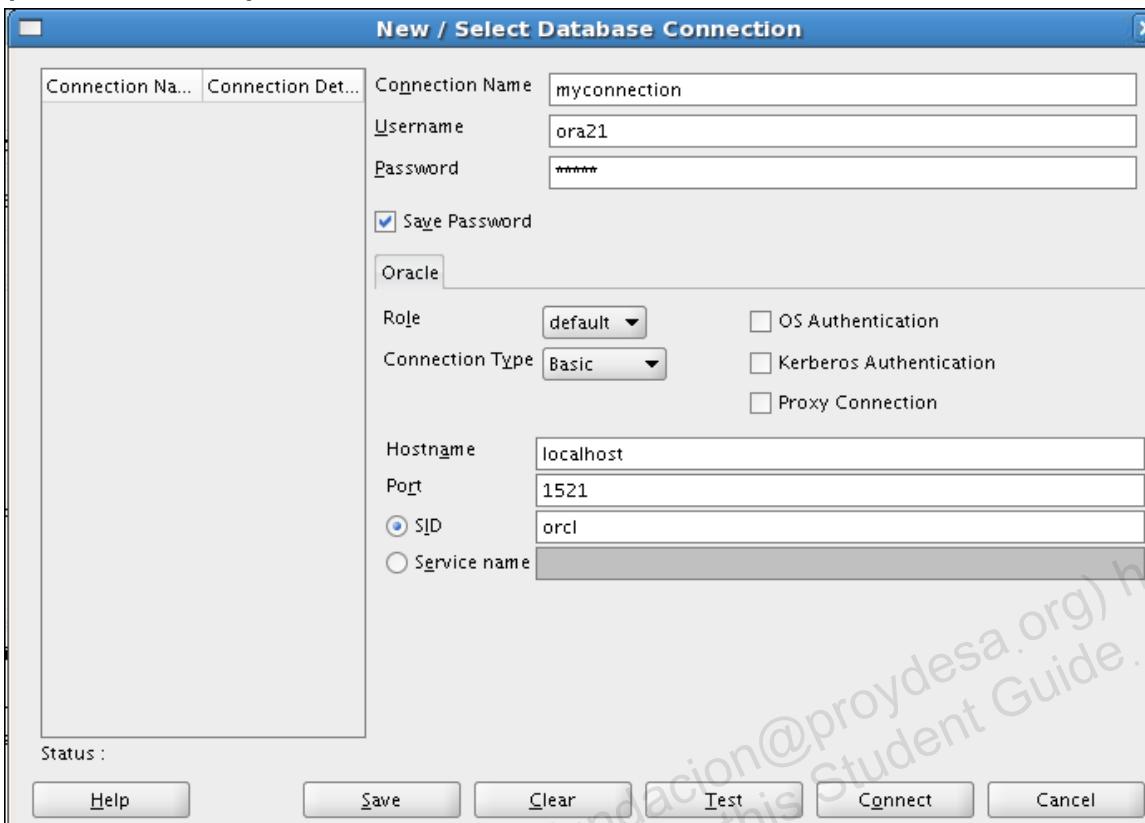


2. Cree una conexión a la base de datos con la siguiente información:

- a. Connection Name: myconnection
- b. Nombre de usuario: oraxxx (pregunte al instructor para que le asigne una cuenta ora fuera del rango de cuentas ora21–ora40.)
- c. Password: oraxxx
- d. Hostname: localhost
- e. Port: 1521
- f. SID: orcl (o el valor proporcionado por el instructor)



Soluciones a la Práctica I-2: Uso de SQL Developer (continuación)



3. Pruebe la nueva conexión. Si el estado es Success, conéctese a la base de datos mediante esta nueva conexión.

a. Haga clic en el botón Test en la ventana New/Select Database Connection.



b. Si el estado es Success, haga clic en el botón Connect.



Examen de Tablas

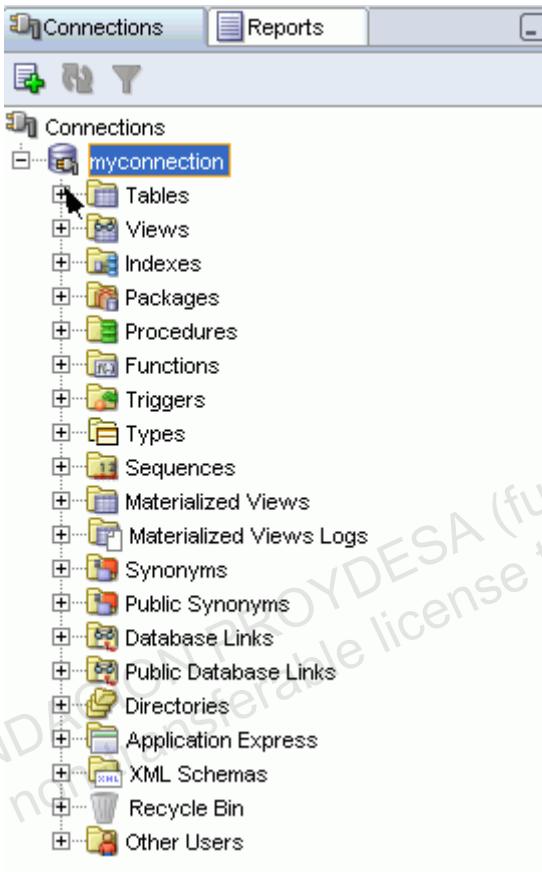
4. Examine la estructura de la tabla EMPLOYEES y muestre sus datos.

a. Amplíe la conexión myconnection haciendo clic en el signo más situado junto a la misma.

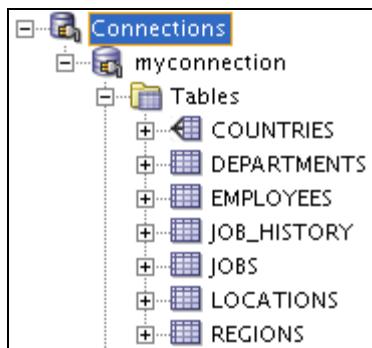
Soluciones a la Práctica I-2: Uso de SQL Developer (continuación)



b. Amplíe el ícono Tables haciendo clic en el signo más situado junto al mismo.



Soluciones a la Práctica I-2: Uso de SQL Developer (continuación)



- c. Muestre la estructura de la tabla EMPLOYEES.

Haga clic en la tabla **EMPLOYEES**. El separador Columns muestra las columnas de la tabla EMPLOYEES del siguiente modo:

Column Name	Data Type	Nullable	Data Default	COLUMN ID	Primary Key	COMMENTS
EMPLOYEE_ID	NUMBER(6,0)	No	(null)	1	1	Primary key of emplo
FIRST_NAME	VARCHAR2(20 BYTE)	Yes	(null)	2		(null) First name of the emp
LAST_NAME	VARCHAR2(25 BYTE)	No	(null)	3		(null) Last name of the emp
EMAIL	VARCHAR2(25 BYTE)	No	(null)	4		(null) Email id of the emp
PHONE_NUMBER	VARCHAR2(20 BYTE)	Yes	(null)	5		(null) Phone number of the
HIRE_DATE	DATE	No	(null)	6		(null) Date when the emplo
JOB_ID	VARCHAR2(10 BYTE)	No	(null)	7		(null) Current job of the em
SALARY	NUMBER(8,2)	Yes	(null)	8		(null) Monthly salary of the
COMMISSION_PCT	NUMBER(2,2)	Yes	(null)	9		(null) Commission percent:
MANAGER_ID	NUMBER(6,0)	Yes	(null)	10		(null) Manager id of the em
DEPARTMENT_ID	NUMBER(4,0)	Yes	(null)	11		(null) Department id where

- d. Visualice los datos de la tabla DEPARTMENTS.

En el navegador de conexiones, haga clic en la tabla **DEPARTMENTS**.

A continuación, haga clic en el separador Data.

Soluciones a la Práctica I-2: Uso de SQL Developer (continuación)

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	30	Purchasing	114	1700
4	40	Human Resources	203	2400
5	50	Shipping	121	1500
6	60	IT	103	1400
7	70	Public Relations	204	2700

5. Ejecute algunas sentencias SELECT básicas para consultar los datos en la tabla EMPLOYEES en el área de la hoja de trabajo de SQL. Utilice los iconos Execute Statement (o pulse F9) y Run Script (o F5) para ejecutar las sentencias SELECT. Revise los resultados de ambos métodos de ejecución de sentencias SELECT en las páginas con separadores adecuadas.

- a. Escriba una consulta para seleccionar el apellido y salario de los empleados cuyo salario sea inferior o igual a 3.000 dólares.

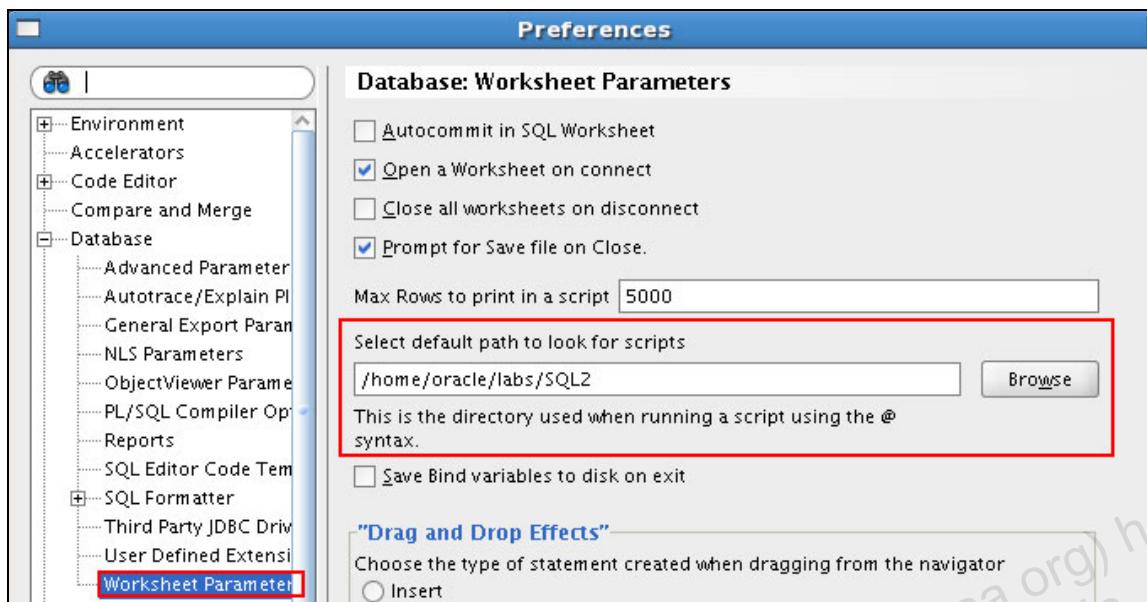
```
SELECT last_name, salary
FROM employees
WHERE salary <= 3000;
```

- b. Escriba una consulta para mostrar el apellido, el ID de cargo y la comisión de todos los empleados que no tienen derecho a recibir una comisión.

```
SELECT last_name, job_id, commission_pct
FROM employees
WHERE commission_pct IS NULL;
```

6. Defina su preferencia de rutas de acceso del script en /home/oracle/labs/sql2.
- Seleccione Tools > Preferences > Database > Worksheet Parameters.
 - Introduzca el valor en el campo **Select default path to look for scripts**. A continuación, haga clic en OK.

Soluciones a la Práctica I-2: Uso de SQL Developer (continuación)

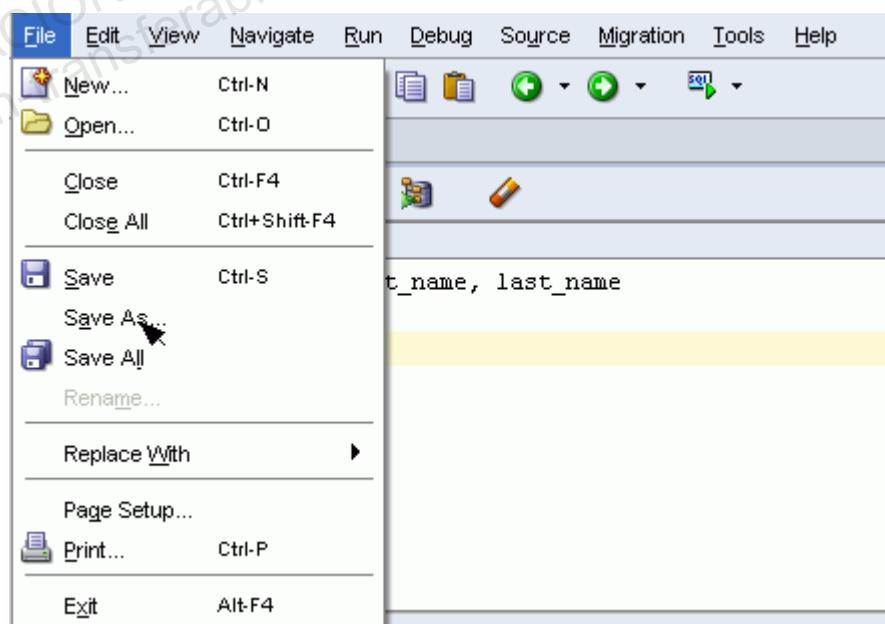


7. Introduzca la siguiente sentencia SQL:

```
SELECT employee_id, first_name, last_name
FROM employees;
```

8. Guarde la sentencia SQL en un script mediante la opción de menú File > Save As.

a. Seleccione File > Save As.

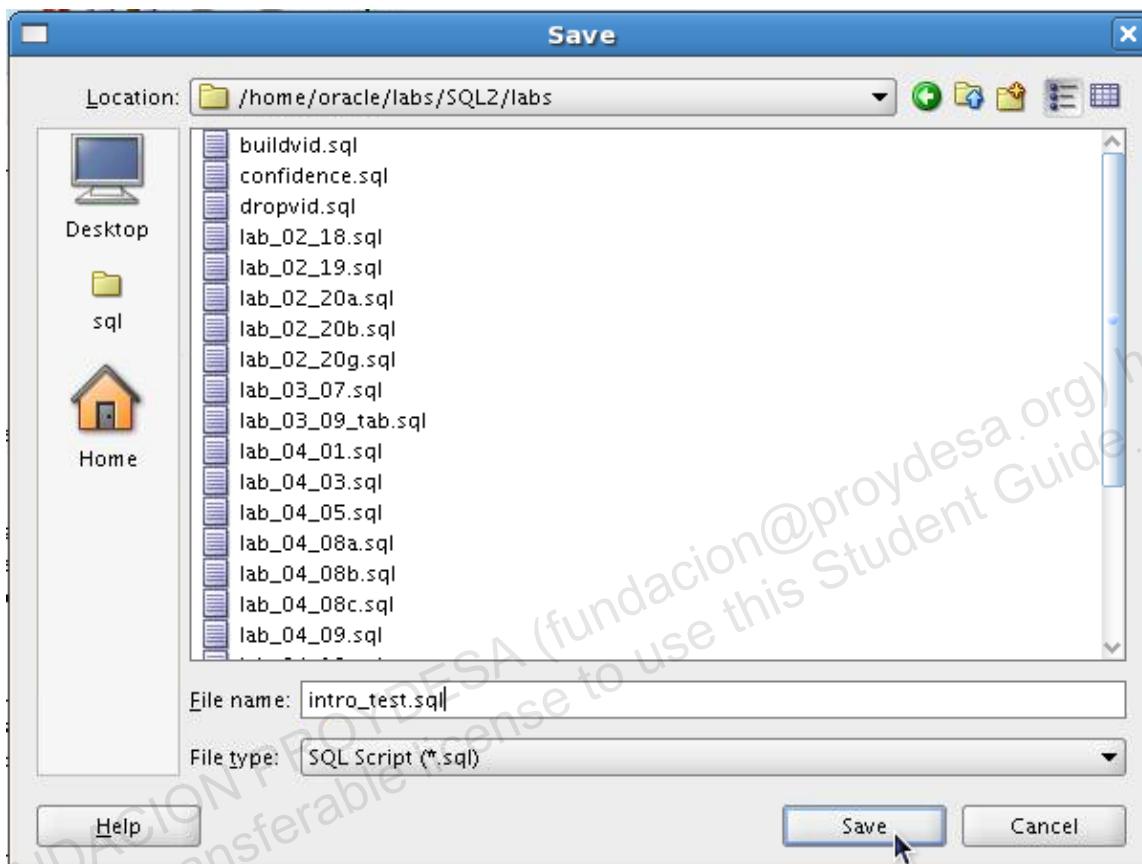


Soluciones a la Práctica I-2: Uso de SQL Developer (continuación)

b. Asigne el nombre **intro_test.sql** al archivo.

Introduzca **intro_test.sql** en el cuadro de texto **File_name**.

c. Coloque el archivo en la carpeta **/home/oracle/labs/SQL2/labs**.

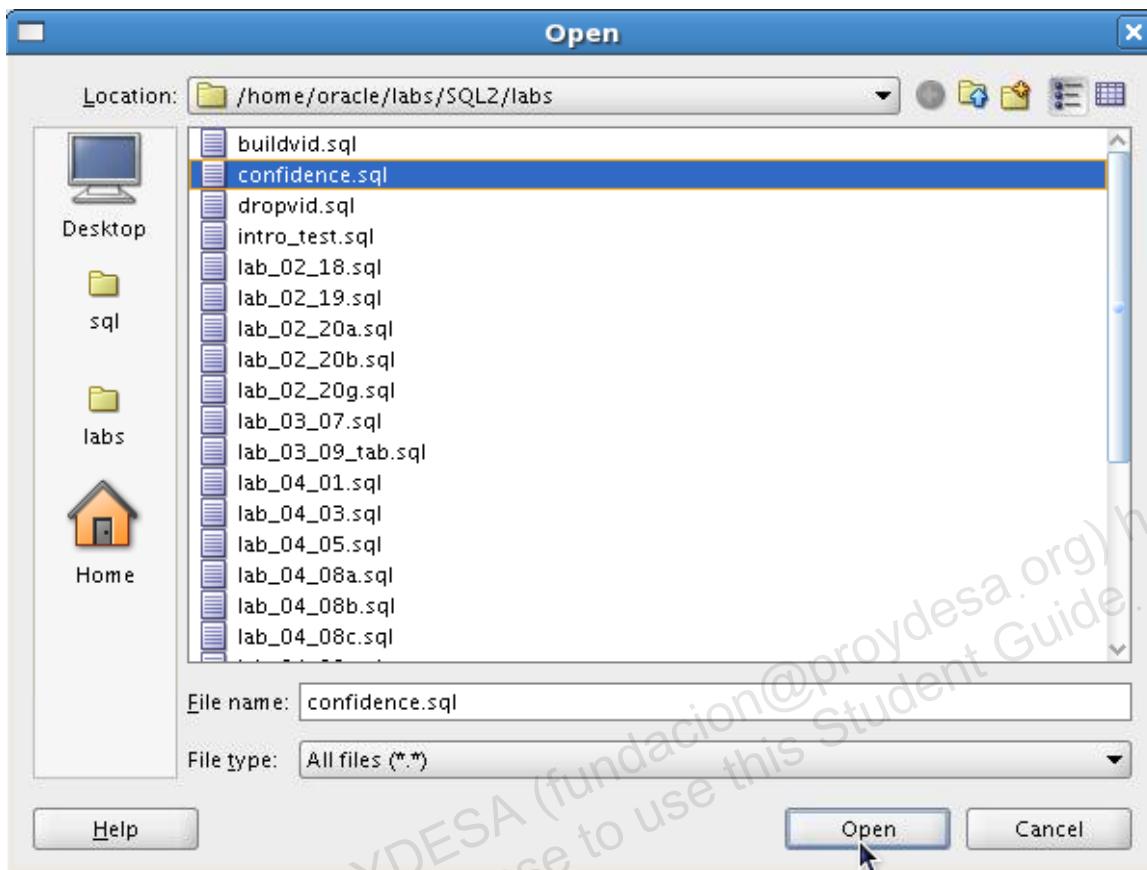


A continuación, haga clic en Save.

9. Abra y ejecute **confidence.sql** de la carpeta **/home/oracle/labs/SQL2/labs** y observe la salida.

Soluciones a la Práctica I-2: Uso de SQL Developer (continuación)

Abra el script `confidence.sql` mediante la opción de menú File > Open.



A continuación, pulse F5 para ejecutar el script.

A continuación se muestra el resultado esperado:

```
COUNT (*)
-----
8

1 rows selected

COUNT (*)
-----
107

1 rows selected

COUNT (*)
-----
25

1 rows selected
```

Soluciones a la Práctica I-2: Uso de SQL Developer (continuación)

```
COUNT (*)
-----
4

1 rows selected

COUNT (*)
-----
23

1 rows selected

COUNT (*)
-----
27

1 rows selected

COUNT (*)
-----
19

1 rows selected

COUNT (*)
-----
10

1 rows selected
```

Prácticas y Soluciones de la Lección 1

Práctica 1-1: Control del Acceso de los Usuarios

1. ¿Qué privilegio se le debe otorgar a un usuario para conectarse al servidor de Oracle?
¿Es un privilegio de sistema o un privilegio de objeto?

2. ¿Qué privilegio se le debe a otorgar a un usuario para crear tablas?

3. Si crea una tabla, ¿quién puede transferir privilegios a otros usuarios en la tabla?

4. Es el DBA. Crea usuarios que necesitan los mismos privilegios de sistema.
¿Qué debe utilizar para facilitar su trabajo?

5. ¿Qué comando utiliza para cambiar la contraseña?

6. User21 es el propietario de la tabla EMP y otorga el privilegio DELETE a User22 mediante la cláusula WITH GRANT OPTION. User22 le otorga el privilegio DELETE en EMP a User23. User21 ahora averigua que User23 tiene el privilegio y lo revoca de User22. ¿Qué usuario puede ahora suprimir de la tabla EMP?

7. Desea otorgar a SCOTT el privilegio para actualizar los datos en la tabla DEPARTMENTS. También desea activar a SCOTT para que pueda otorgar este privilegio a otros usuarios. ¿Qué comando utiliza?

Para completar la pregunta 8 y posteriores, debe conectarse a la base de datos mediante SQL Developer. Si todavía no está conectado, realice lo siguiente para hacerlo:

1. Haga clic en el ícono del escritorio de SQL Developer.
2. En el navegador de conexiones, utilice la cuenta `oraxxx` y la contraseña correspondiente proporcionada por el instructor para conectarse a la base de datos.
8. Otorgue el privilegio de consulta a otro usuario en la tabla. A continuación, verifique que el usuario puede utilizar el privilegio.
Nota: para este ejercicio, trabaje en equipo con otro grupo. Por ejemplo, si es el usuario `ora21`, trabaje en equipo con otro usuario `ora22`.
 - a. Otorgue un privilegio a otro usuario para ver los registros en la tabla REGIONS. Incluya una opción para este usuario para que además otorgue este privilegio a otros usuarios.
 - b. Pida al usuario que consulte la tabla REGIONS.
 - c. Pida al usuario que transfiera el privilegio de consulta a un tercer usuario (por ejemplo, `ora23`).

Práctica 1-1: Control del Acceso de los Usuarios (continuación)

- d. Recupere el privilegio del usuario que realiza el paso b.
- Nota:** cada equipo puede ejecutar los ejercicios 9 y 10 de forma independiente.
9. Otorgue los privilegios de manipulación de datos y consulta a otro usuario en la tabla COUNTRIES. Asegúrese de que el usuario no puede transferir estos privilegios a otros usuarios.
 10. Recupere los privilegios en la tabla COUNTRIES otorgados a otro usuario.
- Nota:** para realizar los ejercicios del 11 al 17, trabaje en equipo con otro grupo.
11. Otorgue acceso a otro usuario a la tabla DEPARTMENTS. Pida al usuario que le otorgue acceso de consulta a su tabla DEPARTMENTS.
 12. Consulte todas las filas de la tabla DEPARTMENTS.

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10 Administration		200	1700
2	20 Marketing		201	1800
3	30 Purchasing		114	1700
4	40 Human Resources		203	2400
5	50 Shipping		121	1500
6	60 IT		103	1400
7	70 Public Relations		204	2700
8	80 Sales		145	2500

- ...
13. Agregue una nueva fila a la tabla DEPARTMENTS. El equipo 1 debe agregar Education como departamento número 500. El equipo 2 debe agregar Human Resources como departamento número 510. Consulte la tabla del otro equipo.
 14. Cree un sinónimo para la tabla DEPARTMENTS del otro equipo.
 15. Consulte todas las filas de la tabla DEPARTMENTS del otro equipo mediante el sinónimo.

Team 1 SELECT statement results:

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
15	150 Shared Services		(null)	1700
16	160 Benefits		(null)	1700
17	170 Manufacturing		(null)	1700
18	180 Construction		(null)	1700
19	190 Contracting		(null)	1700
20	200 Operations		(null)	1700
21	210 IT Support		(null)	1700
22	220 NOC		(null)	1700
23	230 IT Helpdesk		(null)	1700
24	240 Government Sales		(null)	1700
25	250 Retail Sales		(null)	1700
26	260 Recruiting		(null)	1700
27	270 Payroll		(null)	1700
28	510 Human Resources		(null)	(null)

Práctica 1-1: Control del Acceso de los Usuarios (continuación)

Team 2 SELECT statement results:

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
16	160 Benefits	(null)	1700
17	170 Manufacturing	(null)	1700
18	180 Construction	(null)	1700
19	190 Contracting	(null)	1700
20	200 Operations	(null)	1700
21	210 IT Support	(null)	1700
22	220 NOC	(null)	1700
23	230 IT Helpdesk	(null)	1700
24	240 Government Sales	(null)	1700
25	250 Retail Sales	(null)	1700
26	260 Recruiting	(null)	1700
27	270 Payroll	(null)	1700
28	500 Education	(null)	(null)

16. Revoque el privilegio SELECT del otro equipo.
17. Elimine la fila insertada en la tabla DEPARTMENTS en el paso 13 y guarde los cambios.

Soluciones a la Práctica 1-1: Control del Acceso de los Usuarios

Para completar la pregunta 8 y posteriores, debe conectarse a la base de datos mediante SQL Developer.

1. ¿Qué privilegio se le debe otorgar a un usuario para conectarse al servidor de Oracle? ¿Es un privilegio de sistema o de objeto?

Privilegio de sistema CREATE SESSION

2. ¿Qué privilegio se le debe a otorgar a un usuario para crear tablas?

Privilegio CREATE TABLE

3. Si crea una tabla, ¿quién puede transferir privilegios a otros usuarios en la tabla?

Usted y cualquier usuario al que le haya otorgado dichos privilegios mediante WITH GRANT OPTION

4. Es el DBA. Cree usuarios que necesiten los mismos privilegios de sistema.

¿Que debe utilizar para facilitar su trabajo?

Crear un rol que contenga los privilegios de sistema y otorgar el rol a los usuarios.

5. ¿Qué comando utiliza para cambiar la contraseña?

Sentencia ALTER USER

6. User21 es el propietario de la tabla EMP y otorga los privilegios DELETE a User22 mediante la cláusula WITH GRANT OPTION. User22 le otorga los privilegios DELETE en EMP a User23. User21 ahora averigua que User23 tiene el privilegio y lo revoca de User22. ¿Qué usuario puede ahora suprimir datos de la tabla EMP?

Sólo User21

7. Desea otorgar a SCOTT el privilegio para actualizar los datos en la tabla DEPARTMENTS. También desea activar a SCOTT para que pueda otorgar este privilegio a otros usuarios. ¿Qué comando utiliza?

GRANT UPDATE ON departments TO scott WITH GRANT OPTION;

Soluciones a la Práctica 1-1: Control del Acceso de los Usuarios (continuación)

8. Otorgue el privilegio de consulta a otro usuario en la tabla. A continuación, verifique que el usuario puede utilizar el privilegio.
- Nota:** para este ejercicio, trabaje en equipo con otro grupo. Por ejemplo, si es el usuario ora21, trabaje en equipo con otro usuario ora22.
- Otorgue un privilegio a otro usuario para ver los registros en la tabla REGIONS. Incluya una opción para este usuario para que además otorgue este privilegio a otros usuarios.

El equipo 1 ejecuta esta sentencia:

```
GRANT select
ON regions
TO <team2_oraxx> WITH GRANT OPTION;
```

- Pida al usuario que consulte la tabla REGIONS.

El equipo 2 ejecuta esta sentencia:

```
SELECT * FROM <team1_oraxx>.regions;
```

- Pida al usuario que transfiera el privilegio de consulta a un tercer usuario (por ejemplo, ora23).

El equipo 2 ejecuta esta sentencia:

```
GRANT select
ON <team1_oraxx>.regions
TO <team3_oraxx>;
```

- Recupere el privilegio del usuario que realiza el paso b.

El equipo 1 ejecuta esta sentencia.

```
REVOKE select
ON regions
FROM <team2_oraxx>;
```

- Otorgue los privilegios de manipulación de datos y consulta a otro usuario en la tabla COUNTRIES. Asegúrese de que el usuario no puede transferir estos privilegios a otros usuarios.

El equipo 1 ejecuta esta sentencia.

```
GRANT select, update, insert
ON COUNTRIES
TO <team2_oraxx>;
```

Soluciones a la Práctica 1-1: Control del Acceso de los Usuarios (continuación)

10. Recupere los privilegios en la tabla COUNTRIES otorgados a otro usuario.

El equipo 1 ejecuta esta sentencia:

```
REVOKE select, update, insert ON COUNTRIES FROM <team2_oraxx>;
```

Nota: para realizar los ejercicios del 11 al 17, trabaje en equipo con otro grupo.

11. Otorgue acceso a otro usuario a la tabla DEPARTMENTS. Pida al usuario que le otorgue acceso de consulta a su tabla DEPARTMENTS.

El equipo 2 ejecuta la sentencia GRANT.

```
GRANT select
ON departments
TO <team1_oraxx>;
```

El equipo 1 ejecuta la sentencia GRANT.

```
GRANT select
ON departments
TO <team2_oraxx>;
```

Aquí, <equipo1_oraxx> es el nombre de usuario del equipo 1 <equipo2_oraxx> del equipo 2.

12. Consulte todas las filas de la tabla DEPARTMENTS.

```
SELECT *
FROM departments;
```

13. Agregue una nueva fila a la tabla DEPARTMENTS. El equipo 1 debe agregar Education como departamento número 500. El equipo 2 debe agregar Human Resources como departamento número 510. Consulte la tabla del otro equipo.

El equipo 1 ejecuta esta sentencia INSERT.

```
INSERT INTO departments (department_id,
department_name)
VALUES (500, 'Education');
COMMIT;
```

El equipo 2 ejecuta esta sentencia INSERT.

```
INSERT INTO departments (department_id,
department_name)
VALUES (510, 'Human Resources');
COMMIT;
```

Soluciones a la Práctica 1-1: Control del Acceso de los Usuarios (continuación)

14. Cree un sinónimo para la tabla DEPARTMENTS del otro equipo.

El equipo 1 crea un sinónimo denominado team2.

```
CREATE SYNONYM team2
    FOR <team2_oraxx>.DEPARTMENTS;
```

El equipo 2 crea un sinónimo denominado team1.

```
CREATE SYNONYM team1
    FOR <team1_oraxx>. DEPARTMENTS;
```

15. Consulte todas las filas de la tabla DEPARTMENTS del otro equipo mediante el sinónimo.

El equipo 1 ejecuta esta sentencia SELECT.

```
SELECT *
    FROM      team2;
```

El equipo 2 ejecuta esta sentencia SELECT.

```
SELECT *
    FROM      team1;
```

16. Revoque el privilegio SELECT del otro equipo.

El equipo 1 revoca el privilegio.

```
REVOKE select
    ON departments
    FROM <team2_oraxx>;
```

El equipo 2 revoca el privilegio.

```
REVOKE select
    ON departments
    FROM <team1_oraxx>;
```

Soluciones a la Práctica 1-1: Control del Acceso de los Usuarios (continuación)

17. Elimine la fila insertada en la tabla DEPARTMENTS en el paso 8 y guarde los cambios.

El equipo 1 ejecuta esta sentencia DELETE.

```
DELETE FROM departments  
WHERE department_id = 500;  
COMMIT;
```

El equipo 2 ejecuta esta sentencia DELETE.

```
DELETE FROM departments  
WHERE department_id = 510;  
COMMIT;
```

Prácticas y Soluciones de la Lección 2

Práctica 2-1: Gestión de Objetos de Esquema

En esta práctica, utilizará el comando `ALTER TABLE` para modificar columnas y agregar restricciones. Utilizará el comando `CREATE INDEX` para crear índices al crear una tabla, junto con el comando `CREATE TABLE`. Creará tablas externas.

1. Cree la tabla DEPT2 según el siguiente gráfico de instancias de tabla. Introduzca la sintaxis en la hoja de trabajo de SQL. A continuación, ejecute la sentencia para crear la tabla. Confirme que se ha creado la tabla.

Column Name	ID	NAME
Key Type		
Nulls/Unique		
FK Table		
FK Column		
Data type	NUMBER	VARCHAR2
Length	7	25

Name	Null	Type
ID		NUMBER(7)
NAME		VARCHAR2(25)
2 rows selected		

2. Rellene la tabla DEPT2 con datos de la tabla DEPARTMENTS. Incluya sólo las columnas que necesite.
3. Cree la tabla EMP2 según el siguiente gráfico de instancias de tabla. Introduzca la sintaxis en la hoja de trabajo de SQL. A continuación, ejecute la sentencia para crear la tabla. Confirme que se ha creado la tabla.

Column Name	ID	LAST_NAME	FIRST_NAME	DEPT_ID
Key Type				
Nulls/Unique				
FK Table				
FK Column				
Data type	NUMBER	VARCHAR2	VARCHAR2	NUMBER
Length	7	25	25	7

Práctica 2-1: Gestión de Objetos de Esquema (continuación)

Name	Null	Type
ID		NUMBER(7)
LAST_NAME		VARCHAR2(25)
FIRST_NAME		VARCHAR2(25)
DEPT_ID		NUMBER(7)
<i>4 rows selected</i>		

4. Modifique la tabla EMP2 para permitir apellidos de empleado más largos. Confirme la modificación.

Name	Null	Type
ID		NUMBER(7)
LAST_NAME		VARCHAR2(50)
FIRST_NAME		VARCHAR2(25)
DEPT_ID		NUMBER(7)
<i>4 rows selected</i>		

5. Cree la tabla EMPLOYEES2 según la estructura de la tabla EMPLOYEES. Incluya sólo las columnas EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY y DEPARTMENT_ID. Asigne a las columnas de la tabla los nombres ID, FIRST_NAME, LAST_NAME, SALARY y DEPT_ID, respectivamente.
6. Borre la tabla EMP2.
7. Consulte la papelera de reciclaje para ver si está la tabla.

ORIGINAL_NAME	OPERATION	DROPTIME
17 EMP_NEW_SAL	DROP	2009-05-22:14:44:15
18 EMP2	DROP	2009-05-22:14:57:57

8. Restaure la tabla EMP2 a un estado anterior a la sentencia DROP.

Name	Null	Type
ID		NUMBER(7)
LAST_NAME		VARCHAR2(50)
FIRST_NAME		VARCHAR2(25)
DEPT_ID		NUMBER(7)
<i>4 rows selected</i>		

9. Borre la columna FIRST_NAME de la tabla EMPLOYEES2. Confirme la modificación comprobando la descripción de la tabla.

Práctica 2-1: Gestión de Objetos de Esquema (continuación)

Name	Null	Type
ID		NUMBER(6)
LAST_NAME	NOT NULL	VARCHAR2(25)
SALARY		NUMBER(8, 2)
DEPT_ID		NUMBER(4)
4 rows selected		

10. En la tabla EMPLOYEES2, marque la columna DEPT_ID como UNUSED. Confirme la modificación comprobando la descripción de la tabla.

Name	Null	Type
ID		NUMBER(6)
LAST_NAME	NOT NULL	VARCHAR2(25)
SALARY		NUMBER(8, 2)
3 rows selected		

11. Borre la columna UNSUSED de la tabla EMPLOYEES2. Confirme la modificación comprobando la descripción de la tabla.
12. Agregue una restricción PRIMARY KEY de nivel de tabla a la tabla EMP2 en la columna ID. Se le debe asignar un nombre a la restricción en el momento de la creación. Asigne el nombre my_emp_id_pk a la restricción.
13. Cree una restricción PRIMARY KEY en la tabla DEPT2 mediante la columna ID. Se le debe asignar un nombre a la restricción en el momento de la creación. Asigne el nombre my_dept_id_pk a la restricción.
14. Agregue una referencia de clave ajena en la tabla EMP2 para garantizar que el empleado no se ha asignado a un departamento que no existe. Asigne el nombre my_emp_dept_id_fk a la restricción.
15. Borre la tabla EMP2 . Agregue una columna COMMISSION del tipo de dato NUMBER, precisión 2, escala 2. Agregue una restricción a la columna COMMISSION que garantice que el valor de comisión sea superior a cero.
16. Borre las tablas EMP2 y DEPT2 para que no se puedan restaurar. Verifique la papelera de reciclaje.
17. Cree la tabla DEPT_NAMED_INDEX según el siguiente gráfico de instancias de tabla. Asigne un nombre al índice para la columna PRIMARY KEY como DEPT_PK_IDX.

Column Name	Deptno	Dname
Primary Key	Yes	
Data Type	Number	VARCHAR2
Length	4	30

Práctica 2-1: Gestión de Objetos de Esquema (continuación)

18. Cree una tabla externa library_items_ext. Utilice el controlador de acceso ORACLE_LOADER.

Nota: El directorio emp_dir y el archivo library_items.dat ya se han creado para este ejercicio. library_items.dat tiene registros con el siguiente formato:

2354, 2264, 13.21, 150,
 2355, 2289, 46.23, 200,
 2355, 2264, 50.00, 100,

- Abra el archivo lab_02_18.sql. Observe el fragmento de código para crear la tabla externa library_items_ext. A continuación, sustituya <TODO1>, <TODO2>, <TODO3>, y <TODO4> por lo que corresponda y guarde el archivo como lab_02_18_soln.sql. Ejecute el script para crear la tabla externa.
- Consulte la tabla library_items_ext.

	CATEGOR...	BOO...	BOOK_P...	QUAN...
1	2354	2264	13.21	150
2	2355	2289	46.23	200
3	2355	2264	50	100

19. El departamento de HR necesita un informe de las direcciones de todos los departamentos. Cree una tabla externa como dept_add_ext mediante el controlador de acceso ORACLE_DATAPUMP. El informe debe mostrar el ID de ubicación, dirección, ciudad, estado o provincia y país en la salida. Utilice NATURAL JOIN para producir los resultados.

Nota: ya se ha creado el directorio emp_dir para este ejercicio.

- Abra el archivo lab_02_19.sql. Observe el fragmento de código para crear la tabla externa dept_add_ext. A continuación, sustituya <TODO1>, <TODO2> y <TODO3> por el código adecuado. Sustituya <oraxx_emp4.exp> y <oraxx_emp5.exp> por los nombres de archivo adecuados. Por ejemplo, si es el usuario ora21, sus nombres de archivos serán ora21_emp4.exp y ora21_emp5.exp. Guarde el script como lab_02_19_soln.sql.
- Ejecute el script lab_02_19_soln.sql para crear la tabla externa.
- Consulte la tabla dept_add_ext.

Práctica 2-1: Gestión de Objetos de Esquema (continuación)

	LOCAT...	STREET_ADDRESS	CITY	STATE_PROVINCE	COUNTRY_NAME
1	1000	1297 Via Cola di Rie	Roma	(null)	Italy
2	1100	93091 Calle della Testa	Venice	(null)	Italy
3	1200	2017 Shinjuku-ku	Tokyo	Tokyo Prefecture	Japan
4	1300	9450 Kamiya-cho	Hiroshima	(null)	Japan
5	1400	2014 Jabberwocky Rd	Southlake	Texas	United States of Amer
6	1500	2011 Interiors Blvd	South San Francisco	California	United States of Amer
7	1600	2007 Zagora St	South Brunswick	New Jersey	United States of Amer
8	1700	2004 Charade Rd	Seattle	Washington	United States of Amer

Nota: al realizar el paso anterior, se crean dos archivos `oraxx_emp4.exp` y `oraxx_emp5.exp` en el directorio por defecto `emp_dir`.

20. Cree la tabla `emp_books` y rellénela con datos. Defina la clave primaria como diferida y observe lo que ocurre al final de la transacción.

- a. Ejecute el archivo `lab_02_20_a.sql` para crear la tabla `emp_books`.
Observe que la clave primaria `emp_books_pk` no se ha creado como diferible.

```
create table succeeded.
```

- b. Ejecute el archivo `lab_02_20_b.sql` para crear la tabla `emp_books`.
¿Qué observa?

```
1 rows inserted

Error starting at line 2 in command:
insert into emp_books values(300,'Change Management')
Error report:
SQL Error: ORA-00001: unique constraint (ORA21.EMP_BOOKS_PK) violated
00001. 00000 -  "unique constraint (%s.%s) violated"
*Cause: An UPDATE or INSERT statement attempted to insert a duplicate key.
For Trusted Oracle configured in DBMS MAC mode, you may see
this message if a duplicate entry exists at a different level.
*Action: Either remove the unique restriction or do not insert the key.
```

- c. Defina la restricción `emp_books_pk` como diferida. ¿Qué observa?

```
Error starting at line 1 in command:
set constraint emp_books_pk deferred
Error report:
SQL Error: ORA-02447: cannot defer a constraint that is not deferrable
02447. 00000 -  "cannot defer a constraint that is not deferrable"
*Cause: An attempt was made to defer a nondeferrable constraint
*Action: Drop the constraint and create a new one that is deferrable
```

- d. Borre la restricción `emp_books_pk`.

```
alter table emp_books succeeded.
```

Práctica 2-1: Gestión de Objetos de Esquema (continuación)

- e. Modifique la definición de tabla emp_books para agregar la restricción emp_books_pk como diferida esta vez.

```
alter table emp_books succeeded.
```

- f. Defina la restricción emp_books_pk como diferida.

```
set constraint succeeded.
```

- g. Ejecute el archivo lab_02_20_g.sql para crear la tabla emp_books. ¿Qué observa?

```
1 rows inserted  
1 rows inserted  
1 rows inserted
```

- h. Confirme la transacción. ¿Qué observa?

```
Error report:  
SQL Error: ORA-02091: transaction rolled back  
ORA-00001: unique constraint (ORA21.EMP_BOOKS_PK) violated  
02091. 00000 - "transaction rolled back"  
*Cause: Also see error 2092. If the transaction is aborted at a remote  
site then you will only see 2091; if aborted at host then you will  
see 2092 and 2091.  
*Action: Add rollback segment and retry the transaction.
```

Soluciones a la Práctica 2-1: Gestión de Objetos de Esquema

1. Cree la tabla DEPT2 según el siguiente gráfico de instancias de tabla. Introduzca la sintaxis en la hoja de trabajo de SQL. A continuación, ejecute la sentencia para crear la tabla. Confirme que se ha creado la tabla.

Column Name	ID	NAME
Key Type		
Nulls/Unique		
FK Table		
FK Column		
Data type	NUMBER	VARCHAR2
Length	7	25

```
CREATE TABLE dept2
  (id          NUMBER (7) ,
   name        VARCHAR2 (25)) ;
```

```
DESCRIBE dept2
```

2. Rellene la tabla DEPT2 con datos de la tabla DEPARTMENTS. Incluya sólo las columnas que necesite.

```
INSERT INTO dept2
SELECT department_id, department_name
FROM departments;
```

3. Cree la tabla EMP2 según el siguiente gráfico de instancias de tabla. Introduzca la sintaxis en la hoja de trabajo de SQL. A continuación, ejecute la sentencia para crear la tabla. Confirme que se ha creado la tabla.

Column Name	ID	LAST_NAME	FIRST_NAME	DEPT_ID
Key Type				
Nulls/Unique				
FK Table				
FK Column				
Data type	NUMBER	VARCHAR2	VARCHAR2	NUMBER
Length	7	25	25	7

Soluciones a la Práctica 2-1: Gestión de Objetos de Esquema (continuación)

```
CREATE TABLE emp2
(id          NUMBER(7),
last_name    VARCHAR2(25),
first_name   VARCHAR2(25),
dept_id      NUMBER(7));

DESCRIBE emp2
```

4. Modifique la tabla EMP2 para permitir apellidos de empleado más largos. Confirme la modificación.

```
ALTER TABLE emp2
MODIFY (last_name    VARCHAR2(50));

DESCRIBE emp2
```

5. Cree la tabla EMPLOYEES2 según la estructura de la tabla EMPLOYEES. Incluya sólo las columnas EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY y DEPARTMENT_ID. Asigne a las columnas de la tabla los nombres ID, FIRST_NAME, LAST_NAME, SALARY y DEPT_ID, respectivamente.

```
CREATE TABLE employees2 AS
SELECT employee_id id, first_name, last_name, salary,
       department_id dept_id
  FROM employees;
```

6. Borre la tabla EMP2.

```
DROP TABLE emp2;
```

Soluciones a la Práctica 2-1: Gestión de Objetos de Esquema (continuación)

7. Consulte la papelera de reciclaje para ver si está la tabla.

```
SELECT original_name, operation, droptime  
FROM recyclebin;
```

8. Restaure la tabla EMP2 a un estado anterior a la sentencia DROP.

```
FLASHBACK TABLE emp2 TO BEFORE DROP;  
DESC emp2;
```

9. Borre la columna FIRST_NAME de la tabla EMPLOYEES2. Confirme la modificación comprobando la descripción de la tabla.

```
ALTER TABLE employees2  
DROP COLUMN first_name;  
  
DESCRIBE employees2
```

10. En la tabla EMPLOYEES2, marque la columna DEPT_ID como UNUSED. Confirme la modificación comprobando la descripción de la tabla.

```
ALTER TABLE employees2  
SET UNUSED (dept_id);  
  
DESCRIBE employees2
```

11. Borre todas las columnas UNUSED de la tabla EMPLOYEES2. Confirme la modificación comprobando la descripción de la tabla.

```
ALTER TABLE employees2  
DROP UNUSED COLUMNS;  
  
DESCRIBE employees2
```

Soluciones a la Práctica 2-1: Gestión de Objetos de Esquema (continuación)

12. Agregue una restricción PRIMARY KEY de nivel de tabla a la tabla EMP2 en la columna ID. Se le debe asignar un nombre a la restricción en el momento de la creación. Asigne el nombre my_emp_id_pk a la restricción.

```
ALTER TABLE emp2
ADD CONSTRAINT my_emp_id_pk PRIMARY KEY (id);
```

13. Cree una restricción PRIMARY KEY en la tabla DEPT2 mediante la columna ID. Se le debe asignar un nombre a la restricción en el momento de la creación. Asigne el nombre my_dept_id_pk a la restricción.

```
ALTER TABLE dept2
ADD CONSTRAINT my_dept_id_pk PRIMARY KEY(id);
```

14. Agregue una referencia de clave ajena en la tabla EMP2 para garantizar que el empleado no se ha asignado a un departamento que no existe. Asigne el nombre my_emp_dept_id_fk a la restricción.

```
ALTER TABLE emp2
ADD CONSTRAINT my_emp_dept_id_fk
FOREIGN KEY (dept_id) REFERENCES dept2(id);
```

15. Modifique la tabla EMP2. Agregue una columna COMMISSION del tipo de dato NUMBER, precisión 2, escala 2. Agregue una restricción a la columna COMMISSION que garantice que el valor de comisión sea superior a cero.

```
ALTER TABLE emp2
ADD commission NUMBER(2,2)
CONSTRAINT my_emp_comm_ck CHECK (commission > 0);
```

16. Borre las tablas EMP2 y DEPT2 para que no se puedan restaurar. Compruebe la papelera de reciclaje.

```
DROP TABLE emp2 PURGE;
DROP TABLE dept2 PURGE;

SELECT original_name, operation, droptime
FROM recyclebin;
```

17. Cree la tabla DEPT_NAMED_INDEX según el siguiente gráfico de instancias de tabla. Asigne un nombre al índice para la columna PRIMARY KEY como DEPT_PK_IDX.

Column Name	Deptno	Dname
Primary Key	Yes	
Data Type	Number	VARCHAR2
Length	4	30

Soluciones a la Práctica 2-1: Gestión de Objetos de Esquema (continuación)

```
CREATE TABLE DEPT_NAMED_INDEX
  (deptno NUMBER(4)
   PRIMARY KEY USING INDEX
   (CREATE INDEX dept_pk_idx ON
    DEPT_NAMED_INDEX(deptno)),
  dname VARCHAR2(30));
```

18. Cree una tabla externa library_items_ext. Utilice el controlador de acceso ORACLE_LOADER.

Nota: ya se han creado los directorios emp_dir y library_items.dat para este ejercicio. Asegúrese de que el archivo externo y la base de datos están en la misma máquina.

library_items.dat tiene registros con el siguiente formato:

```
2354, 2264, 13.21, 150,
2355, 2289, 46.23, 200,
2355, 2264, 50.00, 100,
```

- Abra el archivo lab_02_18.sql. Observe el fragmento de código para crear la tabla externa library_items_ext. A continuación, sustituya <TODO1>, <TODO2>, <TODO3> y <TODO4> por lo que corresponda y guarde el archivo como lab_02_18_soln.sql.
Ejecute el script para crear la tabla externa.

```
CREATE TABLE library_items_ext ( category_id number(12)
                                , book_id number(6)
                                , book_price number(8,2)
                                , quantity number(8)
                                )
ORGANIZATION EXTERNAL
  (TYPE ORACLE_LOADER
  DEFAULT DIRECTORY emp_dir
  ACCESS PARAMETERS (RECORDS DELIMITED BY NEWLINE
                     FIELDS TERMINATED BY ',')
  LOCATION ('library_items.dat')
  )
REJECT LIMIT UNLIMITED;
```

Soluciones a la Práctica 2-1: Gestión de Objetos de Esquema (continuación)

b. Consulte la tabla `library_items_ext`.

```
SELECT * FROM library_items_ext;
```

19. El departamento de HR necesita un informe de las direcciones de todos los departamentos. Cree una tabla externa como `dept_add_ext` mediante el controlador de acceso ORACLE_DATAPUMP. El informe debe mostrar el ID de ubicación, la dirección, la ciudad, el estado o la provincia y el país en la salida. Utilice NATURAL JOIN para producir los resultados.

Nota: ya se ha creado el directorio `emp_dir` para este ejercicio. Asegúrese de que el archivo externo y la base de datos están en la misma máquina.

- a. Abra el archivo `lab_02_19.sql`. Observe el fragmento de código para crear la tabla externa `dept_add_ext`. A continuación, sustituya `<TODO1>`, `<TODO2>` y `<TODO3>` por el código adecuado. Sustituya `<oraxx_emp4.exp>` y `<oraxx_emp5.exp>` por los nombres de archivo adecuados. Por ejemplo, si es el usuario ora21, sus nombres de archivos serán `ora21_emp4.exp` y `ora21_emp5.exp`. Guarde el script como `lab_02_19_soln.sql`.

```
CREATE TABLE dept_add_ext (location_id,
                           street_address, city,
                           state_province,
                           country_name)
ORGANIZATION EXTERNAL (
  TYPE ORACLE_DATAPUMP
  DEFAULT DIRECTORY emp_dir
  LOCATION ('oraxx_emp4.exp', 'oraxx_emp5.exp')
  PARALLEL
  AS
  SELECT location_id, street_address, city, state_province,
         country_name
  FROM locations
  NATURAL JOIN countries;
```

Nota: al realizar el paso anterior, se crean dos archivos `oraxx_emp4.exp` y `oraxx_emp5.exp` en el directorio por defecto `emp_dir`.

Ejecute el script `lab_02_19_soln.sql` para crear la tabla externa.

Soluciones a la Práctica 2-1: Gestión de Objetos de Esquema (continuación)

- b. Consulte la tabla dept_add_ext.

```
SELECT * FROM dept_add_ext;
```

20. Cree la tabla emp_books y rellénela con datos. Defina la clave primaria como diferida y observe lo que ocurre al final de la transacción.

- a. Ejecute el script lab_02_20a.sql para crear la tabla emp_books. Observe que la clave primaria emp_books_pk no se ha creado como diferible.

```
CREATE TABLE emp_books (book_id number,
                       title varchar2(20), CONSTRAINT
emp_books_pk PRIMARY KEY (book_id));
```

- b. Ejecute el archivo lab_02_20b.sql para llenar los datos en la tabla emp_books.

¿Qué observa?

```
INSERT INTO emp_books VALUES(300,'Organizations');
INSERT INTO emp_books VALUES(300,'Change Management');
```

Se inserta la primera fila. Sin embargo, aparece el error ora-00001 con la inserción de la segunda fila.

- c. Defina la restricción emp_books_pk como diferida. ¿Qué observa?

```
SET CONSTRAINT emp_books_pk DEFERRED;
```

Verá el siguiente error: “ORA-02447: Cannot defer a constraint that is not deferrable.”

- d. Borre la restricción emp_books_pk.

```
ALTER TABLE emp_books DROP CONSTRAINT emp_books_pk;
```

- e. Modifique la definición de tabla emp_books para agregar la restricción emp_books_pk como diferible esta vez.

```
ALTER TABLE emp_books ADD (CONSTRAINT emp_books_pk PRIMARY KEY
(book_id) DEFERRABLE);
```

- f. Defina las restricciones emp_books_pk como diferidas.

```
SET CONSTRAINT emp_books_pk DEFERRED;
```

Soluciones a la Práctica 2-1: Gestión de Objetos de Esquema (continuación)

- g. Ejecute el script `lab_02_20g.sql` para llenar los datos en la tabla `emp_books`.

¿Qué observa?

```
INSERT INTO emp_books VALUES(300,'Change Management');  
INSERT INTO emp_books VALUES (300,'Personality');  
INSERT INTO emp_books VALUES (350,'Creativity');
```

Verá que se han insertado todas las filas.

- h. Confirme la transacción. ¿Qué observa?

```
COMMIT;
```

Verá que se ha realizado un rollback de la transacción

Prácticas y Soluciones de la Lección 3

Práctica 3-1: Gestión de Objetos con Vistas de Diccionario de Datos

En esta práctica, consultará las vistas de diccionario para obtener información sobre el esquema.

1. Consulte la vista de diccionario de datos USER_TABLES para ver información sobre las tablas que posee.

TABLE_NAME
1 REGIONS
2 LOCATIONS
3 DEPARTMENTS
4 JOBS
5 EMPLOYEES
6 JOB_HISTORY
7 EMP_NEW_SAL
8 EMPLOYEES2
9 DEPT_NAMED_INDEX

...

2. Consulte la vista de diccionario de datos ALL_TABLES para ver información sobre las tablas a las que puede acceder. Excluya las tablas que posee.

Nota: es posible que su lista no coincida exactamente con la siguiente lista:

TABLE_NAME	OWNER
1 DUAL	SYS
2 SYSTEM_PRIVILEGE_MAP	SYS
3 TABLE_PRIVILEGE_MAP	SYS

...

98 PLAN_TABLE\$	SYS
99 WRI\$_ADV_ASA_RECO_DATA	SYS
100 PSTUBTBL	SYS

3. Para una tabla especificada, cree un archivo de comandos que registre los nombres de columna, tipos de dato y longitudes de los tipos de dato, así como de si se permiten valores nulos. Solicite al usuario que introduzca el nombre de la tabla. Otorgue los alias adecuados a las columnas DATA_PRECISION y DATA_SCALE. Guarde este script en un archivo con el nombre lab_03_01.sql.

Por ejemplo, si el usuario introduce DEPARTMENTS, se produce la siguiente salida:

Práctica 3-1: Gestión de Objetos con Vistas de Diccionario de Datos (continuación)

COLUMN_NAME	DATA_TYPE	DATA_LENGTH	PRECISION	SCALE	NULLABLE
1 DEPARTMENT_ID	NUMBER	22	4	0	N
2 DEPARTMENT_NAME	VARCHAR2	30	(null)	(null)	N
3 MANAGER_ID	NUMBER	22	6	0	Y
4 LOCATION_ID	NUMBER	22	4	0	Y

4. Cree un archivo de comandos que registre el nombre de columna, nombre de restricción, tipo de restricción, condición de búsqueda y estado de una tabla concreta. Debe unir las tablas USER_CONSTRAINTS y USER_CONS_COLUMNS para obtener toda esta información. Solicite al usuario que introduzca el nombre de la tabla. Guarde el script en un archivo con nombre lab_03_04.sql.
Por ejemplo, si el usuario introduce DEPARTMENTS, se obtienen los siguientes resultados de salida:

COLUMN_NAME	CONSTRAINT_NAME	CONSTR...	SEARCH_CONDITION	STATUS
1 DEPARTMENT_NAME	DEPT_NAME_NN	C	"DEPARTMENT_NAME" IS NOT ...	ENABLED
2 DEPARTMENT_ID	DEPT_ID_PK	P	(null)	ENABLED
3 LOCATION_ID	DEPT_LOC_FK	R	(null)	ENABLED
4 MANAGER_ID	DEPT_MGR_FK	R	(null)	ENABLED

5. Agregue un comentario a la tabla DEPARTMENTS. A continuación, consulte la vista USER_TAB_COMMENTS para verificar que está el comentario.

COMMENTS
1 Company department information including name, code, and location.

6. Cree un sinónimo para la tabla EMPLOYEES. Llámelo EMP. Y busque los nombres de todos los sinónimos del esquema.

SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK
1 TEAM2	ORA22	DEPARTMENTS	(null)
2 EMP	ORA21	EMPLOYEES	(null)

7. Ejecute lab_03_07.sql para crear la vista dept50 para este ejercicio. Debe determinar los nombres y definiciones de todas las vistas del esquema. Cree un informe que recupere la información sobre vistas: El nombre de vista y el texto de la vista USER_VIEWS del diccionario de datos.

Nota: EMP_DETAILS_VIEW se ha creado como parte del esquema.

Nota: puede ver la definición completa de la vista si utiliza Run Script (o pulsa F5) en SQL Developer. Si utiliza Execute Statement (o pulsa F9) en SQL Developer, desplácese horizontalmente en el panel de resultados. Si utiliza SQL*Plus, para ver más contenido de una columna LONG, utilice el comando SET LONG *n*, donde *n* es el valor del número de caracteres de la columna LONG que desea ver.

Práctica 3-1: Gestión de Objetos con Vistas de Diccionario de Datos (continuación)

VIEW_NAME	TEXT
1 DEPT50	SELECT employee_id empno, last_name employee, department_id deptno
2 EMP_DETAILS_VIEW	SELECT e.employee_id, e.job_id, e.manager_id, e.department_id, d.location_id,

8. Busque los nombres de las secuencias. Escriba una consulta en el archivo de comandos para visualizar la siguiente información sobre las secuencias: Nombre de secuencia, valor máximo, tamaño de incremento y último número. Asigne al script el nombre lab_03_08.sql. Ejecute la sentencia del archivo de comandos.

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
1 DEPARTMENTS_SEQ	9990	10	280
2 EMPLOYEES_SEQ	99999999999999999999999999999999	1	207
3 LOCATIONS_SEQ	9900	100	3300

Ejecute el script lab_03_09_tab.sql como requisito previo para los ejercicios 9 a 11. Asimismo, abra el script para copiar el código y pegarlo en la hoja de trabajo de SQL. A continuación, ejecute el script. Este script:

- Borra si hay tablas existentes DEPT2 y EMP2
- Crea las tablas DEPT2 y EMP2

Nota: en la práctica 2, ya debe haber borrado las tablas DEPT2 y EMP2 para que no se puedan restaurar.

9. Confirme que las tablas DEPT2 y EMP2 se han almacenado en el diccionario de datos.

TABLE_NAME
1 DEPT2
2 EMP2

10. Confirme que se han agregado las restricciones consultando la vista USER_CONSTRAINTS. Tenga en cuenta los tipos y nombre se las restricciones.

CONSTRAINT_NAME	CONSTRAINT_TYPE
1 MY_DEPT_ID_PK	P
2 MY_EMP_ID_PK	P
3 MY_EMP_DEPT_ID_FK	R

11. Muestre los tipos y nombres de objeto de la vista de diccionario de datos USER_OBJECTS de las tablas EMP2 y DEPT2.

Práctica 3-1: Gestión de Objetos con Vistas de Diccionario de Datos (continuación)

12. Cree la tabla SALES_DEPT según el siguiente gráfico de instancias de tabla. Asigne un nombre al índice para la columna PRIMARY KEY como SALES_PK_IDX. A continuación, consulte la vista de diccionario de datos para averiguar el nombre de índice, de tabla y si el índice es único.

Column Name	Team_Id	Location
Primary Key	Yes	
Data Type	Number	VARCHAR2
Length	3	30

INDEX_NAME	TABLE_NAME	UNIQUENESS
SALES_PK_IDX	SALES_DEPT	NONUNIQUE

Soluciones a la Práctica 3-1: Gestión de Objetos con Vistas de Diccionario de Datos

1. Consulte la vista de diccionario de datos para ver información sobre las tablas que posee.

```
SELECT table_name
      FROM user_tables;
```

2. Consulte la vista de diccionario de datos para ver información sobre las tablas a las que puede acceder. Excluya las tablas que posee.

```
SELECT table_name, owner
      FROM all_tables
     WHERE owner <>'ORAXXX';
```

3. Para una tabla especificada, cree un script que registre los nombres de columna, tipos de dato y longitudes de los tipos de dato, así como de si se permiten valores nulos.

Solicite al usuario que introduzca el nombre de la tabla. Otorgue los alias adecuados a las columnas DATA_PRECISION y DATA_SCALE. Guarde este script en un archivo con el nombre lab_03_01.sql.

```
SELECT column_name, data_type, data_length,
       data_precision PRECISION, data_scale SCALE, nullable
  FROM user_tab_columns
 WHERE table_name = UPPER('&tab_name');
```

Para probar, ejecute el script e introduzca DEPARTMENTS como nombre de la tabla.

4. Cree un archivo de comandos que registre el nombre de columna, nombre de restricción, tipo de restricción, condición de búsqueda y estado de una tabla concreta. Debe unir las tablas USER_CONSTRAINTS y USER_CONS_COLUMNS para obtener toda esta información. Solicite al usuario que introduzca el nombre de la tabla.

Guarde el script en un archivo con el nombre lab_03_04.sql.

```
SELECT ucc.column_name, uc.constraint_name,
       uc.constraint_type,
              uc.search_condition, uc.status
  FROM user_constraints uc JOIN user_cons_columns ucc
  ON uc.table_name = ucc.table_name
 AND uc.constraint_name = ucc.constraint_name
 AND uc.table_name = UPPER('&tab_name');
```

Para probar, ejecute el script e introduzca DEPARTMENTS como nombre de la tabla.

Soluciones a la Práctica 3-1: Gestión de Objetos con Vistas de Diccionario de Datos (continuación)

5. Agregue un comentario a la tabla DEPARTMENTS. A continuación, consulte la vista USER_TAB_COMMENTS para verificar que está el comentario.

```
COMMENT ON TABLE departments IS
  'Company department information including name, code, and
location.';

SELECT COMMENTS
FROM   user_tab_comments
WHERE  table_name = 'DEPARTMENTS';
```

6. Cree un sinónimo para la tabla EMPLOYEES. Llámelo EMP. A continuación, busque los nombres de todos los sinónimos del esquema.

```
CREATE SYNONYM emp FOR EMPLOYEES;
SELECT *
FROM   user_synonyms;
```

7. Ejecute lab_03_07.sql para crear la vista dept50 para este ejercicio. Debe determinar los nombres y las definiciones de todas las vistas del esquema. Cree un informe que recupere la información sobre vistas: El nombre de vista y el texto de la vista USER_VIEWS del diccionario de datos.

Nota: EMP_DETAILS_VIEW se ha creado como parte del esquema.

Nota: puede ver la definición completa de la vista si utiliza Run Script (o pulsa F5) en SQL Developer. Si utiliza Execute Statement (o pulsa F9) en SQL Developer, desplácese horizontalmente en el panel de resultados. Si utiliza SQL*Plus, para ver más contenido de una columna LONG, utilice el comando SET LONG *n*, donde *n* es el valor del número de caracteres de la columna LONG que desea ver.

```
SELECT view_name, text
FROM   user_views;
```

Soluciones a la Práctica 3-1: Gestión de Objetos con Vistas de Diccionario de Datos (continuación)

8. Busque los nombres de las secuencias. Escriba una consulta en el archivo de comandos para visualizar la siguiente información sobre las secuencias: Nombre de secuencia, valor máximo, tamaño de incremento y último número. Asigne al script el nombre lab_03_08.sql. Ejecute la sentencia del archivo de comandos.

```
SELECT sequence_name, max_value, increment_by, last_number
FROM user_sequences;
```

Ejecute el script lab_03_09_tab.sql como requisito previo para los ejercicios 9 a 11. Asimismo, abra el script para copiar el código y pegarlo en la hoja de trabajo de SQL. A continuación, ejecute el script. Este script:

- Borra las tablas DEPT2 y EMP2
- Crea las tablas DEPT2 y EMP2

Nota: en la práctica 2, ya debe haber borrado las tablas DEPT2 y EMP2 para que no se puedan restaurar.

9. Confirme que las tablas DEPT2 y EMP2 se han almacenado en el diccionario de datos.

```
SELECT table_name
FROM user_tables
WHERE table_name IN ('DEPT2', 'EMP2');
```

10. Consulte el diccionario de datos para averiguar los tipos y nombres de restricción de ambas tablas.

```
SELECT constraint_name, constraint_type
FROM user_constraints
WHERE table_name IN ('EMP2', 'DEPT2');
```

11. Consulte el diccionario de datos para mostrar los tipos y nombres de objeto de ambas tablas.

```
SELECT object_name, object_type
FROM user_objects
WHERE object_name LIKE 'EMP%'
OR object_name LIKE 'DEPT%';
```

Soluciones a la Práctica 3-1: Gestión de Objetos con Vistas de Diccionario de Datos (continuación)

12. Cree la tabla SALES_DEPT según el siguiente gráfico de instancias de tabla.

Asigne un nombre al índice para la columna PRIMARY KEY como SALES_PK_IDX.

A continuación, consulte la vista de diccionario de datos para averiguar el nombre de índice, de tabla y si el índice es único.

Column Name	Team_Id	Location
Primary Key	Yes	
Data Type	Number	VARCHAR2
Length	3	30

```

CREATE TABLE SALES_DEPT
  (team_id NUMBER(3)
   PRIMARY KEY USING INDEX
   (CREATE INDEX sales_pk_idx ON
    SALES_DEPT(team_id)),
   location VARCHAR2(30));

SELECT INDEX_NAME, TABLE_NAME, UNIQUENESS
FROM USER_INDEXES
WHERE TABLE_NAME = 'SALES_DEPT';

```

Prácticas y Soluciones de la Lección 4

Práctica 4-1: Manipulación de Juegos de Datos Grandes

En esta práctica, realizará operaciones INSERT y MERGE de varias tablas y realizará un seguimiento de las versiones de fila.

1. Ejecute el script lab_04_01.sql en la carpeta lab para crear la tabla SAL_HISTORY.
2. Muestre la estructura de la tabla SAL_HISTORY.

Name	Null	Type
EMPLOYEE_ID		NUMBER(6)
HIRE_DATE		DATE
SALARY		NUMBER(8, 2)

3 rows selected

3. Ejecute el script lab_04_01.sql en la carpeta lab para crear la tabla MGR_HISTORY.
4. Muestre la estructura de la tabla MGR_HISTORY.

Name	Null	Type
EMPLOYEE_ID		NUMBER(6)
MANAGER_ID		NUMBER(6)
SALARY		NUMBER(8, 2)

3 rows selected

5. Ejecute el script lab_04_05.sql en la carpeta lab para crear la tabla SPECIAL_SAL.
6. Muestre la estructura de la tabla SPECIAL_SAL.

Name	Null	Type
EMPLOYEE_ID		NUMBER(6)
SALARY		NUMBER(8, 2)

2 rows selected

7. a. Escriba una consulta para realizar lo siguiente:
 - Recupere los detalles como el ID de empleado, fecha de contratación, salario e ID de gestor de aquellos empleados cuyo ID de empleado sea inferior o igual a 125 de la tabla EMPLOYEES.
 - Si el salario es mayor de 20.000 dólares, inserte los detalles como el ID de empleado y salario en la tabla SPECIAL_SAL.

Práctica 4-1: Manipulación de Juegos de Datos Grandes (continuación)

- Inserte los detalles como el ID de empleado, la fecha de contratación y el salario en la tabla SAL_HISTORY.
- Inserte los detalles como el ID de empleado, el ID de gestor y el salario en la tabla MGR_HISTORY.

- b. Muestre los registros de la tabla SPECIAL_SAL.

	EMPLOYEE_ID	SALARY
1	100	24000

- c. Muestre los registros de la tabla SAL_HISTORY.

	EMPLOYEE_ID	HIRE_DATE	SALARY
1	101	21-SEP-89	17000
2	102	13-JAN-93	17000
3	103	03-JAN-90	9000
4	104	21-MAY-91	6000
5	105	25-JUN-97	4800
6	106	05-FEB-98	4800
7	107	07-FEB-99	4200

- d. Muestre los registros de la tabla MGR_HISTORY.

	EMPLOYEE_ID	MANAGER_ID	SALARY
1	101	100	17000
2	102	100	17000
3	103	102	9000
4	104	103	6000
5	105	103	4800
6	106	103	4800
7	107	103	4200

8. a. Ejecute el script lab_04_08_a.sql en la carpeta lab para crear la tabla SALES_WEEK_DATA.
- b. Ejecute el script lab_04_08b.sql en la carpeta lab para insertar registros en la tabla SALES_WEEK_DATA .

Práctica 4-1: Manipulación de Juegos de Datos Grandes (continuación)

- c. Visualice la estructura de la tabla SALES_WEEK_DATA.

Name	Null	Type
ID		NUMBER(6)
WEEK_ID		NUMBER(2)
QTY_MON		NUMBER(8,2)
QTY_TUE		NUMBER(8,2)
QTY_WED		NUMBER(8,2)
QTY_THUR		NUMBER(8,2)
QTY_FRI		NUMBER(8,2)

7 rows selected

- d. Visualice los registros de la tabla SALES_WEEK_DATA.

ID	WEEK_ID	QTY_MON	QTY_TUE	QTY_WED	QTY_THUR	QTY_FRI
1	200	6	2050	2200	1700	1200

- e. Ejecute el script lab_04_08_e.sql en la carpeta lab para crear la tabla EMP_SALES_INFO .

- f. Muestre la estructura de la tabla EMP_SALES_INFO .

Name	Null	Type
ID		NUMBER(6)
WEEK		NUMBER(2)
QTY_SALES		NUMBER(8,2)

3 rows selected

- g. Escriba una consulta para realizar lo siguiente:

- Recupere los detalles como ID, ID de semana, cantidad de ventas el lunes, el martes, el miércoles, el jueves y el viernes de la tabla SALES_WEEK_DATA .
- Cree una transformación para que cada registro recuperado de la tabla SALES_WEEK_DATA se convierta en varios registros de la tabla EMP_SALES_INFO .Indicación:
utilice la sentencia INSERT de giro.

- h. Visualice los registros de la tabla EMP_SALES_INFO .

ID	WEEK	QTY_SALES
1	200	6
		2050
2	200	6
		2200
3	200	6
		1700
4	200	6
		1200
5	200	6
		3000

Práctica 4-1: Manipulación de Juegos de Datos Grandes (continuación)

9. Ha almacenado los datos de empleados antiguos en un archivo plano denominado emp.data. Desea almacenar los nombres e ID de correo electrónico de todos los empleados, antiguos y presentes, de una tabla. Para ello, primero cree una tabla externa denominada EMP_DATA mediante el archivo de origen emp.dat en el directorio emp_dir. Utilice el script lab_04_09.sql para realizar esta tarea.
10. A continuación, ejecute el script lab_04_10.sql para crear la tabla EMP_HIST.
- Aumente el tamaño de la columna de correo electrónico a 45.
 - Fusione los datos en la tabla EMP_DATA creada en la práctica anterior en los datos de la tabla EMP_HIST. Suponga que los datos de la tabla externa EMP_DATA son los más actualizados. Si una fila de la tabla EMP_DATA coincide con la tabla EMP_HIST, actualice la columna de correo electrónico de la tabla EMP_HIST para hacer coincidir la fila de la tabla EMP_DATA. Si una fila de la tabla EMP_DATA, no coincide, insértela en la tabla EMP_HIST. Las filas se consideran coincidentes si los apellidos y nombres del empleado son idénticos.
 - Recupere las filas de EMP_HIST después de la fusión.

	FIRST_NAME	LAST_NAME	EMAIL
1	Ellen	Abel	EABEL
2	Sundar	Ande	SANDE
3	Mozhe	Atkinson	MATKINSO
4	David	Austin	DAUSTIN
5	Hermann	Baer	HBAER
6	Shelli	Baida	SBAIDA
7	Amit	Banda	ABANDA
8	Elizabeth	Bates	EBATES
9	Sarah	Bell	SBELL
10	David	Bernstein	DBERNSTE
11	Laura	Bissot	LBISSOT

11. Cree la tabla EMP3 mediante el script lab_04_11.sql. En la tabla EMP3, cambie el departamento para Kochhar a 60 y confírmelo los cambios. A continuación, cambie el departamento de Kochhar a 50 y confírmelo el cambio. Realice un seguimiento de los cambios en Kochhar mediante la función Row Versions.

	START_DATE	END_DATE	DEPARTMENT_ID
1	18-JUN-09 06.04.26.000000000 PM (null)		50
2	18-JUN-09 06.04.26.000000000 PM	18-JUN-09 06.04.26.000000000 PM	60
3	(null)	18-JUN-09 06.04.26.000000000 PM	90

Soluciones a la Práctica 4-1: Manipulación de Juegos de Datos Grandes

- Ejecute el script lab_04_01.sql en la carpeta lab para crear la tabla SAL_HISTORY.

- Muestre la estructura de la tabla SAL_HISTORY.

```
DESC sal_history
```

- Ejecute el script lab_04_01.sql en la carpeta lab para crear la tabla MGR_HISTORY.

- Muestre la estructura de la tabla MGR_HISTORY.

```
DESC mgr_history
```

- Ejecute el script lab_04_05.sql en la carpeta lab para crear la tabla SPECIAL_SAL.

- Muestre la estructura de la tabla SPECIAL_SAL.

```
DESC special_sal
```

- a. Escriba una consulta para realizar lo siguiente:

- Recupere los detalles como el ID de empleado, fecha de contratación, salario e ID de gestor de aquellos empleados cuyo ID de empleado sea inferior o igual a 125 de la tabla EMPLOYEES.
- Si el salario es mayor de 20.000 dólares, inserte los detalles como el ID de empleado y salario en la tabla SPECIAL_SAL.
- Inserte los detalles como el ID de empleado, fecha de contratación y salario en la tabla SAL_HISTORY.
- Inserte los detalles como el ID de empleado, ID de gestor y salario en la tabla MGR_HISTORY.

Soluciones a la Práctica 4-1: Manipulación de Juegos de Datos Grandes (continuación)

```
INSERT ALL
WHEN SAL > 20000 THEN
INTO special_sal VALUES (EMPID, SAL)
ELSE
INTO sal_history VALUES(EMPID, HIREDATE, SAL)
INTO mgr_history VALUES(EMPID, MGR, SAL)
SELECT employee_id EMPID, hire_date HIREDATE,
salary SAL, manager_id MGR
FROM employees
WHERE employee_id < 125;
```

- b. Muestre los registros de la tabla SPECIAL_SAL.

```
SELECT * FROM special_sal;
```

- c. Muestre los registros de la tabla SAL_HISTORY.

```
SELECT * FROM sal_history;
```

- d. Muestre los registros de la tabla MGR_HISTORY.

```
SELECT * FROM mgr_history;
```

8. a. Ejecute el script lab_04_08a.sql en la carpeta lab para crear la tabla SALES_WEEK_DATA.

non-translated version of the page watermark is present in the background of this section

b. Ejecute el script lab_04_08b.sql en la carpeta lab para insertar registros en la tabla

SALES_WEEK_DATA.

- c. Muestre la estructura de la tabla SALES_WEEK_DATA.

```
DESC sales_week_data
```

- d. Muestre los registros de la tabla SALES_WEEK_DATA.

```
SELECT * FROM SALES_WEEK_DATA;
```

Soluciones a la Práctica 4-1: Manipulación de Juegos de Datos Grandes (continuación)

- e. Ejecute el script lab_04_08_e.sql en la carpeta lab para crear la tabla EMP_SALES_INFO.
- f. Muestre la estructura de la tabla EMP_SALES_INFO.

```
DESC emp_sales_info
```

- g. Escriba una consulta para realizar lo siguiente:

- Recupere los detalles como ID de empleado, ID de semana, cantidad de ventas el lunes, el martes, el miércoles, el jueves, el viernes de la tabla SALES_WEEK_DATA.
- Cree una transformación para que cada registro recuperado de la tabla SALES_WEEK_DATA se convierta en varios registros de la tabla EMP_SALES_INFO.

Indicación: utilice la sentencia INSERT de giro.

```
INSERT ALL
  INTO emp_sales_info VALUES (id, week_id, QTY_MON)
  INTO emp_sales_info VALUES (id, week_id, QTY_TUE)
  INTO emp_sales_info VALUES (id, week_id, QTY_WED)
  INTO emp_sales_info VALUES (id, week_id, QTY_THUR)
  INTO emp_sales_info VALUES (id, week_id, QTY_FRI)
SELECT ID, week_id, QTY_MON, QTY_TUE, QTY_WED,
      QTY_THUR, QTY_FRI FROM sales_week_data;
```

- h. Muestre los registros de la tabla SALES_INFO.

```
SELECT * FROM emp_sales_info;
```

Soluciones a la Práctica 4-1: Manipulación de Juegos de Datos Grandes (continuación)

9. Ha almacenado los datos de empleados antiguos en un archivo plante denominado emp.data. Desea almacenar los nombres e ID de correo electrónico de todos los empleados, antiguos y presentes, de una tabla. Para ello, primero cree una tabla externa denominada EMP_DATA mediante el archivo de origen emp.dat en el directorio emp_dir. Puede utilizar el script lab_04_09.sql para realizar esta acción.

```
CREATE TABLE emp_data
  (first_name  VARCHAR2(20)
  , last_name   VARCHAR2(20)
  , email       VARCHAR2(30)
  )
ORGANIZATION EXTERNAL
(
  TYPE oracle_loader
  DEFAULT DIRECTORY emp_dir
  ACCESS PARAMETERS
  (
    RECORDS DELIMITED BY NEWLINE CHARACTERSET US7ASCII
    NOBADFILE
    NOLOGFILE
    FIELDS
    ( first_name POSITION ( 1:20) CHAR
    , last_name POSITION (22:41) CHAR
    , email      POSITION (43:72) CHAR )
  )
  LOCATION ('emp.dat') ) ;
```

10. A continuación, ejecute el script lab_04_10.sql para crear la tabla EMP_HIST.
- Aumente el tamaño de la columna de correo electrónico a 45.

```
ALTER TABLE emp_hist MODIFY email varchar(45);
```

- Fusione los datos en la tabla EMP_DATA creada en la práctica anterior en los datos de la tabla EMP_HIST. Suponga que los datos de la tabla externa EMP_DATA son los más actualizados. Si una fila de la tabla EMP_DATA coincide con la tabla EMP_HIST, actualice la columna de correo electrónico de la tabla EMP_HIST para hacer coincidir la fila de la tabla EMP_DATA. Si una fila de la tabla EMP_DATA, no coincide, insértela en la tabla EMP_HIST. Las filas se consideran coincidentes si los apellidos y nombres del empleado son idénticos.

```
MERGE INTO EMP_HIST f USING EMP_DATA h
ON (f.first_name = h.first_name
AND f.last_name = h.last_name)
```

Soluciones a la Práctica 4-1: Manipulación de Juegos de Datos Grandes (continuación)

```

WHEN MATCHED THEN
    UPDATE SET f.email = h.email
WHEN NOT MATCHED THEN
    INSERT (f.first_name
        , f.last_name
        , f.email)
VALUES (h.first_name
        , h.last_name
        , h.email);

```

- c. Recupere las filas de EMP_HIST después de la fusión.

```
SELECT * FROM emp_hist;
```

11. Cree la tabla EMP3 mediante el script lab_04_11.sql. En la tabla EMP3, cambie el departamento para Kochhar a 60 y confirme los cambios. A continuación, cambie el departamento de Kochhar a 50 y confirme el cambio. Realice un seguimiento de los cambios en Kochhar mediante la función Row Versions.

```

UPDATE emp3 SET department_id = 60
WHERE last_name = 'Kochhar';
COMMIT;
UPDATE emp3 SET department_id = 50
WHERE last_name = 'Kochhar';
COMMIT;

```

```

SELECT VERSIONS_STARTTIME "START_DATE",
       VERSIONS_ENDTIME "END_DATE",   DEPARTMENT_ID
  FROM EMP3
 WHERE VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE
   AND LAST_NAME = 'Kochhar';

```

Prácticas y Soluciones de la Lección 5

Práctica 5-1: Gestión de Datos Situados en Distintas Zonas Horarias

En esta práctica, mostrará los desplazamientos de zona horaria, CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP. También definirá las zonas horarias y utilizará la función EXTRACT.

1. Modifique la sesión para definir NLS_DATE_FORMAT en DD-MON-YYYY HH24:MI:SS.
2. a. Escriba consultas para mostrar los desplazamientos de zona horaria (TZ_OFFSET) de las siguientes zonas horarias.

- EE.UU./Nuevo Pacífico

<code>TZ_OFFSET('US/PACIFIC-NEW')</code>
1 -07:00

- Singapur

<code>TZ_OFFSET('SINGAPORE')</code>
1 +08:00

- Egipto

<code>TZ_OFFSET('EGYPT')</code>
1 +03:00

- b. Modifique la sesión para definir el valor del parámetro TIME_ZONE en el desplazamiento de zona horaria de EE.UU./Nuevo Pacífico.
- c. Muestre CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP para esta sesión.
- d. Modifique la sesión para definir el valor del parámetro TIME_ZONE en el desplazamiento de zona horaria de Singapur.
- e. Muestre CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP para esta sesión.

Nota: la salida puede ser diferente según la fecha de ejecución del comando.

<code>CURRENT_DATE</code>	<code>CURRENT_TIMESTAMP</code>	<code>LOCALTIMESTAMP</code>
1 23-JUN-2009 15:12:08	23-JUN-09 03.12.08.000000000 PM	+08:00 23-JUN-09 03.12.08.000000000 PM

Nota: observe que en la pregunta anterior CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP son sensibles a mayúsculas/minúsculas en la zona horaria de la sesión.

3. Escriba una consulta para mostrar DBTIMEZONE y SESSIONTIMEZONE.

<code>DBTIMEZONE</code>	<code>SESSIONTIMEZONE</code>
1 +00:00	+08:00

Práctica 5-1: Gestión de Datos Situados en Distintas Zonas Horarias (continuación)

4. Escriba una consulta para extraer YEAR de la columna HIRE_DATE de la tabla EMPLOYEES para aquellos empleados que trabajan en el departamento 80.

	LAST_NAME	EXTRACT(YEARFROMHIRE_DATE)
1	Russell	1996
2	Partners	1997
3	Errazuriz	1997
4	Cambrault	1999
5	Zlotkey	2000
6	Tucker	1997
7	Bernstein	1997

5. Modifique la sesión para definir NLS_DATE_FORMAT en DD-MON-YYYY.
 6. Examine y ejecute el script lab_05_06.sql para crear la tabla SAMPLE_DATES y rellenarla.

- a. Seleccione en la tabla y visualice los datos.

	DATE_COL
1	23-JUN-2009

- b. Modifique el tipo de dato de la columna DATE_COL y cámbielo a TIMESTAMP. Seleccione en la tabla para visualizar los datos.

	DATE_COL
1	23-JUN-09 02.14.52.000000000 PM

- c. Intente modificar el tipo de dato de la columna DATE_COL y cámbielo a TIMESTAMP WITH TIME ZONE. ¿Qué sucede?
7. Cree una consulta para recuperar los apellidos de la tabla EMPLOYEES y calcule el estado de revisión. Si el año de contratación fue 1998, muestre Needs Review para el estado de revisión; de lo contrario, muestre not this year! Asigne el nombre Review a la columna de estado de revisión. Ordene los resultados por la columna HIRE_DATE.
- Indicación:** utilice una expresión CASE con la función EXTRACT para calcular el estado de revisión.

Práctica 5-1: Gestión de Datos Situados en Distintas Zonas Horarias (continuación)

LAST_NAME	Review
King	not this year!
Whalen	not this year!
Kochhar	not this year!
Hunold	not this year!
Ernst	not this year!
De Haan	not this year!
Mavris	not this year!

...

8. Cree una consulta para imprimir los apellidos y el número de años de servicio de cada empleado. Si el empleado ha estado contratado durante cinco o más años, imprima 5 years of service. Si el empleado ha estado contratado durante 10 o más años, imprima 10 years of service. Si el empleado ha estado contratado durante 15 o más años, imprima 15 years of service. Si no coincide ninguna de estas condiciones, imprima maybe next year! Ordene los resultados por la columna HIRE_DATE. Utilice la tabla EMPLOYEES.

Indicación: utilice las expresiones CASE y TO_YMINTERVAL.

LAST_NAME	HIRE_DATE	SYSDATE	Awards
OConnell	21-JUN-1999	23-JUN-2009	10 years of service
Grant	13-JAN-2000	23-JUN-2009	5 years of service
Whalen	17-SEP-1987	23-JUN-2009	15 years of service
Hartstein	17-FEB-1996	23-JUN-2009	10 years of service
Fay	17-AUG-1997	23-JUN-2009	10 years of service
Mavris	07-JUN-1994	23-JUN-2009	15 years of service

...

Soluciones a la Práctica 5-1: Gestión de Datos Situados en Distintas Zonas Horarias

- Modifique la sesión para definir NLS_DATE_FORMAT en DD-MON-YYYY HH24:MI:SS.

```
ALTER SESSION SET NLS_DATE_FORMAT =
'DD-MON-YYYY HH24:MI:SS';
```

- a. Escriba consultas para mostrar los desplazamientos de zona horaria (TZ_OFFSET) de las siguientes zonas horarias: *EE.UU./Nuevo Pacífico, Singapur y Egipto*.

EE.UU./Nuevo Pacífico

```
SELECT TZ_OFFSET ('US/Pacific-New') from dual;
```

Singapur

```
SELECT TZ_OFFSET ('Singapore') from dual;
```

Egipto

```
SELECT TZ_OFFSET ('Egypt') from dual;
```

- b. Modifique la sesión para definir el valor del parámetro TIME_ZONE en el desplazamiento de zona horaria de EE.UU./Nuevo Pacífico.

```
ALTER SESSION SET TIME_ZONE = '-7:00';
```

- c. Muestre CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP para esta sesión.

Nota: la salida puede ser diferente según la fecha de ejecución del comando.

```
SELECT CURRENT_DATE, CURRENT_TIMESTAMP,
LOCALTIMESTAMP FROM DUAL;
```

- d. Modifique la sesión para definir el valor del parámetro TIME_ZONE en el desplazamiento de zona horaria de Singapur.

```
ALTER SESSION SET TIME_ZONE = '+8:00';
```

- e. Muestre CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP para esta sesión.

Nota: la salida puede ser diferente según la fecha de ejecución del comando.

```
SELECT CURRENT_DATE, CURRENT_TIMESTAMP,
LOCALTIMESTAMP FROM DUAL;
```

Soluciones a la Práctica 5-1: Gestión de Datos Situados en Distintas Zonas Horarias (continuación)

Nota: observe que en la pregunta anterior CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP son sensibles a mayúsculas/minúsculas en la zona horaria de la sesión.

- Escriba una consulta para mostrar DBTIMEZONE y SESSIONTIMEZONE.

```
SELECT DBTIMEZONE, SESSIONTIMEZONE
FROM DUAL;
```

- Escriba una consulta para extraer YEAR de la columna HIRE_DATE de la tabla EMPLOYEES para aquellos empleados que trabajan en el departamento 80.

```
SELECT last_name, EXTRACT (YEAR FROM HIRE_DATE)
FROM employees
WHERE department_id = 80;
```

- Modifique la sesión para definir NLS_DATE_FORMAT en DD-MON-YYYY.

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MON-YYYY';
```

- Examine y ejecute el script lab_05_06.sql para crear la tabla SAMPLE_DATES y llenarla.

- Seleccione en la tabla y visualice los datos.

```
SELECT * FROM sample_dates;
```

- Modifique el tipo de dato de la columna DATE_COL y cámbielo a TIMESTAMP. Seleccione en la tabla para visualizar los datos.

```
ALTER TABLE sample_dates MODIFY date_col TIMESTAMP;
SELECT * FROM sample_dates;
```

- Intente modificar el tipo de dato de la columna DATE_COL y cámbielo a TIMESTAMP WITH TIME ZONE. ¿Qué sucede?

```
ALTER TABLE sample_dates MODIFY date_col
TIMESTAMP WITH TIME ZONE;
```

Soluciones a la Práctica 5-1: Gestión de Datos Situados en Distintas Zonas Horarias (continuación)

No podrá cambiar el tipo de dato de la columna DATE_COL porque el servidor de Oracle no le permite convertir de TIMESTAMP a TIMESTAMP WITH TIMEZONE mediante la sentencia ALTER.

- Cree una consulta para recuperar los apellidos de la tabla EMPLOYEES y calcule el estado de revisión. Si el año de contratación fue 1998, muestre Needs Review para el estado de revisión; de lo contrario, muestre not this year! Asigne el nombre Review a la columna de estado de revisión. Ordene los resultados por la columna HIRE_DATE.

Indicación: utilice una expresión CASE con la función EXTRACT para calcular el estado de revisión.

```
SELECT e.last_name
      , (CASE extract(year from e.hire_date)
          WHEN 1998 THEN 'Needs Review'
          ELSE 'not this year!'
        END )           AS "Review "
   FROM employees e
  ORDER BY e.hire_date;
```

- Cree una consulta para imprimir los apellidos y el número de años de servicio de cada empleado. Si el empleado ha estado contratado durante cinco o más años, imprima 5 years of service. Si el empleado ha estado contratado durante cinco o más años, imprima 10 years of service. Si el empleado ha estado contratado durante 15 o más años, imprima 15 years of service. Si no coincide ninguna de estas condiciones, imprima maybe next year! Ordene los resultados por la columna HIRE_DATE. Utilice la tabla EMPLOYEES.

Indicación: utilice las expresiones CASE y TO_YMINTERVAL.

```
SELECT e.last_name, hire_date, sysdate,
      (CASE
        WHEN (sysdate -TO_YMINTERVAL('15-0'))>=
             hire_date THEN      '15 years of service'
        WHEN (sysdate -TO_YMINTERVAL('10-0'))>= hire_date
             THEN      '10 years of service'
        WHEN (sysdate - TO_YMINTERVAL('5-0'))>= hire_date
             THEN '5 years of service'
             ELSE 'maybe next year!'
       END) AS "Awards"
   FROM   employees e;
```

Prácticas y Soluciones de la Lección 6

Práctica 6-1: Recuperación de Datos mediante Subconsultas

En esta práctica, escribirá subconsultas de varias columnas y subconsultas escalares y correlacionadas. También resolverá problemas escribiendo la cláusula WITH.

1. Escriba una consulta para mostrar el apellido, el número de departamento y el salario de los empleados cuyo número de departamento y salario coincidan con el número de departamento y salario de los empleados que perciban una comisión.

	LAST_NAME	DEPARTMENT_ID	SALARY
1	Russell	80	14000
2	Partners	80	13500
3	Errazuriz	80	12000

2. Muestre el apellido, nombre de departamento y salario de los empleados cuyo salario y comisión coincidan con el salario y comisión de los empleados ubicados en el ID de ubicación 1700.

	LAST_NAME	DEPARTMENT_NAME	SALARY
1	Whalen	Administration	4400
2	Higgins	Accounting	12000
3	Greenberg	Finance	12000
4	Gietz	Accounting	8300

3. Cree una consulta para mostrar el apellido, fecha de contratación y salario de todos los empleados con el mismo salario y comisión que Kochhar.

Nota: no muestre a Kochhar en el juego de resultados.

	LAST_NAME	HIRE_DATE	SALARY
1	De Haan	13-JAN-1993	17000

4. Cree una consulta para mostrar los empleados que ganen un salario superior al salario de todos los directores de ventas (`JOB_ID = 'SA_MAN'`). Ordene los resultados de mayor a menor.

Práctica 6-1: Recuperación de Datos mediante Subconsultas (continuación)

	LAST_NAME	JOB_ID	SALARY
1	King	AD_PRES	24000
2	De Haan	AD_VP	17000
3	Kochhar	AD_VP	17000

5. Muestre los detalles como el ID de empleado, apellido e ID de departamento de aquellos empleados que viven en ciudades cuyos nombres comienzan por *T*.

	EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
1		202 Fay	20
2		201 Hartstein	20

6. Escriba una consulta para buscar todos los empleados que ganen más del salario medio en sus departamentos. Muestre el apellido, salario, ID de departamento y salario medio del departamento. Ordene el salario medio y redondee a dos decimales. Utilice los alias de las columnas recuperadas por la consulta como se muestra en la salida de ejemplo.

7. Busque todos los empleados que no sean supervisores.
a. En primer lugar, realice esta acción con el operador NOT EXISTS.

Práctica 6-1: Recuperación de Datos mediante Subconsultas (continuación)

	LAST_NAME
1	Abel
2	Ande
3	Atkinson
4	Austin
5	Baer
6	Baida

- b. ¿Se puede realizar con el operador NOT IN? ¿Cómo? o ¿por qué no?
8. Escriba una consulta para mostrar los apellidos de los empleados que ganan menos del salario medio en sus departamentos.

	LAST_NAME
1	Chen
2	Sciarra
3	Urman
4	Popp
5	Khoo
6	Baida

9. Escriba una consulta para mostrar los apellidos de los empleados que tiene uno o más compañeros en sus departamentos con las fechas de contratación más antiguas pero los salarios más altos.

	LAST_NAME
1	Vargas
2	Patel
3	Olson
4	Marlow
5	Landry
6	Perkins

10. Escriba una consulta para mostrar el ID de empleado, los apellidos y los nombres de departamento de todos los empleados.

Nota: utilice una subconsulta escalar para recuperar el nombre de departamento en la sentencia SELECT.

	EMPLOYEE_ID	LAST_NAME	DEPARTMENT
1	205	Higgins	Accounting
2	206	Gietz	Accounting
3	200	Whalen	Administration
4	100	King	Executive
5	101	Kochhar	Executive

...

**Práctica 6-1: Recuperación de Datos mediante Subconsultas
(continuación)**

105	196 Walsh	Shipping
106	197 Feeney	Shipping
107	178 Grant	(null)

11. Escriba una consulta para mostrar los nombres de departamentos de los departamentos cuyo costo del salario total esté por encima de un octavo (1/8) del costo total de toda la compañía. Utilice la cláusula WITH para escribir esta consulta. Asigne el nombre SUMMARY a la consulta.

	DEPARTMENT_NAME	DEPT_TOTAL
1	Sales	304500
2	Shipping	156400

Soluciones a la Práctica 6-1: Recuperación de Datos mediante Subconsultas

- Escriba una consulta para mostrar el apellido, el número de departamento y salario de los empleados cuyo número de departamento y salario coincidan con el número de departamento y salario de los empleados que perciban una comisión.

```
SELECT last_name, department_id, salary
FROM employees
WHERE (salary, department_id) IN
    (SELECT salary, department_id
     FROM employees
     WHERE commission_pct IS NOT NULL);
```

- Muestre el apellido, nombre de departamento y salario de los empleados cuyo salario y comisión coincidan con el salario y comisión de los empleados ubicados en el ID de ubicación 1700.

```
SELECT e.last_name, d.department_name, e.salary
FROM employees e, departments d
WHERE e.department_id = d.department_id
AND (salary, NVL(commission_pct,0)) IN
    (SELECT salary, NVL(commission_pct,0)
     FROM employees e, departments d
     WHERE e.department_id = d.department_id
     AND d.location_id = 1700);
```

- Cree una consulta para mostrar el apellido, fecha de contratación y salario de todos los empleados con el mismo salario y comisión que Kochhar.

Nota: no muestre a Kochhar en el juego de resultados.

```
SELECT last_name, hire_date, salary
FROM employees
WHERE (salary, NVL(commission_pct,0)) IN
    (SELECT salary, NVL(commission_pct,0)
     FROM employees
     WHERE last_name = 'Kochhar')
AND last_name != 'Kochhar';
```

- Cree una consulta para mostrar los empleados que ganen un salario superior al salario de todos los directores de ventas (JOB_ID = 'SA_MAN'). Ordene los resultado de mayor a menor.

```
SELECT last_name, job_id, salary
FROM employees
WHERE salary > ALL
    (SELECT salary
     FROM employees
     WHERE job_id = 'SA_MAN')
ORDER BY salary DESC;
```

Soluciones a la Práctica 6-1: Recuperación de Datos mediante Subconsultas (continuación)

5. Muestre los detalles como el ID de empleado, apellido e ID de departamento de aquellos empleados que viven en ciudades cuyos nombres comienzan por *T*.

```
SELECT employee_id, last_name, department_id
FROM employees
WHERE department_id IN (SELECT department_id
                          FROM departments
                          WHERE location_id IN
                                (SELECT location_id
                                 FROM locations
                                 WHERE city LIKE 'T%'));
```

6. Escriba una consulta para buscar todos los empleados que ganen más del salario medio en sus departamentos. Muestre el apellido, salario, ID de departamento y salario medio del departamento. Ordene por salario medio. Utilice los alias de las columnas recuperadas por la consulta como se muestra en la salida de ejemplo.

```
SELECT e.last_name ename, e.salary salary,
       e.department_id deptno, AVG(a.salary) dept_avg
  FROM employees e, employees a
 WHERE e.department_id = a.department_id
   AND e.salary > (SELECT AVG(salary)
                   FROM employees
                   WHERE department_id = e.department_id )
 GROUP BY e.last_name, e.salary, e.department_id
 ORDER BY AVG(a.salary);
```

Soluciones a la Práctica 6-1: Recuperación de Datos mediante Subconsultas (continuación)

7. Busque todos los empleados que no sean supervisores.
- En primer lugar, realice esta acción con el operador NOT EXISTS.

```
SELECT outer.last_name
FROM employees outer
WHERE NOT EXISTS (SELECT 'X'
                   FROM employees inner
                   WHERE inner.manager_id =
                         outer.employee_id);
```

- ¿Se puede realizar con el operador NOT IN? ¿Cómo? o ¿por qué no?

```
SELECT outer.last_name
FROM employees outer
WHERE outer.employee_id
NOT IN (SELECT inner.manager_id
        FROM employees inner);
```

Esta solución alternativa no es adecuada. La subconsulta selecciona un valor NULL, de manera que la consulta no devuelve ninguna fila. El motivo es que todas las condiciones que comparan un valor NULL tienen un resultado NULL. Siempre que los valores NULL formen parte del juego de valores, *no* utilice NOT IN como sustituto de NOT EXISTS.

8. Escriba una consulta para mostrar los apellidos de los empleados que ganan menos del salario medio en sus departamentos.

```
SELECT last_name
FROM employees outer
WHERE outer.salary < (SELECT AVG(inner.salary)
                      FROM employees inner
                      WHERE inner.department_id
                            = outer.department_id);
```

Soluciones a la Práctica 6-1: Recuperación de Datos mediante Subconsultas (continuación)

9. Escriba una consulta para mostrar los apellidos de los empleados que tiene uno o más compañeros en sus departamentos con las fechas de contratación más antiguas pero los salarios más altos.

```
SELECT last_name
  FROM employees outer
 WHERE EXISTS (SELECT 'X'
                  FROM employees inner
                 WHERE inner.department_id =
                      outer.department_id
                   AND inner.hire_date > outer.hire_date
                   AND inner.salary > outer.salary);
```

10. Escriba una consulta para mostrar el ID de empleado, apellidos y nombres de departamento de todos los empleados.

Nota: utilice una subconsulta escalar para recuperar el nombre de departamento en la sentencia SELECT.

```
SELECT employee_id, last_name,
       (SELECT department_name
          FROM departments d
         WHERE e.department_id =
              d.department_id) department
     FROM employees e
    ORDER BY department;
```

11. Escriba una consulta para mostrar los nombres de departamentos de los departamentos cuyo costo del salario total esté por encima de un octavo (1/8) del costo total de toda la compañía. Utilice la cláusula WITH para escribir esta consulta. Asigne el nombre SUMMARY a la consulta.

```
WITH
summary AS (
  SELECT d.department_name, SUM(e.salary) AS dept_total
    FROM employees e, departments d
   WHERE e.department_id = d.department_id
 GROUP BY d.department_name)
SELECT department_name, dept_total
  FROM summary
 WHERE dept_total > ( SELECT SUM(dept_total) * 1/8
                           FROM summary )
 ORDER BY dept_total DESC;
```

Prácticas y Soluciones de la Lección 7

Práctica 7-1: Soporte para Expresiones Regulares

En esta práctica, utilice funciones de expresiones regulares para buscar, sustituir y manipular datos. Cree también una nueva tabla CONTACTS y agregue una restricción CHECK a la columna p_number para garantizar que se introducen los números de teléfono en la base de datos en un formato estándar específico.

1. Escriba una consulta para buscar la tabla EMPLOYEES de todos los empleados cuyo nombre comience por “Ki” o “Ko.”

	FIRST_NAME	LAST_NAME
1	Janette	King
2	Steven	King
3	Neena	Kochhar

2. Cree una consulta que elimine los espacios en la columna STREET_ADDRESS de la tabla LOCATIONS de la visualización. Utilice “Street Address” como cabecera de columna.

	Street Address
1	1297ViaColadiRie
2	93091CalledellaTesta
3	2017Shinjuku-ku
4	9450Kamiya-cho
5	2014JabberwockyRd
6	2011InteriorBlvd
7	2007ZagoraSt

3. Cree un consulta que muestre “St” sustituido por “Street” en la columna STREET_ADDRESS de la tabla LOCATIONS. Tenga cuidado de que no aplicarlo a ninguna fila en la que ya aparezca “Street”. Muestre sólo las filas afectadas.

	REGEXP_REPLACE(STREET_ADDRESS,'ST\$','STREET')
1	2007 Zagora Street
2	6092 Boxwood Street
3	12-98 Victoria Street
4	8204 Arthur Street

4. Cree una tabla de contactos y agregue una restricción de control a la columna p_number para aplicar la siguiente máscara de formato para garantizar que se introducen los números de teléfono en la base de datos con el siguiente formato estándar. (XXX) XXX-XXXX. La tabla debe tener las siguientes columnas:

- l_name varchar2 (30)
- p_number varchar2 (30)

Práctica 7-1: Soporte para Expresiones Regulares (continuación)

5. Ejecute el script SQL `lab_07_05.sql` para insertar los siguientes siete números de teléfono en la tabla `contacts`. ¿Qué números se agregan?

l_name Column Value	p_number Column Value
NULL	'(650) 555-5555'
NULL	'(215) 555-3427'
NULL	'650 555-5555'
NULL	'650 555 5555'
NULL	'650-555-5555'
NULL	'(650) 555-5555'
NULL	' (650) 555-5555'

6. Escriba una consulta para buscar el número de incidencias del patrón de ADN `ctc` de la cadena `gtctcggtctcgttctgtctgtcgttctg`. Ignore la sensibilidad a mayúsculas/minúsculas.

COUNT_DNA
1 2

Soluciones a la Práctica 7-1: Soporte para Expresiones Regulares

- Escriba una consulta para buscar la tabla EMPLOYEES de todos los empleados cuyo nombre comience por “Ki” o “Ko.”

```
SELECT first_name, last_name
FROM employees
WHERE REGEXP_LIKE (last_name, '^K(i|o).');
```

- Cree una consulta que elimine los espacios en la columna STREET_ADDRESS de la tabla LOCATIONS de la visualización. Utilice “Street Address” como cabecera de columna.

```
SELECT regexp_replace(street_address, ' ', '') AS "Street
Address"
FROM locations;
```

- Cree una consulta que muestre “St” sustituido por “Street” en la columna STREET_ADDRESS de la tabla LOCATIONS. Tenga cuidado de que no aplicarlo a ninguna fila que en la que ya aparezca “Street”. Muestre sólo las filas afectadas.

```
SELECT regexp_replace(street_address, 'St$', 'Street')
FROM locations
WHERE regexp_like(street_address, 'St');
```

- Cree una tabla de contactos y agregue una restricción de control a la columna p_number para aplicar la siguiente máscara de formato para garantizar que se introducen los números de teléfono en la base de datos con el siguiente formato estándar. **(XXX) XXX-XXXX**. La tabla debe tener las siguientes columnas:

- l_name varchar2(30)
- p_number varchar2 (30)

```
CREATE TABLE contacts
(
    l_name      VARCHAR2(30),
    p_number    VARCHAR2(30)
        CONSTRAINT p_number_format
        CHECK ( REGEXP_LIKE ( p_number, '^\\(\\d{3}\\) \\d{3}-\\d{4}$' ) )
);
```

Soluciones a la Práctica 7-1: Soporte para Expresiones Regulares (continuación)

5. Ejecute el script SQL lab_07_05.sql para insertar los siguientes siete números de teléfono en la tabla de contactos. ¿Qué números se agregan?
Sólo las dos primeras sentencias INSERT utilizan un formato que se ajusta a la restricción c_contacts_pnf; las sentencias restantes generan errores de la restricción CHECK.
6. Escriba una consulta para buscar el número de incidencias del patrón de ADN ctc de la cadena gtctcgtctcggtctgtctgtcggtctg. Utilice el alias Count_DNA. Ignore la sensibilidad a mayúsculas/minúsculas. **Esta función, introducida en 11g versión 2, devuelve el número de veces que se encuentra una coincidencia de patrón en la cadena de entrada.**

```
SELECT REGEXP_COUNT('gtctcgtctcggtctgtctgtcggtctg', 'ctc')
AS Count_DNA
FROM dual;
```


B

Descripciones de las Tablas

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Descripción de Esquema

Descripción General

Los esquemas de ejemplo de Oracle Database ilustran una compañía de ejemplo que opera en todo el mundo para surtir pedidos de varios productos distintos. La compañía tiene tres divisiones:

- **Human Resources:** realiza un seguimiento de la información sobre los empleados y las instalaciones.
- **Order Entry:** realiza un seguimiento de los inventarios y las ventas de productos a través de distintos canales.
- **Sales History:** realiza un seguimiento de los datos estadísticos de negocio para facilitar las decisiones de negocio.

Cada una de estas divisiones se representa mediante un esquema. En este curso tendrá acceso a los objetos de todos los esquemas. No obstante, el énfasis de los ejemplos, demostraciones y prácticas está en el esquema de Human Resources (HR).

Todos los scripts necesarios para crear los esquemas de ejemplo están en la carpeta \$ORACLE_HOME/demo/schema/ folder.

Human Resources (HR)

Se trata del esquema que se utiliza en este curso. En los registros de recursos humanos (HR), cada empleado tiene un número de identificación, una dirección de correo electrónico, un código de identificación de cargo, un salario y un gestor. Algunos empleados ganan comisiones además de su salario.

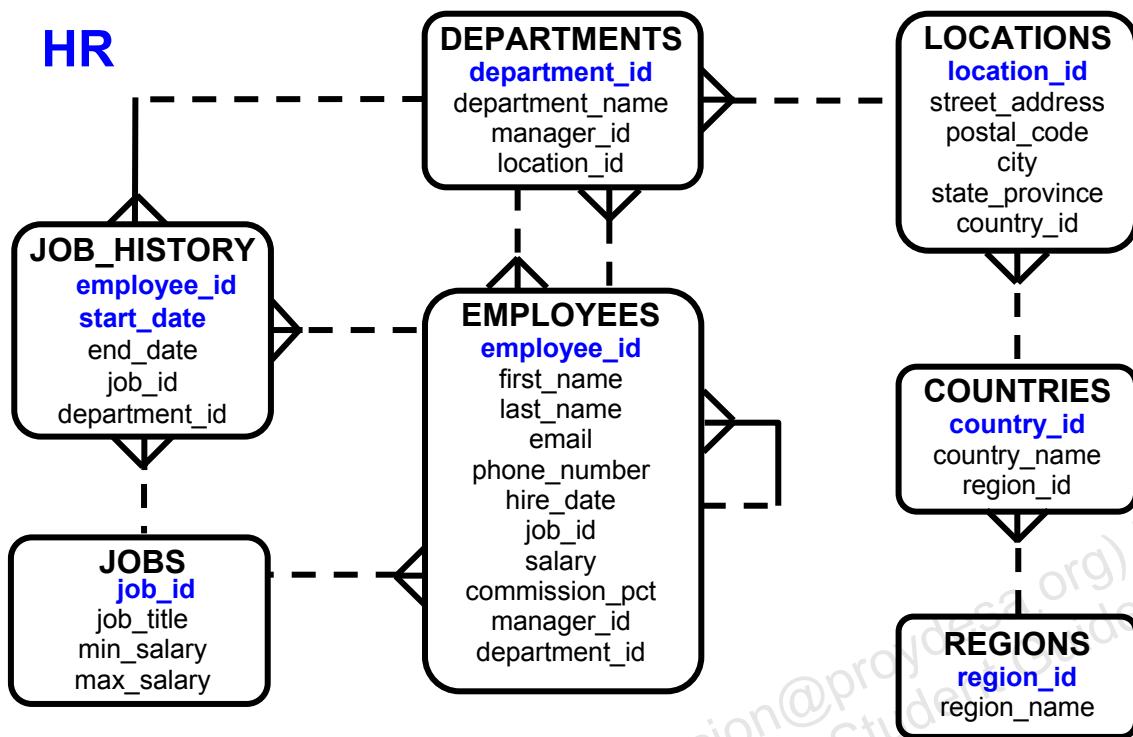
La compañía también registra información sobre cargos dentro de la organización. Cada cargo dispone de un código de identificación, un cargo y un rango de salario máximo y mínimo. Algunos empleados llevan bastante tiempo en la compañía y han desempeñado distintos trabajos en ella. Cuando se reasigna un empleado, se registran la duración del trabajo del empleado, el número de identificación del cargo y el departamento.

La compañía de ejemplo se extiende por distintas regiones, por lo que registra la ubicación de sus almacenes y departamentos. Cada empleado está asignado a un departamento y cada departamento se identifica mediante un número de departamento único o una abreviatura. Cada departamento está asociado a una ubicación y cada ubicación tiene una dirección completa que incluye el nombre de la calle, el código postal, la ciudad, el estado o la provincia y el código de país.

En las ubicaciones de los departamentos y almacenes, la compañía registra detalles como el nombre de país, el símbolo y nombre de divisa y la región en la que está ubicado geográficamente el país.

Diagrama de Relación de Entidades de HR

Unauthorized reproduction or distribution prohibited. Copyright© 2013, Oracle and/or its affiliates.



Descripciones de las Tablas de Human Resources (HR)

DESCRIBE countries

Name	Null	Type
COUNTRY_ID	NOT NULL	CHAR(2)
COUNTRY_NAME		VARCHAR2(40)
REGION_ID		NUMBER

SELECT * FROM countries;

	COUNTRY_ID	COUNTRY_NAME	REGION_ID
1	CA	Canada	2
2	DE	Germany	1
3	UK	United Kingdom	1
4	US	United States of America	2

Descripciones de las Tablas de Human Resources (HR) (continuación)

DESCRIBE departments

Name	Null	Type
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)

SELECT * FROM departments;

#	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500
6	90	Executive	100	1700
7	110	Accounting	205	1700
8	190	Contracting	(null)	1700

Descripciones de las Tablas de Human Resources (HR) (continuación)

DESCRIBE employees

Name	Null	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

SELECT * FROM employees;

#	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMIS...	MANAGER_ID	DEPARTMENT_ID
1	100	Steven	King	SKING	515.123.4567	17-JUN-87	AD_PRES	24000	(null)	(null)	90
2	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	17000	(null)	100	90
3	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	17000	(null)	100	90
4	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-90	IT_PROG	9000	(null)	102	60
5	104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-91	IT_PROG	6000	(null)	103	60
6	107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-99	IT_PROG	4200	(null)	103	60
7	124	Kevin	Moungos	KMOURGOS	650.123.5234	16-NOV-99	ST_MAN	5800	(null)	100	50
8	141	Trenna	Rajs	TRAJS	650.121.8009	17-OCT-95	ST_CLERK	3500	(null)	124	50
9	142	Curtis	Davies	CDAVIES	650.121.2994	29-JAN-97	ST_CLERK	3100	(null)	124	50
10	143	Randall	Matos	RMATOS	650.121.2874	15-MAR-98	ST_CLERK	2600	(null)	124	50
11	144	Peter	Vargas	PVARGAS	650.121.2004	09-JUL-98	ST_CLERK	2500	(null)	124	50
12	149	Eleni	Zlotkey	EZLOTKEY	011.44.1344.429018	29-JAN-00	SA_MAN	10500	0.2	100	80
13	174	Ellen	Abel	EABEL	011.44.1644.429267	11-MAY-96	SA REP	11000	0.3	149	80
14	176	Jonathon	Taylor	JTAYLOR	011.44.1644.429265	24-MAR-98	SA REP	8600	0.2	149	80
15	178	Kimberely	Grant	KGRANT	011.44.1644.429263	24-MAY-99	SA REP	7000	0.15	149	(null)
16	200	Jennifer	Whalen	JWHALEN	515.123.4444	17-SEP-87	AD_ASST	4400	(null)	101	10
17	201	Michael	Hartstein	MHARTSTE	515.123.5555	17-FEB-96	MKT_MAN	13000	(null)	100	20
18	202	Pat	Fay	PFAY	603.123.6666	17-AUG-97	MKT REP	6000	(null)	201	20
19	205	Shelley	Higgins	SHIGGINS	515.123.8080	07-JUN-94	AC_MGR	12000	(null)	101	110
20	206	William	Gietz	WGRIETZ	515.123.8181	07-JUN-94	AC ACC...	8300	(null)	205	110

Descripciones de las Tablas de Human Resources (HR) (continuación)

DESCRIBE job_history

DESCRIBE job_history		Null	Type
Name			
EMPLOYEE_ID		NOT NULL	NUMBER(6)
START_DATE		NOT NULL	DATE
END_DATE		NOT NULL	DATE
JOB_ID		NOT NULL	VARCHAR2(10)
DEPARTMENT_ID			NUMBER(4)

SELECT * FROM job_history

#	EMPLOYEE_ID	START_DATE	END_DATE	#	JOB_ID	#	DEPARTMENT_ID
1	102	13-JAN-93	24-JUL-98	IT_PROG			60
2	101	21-SEP-89	27-OCT-93	AC_ACCOUNT			110
3	101	28-OCT-93	15-MAR-97	AC_MGR			110
4	201	17-FEB-96	19-DEC-99	MK_REP			20
5	114	24-MAR-98	31-DEC-99	ST_CLERK			50
6	122	01-JAN-99	31-DEC-99	ST_CLERK			50
7	200	17-SEP-87	17-JUN-93	AD_ASST			90
8	176	24-MAR-98	31-DEC-98	SA_REP			80
9	176	01-JAN-99	31-DEC-99	SA_MAN			80
10	200	01-JUL-94	31-DEC-98	AC_ACCOUNT			90

Descripciones de las Tablas de Human Resources (HR) (continuación)

DESCRIBE jobs

Name	Null	Type
JOB_ID	NOT NULL	VARCHAR2(10)
JOB_TITLE	NOT NULL	VARCHAR2(35)
MIN_SALARY		NUMBER(6)
MAX_SALARY		NUMBER(6)

SELECT * FROM jobs

	JOB_ID	JOB_TITLE	MIN_SALARY	MAX_SALARY
1	AD_PRES	President	20000	40000
2	AD_VP	Administration Vice President	15000	30000
3	AD_ASST	Administration Assistant	3000	6000
4	AC_MGR	Accounting Manager	8200	16000
5	AC_ACCOUNT	Public Accountant	4200	9000
6	SA_MAN	Sales Manager	10000	20000
7	SA_REP	Sales Representative	6000	12000
8	ST_MAN	Stock Manager	5500	8500
9	ST_CLERK	Stock Clerk	2000	5000
10	IT_PROG	Programmer	4000	10000
11	MK_MAN	Marketing Manager	9000	15000
12	MK_REP	Marketing Representative	4000	9000

Descripciones de las Tablas de Human Resources (HR) (continuación)

DESCRIBE locations

Name	Null	Type
LOCATION_ID	NOT NULL	NUMBER(4)
STREET_ADDRESS		VARCHAR2(40)
POSTAL_CODE		VARCHAR2(12)
CITY	NOT NULL	VARCHAR2(30)
STATE_PROVINCE		VARCHAR2(25)
COUNTRY_ID		CHAR(2)

SELECT * FROM locations

#	LOCATION_ID	STREET_ADDRESS	POSTAL_CODE	CITY	STATE_PROVINCE	COUNTRY_ID
1	1400	2014 Jabberwocky Rd	26192	Southlake	Texas	US
2	1500	2011 Interiors Blvd	99236	South San Francisco	California	US
3	1700	2004 Charade Rd	98199	Seattle	Washington	US
4	1800	460 Bloor St. W.	ON M5S 1X8	Toronto	Ontario	CA
5	2500	Magdalen Centre, The Oxford Science Park	OX9 9ZB	Oxford	Oxford	UK

Descripciones de las Tablas de Human Resources (HR) (continuación)

DESCRIBE regions

Name	Null	Type
REGION_ID	NOT NULL	NUMBER
REGION_NAME		VARCHAR2(25)

SELECT * FROM regions

	REGION_ID	REGION_NAME
1	1	Europe
2	2	Americas
3	3	Asia
4	4	Middle East and Africa

C

Uso de SQL Developer

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

Al finalizar este apéndice, debería estar capacitado para lo siguiente:

- Mostrar las funciones clave de Oracle SQL Developer
- Identificar las opciones de menú de Oracle SQL Developer
- Crear una conexión a la base de datos
- Gestionar objetos de bases de datos
- Utilizar la hoja de trabajo de SQL
- Guardar y ejecutar scripts SQL
- Crear y guardar informes



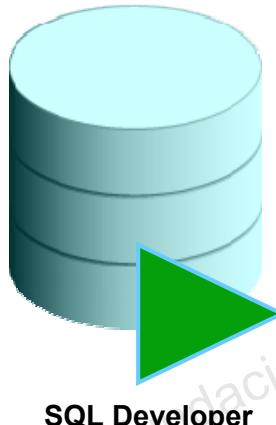
Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

En este apéndice, se ofrece una introducción a la herramienta gráfica SQL Developer. Aprenderá cómo utilizar SQL Developer para las tareas de desarrollo de la base de datos. Aprenderá cómo utilizar la hoja de trabajo de SQL para ejecutar sentencias y scripts SQL.

¿Qué es Oracle SQL Developer?

- Oracle SQL Developer es una herramienta gráfica que mejora la productividad y simplifica las tareas de desarrollo de la base de datos.
- Puede conectarse a cualquier esquema de Oracle Database de destino mediante la autenticación estándar de Oracle Database.



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

¿Qué es Oracle SQL Developer?

Oracle SQL Developer es una herramienta gráfica gratuita diseñada para mejorar la productividad y simplificar el desarrollo de las tareas diarias de la base de datos. Con sólo unos clics, puede crear y depurar fácilmente los procedimientos almacenados, probar sentencias SQL y visualizar los planes del optimizador.

SQL Developer, la herramienta visual para el desarrollo de la base de datos, simplifica las siguientes tareas:

- Examen y gestión de objetos de la base de datos
- Ejecución de sentencias y scripts SQL
- Edición y depuración de sentencias PL/SQL
- Creación de informes

Puede conectarse a cualquier esquema de Oracle Database de destino mediante la autenticación estándar de Oracle Database. Una vez conectado, puede realizar operaciones en los objetos de la base de datos.

SQL Developer versión 1.2 está estrechamente integrado con *Developer Migration Workbench*, que proporciona a los usuarios un punto único para examinar los objetos y datos de base de datos en bases de datos de terceros y migrar desde estas bases de datos a Oracle. También se puede conectar a los esquemas para las bases de datos de terceros (no de Oracle) seleccionadas, como MySQL, Microsoft SQL Server y Microsoft Access, y puede ver los metadatos y datos de estas bases de datos. Además, SQL Developer incluye soporte para Oracle Application Express 3.0.1 (Oracle APEX).

Especificaciones de SQL Developer

- Incluido con Oracle Database 11g Versión 2
- Desarrollado en Java
- Soporta plataformas Windows, Linux y Mac OS X
- Conectividad por defecto mediante el controlador Java Database Connectivity (JDBC) Thin
- Se conecta a Oracle Database versión 9.2.0.1 y posteriores
- Descarga gratuita desde el siguiente enlace:
 - http://www.oracle.com/technology/products/database/sql_developer/index.html



Copyright © 2010, Oracle. Todos los derechos reservados.

Especificaciones de SQL Developer

Oracle SQL Developer 1.5 se incluye con Oracle Database 11g Versión 2. SQL Developer está desarrollado en Java aprovechando el entorno de desarrollo integrado (IDE) de Oracle JDeveloper. Por lo tanto, es una herramienta entre plataformas. La herramienta se ejecuta en plataformas Windows, Linux y Mac OS (sistema operativo) X.

La conectividad por defecto a la base de datos se realiza a través del controlador JDBC Thin; por lo tanto, no es necesario el directorio raíz de Oracle. SQL Developer no necesita un instalador y simplemente necesita descomprimir los archivos descargados. Con SQL Developer, los usuarios se pueden conectar a Oracle Databases 9.2.0.1 y versiones posteriores y a todas las ediciones de Oracle Database, incluida Express Edition.

Nota

Para versiones de Oracle Database anteriores a Oracle Database 11g Versión 2, tendrá que descargar e instalar SQL Developer. SQL Developer 1.5 se puede descargar gratuitamente desde el siguiente enlace:

http://www.oracle.com/technology/products/database/sql_developer/index.html.

Para obtener más instrucciones sobre cómo instalar SQL Developer, puede visitar el siguiente enlace:

http://download.oracle.com/docs/cd/E12151_01/index.htm



Interfaz de SQL Developer 1.5

La interfaz de SQL Developer 1.5 contiene tres separadores de navegación principales, de izquierda a derecha:

- **Separador Connections:** si utiliza este separador, puede examinar los objetos y usuarios de la base de datos a los que tiene acceso.
- **Separador Files:** identificado mediante el icono de carpeta Files, este separador permite acceder a los archivos desde la máquina local sin tener que utilizar el menú File > Open.
- **Separador Reports:** identificado mediante el icono Reports, este separador permite ejecutar los informes predefinidos o crear y aplicar sus propios informes.

Navegación General y Uso

SQL Developer utiliza la parte izquierda de la navegación para buscar y seleccionar objetos, y la parte derecha para mostrar información sobre los objetos seleccionados. También puede personalizar muchos aspectos de la apariencia y del comportamiento de SQL Developer mediante la definición de preferencias.

Nota: debe definir al menos una conexión para conectarse a un esquema de base de datos y emitir consultas SQL o ejecutar procedimientos/funciones.

Interfaz de SQL Developer 1.5 (continuación)

Menús

Los siguientes menús contienen entradas estándar, además de entradas de funciones específicas de SQL Developer.

- **Vista:** contiene opciones que afectan a lo que se muestra en la interfaz de SQL Developer.
- **Navigate:** contiene opciones para acceder a los diferentes paneles y ejecutar los subprogramas.
- **Run:** contiene las opciones Run File y Execution Profile, que son relevantes en la selección de una función o un procedimiento, además de las opciones de depuración.
- **Source:** contiene opciones para utilizarlas al editar funciones y procedimientos.
- **Versioning:** proporciona soporte integrado para los siguientes sistemas de control de versiones y orígenes: Concurrent Versions System (CVS) y Subversion.
- **Migration:** contiene opciones relacionadas con la migración de bases de datos de terceros a Oracle.
- **Tools:** llama a las herramientas de SQL Developer, como SQL*Plus, Preferences y SQL Worksheet.

Nota: el menú Run también contiene opciones relevantes para la selección de una función o un procedimiento para depuración. Son las mismas opciones que las del menú Debug de la versión 1.2.

Creación de una Conexión a la Base Datos

- Debe tener al menos una conexión a la base de datos para utilizar SQL Developer.
- Puede crear y probar conexiones para varios:
 - Bases de Datos
 - Esquemas
- SQL Developer importa automáticamente las conexiones definidas en el archivo `tnsnames.ora` al sistema.
- Puede exportar conexiones a un archivo Extensible Markup Language (XML).
- Cada conexión a la base de datos adicional creada se muestra en la jerarquía del navegador de conexiones.



Copyright © 2010, Oracle. Todos los derechos reservados.

Creación de una Conexión a la Base Datos

Una conexión es un objeto de SQL Developer que especifica la información necesaria para conectarse a una base de datos concreta como usuario específico de dicha base de datos. Para utilizar SQL Developer, debe tener al menos una conexión a la base de datos, que puede ser existente, creada o importada.

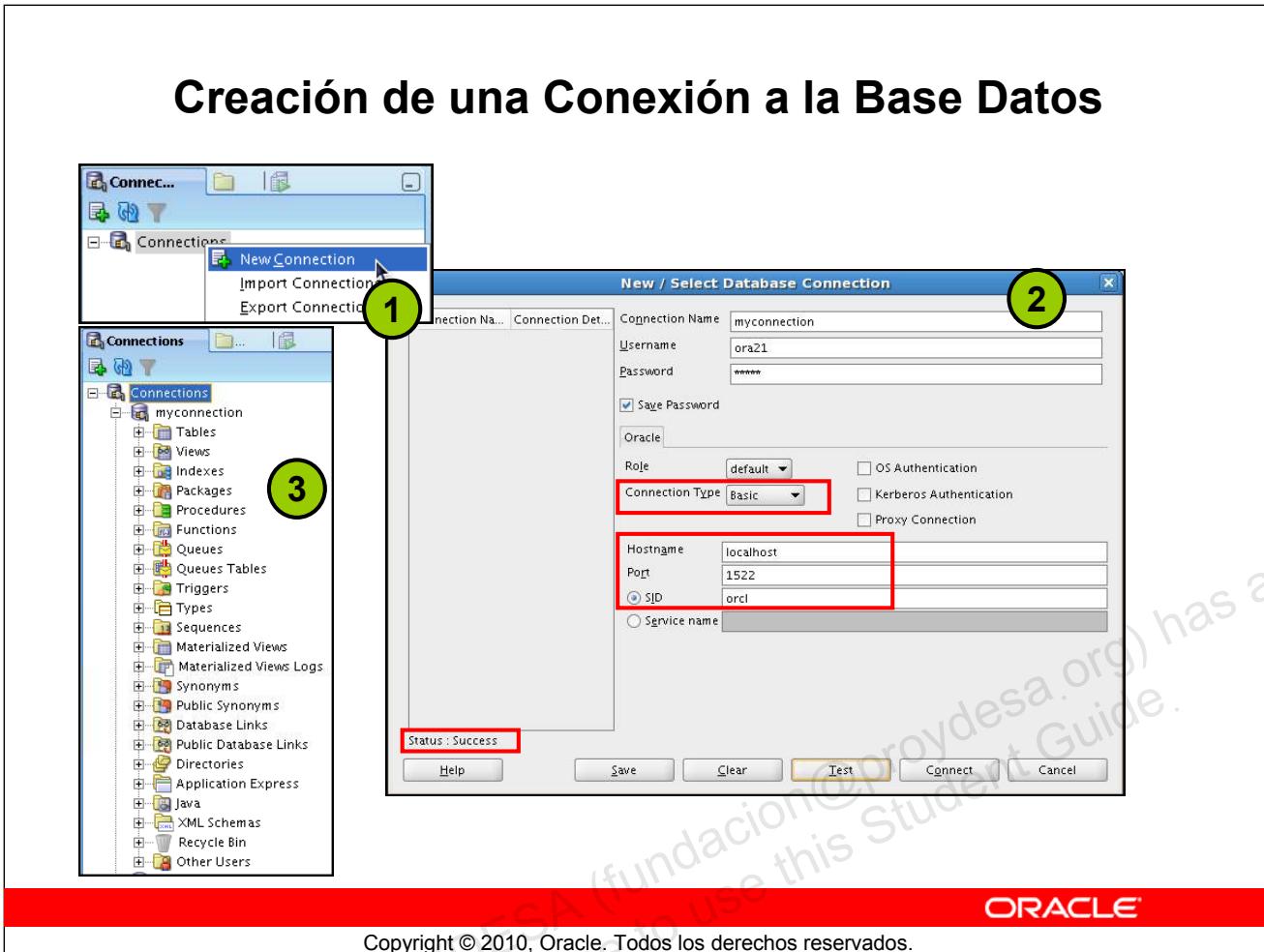
Puede crear y probar conexiones para varias bases de datos y esquemas.

Por defecto, el archivo `tnsnames.ora` está ubicado en el directorio `$ORACLE_HOME/network/admin`, pero también puede estar en el directorio especificado por la variable de entorno `TNS_ADMIN` o el valor de registro. Al iniciar SQL Developer, cuando se muestra el cuadro de diálogo Database Connections, SQL Developer importa automáticamente las conexiones definidas en el archivo `tnsnames.ora` al sistema.

Nota: en los sistemas Windows, si existe el archivo `tnsnames.ora`, pero SQL Developer no está utilizando sus conexiones, defina `TNS_ADMIN` como una variable de entorno del sistema.

Puede exportar conexiones a un archivo XML para volver a utilizarlas más tarde.

Puede crear conexiones adicionales como usuarios diferentes a la misma base de datos o conectarse a diferentes bases de datos.



Creación de una Conexión a la Base Datos (continuación)

Para crear una conexión a la base de datos, realice los siguientes pasos:

1. En la página con separadores Connections, haga clic con el botón derecho del mouse en **Connections** y seleccione **New Connection**.
2. En la ventana New/Select Database Connection, introduzca el nombre de la conexión. Introduzca el nombre de usuario y la contraseña del esquema al que desea conectarse.
 - a) En la lista desplegable Role, puede seleccionar el valor por defecto o SYSDBA (debe seleccionar SYSDBA para el usuario sys o cualquier usuario con privilegios de administrador de la base de datos).
 - b) Puede seleccionar el tipo de conexión como:
 - **Básicas:** en este tipo, introduzca el nombre del host y el SID de la base de datos a la que desea conectarse. El puerto ya está definido en 1521. También puede introducir el nombre del servicio directamente si utiliza una conexión a la base de datos remota.
 - **TNS:** puede seleccionar cualquiera de los alias de base de datos importados del archivo tnsnames.ora.
 - **LDAP:** puede consultar los servicios de base de datos en Oracle Internet Directory, que es un componente de Oracle Identity Management.
 - **Advanced:** puede definir una URL de JDBC personalizada para conectarse a la base de datos.
 - c) Haga clic en Test para asegurarse de que la conexión se ha definido correctamente.
 - d) Haga clic en Connect.

Creación de una Conexión a la Base Datos (continuación)

Si activa la casilla de control Save Password, la contraseña se guarda en un archivo XML. De este modo, cuando cierre la conexión a SQL Developer y la vuelva a abrir, no se le pedirá la contraseña.

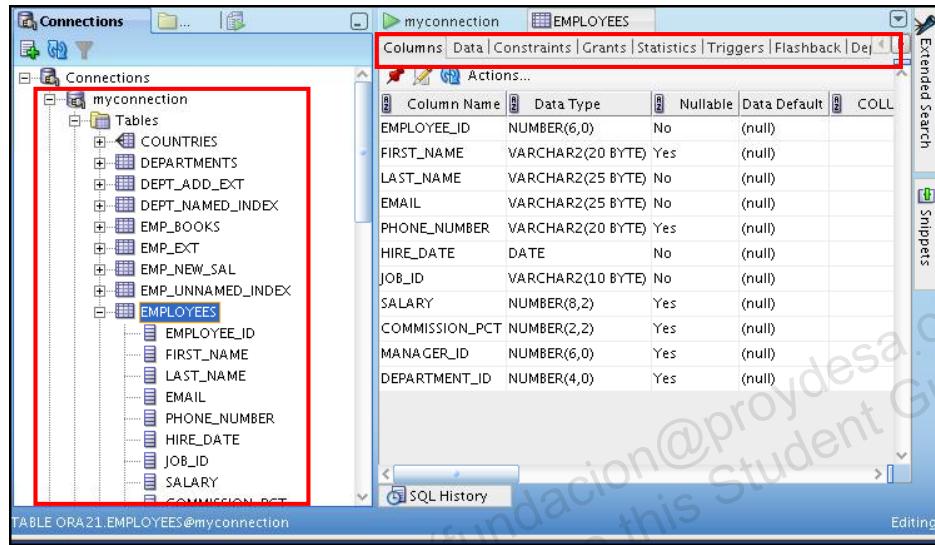
3. La conexión se agrega al navegador de conexiones. Puede ampliar la conexión para ver los objetos de base de datos y las definiciones de objetos, por ejemplo, dependencias, detalles, estadísticas, etc.

Nota: en la misma ventana New>Select Database Connection, puede definir conexiones a orígenes de datos no Oracle mediante los separadores Access, MySQL y SQL Server. Sin embargo, se trata de conexiones de sólo lectura que permiten examinar los objetos y datos de dicho origen de datos.

Examen de Objetos de Bases de Datos

Utilizar el navegador de conexiones para:

- Examinar los objetos de un esquema de base de datos
- Revisar las definiciones de objetos de forma rápida



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Examen de Objetos de Bases de Datos

Una vez creada la conexión a la base de datos, puede utilizar el navegador de conexiones para examinar los objetos de un esquema de base de datos entre los que se incluyen tablas, vistas, índices, paquetes, procedimientos, disparadores y tipos.

Puede ver la definición de los objetos desglosados en separadores de información que se transfieren al diccionario de datos. Por ejemplo, si selecciona una tabla en el navegador, se muestran los detalles sobre las columnas, restricciones, permisos, estadísticas, disparadores, etc., en una página con separadores fáciles de leer.

Si desea ver la definición de la tabla EMPLOYEES como se muestra en la diapositiva, realice los siguientes pasos:

1. Amplíe el nodo Connections en el navegador de conexiones.
2. Amplíe Tables.
3. Haga clic en EMPLOYEES. Por defecto, está seleccionado el separador Columns. Se muestra la descripción de las columnas de la tabla. Mediante el separador Data, puede ver los datos de la tabla e introducir las nuevas filas, datos de actualización y confirmar estos cambios en la base de datos.

Visualización de la Estructura de la Tabla

Utilizar el comando DESCRIBE para mostrar la estructura de una tabla:

The screenshot shows the Oracle SQL Developer interface. At the top, there's a toolbar with various icons. Below it, a status bar displays "myconnection" and "1.69686902 seconds". The main area has a title bar "DESC EMPLOYEES". Below the title bar is a toolbar with icons for Results, Script Output, Explain, Autotrace, DBMS Output, and OWA Output. The main content area displays the results of the DESCRIBE command for the EMPLOYEES table. The output is a table with three columns: Name, Null, and Type. The columns are separated by dashed lines. The data rows are as follows:

Name	Null	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

11 rows selected

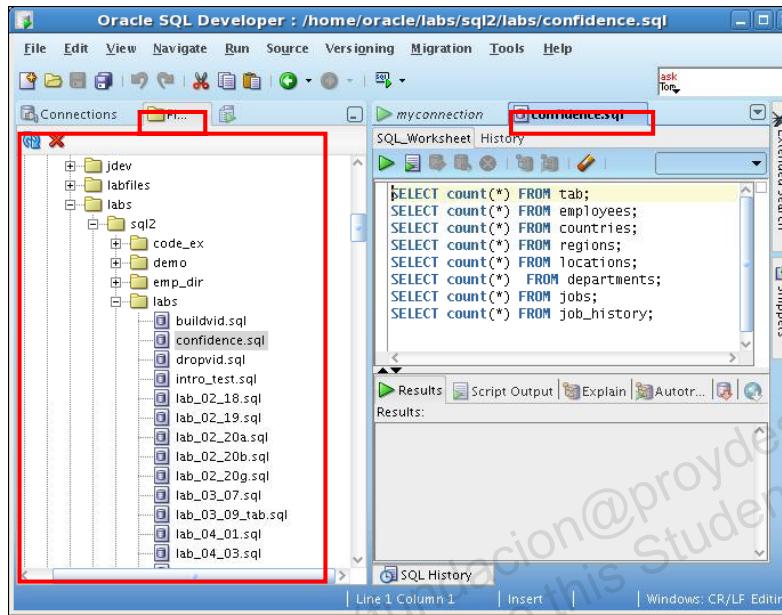
Copyright © 2010, Oracle. Todos los derechos reservados.

Visualización de la Estructura de la Tabla

En SQL Developer, también puede visualizar la estructura de una tabla mediante el comando DESCRIBE. El resultado del comando es una visualización de los nombres de columna y tipos de dato, así como una indicación de si una columna debe contener datos.

Examen de Archivos

Utilizar el navegador de archivos para explorar el sistema de archivos y abrir archivos del sistema.



Copyright © 2010, Oracle. Todos los derechos reservados.

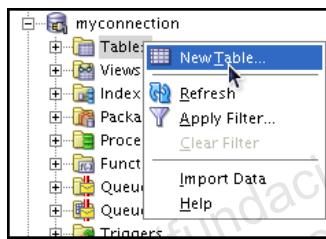
Examen de Objetos de Bases de Datos

Puede utilizar el navegador de archivos para examinar y abrir archivos del sistema.

- Para ver el navegador de archivos, haga clic en el separador Files o en View > Files.
- Para ver el contenido de un archivo, haga clic dos veces en un nombre de archivo para mostrar su contenido en el área de la hoja de trabajo de SQL.

Creación de un Objeto de Esquema

- SQL Developer soporta la creación de cualquier objeto de esquema mediante:
 - Ejecución de una sentencia SQL en SQL Worksheet
 - Uso del menú contextual
- Permite editar los objetos mediante un cuadro de diálogo de edición o con uno de los muchos menús sensibles al contexto.
- Permite ver el lenguaje de definición de datos (DDL) para los ajustes, como la creación de un nuevo objeto o la edición de un objeto de esquema existente.



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

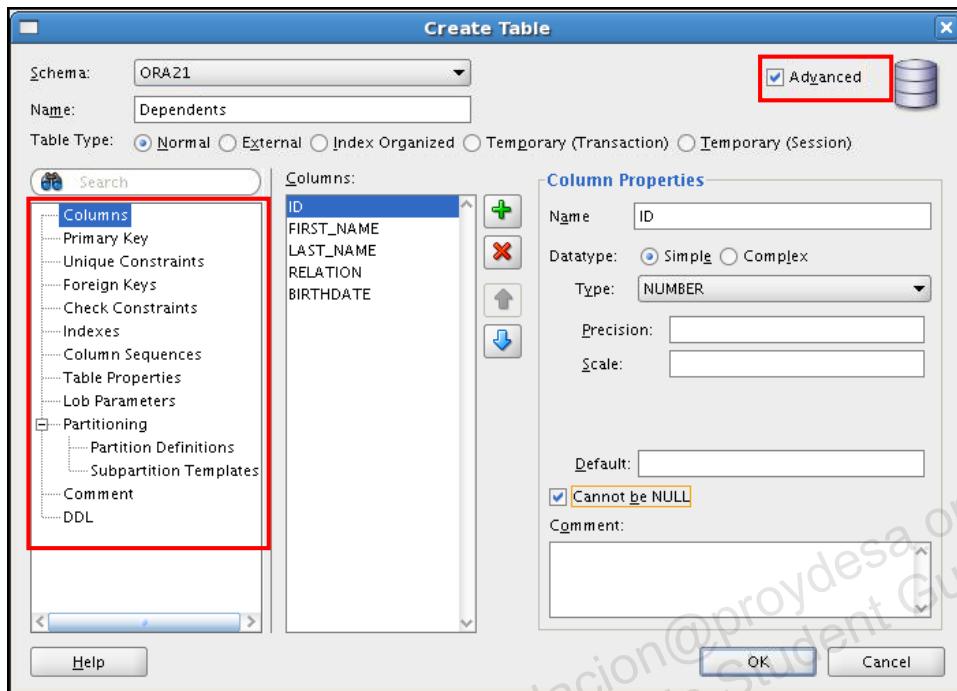
Creación de un Objeto de Esquema

SQL Developer soporta la creación de cualquier objeto de esquema mediante la ejecución de una sentencia SQL en la hoja de trabajo de SQL. Por otro lado, puede crear objetos mediante los menús contextuales. Cuando se crea, puede editar los objetos con un cuadro de diálogo de edición o con uno de los muchos menús sensibles al contexto.

Una vez que se han creado los nuevos objetos o se han editado los existentes, el DDL de estos ajustes estará disponible para la revisión. Una opción Export DDL está disponible si desea crear el DDL completo para uno o más objetos en el esquema.

La diapositiva muestra cómo crear una tabla mediante el menú contextual. Para abrir un cuadro de diálogo para la creación de una nueva tabla, haga clic con el botón derecho del mouse en Tables y seleccione New Table. Los cuadros de diálogo para la creación y edición de objetos de la base de datos tienen varios separadores, cada uno de ellos contiene un agrupamiento lógico de propiedades para dicho tipo de objeto.

Creación de una Nueva Tabla: Ejemplo



Copyright © 2010, Oracle. Todos los derechos reservados.

ORACLE

Creación de una Nueva Tabla: Ejemplo

En el cuadro de diálogo Create Table, si no activa la casilla de control Advanced , puede crear una tabla rápidamente mediante la especificación de columnas y algunas funciones utilizadas con frecuencia.

Si activa la casilla de control Advanced , el cuadro de diálogo Create Table cambia a otro con varias opciones, en las que puede especificar un juego ampliado de funciones mientras crea la tabla.

En el ejemplo de la diapositiva se muestra cómo crear la tabla DEPENDENTS mediante la activación de la casilla de control Advanced.

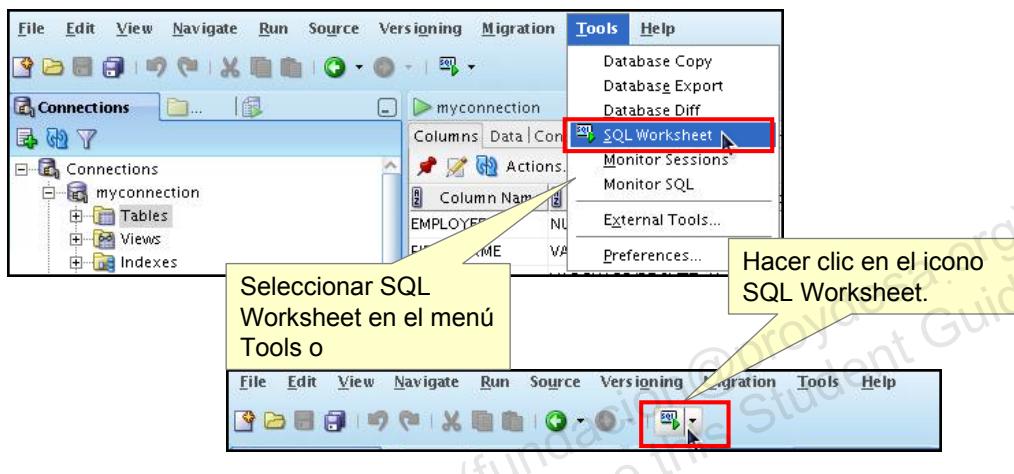
Para crear una nueva tabla, realice los siguientes pasos:

1. En el navegador de conexiones, haga clic con el botón derecho del mouse en Tables.
2. Seleccione Create TABLE.
3. En el cuadro de diálogo Create Table, seleccione Advanced.
4. Especifique la información de columna.
5. Haga clic en OK.

Aunque no es necesario, también debe especificar una clave primaria con el separador Primary Key en el cuadro de diálogo. En ocasiones, puede que desee editar la tabla que ha creado; para ello, haga clic con el botón derecho del mouse en la tabla del navegador de conexiones y seleccione Edit.

Uso de la Hoja de Trabajo de SQL

- Utilizar la hoja de trabajo de SQL para introducir y ejecutar sentencias SQL, PL/SQL y SQL *Plus.
- Especificar las acciones que se pueden procesar mediante la conexión a la base de datos asociada a la hoja de trabajo.



Copyright © 2010, Oracle. Todos los derechos reservados.

Uso de la Hoja de Trabajo de SQL

Al conectarse a una base de datos, automáticamente se abre una ventana de hoja de trabajo de SQL para dicha conexión. Puede utilizar la hoja de trabajo de SQL para introducir y ejecutar sentencias SQL, PL/SQL y SQL *Plus. SQL Worksheet soporta sentencias SQL*Plus hasta un determinado grado. Las sentencias SQL*Plus no soportadas por la hoja de trabajo de SQL se ignoran y no se transfieren a la base de datos.

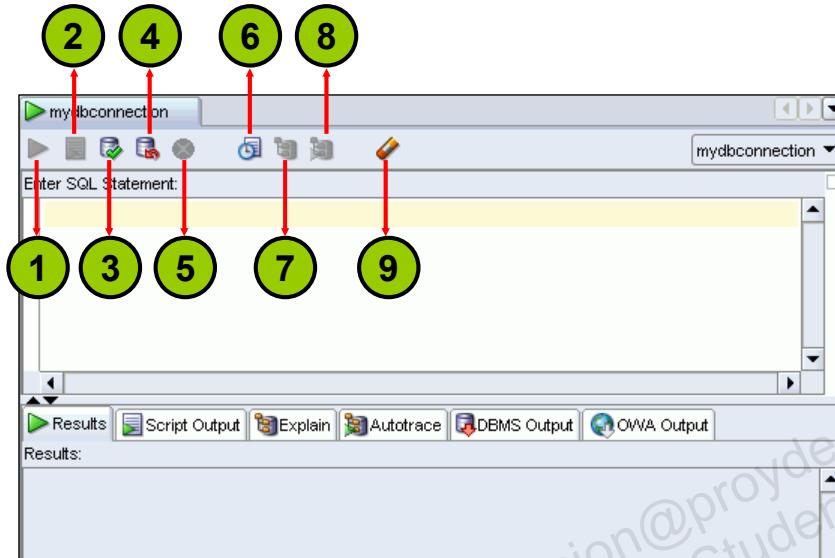
Puede especificar las acciones que se pueden procesar mediante la conexión a la base de datos asociada a la hoja de trabajo, como:

- Creación de una tabla
- Inserción de datos
- Creación y edición de un disparador
- Selección de datos de tablas
- Guardado de datos seleccionados en un archivo

Para visualizar una hoja de trabajo de SQL, utilice una de las siguientes opciones:

- Seleccione Tools > SQL Worksheet.
- Haga clic en el ícono Open SQL Worksheet.

Uso de la Hoja de Trabajo de SQL



Copyright © 2010, Oracle. Todos los derechos reservados.

ORACLE

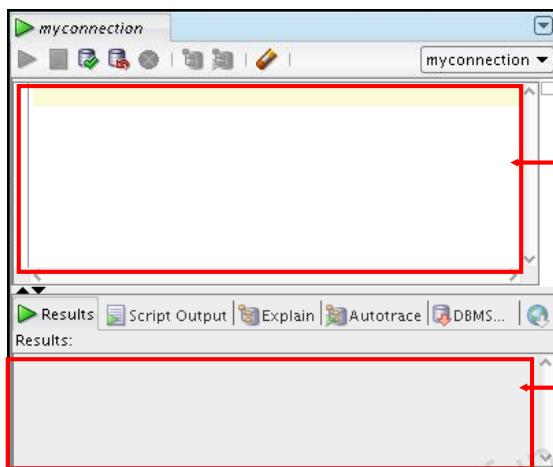
Uso de la Hoja de Trabajo de SQL (continuación)

Puede que desee utilizar teclas de acceso directo o iconos para realizar determinadas tareas, como la ejecución de una sentencia SQL o de un script y la visualización del historial de sentencias SQL ejecutadas. Puede utilizar la barra de herramientas de la hoja de trabajo de SQL que contiene iconos para realizar las siguientes tareas:

1. **Execute Statement:** ejecuta la sentencia en la que está ubicado el cursor del cuadro Enter SQL Statement. Puede utilizar variables de enlace en las sentencias SQL pero no variables de sustitución.
2. **Run Script:** ejecuta todas las sentencias en el cuadro Enter SQL Statement mediante Script Runner. Puede utilizar variables de sustitución en las sentencias SQL pero no variables de enlace.
3. **Commit:** escribe cualquier cambio realizado en la base de datos y finaliza la transacción.
4. **Rollback:** desecha cualquier cambio realizado en la base de datos, sin escribirlos en la base de datos y finaliza la transacción.
5. **Cancel:** para la ejecución de cualquier sentencia que se esté ejecutando actualmente.
6. **SQL History:** muestra un cuadro de diálogo con información sobre las sentencias SQL ejecutadas.
7. **Execute Explain Plan:** genera la ejecución del plan, que puede ver haciendo clic en el separador Explain.
8. **Autotrace:** genera información de rastreo sobre la sentencia.
9. **Clear:** borra la sentencia o sentencias del recuadro Enter SQL Statement.

Uso de la Hoja de Trabajo de SQL

- Utilizar la hoja de trabajo de SQL para introducir y ejecutar sentencias SQL, PL/SQL y SQL *Plus.
- Especificar las acciones que se pueden procesar mediante la conexión a la base de datos asociada a la hoja de trabajo.



Introducir
sentencias SQL.

Los resultados se
muestran aquí.

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Uso de la Hoja de Trabajo de SQL (continuación)

Al conectarse a una base de datos, automáticamente se abre una ventana de hoja de trabajo de SQL para dicha conexión. Puede utilizar la hoja de trabajo de SQL para introducir y ejecutar sentencias SQL, PL/SQL y SQL *Plus. Todos los comandos SQL y PL/SQL están soportados conforme se transfieren directamente desde la hoja de trabajo de SQL a Oracle Database. La hoja de trabajo de SQL debe interpretar los comandos SQL*Plus utilizados en SQL Developer antes de transferirlos a la base de datos.

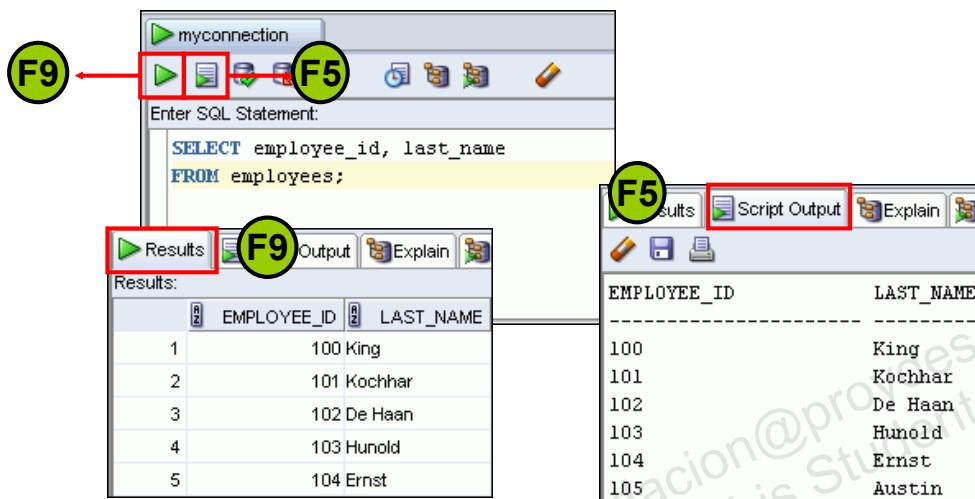
La hoja de trabajo de SQL soporta actualmente varios comandos SQL*Plus. Los comandos que no soporta la hoja de trabajo de SQL se ignoran y no se envían a Oracle Database. A través de la hoja de trabajo de SQL, puede ejecutar sentencias SQL y algunos de los comandos SQL*Plus.

Puede mostrar una hoja de trabajo de SQL mediante alguna de las siguientes dos opciones:

- Seleccione Tools > SQL Worksheet.
- Haga clic en el ícono Open SQL Worksheet.

Ejecución de Sentencias SQL

Uso del recuadro Enter SQL Statement para introducir una o varias sentencias SQL.

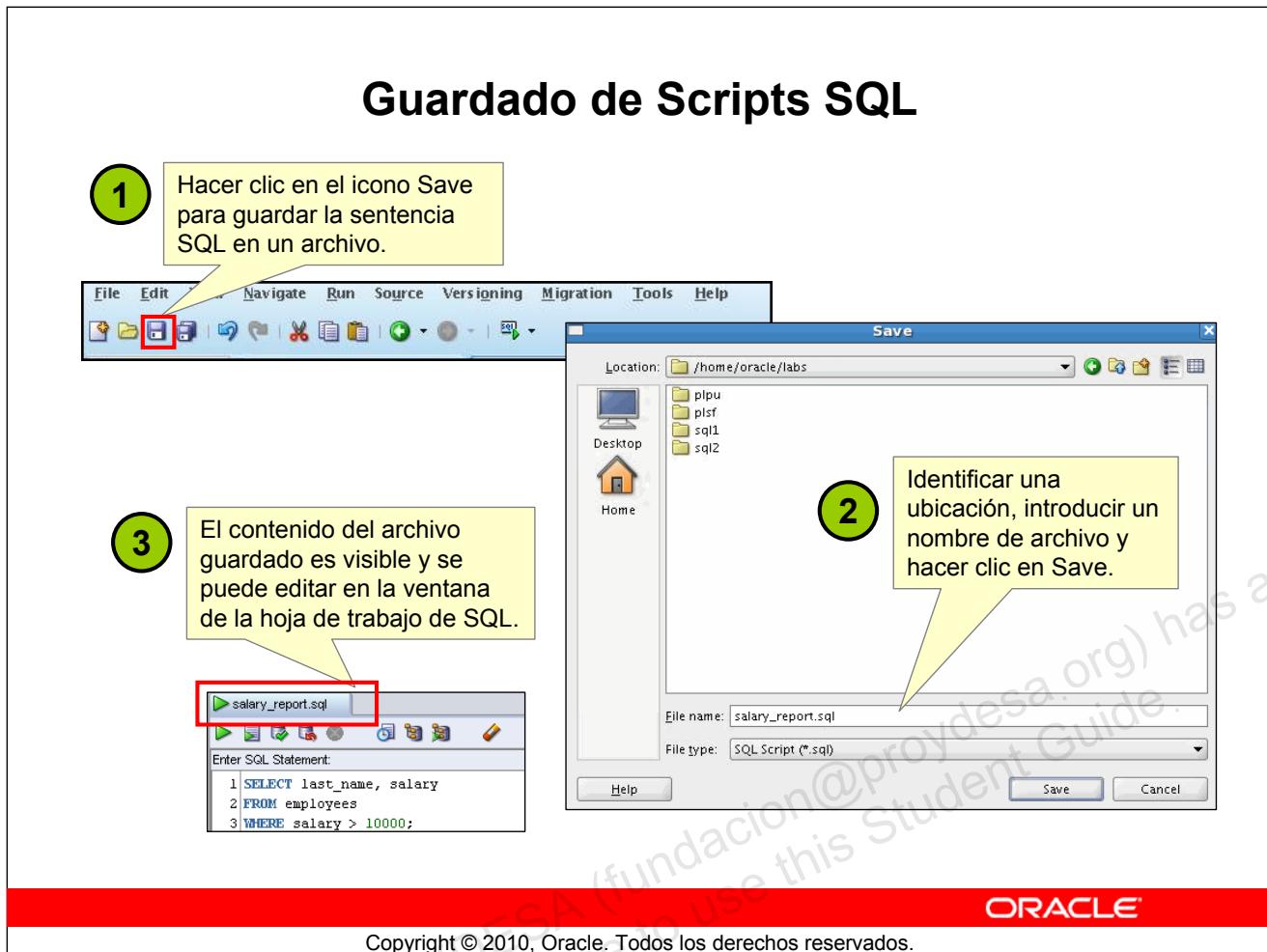


ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Ejecución de Sentencias SQL

En el ejemplo de la diapositiva se muestra la diferencia entre la salida de la misma consulta cuando se utiliza la tecla F9 o Execute Statement y cuando se utiliza F5 o Run Script.



Guardado de Scripts SQL

Puede guardar las sentencias SQL desde la hoja de trabajo de SQL en un archivo de texto. Para guardar el contenido del cuadro Enter SQL Statement, realice los siguientes pasos:

1. Haga clic en el ícono Save o utilice la opción de menú File > Save.
2. En el cuadro de diálogo Guardar de Windows, introduzca un nombre de archivo y la ubicación en la que desea guardarlo.
3. Haga clic en Save.

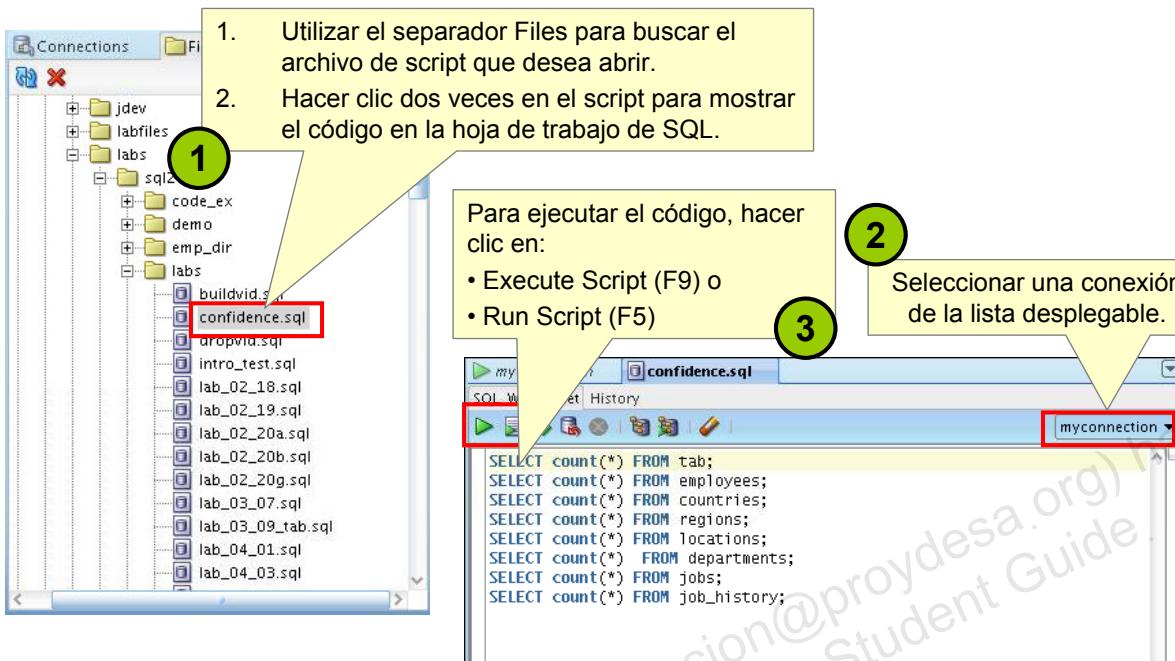
Después de guardar el contenido de un archivo, en la ventana Enter SQL Statement se muestra una página con separadores del contenido del archivo. Puede tener varios archivos abiertos al mismo tiempo. Cada archivo se muestra como una página con separadores.

Script Pathing

Puede seleccionar una ruta de acceso por defecto para buscar y guardar scripts. En Tools > Preferences > Database > Worksheet Parameters, introduzca un valor en el campo “Select default path to look for scripts”.

Ejecución de Archivos de Script Guardados:

Método 1



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Ejecución de Archivos de Script Guardados: Método 1

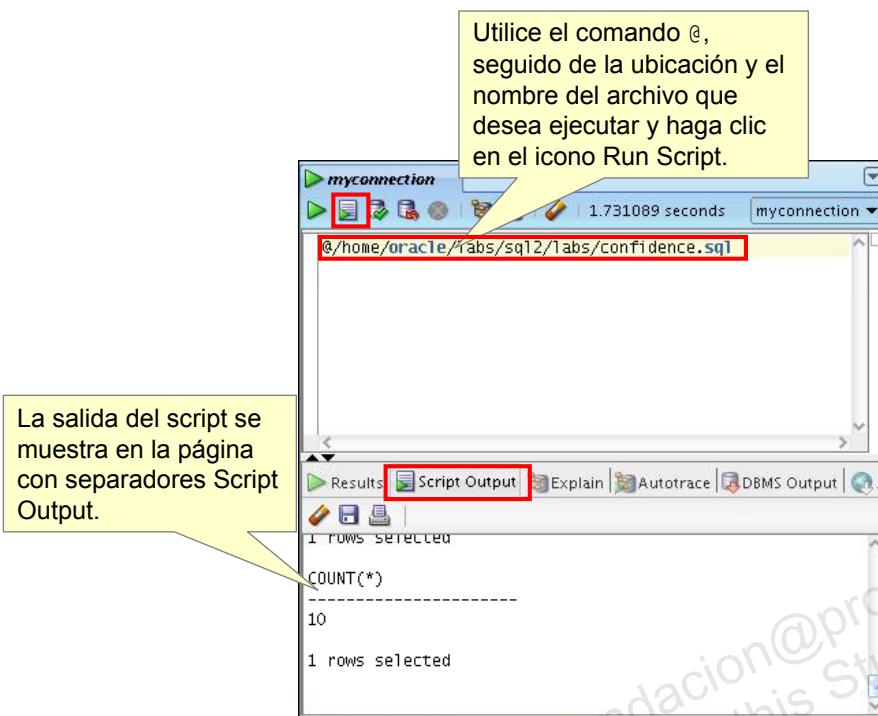
Para abrir un archivo de script y mostrar el código en el área de la hoja de trabajo de SQL, realice los siguientes pasos:

1. En el navegador de archivos, seleccione (o navegue hasta) el archivo de script que desea abrir.
2. Haga clic dos veces para abrir. El código del archivo de script se muestra en el área de la hoja de trabajo de SQL.
3. Seleccione una conexión de la lista desplegable de conexiones.
4. Para ejecutar el código, haga clic en el ícono Run Script (F5) de la barra de herramientas de la hoja de trabajo de SQL. Si no ha seleccionado una conexión de la lista desplegable de conexiones, aparecerá un cuadro de diálogo de conexiones. Seleccione la conexión que desea utilizar para la ejecución del script.

También puede:

1. Seleccione File > Open. Aparece un cuadro de diálogo Open.
2. En el cuadro de diálogo Open, seleccione (o navegue hasta) el archivo de script que desea abrir.
3. Haga clic en Open. El código del archivo de script se muestra en el área de la hoja de trabajo de SQL.
4. Seleccione una conexión de la lista desplegable de conexiones.
5. Para ejecutar el código, haga clic en el ícono Run Script (F5) de la barra de herramientas de la hoja de trabajo de SQL. Si no ha seleccionado una conexión de la lista desplegable de conexiones, aparecerá un cuadro de diálogo de conexiones. Seleccione la conexión que desea utilizar para la ejecución del script.

Ejecución de Archivos de Script Guardados: Método 2



ORACLE

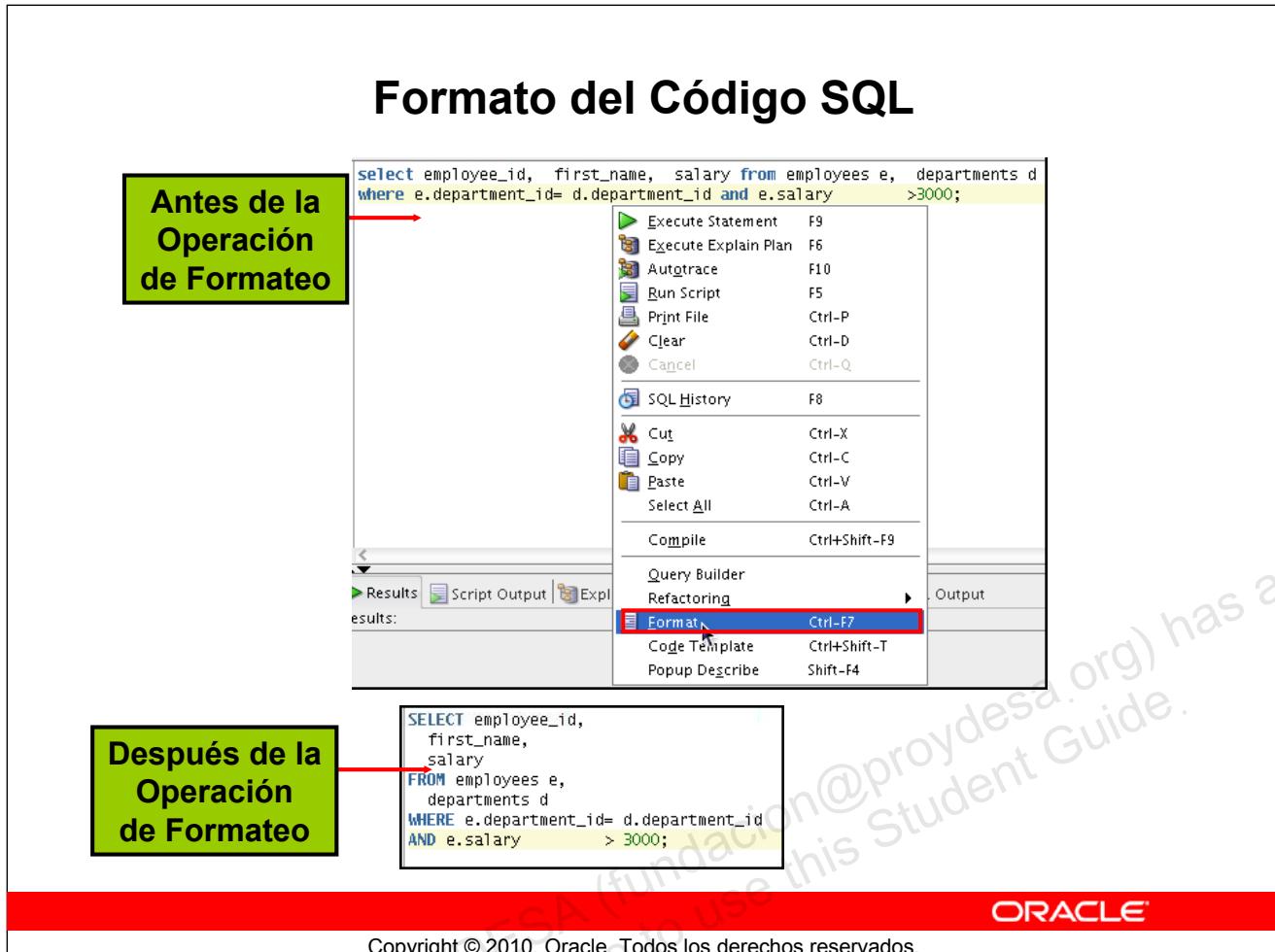
Copyright © 2010, Oracle. Todos los derechos reservados.

Ejecución de Archivos de Script Guardados: Método 2

Para guardar un script SQL, realice los siguientes pasos:

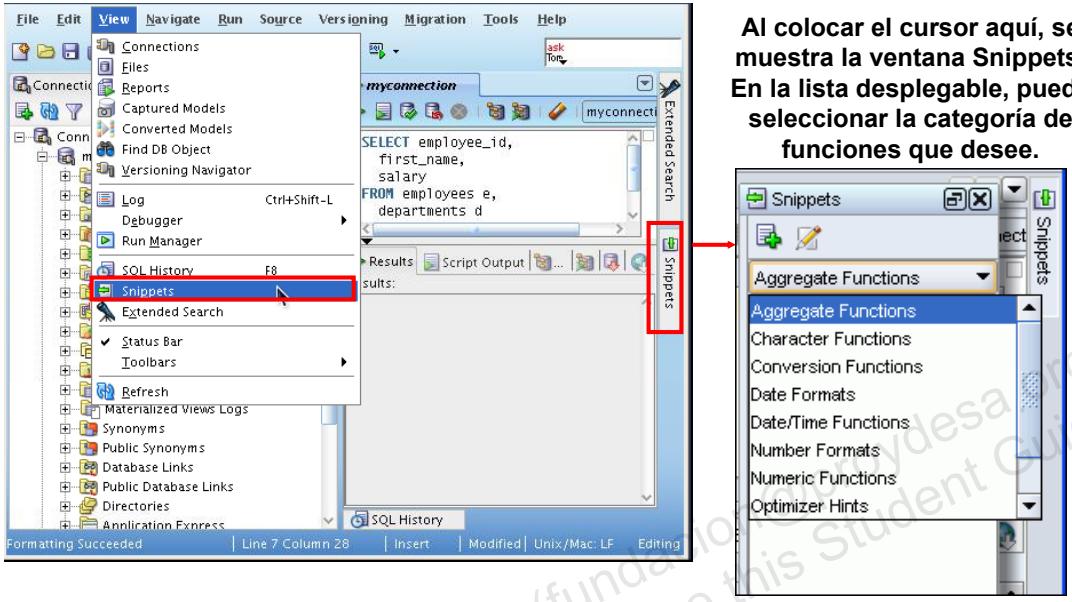
1. Utilice el comando @, seguido de la ubicación y el nombre del archivo que desea ejecutar en la ventana Enter SQL Statement.
2. Haga clic en el icono Run Script.

El resultado de la ejecución del archivo se muestra en la página con separadores Script Output. También puede guardar la salida del script haciendo clic en el icono Save de la página con separadores Script Output. Aparece el cuadro de diálogo Guardar de Windows, donde puede identificar un nombre y una ubicación para el archivo.



Uso de Fragmentos

Los fragmentos son fragmentos de código que pueden ser simplemente sintaxis o ejemplos.



Al colocar el cursor aquí, se muestra la ventana Snippets. En la lista desplegable, puede seleccionar la categoría de funciones que deseé.

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

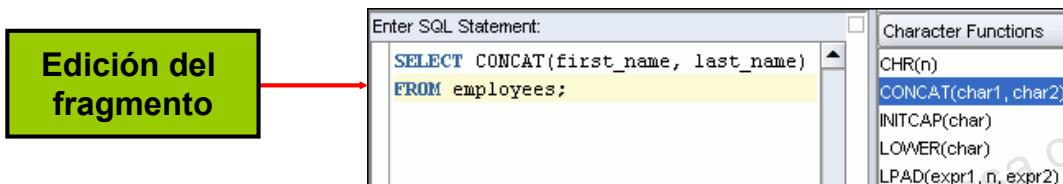
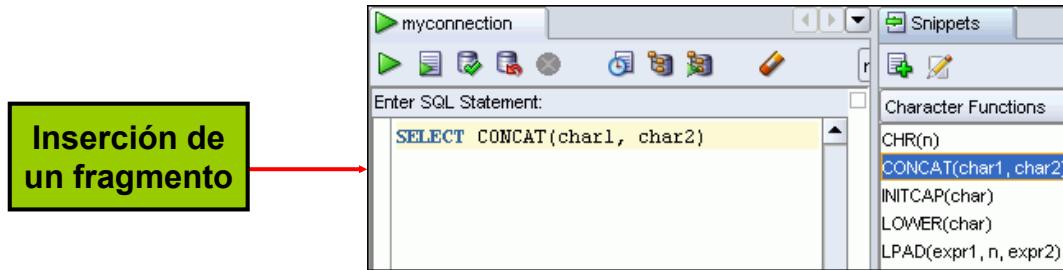
Uso de Fragmentos

Es posible que desee utilizar determinados fragmentos de código al utilizar la hoja de trabajo de SQL o al crear o editar un procedimiento o una función de PL/SQL. SQL Developer dispone de la función denominada Snippets. Los fragmentos son fragmentos de código, como funciones SQL, indicaciones del optimizador y otras técnicas de programación de PL/SQL. Puede arrastrar fragmentos a la ventana Editor.

Para visualizar los fragmentos, seleccione View > Snippets.

Aparece la ventana Snippets a la derecha. Puede utilizar la lista desplegable para seleccionar un grupo. En el margen de la ventana derecha se encuentra un botón Snippets, para poder acceder a la ventana Snippets si se oculta.

Uso de Fragmentos: Ejemplo



Copyright © 2010, Oracle. Todos los derechos reservados.

ORACLE

Uso de Fragmentos: Ejemplo

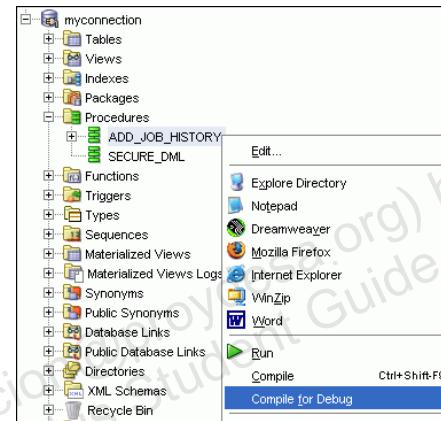
Para insertar un fragmento en el código en la hoja de trabajo de SQL o en otro procedimiento o función PL/SQL, arrastre el fragmento desde la ventana Snippets hasta el lugar deseado del código. A continuación, puede editar la sintaxis para que la función SQL sea válida en el contexto actual. Para ver una descripción breve de una función SQL en una ayuda de burbuja, coloque el cursor sobre el nombre de la función.

En el ejemplo de la diapositiva se muestra que `CONCAT (char1, char2)` se arrastra desde el grupo Character Functions a la ventana Snippets. A continuación, se edita la sintaxis de la función `CONCAT` y el resto de la sentencia se agrega como se describe a continuación:

```
SELECT CONCAT(first_name, last_name)
FROM employees;
```

Depuración de Procedimientos y Funciones

- Utilizar SQL Developer para depurar funciones y procedimientos PL/SQL.
- Utilizar la opción “Compile for Debug” para realizar una compilación PL/SQL para que se pueda depurar el procedimiento.
- Utilizar las opciones del menú Debug para definir puntos de división y realizar tareas Step Into y Step Over.



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Depuración de Procedimientos y Funciones

En SQL Developer, puede depurar procedimientos y funciones PL/SQL. Con las opciones del menú Debug, puede realizar las siguientes tareas de depuración:

- **Find Execution Point** va al siguiente punto de ejecución.
- **Resume** continúa con la ejecución.
- **Step Over** omite el siguiente método y va a la siguiente sentencia después del método.
- **Step Into** va a la primera sentencia del siguiente método.
- **Step Out** deja el método actual y va a la siguiente sentencia.
- **Step to End of Method** va a la última sentencia del método actual.
- **Pause** para la ejecución pero no sale, lo que permite reanudar la ejecución.
- **Terminate** para y sale de la ejecución. No puede reanudar la ejecución desde este punto; en su lugar, para iniciar la ejecución o depuración desde el comienzo de la función o del procedimiento, haga clic en el ícono Run o Debug de la barra de herramientas del separador Source.
- **Garbage Collection** elimina los objetos no válidos de la caché en favor de objetos a los que se accede con mayor frecuencia y más válidos.

Estas opciones también están disponibles como iconos en la barra de herramientas de depuración.

Informes de Bases de Datos

SQL Developer proporciona un número predefinido de informes sobre la base de datos y sus objetos.

Owner	Name	Type	Referenced Owner	Referenced Name
CTXSYS	CTX_CLASSES	VIEW	CTXSYS	DR\$CLASS
CTXSYS	CTX_CLS	PACKAGE	SYS	STANDARD
CTXSYS	CTX_DOC	PACKAGE	SYS	STANDARD
CTXSYS	CTX_INDEX_SETS	VIEW	CTXSYS	DR\$INDEX_SET
CTXSYS	CTX_INDEX_SETS	VIEW	SYS	USER\$
CTXSYS	CTX_INDEX_SET_INDEXES	VIEW	CTXSYS	DR\$INDEX_SET
CTXSYS	CTX_INDEX_SET_INDEXES	VIEW	CTXSYS	DR\$INDEX_SET_INDEX
CTXSYS	CTX_INDEX_SET_INDEXES	VIEW	SYS	USER\$
CTXSYS	CTX_OBJECTS	VIEW	CTXSYS	DR\$CLASS
CTXSYS	CTX_OBJECTS	VIEW	CTXSYS	DR\$OBJECT
CTXSYS	CTX_OBJECT_ATTRIBUTES	VIEW	CTXSYS	DR\$CLASS
CTXSYS	CTX_OBJECT_ATTRIBUTES	VIEW	CTXSYS	DR\$OBJECT
CTXSYS	CTX_OBJECT_ATTRIBUTES	VIEW	CTXSYS	DR\$OBJECT_ATTRIBUTE
CTXSYS	CTX_OBJECT_ATTRIBUTE_LOV	VIEW	CTXSYS	DR\$CLASS
CTXSYS	CTX_OBJECT_ATTRIBUTE_LOV	VIEW	CTXSYS	DR\$OBJECT
CTXSYS	CTX_OBJECT_ATTRIBUTE_LOV	VIEW	CTXSYS	DR\$OBJECT_ATTRIBUTE
CTXSYS	CTX_OBJECT_ATTRIBUTE_LOV	VIEW	CTXSYS	DR\$OBJECT_ATTRIBUTE_LOV
CTXSYS	CTX_PARAMETERS	VIEW	CTXSYS	DR\$PARAMETER

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Informes de Bases de Datos

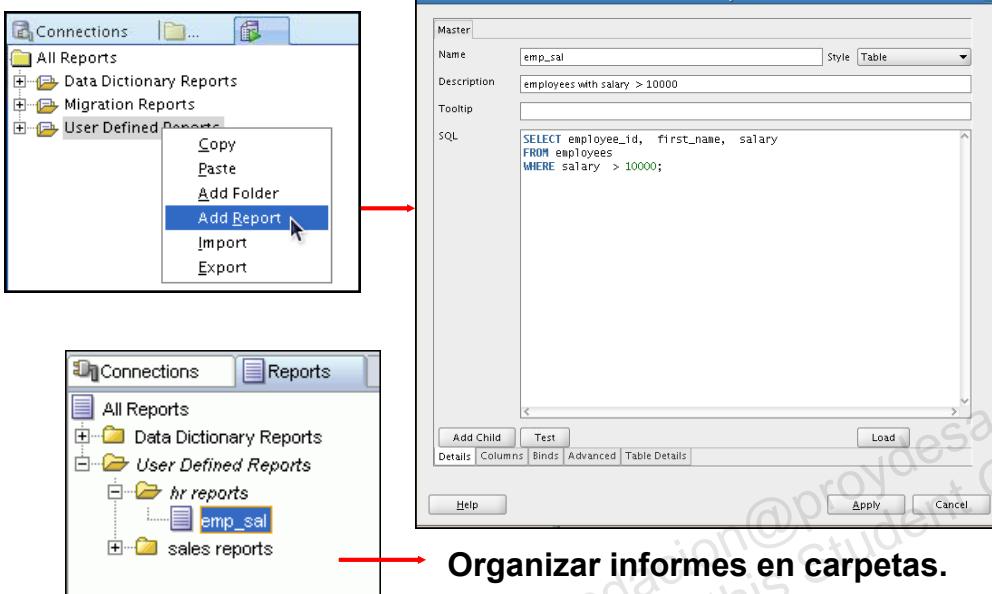
SQL Developer proporciona diferentes informes sobre la base de datos y sus objetos. Estos informes se pueden agrupar en las siguientes categorías:

- Informes About Your Database
- Informes Database Administration
- Informes Table
- Informes PL/SQL
- Informes Security
- Informes XML
- Informes Jobs
- Informes Streams
- Informes All Objects
- Informes Data Dictionary
- Informes User Defined

Para visualizar los informes, haga clic en el separador Reports situado a la izquierda de la ventana. Los informes individuales se muestran en los paneles con separadores situados a la derecha de la ventana y para cada informe puede seleccionar (en una lista desplegable) la conexión a la base de datos para la que desea mostrar el informe. Para los informes sobre objetos, sólo se muestran aquellos objetos que sean visibles para el usuario de la base de datos asociado a la conexión a la base de datos seleccionada y las filas ordenadas por propietario. También puede crear sus propios informes definidos por el usuario.

Creación de un Informe Definido por el Usuario

Crear y guardar informes definidos por el usuario para un uso repetido.



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Creación de un Informe Definido por el Usuario

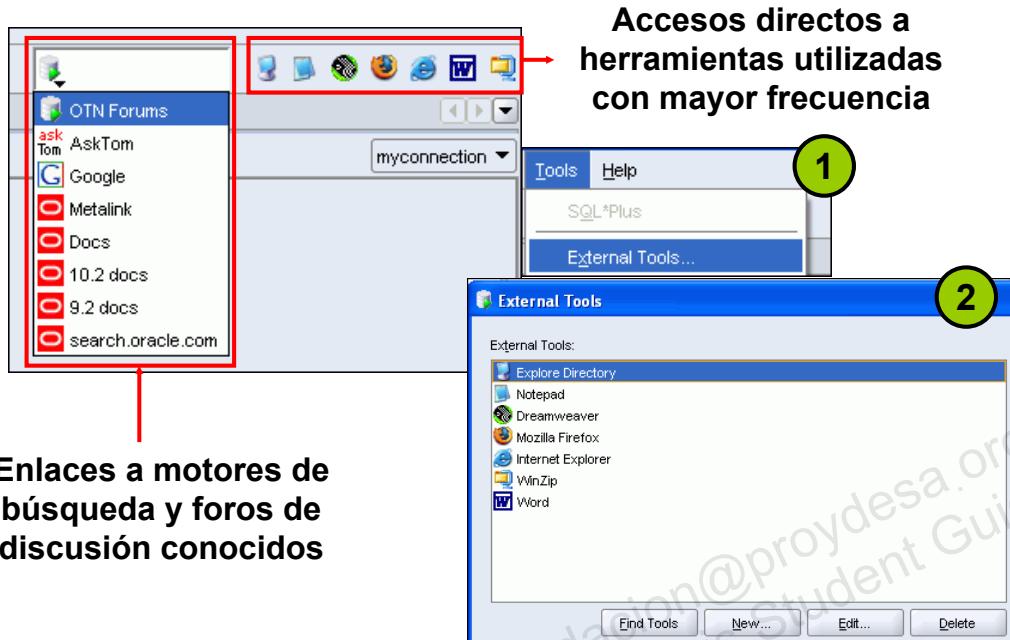
Los informes definidos por el usuario son informes creados por los usuarios de SQL Developer. Para crear un informe definido por el usuario, realice los siguientes pasos:

1. Haga clic con el botón derecho del mouse en el nodo User Defined Reports en Reports y seleccione Add Report.
2. En el cuadro de diálogo Create Report, especifique el nombre del informe y la consulta SQL para recuperar información del informe. A continuación, haga clic en Apply.

En el ejemplo de la diapositiva, se especifica el nombre del informe como `emp_sal`. Se proporciona una descripción opcional que indica que el informe contiene los detalles de los empleados con un salario `salary >= 10000`. La sentencia SQL completa para la recuperación de la información que se mostrará en el informe definido por el usuario se especifica en el cuadro SQL. También puede incluir una ayuda de burbuja opcional que se muestre al colocar el cursor brevemente sobre el nombre del informe en la pantalla del navegador Reports.

Puede organizar los informes definidos por el usuario en carpetas y puede crear una jerarquía de carpetas y subcarpetas. Para crear una carpeta para los informes definidos por el usuario, haga clic con el botón derecho del mouse en el nodo User Defined Reports o en cualquier nombre de carpeta de dicho nodo y seleccione Add Folder. La información sobre los informes definidos por el usuario, incluidas las carpetas de dichos informes, se almacena en un archivo denominado `UserReports.xml` en el directorio de información específica del usuario.

Motores de Búsqueda y Herramientas Externas



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Motores de Búsqueda y Herramientas Externas

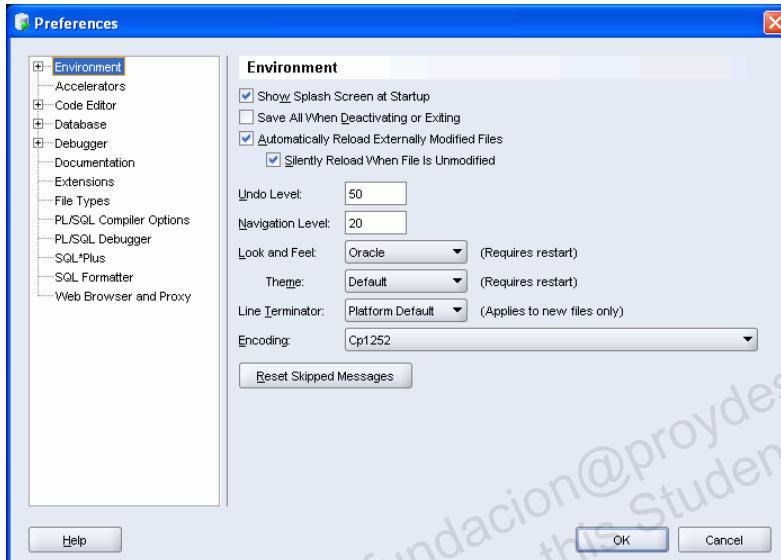
Para mejorar la productividad de los desarrolladores de SQL, SQL Developer ha agregado enlaces rápidos a motores de búsqueda y foros de discusión conocidos, como AskTom, Google, etc. Además, dispone de iconos de acceso directo a algunas de las herramientas más utilizadas, como el Bloc de notas, Microsoft Word y Dreamweaver.

Puede agregar herramientas externas a la lista de herramientas existentes o incluso suprimir accesos directos a las herramientas que no utilice habitualmente. Para ello, realice los siguientes pasos:

1. En el menú Tools, seleccione External Tools.
2. En el cuadro de diálogo External Tools, seleccione New para agregar nuevas herramientas. Seleccione Delete para eliminar cualquier herramienta de la lista.

Definición de Preferencias

- Personalizar la interfaz y el entorno de SQL Developer.
- En el menú Tools, seleccione Preferences.



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Definición de Preferencias

Puede personalizar diferentes aspectos de la interfaz y el entorno de SQL Developer mediante la modificación de las preferencias de SQL Developer según sus preferencias y necesidades. Para modificar las preferencias de SQL Developer, seleccione Tools y, a continuación, Preferences.

Las preferencias se agrupan en las siguientes categorías:

- Environment
- Accelerators (accesos directos del teclado)
- Code Editors
- Database
- Debugger
- Documentation
- Extensions
- File Types
- Migration
- PL/SQL Compilers
- PL/SQL Debugger, etc.

Restablecimiento del Diseño de SQL Developer

```
Terminal
File Edit View Terminal Tabs Help
[oracle@EDRSR5P1 ~]$ locate windowinglayout.xml
/home/oracle/.sqldeveloper/system1.5.4.59.40/o.ide.11.1.1.0.22.49.48/windowinglayout.xml
/home/oracle/.sqldeveloper/system1.5.4.59.41/o.ide.11.1.1.0.22.49.48/windowinglayout.xml
[oracle@EDRSR5P1 ~]$ cd /home/oracle/.sqldeveloper/system1.5.4.59.41/o.ide.11.1.1.0.22.49.48
[oracle@EDRSR5P1 o.ide.11.1.1.0.22.49.48]$ ls
Debugging.layout  Editing.layout  projects  windowinglayout.xml
dtcache.xml      preferences.xml  settings.xml
[oracle@EDRSR5P1 o.ide.11.1.1.0.22.49.48]$ rm windowinglayout.xml
[oracle@EDRSR5P1 o.ide.11.1.1.0.22.49.48]$
```



Copyright © 2010, Oracle. Todos los derechos reservados.

Restablecimiento del Diseño de SQL Developer

Al trabajar con SQL Developer, si desaparece el navegador de conexiones o no puede encajar la ventana Log en su lugar original, realice los siguientes pasos para solucionar el problema:

1. Salga de SQL Developer.
2. Abra una ventana de terminal y utilice el comando `locate` para buscar la ubicación de `windowinglayout.xml`.
3. Vaya al directorio que contenga el archivo `windowinglayout.xml` y suprímalo.
4. Reinicie SQL Developer.

Resumen

En este apéndice, debe haber aprendido cómo utilizar SQL Developer para realizar las siguientes acciones:

- Examinar, crear y editar objetos de bases de datos
- Ejecutar sentencias SQL y scripts en la hoja de trabajo de SQL
- Crear y guardar informes personalizados



Copyright © 2010, Oracle. Todos los derechos reservados.

Resumen

SQL Developer es una herramienta gráfica gratuita para simplificar las tareas de desarrollo de la base de datos. Con SQL Developer, puede examinar, crear y editar objetos de bases de datos. Puede utilizar SQL Worksheet para ejecutar archivos de comandos y sentencias SQL. SQL Developer permite crear y guardar su propio juego especial de informes para un uso repetido.

Uso de SQL*Plus

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

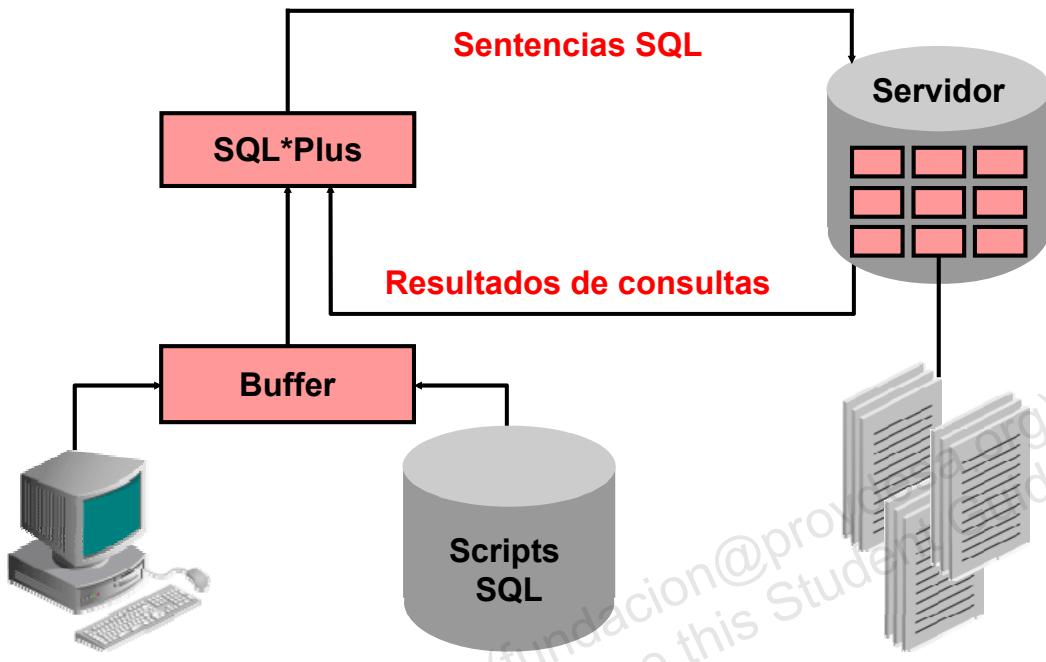
Al finalizar este apéndice, debería estar capacitado para lo siguiente:

- Conectarse a SQL*Plus
- Editar comandos SQL
- Formatear la salida con comandos SQL*Plus
- Interactuar con scripts



Copyright © 2010, Oracle. Todos los derechos reservados.

Interacción de SQL y SQL*Plus



Copyright © 2010, Oracle. Todos los derechos reservados.

ORACLE

SQL y SQL*Plus

SQL es un lenguaje de comandos que se utiliza para la comunicación con el servidor de Oracle desde cualquier herramienta o aplicación. Oracle SQL contiene muchas extensiones. Al introducir una sentencia SQL, ésta se almacena en una parte de la memoria denominada *buffer SQL* y permanece allí hasta que introduzca una nueva sentencia SQL. SQL*Plus es una herramienta de Oracle que reconoce y envía sentencias SQL en Oracle9i Server para su ejecución. Contiene su propio lenguaje de comandos.

Funciones de SQL

- Las pueden utilizar una gran variedad de usuarios, incluidos aquéllos con poca o ninguna experiencia.
- Es un lenguaje que no es de procedimientos.
- Reduce la cantidad de tiempo necesario para crear y mantener sistemas.
- Es un lenguaje como el inglés.

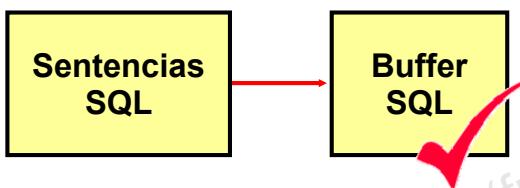
Funciones de SQL*Plus

- Acepta la entrada ad hoc de sentencias.
- Acepta la entrada de SQL de los archivos.
- Proporciona un editor de líneas para modificar sentencias SQL.
- Controla los valores de entorno.
- Formatea resultados de consulta en informes básicos.
- Accede a bases de datos locales y remotas.

Sentencias SQL frente a Comandos SQL*Plus

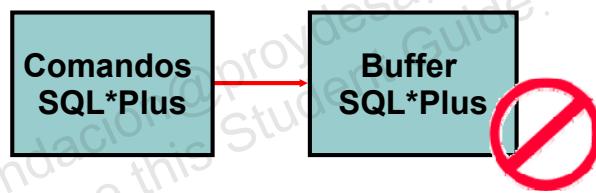
SQL

- Un lenguaje
 - Est醖ar de ANSI
 - Las palabras clave no se pueden abbreviar.
 - Las sentencias manipulan definiciones de tablas y datos en la base de datos.



SQL*Plus

- Un entorno
 - Propiedad de Oracle
 - Las palabras clave se pueden abbreviar.
 - Los comandos no permiten la manipulación de valores en la base de datos.



Copyright © 2010, Oracle. Todos los derechos reservados.

SQL y SQL*Plus (continuación)

En la siguiente tabla compara SQL y SQL*Plus:

SQL	SQL*Plus
Lenguaje para la comunicación con el servidor de Oracle para acceder a los datos.	Reconoce las sentencias SQL y las envía al servidor.
Se basa en SQL del estándar ANSI (American National Standards Institute)	Interfaz propiedad de Oracle para ejecutar sentencias SQL.
Manipula la definiciones de tablas y datos en la base de datos.	No permite la manipulación de valores en la base de datos.
Se introduce en el buffer SQL en una o más líneas.	Se introduce en una línea al mismo tiempo y no se almacena en el buffer SQL.
No tiene ningún carácter de continuación.	Utiliza un guión (-) como carácter de continuación si el comando es más largo que una línea.
No se puede abreviar.	Se puede abreviar.
Utiliza un carácter de terminación para ejecutar comandos inmediatamente.	No necesita caracteres de terminación; ejecuta los comandos inmediatamente.
Utiliza funciones para realizar algunas tareas de formato.	Utiliza comandos para formatear datos.

Visión General de SQL*Plus

- Conectarse a SQL*Plus.
- Describir la estructura de la tabla.
- Editar la sentencia SQL.
- Ejecutar SQL desde SQL*Plus.
- Guardar sentencias SQL en archivos y agregar sentencias SQL a los archivos.
- Ejecutar archivos guardados.
- Cargar comandos del archivo en el buffer para la edición.



Copyright © 2010, Oracle. Todos los derechos reservados.

SQL*Plus

SQL*Plus es un entorno en el que puede:

- Ejecutar sentencias SQL para recuperar, modificar, agregar y eliminar datos de la base de datos.
- Formatear, realizar cálculos, almacenar e imprimir resultados de consulta en forma de informes.
- Crear scripts para almacenar sentencias SQL para un uso repetido en el futuro.

Los comandos SQL*Plus se pueden dividir en las siguientes categorías:

Categoría	Objetivo
Entorno	Afectar al comportamiento general de sentencias SQL para la sesión.
Formato	Formatear resultados de consulta.
Manipulación de archivos	Guardar, cargar y ejecutar scripts.
Ejecución	Enviar sentencias SQL del buffer SQL al servidor de Oracle.
Edición	Modificar sentencias SQL en el buffer.
Interacción	Crear y transferir variables a la sentencia SQL, imprimir valores de variables y mensajes en la pantalla.
Otros	Conectarse a la base de datos, manipular el entorno SQL*Plus y mostrar definiciones de columnas.

Conexión a SQL*Plus

```
Terminal
File Edit View Terminal Tabs Help
[oracle@EDRSR5P1 ~]$sqlplus
SQL*Plus: Release 11.2.0.0.2 Beta on Tue May 26 19:59:06 2009
Copyright (c) 1982, 2009, Oracle. All rights reserved.
Enter user-name: ora21@orcl
Enter password:
Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.0.2 - Beta
With the Partitioning, OLAP, Data Mining and Real Application Testing options
SQL>
```

sqlplus [username[/password[@database]]]

```
Terminal
File Edit View Terminal Tabs Help
[oracle@EDRSR5P1 ~]$sqlplus ora21/ora21@orcl
SQL*Plus: Release 11.2.0.0.2 Beta on Tue May 26 19:58:06 2009
Copyright (c) 1982, 2009, Oracle. All rights reserved.

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.0.2 - Beta
With the Partitioning, OLAP, Data Mining and Real Application Testing options
SQL>
```

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Conexión a SQL*Plus

La forma de llamar a SQL*Plus dependerá del tipo de sistema operativo en que esté ejecutando Oracle Database.

Para conectarse desde un entorno Linux:

- Haga clic con el botón derecho del mouse en el escritorio de Linux y seleccione un terminal.
- Introduzca el comando `sqlplus` que se muestra en la diapositiva.
- Introduzca el nombre de usuario, la contraseña y el nombre de la base de datos.

En la sintaxis:

`username` Nombre de usuario de la base de datos.

`password` Contraseña de la base de datos (la contraseña será visible si la introduce aquí).

`@database` Cadena de conexión de la base de datos.

Nota: para asegurarse de la integridad de la contraseña, no la introduzca en la petición de datos del sistema operativo. En su lugar, introduzca sólo el nombre de usuario. Introduzca la contraseña en la petición de datos de la contraseña.

Visualización de la Estructura de la Tabla

Utilizar el comando SQL*Plus DESCRIBE para mostrar la estructura de una tabla:

```
DESC[RIBE] tablename
```



Copyright © 2010, Oracle. Todos los derechos reservados.

Visualización de la Estructura de la Tabla

En SQL*Plus, puede mostrar la estructura de una tabla mediante el comando DESCRIBE. El resultado del comando es una visualización de los nombres de columna y tipos de dato, así como una indicación de si una columna debe contener datos.

En la sintaxis:

tablename Nombre de cualquier tabla existente, vista o sinónimo al que puede acceder el usuario.

Para describir la tabla DEPARTMENTS, utilice este comando:

```
SQL> DESCRIBE DEPARTMENTS
      Name          Null?    Type
-----+
DEPARTMENT_ID      NOT NULL NUMBER(4)
DEPARTMENT_NAME    NOT NULL VARCHAR2(30)
MANAGER_ID          NUMBER(6)
LOCATION_ID          NUMBER(4)
```

Visualización de la Estructura de la Tabla

```
DESCRIBE departments
```

Name	Null?	Type
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)



Copyright © 2010, Oracle. Todos los derechos reservados.

Visualización de la Estructura de Tabla (continuación)

En el ejemplo de la diapositiva se muestra la información sobre la estructura de la tabla DEPARTMENTS. En el resultado:

Null?: especifica si una columna debe contener datos (NOT NULL indica que una columna debe contener datos).

Type: muestra el tipo de dato de una columna.

Comandos de Edición SQL*Plus

- A [PPEND] *text*
- C [HANGE] / *old* / *new*
- C [HANGE] / *text* /
- CL [EAR] BUFF [ER]
- DEL
- DEL *n*
- DEL *m n*



Copyright © 2010, Oracle. Todos los derechos reservados.

Comandos de Edición SQL*Plus

Los comandos SQL*Plus se introducen en una línea al mismo tiempo y no se almacenan en el buffer SQL.

Comando	Descripción
A [PPEND] <i>text</i>	Agrega <i>text</i> o al final de la línea actual.
C [HANGE] / <i>old</i> / <i>new</i>	Cambia el texto <i>antiguo</i> por el <i>nuevo</i> en la línea actual.
C [HANGE] / <i>text</i> /	Suprime el <i>text</i> o de la línea actual.
CL [EAR] BUFF [ER]	Suprime todas las líneas del buffer SQL.
DEL	Suprime la línea actual.
DEL <i>n</i>	Suprime la línea <i>n</i> .
DEL <i>m n</i>	Suprime de las líneas <i>m</i> hasta la <i>n</i> , inclusive.

Instrucciones

- Si pulsa Intro antes de que haya terminado la ejecución de un comando, SQL*Plus le solicitará un número de línea.
- Termine el buffer SQL introduciendo uno de los caracteres de terminación (punto y coma o barra) o pulsando Intro dos veces. A continuación, aparecerá la petición de datos SQL.

Comandos de Edición SQL*Plus

- `I [NPUT]`
- `I [NPUT] text`
- `L [IST]`
- `L [IST] n`
- `L [IST] m n`
- `R [UN]`
- `n`
- `n text`
- `0 text`



Copyright © 2010, Oracle. Todos los derechos reservados.

Comandos de Edición SQL*Plus (continuación)

Comando	Descripción
<code>I [NPUT]</code>	Inserta un número indefinido de líneas.
<code>I [NPUT] text</code>	Inserta una línea que consta de <i>texto</i> .
<code>L [IST]</code>	Muestra todas las líneas en el buffer SQL.
<code>L [IST] n</code>	Muestra una línea (especificada por <i>n</i>).
<code>L [IST] m n</code>	Muestra un rango de líneas (<i>m</i> a <i>n</i>), inclusive.
<code>R [UN]</code>	Muestra y ejecuta la sentencia SQL actual en el buffer.
<code>n</code>	Especifica la línea para crear la línea actual.
<code>n text</code>	Sustituye la línea <i>n</i> con <i>texto</i> .
<code>0 text</code>	Inserta una línea delante de la línea 1.

Nota: puede introducir sólo un comando SQL*Plus para cada petición de datos SQL. Los comandos SQL*Plus no se almacenan en el buffer. Para que un comando SQL*Plus continúe en la siguiente línea, finalice la primera línea con un guión (-).

Uso de LIST, n y APPEND

```
LIST
 1  SELECT last_name
 2* FROM employees
```

```
1
1* SELECT last_name
```

```
A , job_id
1* SELECT last_name, job_id
```

```
LIST
 1  SELECT last_name, job_id
 2* FROM employees
```

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Uso de LIST, n y APPEND

- Utilice el comando L [IST] para mostrar el contenido del buffer SQL. El asterisco (*) situado junto a la línea 2 en el buffer indica que la línea 2 es la línea actual. Cualquier edición que realice se aplica a la línea actual.
- Introduzca el número (n) de la línea que desea editar para cambiar el número de la línea actual. Se muestra la nueva línea actual.
- Utilice el comando A [PPEND] para agregar texto a la línea actual. Se muestra la línea recién editada. Verifique el nuevo contenido del buffer mediante el comando LIST.

Nota: muchos de los comandos SQL*Plus, incluidos LIST y APPEND, se pueden abreviar sólo con su primera letra. LIST se puede abbreviar con L; APPEND se puede abbreviar con A.

Uso del Comando CHANGE

```
LIST
```

```
1* SELECT * from employees
```

```
c/employees/departments
```

```
1* SELECT * from departments
```

```
LIST
```

```
1* SELECT * from departments
```



Copyright © 2010, Oracle. Todos los derechos reservados.

Uso del Comando CHANGE

- Utilice L [LIST] para mostrar el contenido del buffer.
- Utilice el comando C [HANGE] para modificar el contenido de la línea actual del buffer SQL. En este caso, sustituya la tabla employees por la tabla departments. Se muestra la nueva línea actual.
- Utilice el comando L [LIST] para verificar el nuevo contenido del buffer.

Comandos de Archivos SQL*Plus

- `SAVE filename`
- `GET filename`
- `START filename`
- `@ filename`
- `EDIT filename`
- `SPOOL filename`
- `EXIT`



Copyright © 2010, Oracle. Todos los derechos reservados.

Comandos de Archivos SQL*Plus

Las sentencias SQL se comunican con el servidor de Oracle. Los comandos SQL*Plus controlan el entorno, formatean los resultados de la consulta y gestionan archivos. Puede utilizar los comandos descritos en la siguiente tabla:

Comando	Descripción
<code>SAV[E] filename [.ext]</code> [REP[LACE]APP[END]]	Guarda el contenido actual del buffer SQL en un archivo. Utilice APPEND para agregar a un archivo existente; utilice REPLACE para sobrescribir un archivo existente. La extensión por defecto es .sql.
<code>GET filename [.ext]</code>	Escribe el contenido de un archivo guardado anteriormente en el buffer SQL. La extensión por defecto del nombre de archivo es .sql.
<code>STA[RT] filename [.ext]</code>	Ejecuta el archivo de comandos guardado anteriormente.
<code>@ filename</code>	Ejecuta un archivo de comandos guardado anteriormente (igual que START).
<code>ED[IT]</code>	Llama al editor y guarda el contenido del buffer en un archivo denominado afiedt.buf.
<code>ED[IT] [filename [.ext]]</code>	Llama al editor para editar el contenido de un archivo guardado.
<code>SPO[OL] [filename [.ext]] OFF OUT</code>	Almacena los resultados de la consulta en un archivo. OFF cierra el archivo de spool. OUT cierra el archivo de spool y envía los resultados del archivo a la impresora.
<code>EXIT</code>	Sale de SQL*Plus.

Uso de los Comandos SAVE y START

```
LIST
```

```
1  SELECT last_name, manager_id, department_id  
2* FROM employees
```

```
SAVE my_query
```

```
Created file my_query
```

```
START my_query
```

```
LAST_NAME
```

```
MANAGER_ID DEPARTMENT_ID
```

```
-----  
King
```

```
90
```

```
Kochhar
```

```
100
```

```
90
```

```
...
```

```
107 rows selected.
```



Copyright © 2010, Oracle. Todos los derechos reservados.

Uso de los Comandos SAVE y START

SAVE

Utilice el comando SAVE para almacenar el contenido actual del buffer en un archivo. De esta forma, podrá almacenar los scripts utilizados con frecuencia para utilizarlos en el futuro.

START

Utilice el comando START para ejecutar un script en SQL*Plus. Además, también puede utilizar el símbolo @ para ejecutar un script.

```
@my_query
```

Comando SERVEROUTPUT

- Utilice el comando SET SERVEROUT [PUT] para controlar si se debe mostrar la salida de los procedimientos almacenados o bloques PL/SQL en SQL*Plus.
- El límite de longitud de la línea DBMS_OUTPUT ha aumentado de 255 a 32767 bytes.
- El tamaño por defecto es ahora ilimitado.
- Los recursos no se asignan previamente si se define SERVEROUTPUT.
- Ya que no afecta al rendimiento, utilice UNLIMITED a menos que desee conservar la memoria física.

```
SET SERVEROUT[PUT] {ON | OFF} [SIZE {n | UNLIMITED}]
[FOR[MAT] {WRA[PPED] | WOR[D_WRAPPED] | TRU[NATED]}]
```

Copyright © 2010, Oracle. Todos los derechos reservados.

Comando SERVEROUTPUT

La mayoría de los programas PL/SQL realizan entradas y salidas mediante sentencias SQL para almacenar datos en las tablas de las bases de datos o consultar dichas tablas. Las demás entradas/salidas PL/SQL se realizan a través de API, que interactúan con otros programas. Por ejemplo, el paquete DBMS_OUTPUT tiene procedimientos como PUT_LINE. Para ver el resultado fuera de PL/SQL, se necesita otro programa como SQL*Plus, para leer y visualizar los datos transferidos a DBMS_OUTPUT.

SQL*Plus no muestra los datos de DBMS_OUTPUT a menos que antes emita el comando SQL*Plus SET SERVEROUTPUT ON siguiente:

```
SET SERVEROUTPUT ON
```

Nota

- SIZE define el número de bytes de la salida que se pueden almacenar en buffer en el servidor de Oracle Database. El valor por defecto es UNLIMITED. *n* no puede ser menor que 2.000 o mayor que 1.000.000.
- Para obtener más información sobre SERVEROUTPUT, consulte *Oracle Database PL/SQL User's Guide and Reference 11g* (Guía del Usuario y Referencia de PL/SQL de Oracle Database 11g).

Uso del Comando SQL*Plus SPOOL

```
SPO[OL] [file_name[.ext]] [CRE[ATE] | REP[LACE] |  
APP[END]] | OFF | OUT]
```

Opción	Descripción
file_name[.ext]	Envía la salida al nombre de archivo especificado.
CRE[ATE]	Crea un nuevo archivo con el nombre especificado.
REP[LACE]	Sustituye el contenido de un archivo existente. Si el archivo no existe, REPLACE crea el archivo.
APP[END]	Agrega el contenido del buffer al final del archivo especificado.
OFF	Para el envío de resultados.
OUT	Para el envío de datos y envía el archivo a la impresora estándar (por defecto) de la computadora.

Copyright © 2010, Oracle. Todos los derechos reservados.

Uso del Comando SQL*Plus SPOOL

El comando SPOOL almacena los resultados de la consulta en un archivo o envía el archivo a una impresora. Se ha mejorado el comando SPOOL. Ahora puede adjuntar a (o sustituir) un archivo existente, cuando antes sólo podía utilizar SPOOL para crear (y sustituir) un archivo. REPLACE es el valor por defecto.

Para enviar la salida generada por los comandos en un script sin mostrar la salida en la pantalla, utilice SET TERMOUT OFF. SET TERMOUT OFF no afecta a la salida de los comandos que se ejecutan interactivamente.

Debe utilizar comillas con los nombres de archivo que contengan espacios en blanco. Para crear un archivo HTML válido mediante los comandos SPOOL APPEND, debe utilizar PROMPT o un comando para crear la cabecera y el pie de página de la página HTML. El comando SPOOL APPEND no analiza las etiquetas HTML. Defina SQLPLUSCOMPAT [IBILITY] en 9.2 o anterior para desactivar los parámetros CREATE, APPEND y SAVE.

Uso del Comando AUTOTRACE

- Muestra un informe después de la ejecución correcta de sentencias de manipulación de datos (DML) de SQL, como SELECT, INSERT, UPDATE o DELETE.
- El informe puede incluir ahora estadísticas de ejecución y la ruta de acceso de ejecución de la consulta.

```
SET AUTOT[RACE] {ON | OFF | TRACE[ONLY]} [EXP[LAIN]]  
[STATISTICS]
```

```
SET AUTOTRACE ON  
-- The AUTOTRACE report includes both the optimizer  
-- execution path and the SQL statement execution  
-- statistics
```



Copyright © 2010, Oracle. Todos los derechos reservados.

Uso del Comando AUTOTRACE

EXPLAIN muestra la ruta de acceso de ejecución de consulta mediante la ejecución de EXPLAIN PLAN. STATISTICS muestra estadísticas de sentencias SQL. El formato del informe AUTOTRACE puede variar en función de la versión del servidor al que está conectado y la configuración del servidor. El paquete DBMS_XPLAN proporciona una forma fácil de mostrar la salida del comando EXPLAIN PLAN en varios formatos predefinidos.

Nota

- Para obtener más información sobre el paquete y los subprogramas, consulte *Oracle Database PL/SQL Packages and Types Reference 11g* (Referencia sobre Paquetes y Tipos PL/SQL de Oracle Database 11g).
- Para obtener más información sobre EXPLAIN PLAN, consulte *Oracle Database SQL Reference 11g* (Referencia sobre SQL de Oracle Database 11g).
- Para obtener más información sobre las estadísticas de planes de ejecución, consulte *Oracle Database Performance Tuning Guide 11g* (Guía de Ajuste del Rendimiento de Oracle Database 11g).

Resumen

En este apéndice, debe haber aprendido cómo utilizar SQL*Plus como un entorno para realizar las siguientes acciones:

- Ejecutar sentencias SQL
- Editar sentencias SQL
- Formatear la salida
- Interactuar con scripts



Copyright © 2010, Oracle. Todos los derechos reservados.

Resumen

SQL*Plus es un entorno de ejecución que puede utilizar para enviar comandos SQL al servidor de la base datos para editar y guardar los comandos SQL. Puede ejecutar los comandos desde la petición de datos SQL o desde un archivo de script.

JDeveloper

Uso de JDeveloper

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

Al finalizar este apéndice, debería estar capacitado para lo siguiente:

- Mostrar las funciones clave de Oracle JDeveloper
- Crear una conexión a base de datos en JDeveloper
- Gestionar objetos de base de datos en JDeveloper
- Utilizar JDeveloper para ejecutar comandos SQL
- Crear y ejecutar unidades de programa PL/SQL

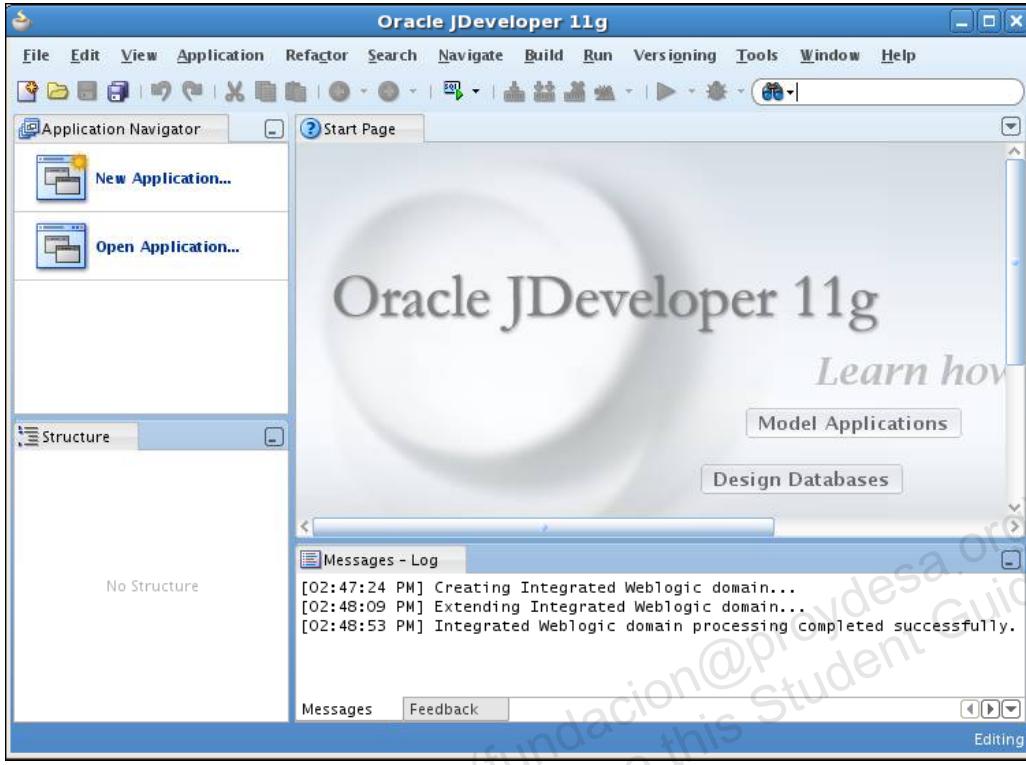


Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

En este apéndice, se presentará la herramienta JDeveloper. Aprenderá cómo utilizar JDeveloper para las tareas de desarrollo de la base de datos.

Oracle JDeveloper



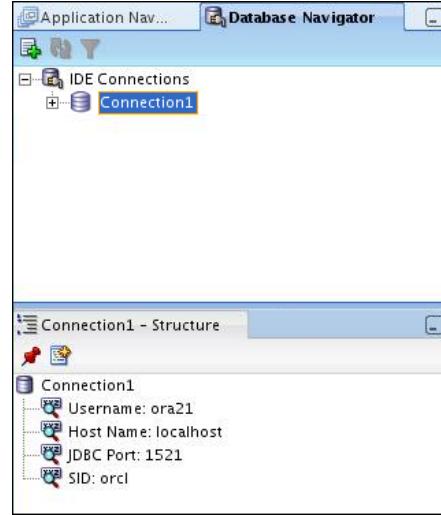
Copyright © 2010, Oracle. Todos los derechos reservados.

Oracle JDeveloper

Oracle JDeveloper es un entorno de desarrollo de integración (IDE) para desarrollar y desplegar aplicaciones Java y servicios web. Soporta cada etapa del ciclo de vida de desarrollo de software (SDLC), del modelado al despliegue. Tiene funciones que permiten utilizar los últimos estándares de la industria para Java, XML y SQL y desarrollar una aplicación.

Oracle JDeveloper 11g inicia un nuevo enfoque al desarrollo J2EE con funciones que permiten un desarrollo visual y declarativo. Este enfoque innovador hace que el desarrollo J2EE sea sencillo y eficaz.

Database Navigator

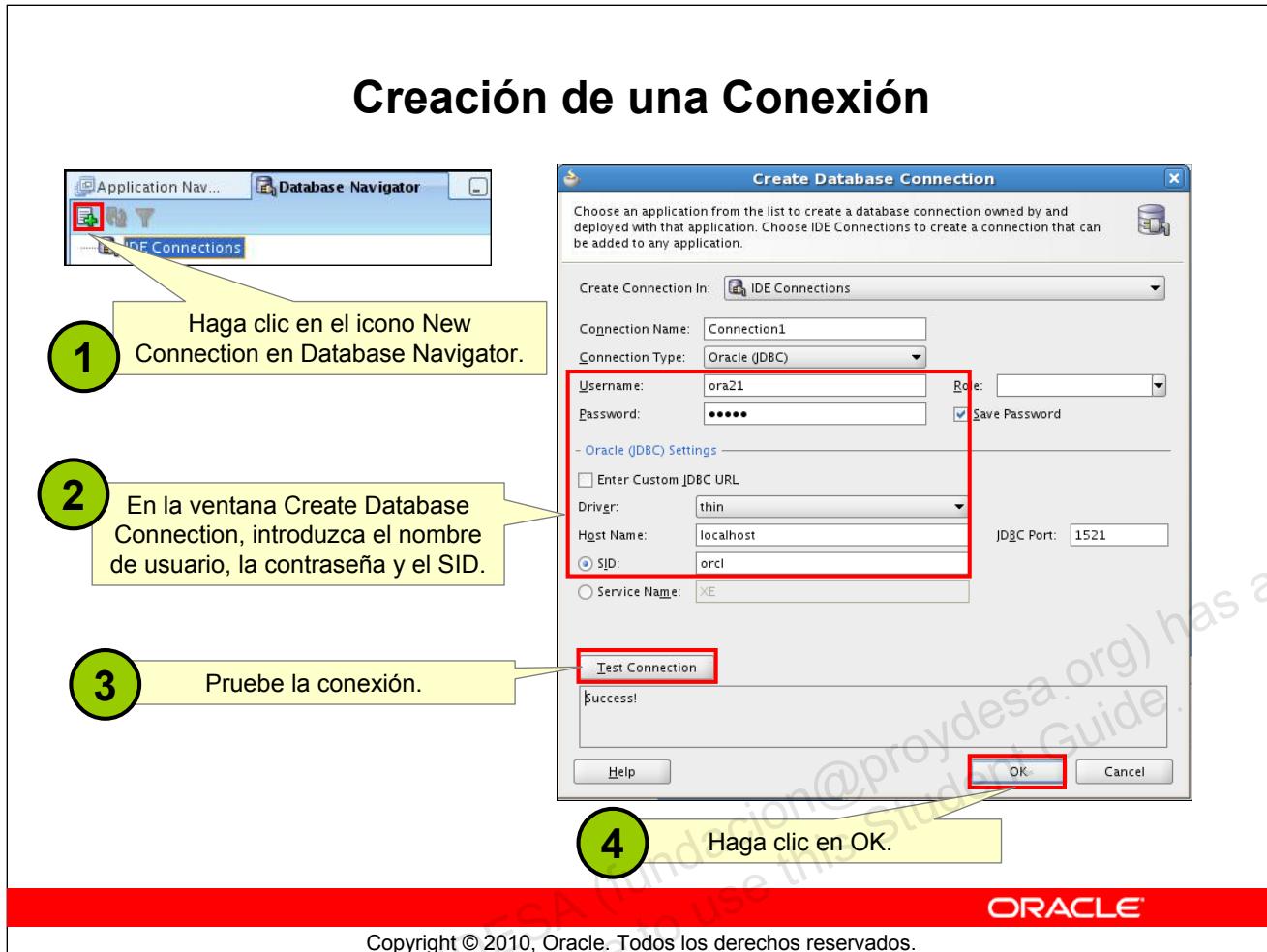


ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Database Navigator

Con Oracle JDeveloper, puede almacenar la información necesaria para conectarse a la base de datos en un objeto denominado “conexión”. Una conexión se almacena como parte de los valores de IDE y se puede exportar e importar para compartirlo fácilmente entre grupos de usuarios. Una conexión tiene diferentes fines, desde el examen de la base de datos y la creación de aplicaciones hasta el despliegue.



Creación de una Conexión

Una conexión es un objeto que especifica la información necesaria para conectarse a una base de datos concreta como usuario específico de dicha base de datos. Puede crear y probar conexiones para varias bases de datos y esquemas.

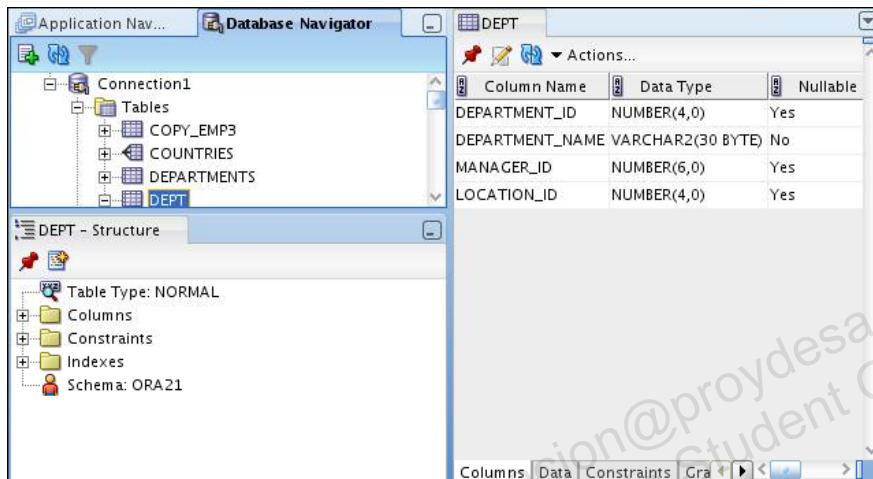
Para crear una conexión a la base de datos, realice los siguientes pasos:

1. Haga clic en el ícono New Connection en Database Navigator.
2. En la ventana Create Database Connection, introduzca el nombre de la conexión. Introduzca el nombre de usuario y la contraseña del esquema al que desea conectarse. Introduzca el SID de la base de datos a la que desea conectarse.
3. Haga clic en Test para asegurarse de que la conexión se ha definido correctamente.
4. Haga clic en OK.

Examen de Objetos de Bases de Datos

Utilizar el navegador de la base de datos para:

- Examinar los objetos de un esquema de base de datos
- Revisar las definiciones de objetos de forma rápida



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Examen de Objetos de Bases de Datos

Una vez creada la conexión a la base de datos, puede utilizar el navegador de la base de datos para examinar los objetos de un esquema de base de datos entre los que se incluyen tablas, vistas, índices, paquetes, procedimientos, disparadores y tipos.

Puede desglosar las definiciones de los objetos en separadores de información que se transfieren al diccionario de datos. Por ejemplo, si selecciona una tabla en el navegador, se muestran los detalles sobre las columnas, restricciones, permisos, estadísticas, disparadores, etc. en una página con separadores fáciles de leer.



Ejecución de Sentencias SQL

Para ejecutar una sentencia SQL, realice los siguientes pasos:

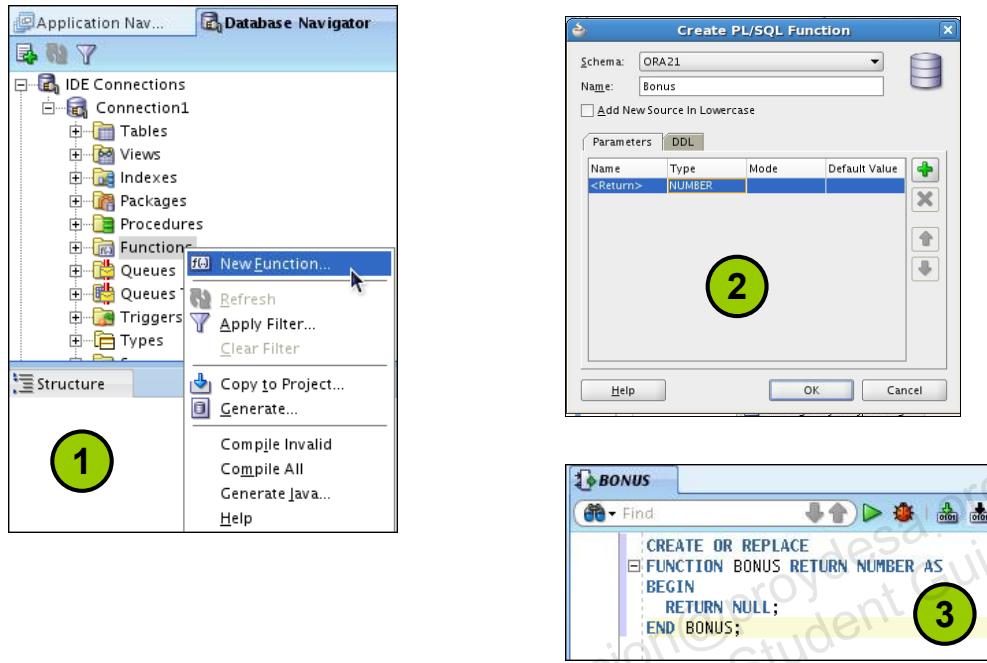
1. Haga clic en el ícono Open SQL Worksheet.
2. Seleccione la conexión.
3. Ejecute el comando SQL haciendo clic en:
 - El botón **Execute statement** o pulsando F9. La salida es la siguiente:

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME
1	100	Steven	King
2	101	Neena	Kochhar

- El botón **Run Script** o pulsando F5. La salida es la siguiente:

EMPLOYEE_ID	FIRST_NAME	LAST_NAME
100	Steven	King

Creación de Unidades de Programa



Esqueleto de la Función

ORACLE

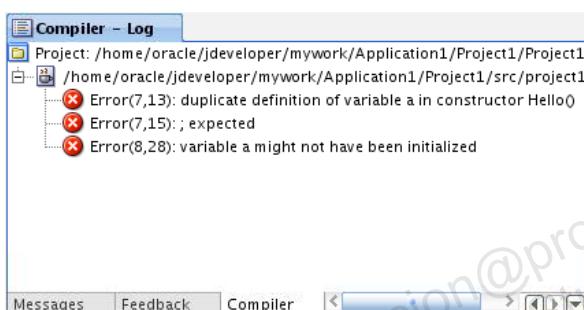
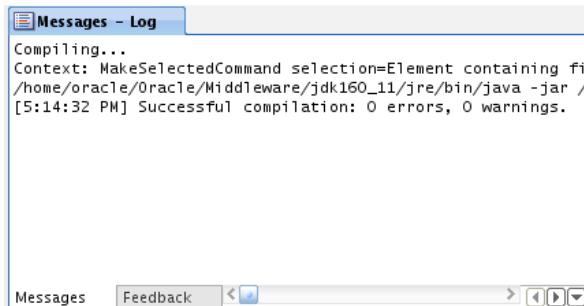
Copyright © 2010, Oracle. Todos los derechos reservados.

Creación de Unidades de Programa

Para crear una unidad de programa PL/SQL:

1. Seleccione View > Database Navigator. Seleccione y amplíe una conexión de base de datos. Haga clic con el botón derecho del mouse en una carpeta que corresponda al tipo de objeto (Procedures, Packages, Functions). Seleccione “New [Procedures|Packages|Functions]”.
2. Introduzca un nombre válido para la función, el paquete o el procedimiento y haga clic en OK.
3. Se crea una estructura básica que se abre en la ventana Code Editor. A continuación, puede editar el subprograma para que se ajuste a sus necesidades.

Compilación



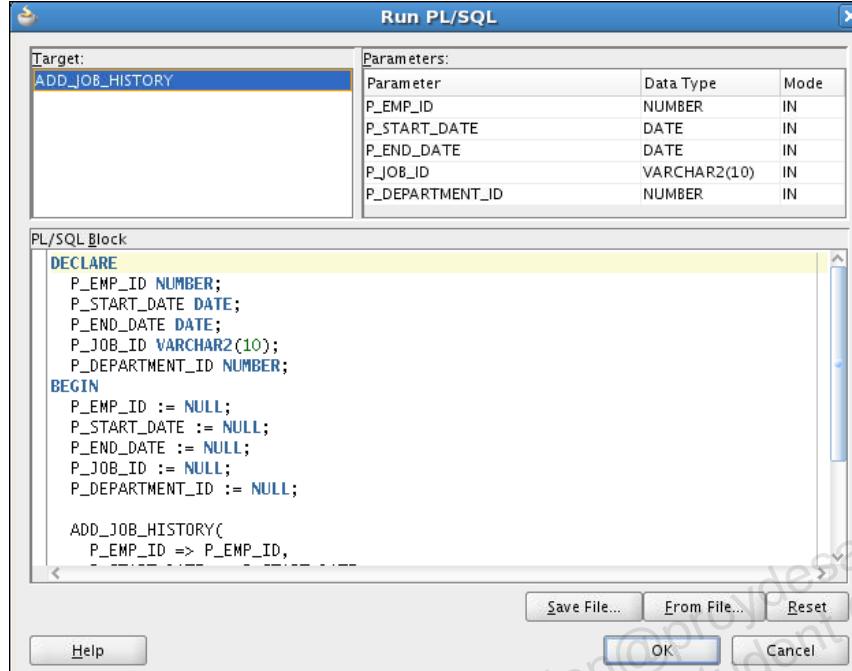
ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Compilación

Después de editar la definición de esqueleto, debe compilar la unidad de programa. Haga clic con el botón derecho del mouse en el objeto PL/SQL que necesita compilar en Connection Navigator y, a continuación, seleccione Compile. También puede pulsar CTRL + MAYÚS + F9 para compilar.

Ejecución de una Unidad de Programa



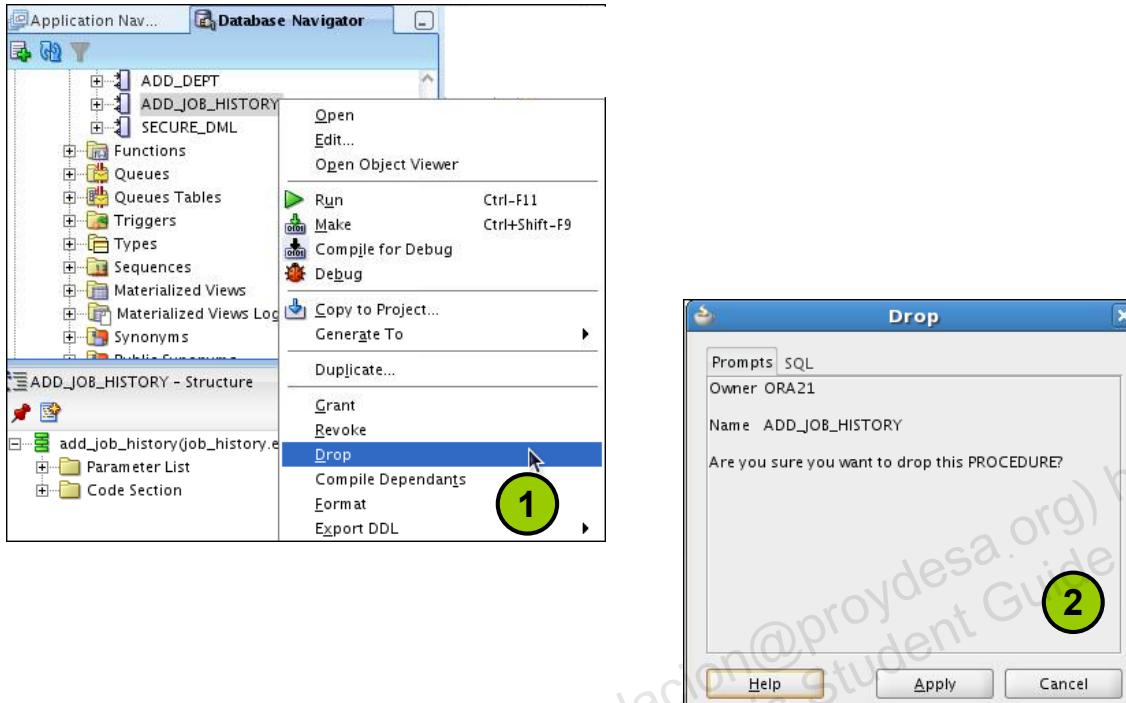
ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Ejecución de una Unidad de Programa

Para ejecutar la unidad de programa, haga clic con el botón derecho del mouse en el objeto y seleccione Run. Aparece el cuadro de diálogo Run PL/SQL. Puede que tenga que cambiar los valores NULL por valores razonables que se transfieren a la unidad de programa. Despues de cambiar los valores, haga clic en OK. La salida se mostrará en la ventana Message-Log.

Borrado de una Unidad de Programa



ORACLE

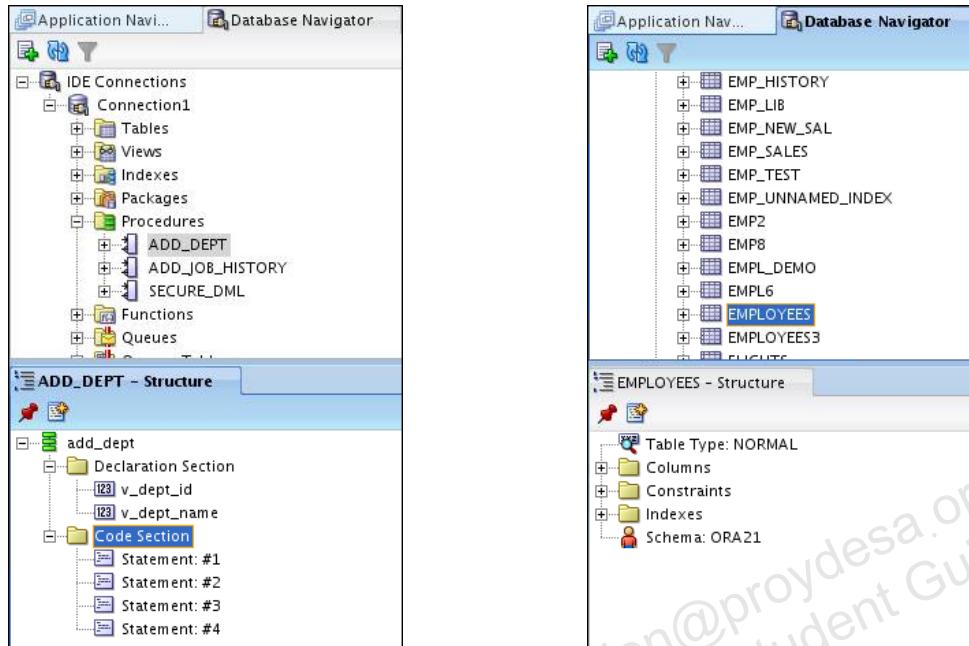
Copyright © 2010, Oracle. Todos los derechos reservados.

Borrado de una Unidad de Programa

Para borrar una unidad de programa:

1. Haga clic con el botón derecho del mouse en el objeto y seleccione Drop.
Aparece el cuadro de diálogo Drop Confirmation.
2. Haga clic en Apply.
El objeto se borra de la base de datos.

Ventana Structure



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

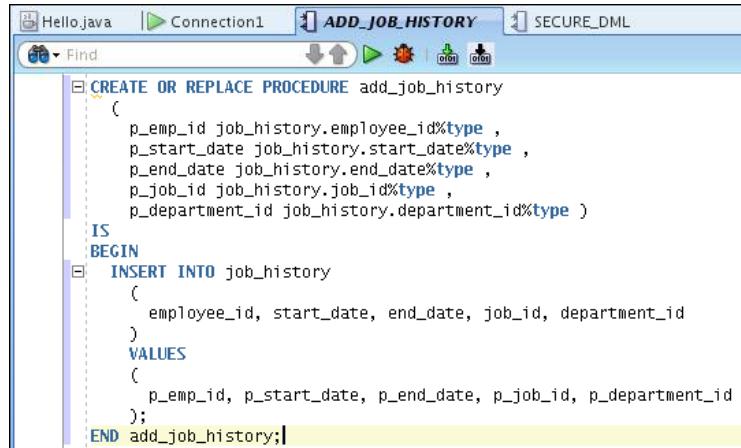
Ventana Structure

La ventana Structure ofrece una vista estructural de los datos del documento seleccionado actualmente en la ventana activa de las ventanas que proporcionan estructura: navegadores, editores, visores y el inspector de propiedades.

En la ventana Structure, puede ver los datos del documento de distintas formas. Las estructuras disponibles están basadas en el tipo de documento. Para un archivo Java, puede ver la estructura del código, la estructura de la interfaz de usuario o los datos de modelo de interfaz de usuario. Para un archivo XML, puede ver la estructura XML, la estructura del diseño o los datos de modelo de interfaz de usuario.

La ventana Structure es dinámica y realiza siempre un seguimiento de la selección actual de la ventana activa (a menos que congele el contenido de la ventana en una vista concreta), ya que está relacionada con el editor actualmente activo. Cuando la selección actual es un nodo del navegador, se asume el editor por defecto. Para cambiar la vista en la estructura de la selección actual, seleccione un separador de estructura distinto.

Ventana del Editor



The screenshot shows the Oracle SQL Developer interface. The title bar has tabs for "Hello.java", "Connection1", "ADD_JOB_HISTORY" (which is the active tab), and "SECURE_DML". Below the title bar is a toolbar with icons for Find, Save, Run, and others. The main area contains the following PL/SQL code:

```
CREATE OR REPLACE PROCEDURE add_job_history
(
    p_emp_id job_history.employee_id%type ,
    p_start_date job_history.start_date%type ,
    p_end_date job_history.end_date%type ,
    p_job_id job_history.job_id%type ,
    p_department_id job_history.department_id%type )
IS
BEGIN
    INSERT INTO job_history
    (
        employee_id, start_date, end_date, job_id, department_id
    )
    VALUES
    (
        p_emp_id, p_start_date, p_end_date, p_job_id, p_department_id
    );
END add_job_history;
```

The Oracle logo is displayed on a red horizontal bar at the bottom of the screen.

Copyright © 2010, Oracle. Todos los derechos reservados.

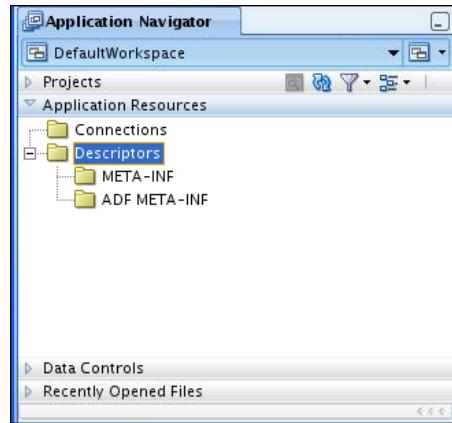
Ventana del Editor

Puede ver todos los archivos de proyectos en una única ventana del editor, abrir varias vistas del mismo archivo o abrir varias vistas de diferentes archivos.

Los separadores situados en la parte superior de la ventana del editor son los separadores del documento. Al seleccionar un separador del documento, dicho documento se enfoca y se coloca en primer plano en la ventana del editor actual.

Los separadores situados en la parte inferior de la ventana del editor para un archivo concreto son los separadores del editor. Al seleccionar un separador del editor, el archivo se abre en ese editor.

Navegador de Aplicaciones

The ORACLE logo in white text on a red background.

Copyright © 2010, Oracle. Todos los derechos reservados.

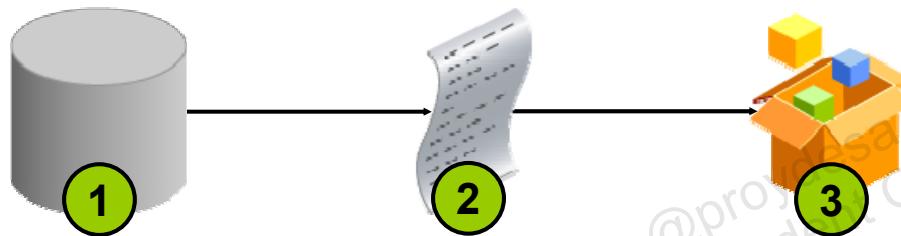
Navegador de Aplicaciones

El navegador de aplicaciones proporciona una vista lógica de la aplicación y de los datos que contiene. El navegador de aplicaciones proporciona una infraestructura a la que las distintas extensiones pueden conectarse y utilizar para organizar los datos y menús de forma abstracta y consistente. Aunque el navegador de aplicaciones puede contener archivos individuales (como archivos de origen Java), está diseñado para consolidar datos complejos. Los tipos de dato complejos, como los objetos de entidades, diagramas Unified Modeling Language (UML), Enterprise JavaBeans (EJB) o servicios Web, aparecen en el navegador como nodos únicos. Los archivos raw que componen estos nodos abstractos aparecen en la ventana Structure.

Despliegue de Procedimientos Java Almacenados

Antes de desplegar procedimientos Java almacenados, realice los siguientes pasos:

1. Cree una conexión a la base de datos.
2. Cree un perfil de despliegue.
3. Despliegue los objetos.



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Despliegue de Procedimientos Java Almacenados

Cree un perfil de despliegue para los procedimientos Java almacenados y, a continuación, despliegue las clases y, si lo desea, cualquier método estático público en JDeveloper con los valores del perfil.

El despliegue en la base de datos utiliza la información proporcionada en Deployment Profile Wizard y dos utilidades de Oracle Database:

- `loadjava` carga la clase Java que contiene los procedimientos almacenados en Oracle Database.
- `publish` genera los envoltorios específicos de llamada PL/SQL para los métodos estáticos públicos cargados. La publicación permite que se llame a los métodos Java como funciones o procedimientos PL/SQL.

Publicación de Java en PL/SQL

The screenshot shows two windows side-by-side. The left window, titled 'TrimLob.java', contains Java code for a class named 'TrimLob' with a main method. The right window, titled 'TRIMLOBPROC', contains a PL/SQL CREATE PROCEDURE statement. A green circle labeled '1' points to the Java code, and a green circle labeled '2' points to the PL/SQL code.

```
public class TrimLob
{
    public static void main (String args []) throws SQLException {
        Connection conn=null;
        if (System.getProperty("oracle.jserver.version") != null)
        {
            conn = DriverManager.getConnection("jdbc:default:connection:");
        }
        else
        {
            DriverManager.registerDriver(new oracle.jdbc.OracleDriver());
            conn = DriverManager.getConnection("jdbc:oracle:thin:scott/tiger");
        }
    }
}
```

This screenshot shows the same two windows as above, but the focus is on the right window which now displays the full PL/SQL CREATE OR REPLACE PROCEDURE statement for 'TRIMLOBPROC'. A green circle labeled '2' points to this code.

```
CREATE OR REPLACE PROCEDURE TRIMLOBPROC
as Language java
name 'TrimLob.main(java.lang.String[])';
```

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Publicación de Java en PL/SQL

La diapositiva muestra el código Java y cómo publicar el código Java en un procedimiento PL/SQL.

¿Cómo Puedo Obtener más Información sobre JDeveloper 11g?

Tema	Dirección Web
Página del Producto Oracle JDeveloper	http://www.oracle.com/technology/products/jdev/index.html
Tutoriales de Oracle JDeveloper 11g	http://www.oracle.com/technology/obe/obe11jdev/11/index.html
Documentación de los productos Oracle JDeveloper 11g	http://www.oracle.com/technology/documentation/jdev.html
Foro de discusión de Oracle JDeveloper 11g	http://forums.oracle.com/forums/forum.jspa?forumID=83



Copyright © 2010, Oracle. Todos los derechos reservados.

Resumen

En este apéndice, debe haber aprendido cómo utilizar JDeveloper para realizar las siguientes acciones:

- Mostrar las funciones clave de Oracle JDeveloper
- Crear una conexión a base de datos en JDeveloper
- Gestionar objetos de base de datos en JDeveloper
- Utilizar JDeveloper para ejecutar comandos SQL
- Crear y ejecutar unidades de programa PL/SQL



Copyright © 2010, Oracle. Todos los derechos reservados.

Generación de Informes Agrupando Datos Relacionados

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

Al finalizar este apéndice, debería estar capacitado para utilizar:

- La operación ROLLUP para generar valores subtotales
- La operación CUBE para generar valores matrices
- La función GROUPING para identificar los valores de fila creados por ROLLUP o CUBE
- GROUPING SETS para generar un juego de resultados único



Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

En este apéndice, aprenderá cómo:

- Agrupar datos para obtener los valores subtotales mediante el operador ROLLUP
- Agrupar datos para obtener los valores matrices mediante el operador ROLLUP
- Utilizar la función GROUPING para identificar el nivel de agregación en el juego de resultados generados por un operador ROLLUP o CUBE
- Utilizar GROUPING SETS para generar un juego de resultados único equivalente al enfoque UNION ALL

Revisión de Funciones de Grupo

- Las funciones de grupo funcionan en juegos de filas para proporcionar un resultado por grupo.

```
SELECT      [column,] group_function(column) . . .
FROM        table
[WHERE      condition]
[GROUP BY   group_by_expression]
[ORDER BY   column];
```

- Ejemplo:

```
SELECT AVG(salary), STDDEV(salary),
       COUNT(commission_pct), MAX(hire_date)
  FROM employees
 WHERE job_id LIKE 'SA%';
```

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Funciones de Grupo

Puede utilizar la cláusula GROUP BY para dividir las filas de la tabla en grupos. A continuación, puede utilizar las funciones de grupo para devolver información de resumen de cada grupo. Las funciones de grupo pueden aparecer en listas de selección en las cláusulas ORDER BY y HAVING. El servidor de Oracle aplica las funciones de grupo a cada grupo de filas y devuelve una fila de resultados única para cada grupo.

Tipos de funciones de grupo: cada una de las funciones del grupo (AVG, SUM, MAX, MIN, COUNT, STDDEV y VARIANCE) acepta un argumento. Las funciones AVG, SUM, STDDEV y VARIANCE sólo operan en valores numéricos. MAX y MIN pueden operar en valores numéricos, de caracteres o de fecha. COUNT devuelve el número de filas no nulas para una determinada expresión. El ejemplo de la diapositiva calcula el salario medio, la desviación estándar del salario, el número de empleados que ganan una comisión y la fecha máxima de contratación para aquellos empleados cuyo JOB_ID empiece por SA.

Instrucciones para Utilizar Funciones de Grupo

- Los tipos de dato para el argumento pueden ser CHAR, VARCHAR2, NUMBER o DATE.
- Todas las funciones de grupo excepto COUNT (*) ignoran los valores nulos. Para sustituir un valor por valores nulos, utilice la función NVL. COUNT devuelve un número o cero.
- El servidor de Oracle ordena implícitamente el juego de resultados en orden ascendente de las columnas del agrupamiento especificadas, al utilizar una cláusula GROUP BY. Para sustituir este orden por defecto, puede utilizar DESC en una cláusula ORDER BY.

Revisión de la Cláusula GROUP BY

- Sintaxis:

```
SELECT      [column,] group_function(column) . . .
FROM        table
[WHERE      condition]
[GROUP BY   group_by_expression]
[ORDER BY   column];
```

- Ejemplo:

```
SELECT      department_id, job_id, SUM(salary),
            COUNT(employee_id)
FROM        employees
GROUP BY   department_id, job_id,
```



Copyright © 2010, Oracle. Todos los derechos reservados.

Revisión de la Cláusula GROUP BY

El servidor de Oracle evalúa el ejemplo ilustrado en la diapositiva de la siguiente forma:

- La cláusula SELECT especifica que se van a recuperar las siguientes columnas:
 - Columnas de ID de departamento e ID de cargo de la tabla EMPLOYEES
 - Suma de todos los salarios y número de empleados de cada grupo especificado en la cláusula GROUP BY
- La cláusula GROUP BY especifica cómo se deben agrupar las filas en la tabla. El salario total y el número de empleados se calcula para cada ID de cargo dentro de cada departamento. Las filas se agrupan por ID de departamento y, a continuación, por cargo dentro de cada departamento.

Revisión de la Cláusula HAVING

- Utilice la cláusula HAVING para especificar los grupos que se deben mostrar.
- A continuación, restrinja los grupos en función de una condición de limitación.

```
SELECT      [column,] group_function(column) ...
FROM        table
[WHERE       condition]
[GROUP BY   group_by_expression]
[HAVING     having_expression]
[ORDER BY   column];
```



Copyright © 2010, Oracle. Todos los derechos reservados.

Cláusula HAVING

Los grupos están formados y las funciones de grupo se calculan antes de aplicar la cláusula HAVING a los grupos. La cláusula HAVING puede preceder a la cláusula GROUP BY, pero se recomienda que coloque la cláusula GROUP BY primero porque es más lógico.

El servidor de Oracle realiza los siguientes pasos al utilizar la cláusula HAVING:

1. Agrupa las filas.
2. Aplica las funciones de grupo a los grupos y muestra los grupos que coinciden con los criterios de la cláusula HAVING.

GROUP BY con los Operadores ROLLUP y CUBE

- Utilice ROLLUP o CUBE con GROUP BY para generar filas superagregadas por columnas de referencia cruzada.
- El agrupamiento ROLLUP genera un juego de resultados que contiene las filas agrupadas normales y los valores subtotales.
- El agrupamiento CUBE genera un juego de resultados que contiene las filas de ROLLUP y las filas matrices.



Copyright © 2010, Oracle. Todos los derechos reservados.

GROUP BY con los Operadores ROLLUP y CUBE

Puede especificar los operadores ROLLUP y CUBE en la cláusula GROUP BY de una consulta. El agrupamiento ROLLUP genera un juego de resultados que contiene las filas agrupadas normales y las filas de subtotal. El operador ROLLUP también calcula una suma total. La operación CUBE de la cláusula GROUP BY agrupa las filas seleccionadas según los valores de todas las posibles combinaciones de expresiones en la especificación y devuelve una única fila de información de resumen para cada grupo. Puede utilizar el operador CUBE para producir filas matrices.

Nota: al trabajar con ROLLUP y CUBE, asegúrese de que las columnas que siguen a la cláusula GROUP BY tienen relaciones significativas, de la vida real entre sí; de lo contrario, los operadores devuelven información irrelevante.

Operador ROLLUP

- ROLLUP es una extensión de la cláusula GROUP BY.
- Utilice la operación ROLLUP para generar agregados acumulados, como los subtotales.

```
SELECT      [column,] group_function(column) . . .
FROM        table
[WHERE      condition]
[GROUP BY   [ROLLUP] group_by_expression]
[HAVING    having_expression];
[ORDER BY   column];
```

Copyright © 2010, Oracle. Todos los derechos reservados.

Operador ROLLUP

El operador ROLLUP proporciona agregados y superagregados para las expresiones de una sentencia GROUP BY. Los escritores de informes pueden utilizar el operador ROLLUP para extraer estadísticas e información de resumen de los juegos de resultados. Los agregados acumulados se pueden utilizar en informes, diagramas y gráficos.

El operador ROLLUP crea agrupamientos desplazándose en una dirección, de derecha a izquierda, por la lista de columnas especificadas en la cláusula GROUP BY. A continuación, aplica la función de agregación a estos agrupamientos.

Nota

- Para producir subtotales en n dimensiones (es decir, n columnas de la cláusula GROUP BY) sin un operador ROLLUP, las sentencias $n+1$ SELECT se deben enlazar mediante UNION ALL. Esto hace que la ejecución de la consulta no sea eficaz porque cada una de las sentencias SELECT provoca el acceso a la tabla. El operador ROLLUP recopila sus resultados con tan sólo un acceso a la tabla. El operador ROLLUP es útil cuando hay varias columnas implicadas en la producción de los subtotales.
- Los subtotales y totales se generan con ROLLUP. CUBE también genera totales, pero se acumula de forma efectiva en cualquier dirección posible, produciendo datos matrices.

Operador ROLLUP: Ejemplo

```
SELECT      department_id, job_id, SUM(salary)
FROM        employees
WHERE       department_id < 60
GROUP BY    ROLLUP(department_id, job_id);
```

#	DEPARTMENT_ID	JOB_ID	SUM(SALARY)
1		10 AD_ASST	4400
2		10 (null)	4400
3		20 MK_MAN	13000
4		20 MK_REP	6000
5		20 (null)	19000
6		30 PU_MAN	11000
7		30 PU_CLERK	13900
8		30 (null)	24900
9		40 HR REP	6500
10		40 (null)	6500
11		50 ST_MAN	36400
12		50 SH_CLERK	64300
13		50 ST_CLERK	55700
14		50 (null)	156400
15	(null) (null)		211200

- 1
- 2
- 3

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Ejemplo de un Operador ROLLUP

En el ejemplo de la diapositiva:

- Se muestran los salarios totales para cada ID de cargo de un departamento para aquellos departamentos cuyo ID sea inferior a 60 a través de la cláusula GROUP BY
- El operador ROLLUP muestra:
 - El salario total de cada departamento cuyo ID sea inferior a 60.
 - El salario total para todos los departamentos cuyo ID sea inferior a 60, independientemente de los ID de cargo.

En este ejemplo, 1 indica un grupo cuyo total se ha calculado mediante DEPARTMENT_ID y JOB_ID, 2 indica un grupo cuyo total se ha calculado sólo mediante DEPARTMENT_ID y 3 indica la suma total.

El operador ROLLUP crea subtotales que se acumulan desde el nivel más detallado hasta la suma total, siguiendo la lista de agrupamiento especificada en la cláusula GROUP BY. En primer lugar, calcula los valores agregados estándar para los grupos especificados en la cláusula GROUP BY (en el ejemplo, la suma de salarios agrupados en cada cargo de un departamento). A continuación, crea subtotales de nivel superior de forma progresiva, de derecha a izquierda a través de la lista de columnas del agrupamiento. (En el ejemplo, se calcula la suma de salarios para cada departamento, seguida de la suma de salarios para todos los departamentos).

- Dadas n expresiones en el operador ROLLUP de la cláusula GROUP BY, la operación genera $n + 1$ (en este caso, $2 + 1 = 3$) agrupamientos.
- Las filas que se basan en los valores de las primeras n expresiones se denominan filas o filas normales y las demás se denominan filas superagregadas.

Operador CUBE

- CUBE es una extensión de la cláusula GROUP BY.
- Puede utilizar el operador CUBE para generar valores matrices con una única sentencia SELECT.

```

SELECT      [column,] group_function(column) ...
FROM        table
[WHERE       condition]
[GROUP BY   [CUBE] group_by_expression]
[HAVING      having_expression]
[ORDER BY   column];
  
```

Copyright © 2010, Oracle. Todos los derechos reservados.

Operador CUBE

El operador CUBE es un conmutador adicional de la cláusula GROUP BY en una sentencia SELECT. El operador CUBE se puede aplicar a todas las funciones agregadas, incluidas AVG, SUM, MAX, MIN y COUNT. Se utiliza para generar juegos de resultados que se suelen utilizar para informes de matriz. ROLLUP genera sólo una parte de las posibles combinaciones de subtotales, mientras que CUBE genera subtotales para todas las posibles combinaciones de agrupamientos especificadas en la cláusula GROUP BY y una suma total.

El operador CUBE se utiliza con una función agregada para generar filas adicionales en un juego de resultados. Las columnas incluidas en la cláusula GROUP BY están incluidas en referencias cruzadas para generar un superjuego de grupos. La función agregada especificada en la lista de selección se aplica a estos grupos para generar valores de resumen para las filas superagregadas adicionales. El número de grupos adicionales del juego de resultados viene determinado por el número de columnas incluidas en la cláusula GROUP BY.

De hecho, todas las combinaciones posibles de columnas o expresiones de la cláusula GROUP BY se utilizan para generar superagregados. Si tiene n columnas o expresiones en la cláusula GROUP BY, habrá 2^n posibles combinaciones de superagregados. Matemáticamente, estas combinaciones forman un cubo de n dimensiones, que es de donde procede el nombre del operador.

Mediante la aplicación o las herramientas de programación, estos valores superagregados se pueden incluir en diagramas y gráficos que muestran los resultados y las relaciones de forma visual y efectiva.

Operador CUBE: Ejemplo

```
SELECT department_id, job_id, SUM(salary)
FROM employees
WHERE department_id < 60
GROUP BY CUBE (department_id, job_id) ;
```

	DEPARTMENT_ID	JOB_ID	SUM(SALARY)
1	(null) (null)		211200
2	(null) HR_REP		6500
3	(null) MK_MAN		13000
4	(null) MK_REP		6000
5	(null) PU_MAN		11000
6	(null) ST_MAN		36400
7	(null) AD_ASST		4400
8	(null) PU_CLERK		13900
9	(null) SH_CLERK		64300
10	(null) ST_CLERK		55700
11	10 (null)		4400
12	10 AD_ASST		4400
13	20 (null)		19000
14	20 MK_MAN		13000
15	20 MK_REP		6000
16	30 (null)		24900

- 1
- 2
- 3
- 4

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Ejemplo de un Operador CUBE

La salida de la sentencia SELECT del ejemplo se puede interpretar de la siguiente forma:

- El salario total para cada cargo de un departamento (para aquellos departamentos cuyo ID sea inferior a 60)
- El salario total de cada departamento cuyo ID sea inferior a 60
- El salario total para cada cargo, independientemente del departamento
- El salario total para aquellos departamentos cuyo ID sea inferior a 60, independientemente de los puestos

En este ejemplo, 1 indica la suma total, 2 indica las filas calculadas sólo por JOB_ID, 3 indica algunas de las filas calculadas por DEPARTMENT_ID y JOB_ID, y 4 indica algunas de las filas calculadas sólo por DEPARTMENT_ID.

El operador CUBE ha realizado también la operación ROLLUP para mostrar los subtotales para aquellos departamentos cuyo ID es inferior a 60 y el salario total para aquellos departamentos cuyo ID sea inferior a 60, independientemente de los puestos. Además, el operador CUBE muestra el salario total para cada cargo, independientemente del departamento.

Nota: al igual que el operador ROLLUP, que genera subtotales en n dimensiones (es decir, n columnas en la cláusula GROUP BY), sin un operador CUBE se necesita que 2^n sentencias SELECT estén enlazadas con UNION ALL. Por lo tanto, un informe con las tres dimensiones necesita que $2^3 = 8$ sentencias SELECT estén enlazadas con UNION ALL.

Función GROUPING

Función GROUPING:

- Se utiliza con el operador CUBE o ROLLUP.
- Se utiliza para buscar grupos que formen el subtotal en una fila.
- Se utiliza para diferenciar los valores NULL almacenados de los valores NULL creados por ROLLUP o CUBE.
- Devuelve 0 o 1.

```
SELECT      [column,] group_function(column) . . ,
            GROUPING(expr)
  FROM       table
  [WHERE     condition]
  [GROUP BY [ROLLUP] [CUBE] group_by_expression]
  [HAVING   having_expression]
  [ORDER BY column];
```

Copyright © 2010, Oracle. Todos los derechos reservados.

Función GROUPING

La función GROUPING se puede utilizar con el operador CUBE o ROLLUP para ayudarle a entender cómo se ha obtenido un valor de resumen.

La función GROUPING utiliza una única columna como su argumento. *expr* en la función GROUPING debe coincidir con una de las expresiones de la cláusula GROUP BY. La función devuelve un valor de 0 o 1.

Los valores devueltos por la función GROUPING son útiles para:

- Determinar el nivel de agregación de un determinado subtotal (es decir, el grupo o los grupos en los que se basa el subtotal)
- Identificar si un valor NULL de la columna de expresión de una fila del juego de resultados indica:
 - Un valor NULL de la tabla base (valor NULL almacenado)
 - Un valor NULL creado por ROLLUP o CUBE (como resultado de una función de grupo en la expresión)

Un valor de 0 devuelto por la función GROUPING basada en una expresión indica una de las siguientes opciones:

- La expresión se ha utilizado para calcular el valor agregado.
- El valor NULL de la columna de expresión es un valor NULL almacenado.

Un valor de 1 devuelto por la función GROUPING basada en una expresión indica una de las siguientes opciones:

- La expresión se ha utilizado para calcular el valor agregado.
- El valor NULL de la columna de expresión lo ha creado ROLLUP o CUBE como resultado de un agrupamiento.

Función GROUPING: Ejemplo

```

SELECT      department_id DEPTID, job_id JOB,
            SUM(salary),
            GROUPING(department_id) GRP_DEPT,
            GROUPING(job_id) GRP_JOB
FROM        employees
WHERE       department_id < 50
GROUP BY   ROLLUP(department_id, job_id);
    
```

The table shows the results of the SQL query. The columns are DEPTID, JOB, SUM(SALARY), GRP_DEPT, and GRP_JOB. The data rows are:

	DEPTID	JOB	SUM(SALARY)	GRP_DEPT	GRP_JOB
1	10	AD_ASST	4400	0	0
2	10	(null)	4400	0	1
3	20	MK_MAN	13000	0	0
4	20	MK_REP	6000	0	0
5	20	(null)	19000	0	1
6	30	PU_MAN	11000	0	0
7	30	PU_CLERK	13900	0	0
8	30	(null)	24900	0	1
9	40	HR REP	6500	0	0
10	40	(null)	6500	0	1
11	(null)	(null)	54800	1	1

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Ejemplo de una Función GROUPING

En el ejemplo de la diapositiva, observe el valor de resumen 4400 de la primera fila (con la etiqueta 1). Este valor de resumen es el salario total para el ID de cargo AD_ASST en el departamento 10. Para calcular este valor de resumen, se han tenido en cuenta las columnas DEPARTMENT_ID y JOB_ID. Por lo tanto, se devuelve un valor de 0 tanto para la expresión GROUPING(department_id) como para GROUPING(job_id).

Observe el valor de resumen 4400 de la segunda fila (con la etiqueta 2). Este valor es el salario total para el departamento 10 y se ha calculado teniendo en cuenta la columna DEPARTMENT_ID; por lo tanto, GROUPING(department_id) ha devuelto un valor de 0. Puesto que no se ha tenido en cuenta la columna JOB_ID para calcular este valor, se ha devuelto un valor de 1 para GROUPING(job_id). Puede observar una salida similar en la quinta fila.

En la última fila, observe el valor 54800 (con la etiqueta 3). Se trata del salario total para aquellos departamentos cuyo ID sea inferior a 50 y todos los puestos. Para calcular este valor de resumen, se han tenido en cuenta las columnas DEPARTMENT_ID y JOB_ID. Por lo tanto, se devuelve un valor de 1 tanto para la expresión GROUPING(department_id) como para GROUPING(job_id).

GROUPING SETS

- La sintaxis GROUPING SETS se utiliza para definir varios agrupamientos en la misma consulta.
- Se calculan todos los agrupamientos especificados en la cláusula GROUPING SETS y los resultados de los agrupamientos individuales se combinan con una operación UNION ALL.
- Eficacia del juego de agrupamientos:
 - Sólo se necesita una pasada rápida por la tabla base.
 - No es necesario escribir sentencias UNION complejas.
 - Cuantos más elementos tiene GROUPING SETS, mayor es la ventaja de rendimiento.



Copyright © 2010, Oracle. Todos los derechos reservados.

GROUPING SETS

GROUPING SETS es una extensión adicional de la cláusula GROUP BY que puede utilizar para especificar varios agrupamientos de datos. Esta acción facilita la agregación eficaz y, por lo tanto, facilita el análisis de los datos a través de varias dimensiones.

Ahora se puede escribir una única sentencia SELECT mediante GROUPING SETS para especificar varios agrupamientos (que pueden incluir también los operadores ROLLUP o CUBE), en lugar de varias sentencias SELECT combinadas a través de operadores UNION ALL. Por ejemplo:

```
SELECT department_id, job_id, manager_id, AVG(salary)
  FROM employees
 GROUP BY
 GROUPING SETS
 ((department_id, job_id, manager_id),
 (department_id, manager_id), (job_id, manager_id));
```

Esta sentencia calcula los agregados en tres agrupamientos:

```
(department_id, job_id, manager_id), (department_id,
manager_id) and (job_id, manager_id)
```

Sin esta función, se necesitan varias consultas combinadas a través de UNION ALL para obtener la salida de la sentencia SELECT anterior. Un enfoque de varias consultas resulta ineficaz porque necesita varias exploraciones de los mismos datos.

GROUPING SETS (continuación)

Comparación del ejemplo anterior con la siguiente alternativa:

```
SELECT department_id, job_id, manager_id, AVG(salary)
FROM employees
GROUP BY CUBE(department_id, job_id, manager_id);
```

Esta sentencia calcula los $8 (2^3)$ agrupamientos, aunque sólo le interesan los grupos $(\text{department_id}, \text{job_id}, \text{manager_id})$, $(\text{department_id}, \text{manager_id})$ y $(\text{job_id}, \text{manager_id})$.

Otra alternativa es la siguiente sentencia:

```
SELECT department_id, job_id, manager_id, AVG(salary)
FROM employees
GROUP BY department_id, job_id, manager_id
UNION ALL
SELECT department_id, NULL, manager_id, AVG(salary)
FROM employees
GROUP BY department_id, manager_id
UNION ALL
SELECT NULL, job_id, manager_id, AVG(salary)
FROM employees
GROUP BY job_id, manager_id;
```

Esta sentencia necesita tres exploraciones de la tabla base, lo que la hace ineficaz.

CUBE y ROLLUP se pueden considerar juegos de agrupamientos con una semántica y unos resultados muy específicos. Las siguientes equivalencias muestran este hecho:

CUBE(a, b, c) es equivalente a	GROUPING SETS ((a, b, c), (a, b), (a, c), (b, c), (a), (b), (c), ())
ROLLUP(a, b, c) es equivalente a	GROUPING SETS ((a, b, c), (a, b), (a), ())

GROUPING SETS: Ejemplo

```
SELECT      department_id, job_id,
            manager_id, AVG(salary)
FROM        employees
GROUP BY    GROUPING SETS
            ((department_id,job_id), (job_id,manager_id));
```

#	DEPARTMENT_ID	JOB_ID	MANAGER_ID	Avg(SALARY)
1	(null)	SH_CLERK	122	3200
2	(null)	AC_MGR	101	12000
3	(null)	ST_MAN	100	7280
4	...	ST_CLERK	121	2675

1

#	DEPARTMENT_ID	JOB_ID	MANAGER_ID	Avg(SALARY)
39	110	AC_MGR	(null)	12000
40	90	AD_PRES	(null)	24000
41	60	IT_PROG	(null)	5760
42	100	FL_MGR	(null)	12000

2

Copyright © 2010, Oracle. Todos los derechos reservados.

ORACLE

GROUPING SETS: Ejemplo

La consulta de la diapositiva calcula los agregados de dos agrupamientos. La tabla está dividida en los siguientes grupos:

- Department ID, Job ID
- Job ID, Manager ID

Se calculan los salarios medios de cada uno de estos grupos. El juego de resultados muestra el salario medio de cada uno de los dos grupos.

En la salida, el grupo marcado como 1 se puede interpretar de la siguiente forma:

- El salario medio de todos los empleados con el ID de cargo SH_CLERK que dependen del gestor 122 es de 3.200.
- El salario medio de todos los empleados con el ID de cargo AC_MGR que dependen del gestor 101 es de 12.000 y así sucesivamente.

El grupo marcado como 2 en la salida se interpreta de la siguiente forma:

- El salario medio de todos los empleados con el ID de cargo AC_MGR del departamento 110 es de 12.000.
- El salario medio de todos los empleados con el ID de cargo AD_PRES del departamento 90 es de 24.000 y así sucesivamente.

GROUPING SETS: Ejemplo (continuación)

El ejemplo de la diapositiva también se puede escribir como:

```
SELECT department_id, job_id, NULL as manager_id,  
       AVG(salary) as AVGSAL  
  FROM employees  
 GROUP BY department_id, job_id  
UNION ALL  
SELECT NULL, job_id, manager_id, avg(salary) as AVGSAL  
  FROM employees  
 GROUP BY job_id, manager_id;
```

En ausencia de un optimizador que compruebe los bloques de consulta para generar el plan de ejecución, la consulta anterior necesitaría dos exploraciones de la tabla base, EMPLOYEES. Esto podría ser muy ineficaz. Por lo tanto, se recomienda el uso de la sentencia GROUPING SETS.

Columnas Compuestas

- Una columna compuesta es una recopilación de columnas que se tratan como una unidad.
ROLLUP (a, **(b, c)**, d)
- Utilice paréntesis en la cláusula GROUP BY para agrupar columnas, de modo que se traten como una unidad al calcular las operaciones ROLLUP o CUBE.
- Cuando se utilizan con ROLLUP o CUBE, las columnas compuestas necesitarían omitir la agregación en ciertos niveles.



Copyright © 2010, Oracle. Todos los derechos reservados.

Columnas Compuestas

Una columna compuesta es una recopilación de columnas que se trata como una unidad durante el cálculo de los agrupamientos. Especifique las columnas entre paréntesis como se muestra en la siguiente sentencia: ROLLUP (a, (b, c), d)

Aquí, (b, c) forma una columna compuesta y se trata como una unidad. En general, las columnas compuestas son útiles en ROLLUP, CUBE y GROUPING SETS. Por ejemplo, en CUBE o ROLLUP, las columnas compuestas necesitarían omitir la agregación en ciertos niveles.

Es decir, GROUP BY ROLLUP (a, (b, c)) es equivalente a:

```
GROUP BY a, b, c UNION ALL
GROUP BY a UNION ALL
GROUP BY ()
```

Aquí, (b, c) se trata como unidad y ROLLUP no se aplica en (b, c). Es como si tuviera un alias, por ejemplo, z como alias de (b, c), y la expresión GROUP BY se reduce a: GROUP BY ROLLUP (a, z).

Nota: GROUP BY () suele ser una sentencia SELECT con valores NULL para las columnas a y b y sólo la función de agregación. Normalmente se utiliza para generar sumas totales.

```
SELECT    NULL, NULL, aggregate_col
FROM      <table_name>
GROUP BY  () ;
```

Columnas Compuestas (continuación)

Comparación con una operación ROLLUP normal como en:

```
GROUP BY ROLLUP(a, b, c)
```

Podría ser:

```
GROUP BY a, b, c UNION ALL
GROUP BY a, b UNION ALL
GROUP BY a UNION ALL
GROUP BY ()
```

Del mismo modo:

```
GROUP BY CUBE((a, b), c)
```

Sería equivalente a:

```
GROUP BY a, b, c UNION ALL
GROUP BY a, b UNION ALL
GROUP BY c UNION ALL
GROUP BY ()
```

En la siguiente tabla se muestra la especificación GROUPING SETS y la especificación GROUP BY equivalente.

Sentencias GROUPING SETS	Sentencias GROUP BY Equivalentes
GROUP BY GROUPING SETS(a, b, c)	GROUP BY a UNION ALL GROUP BY b UNION ALL GROUP BY c
GROUP BY GROUPING SETS(a, b, (b, c)) (La expresión GROUPING SETS tiene una columna compuesta).	GROUP BY a UNION ALL GROUP BY b UNION ALL GROUP BY b, c
GROUP BY GROUPING SETS((a, b, c))	GROUP BY a, b, c
GROUP BY GROUPING SETS(a, (b), ())	GROUP BY a UNION ALL GROUP BY b UNION ALL GROUP BY ()
GROUP BY GROUPING SETS (a,ROLLUP(b, c)) (La expresión GROUPING SETS tiene una columna compuesta).	GROUP BY a UNION ALL GROUP BY ROLLUP(b, c)

Columnas Compuestas: Ejemplo

```
SELECT      department_id, job_id, manager_id,
            SUM(salary)
FROM        employees
GROUP BY    ROLLUP( department_id,(job_id, manager_id));
```

The diagram illustrates the four levels of ROLLUP grouping:

- Level 1 (Root):** Department ID (1, 2, 3, 4, 5, 6, 7) and their corresponding job IDs and manager IDs.
- Level 2 (First Level of Aggregation):** Department ID (1, 2, 3, 4, 5, 6, 7) and their corresponding sum of salaries.
- Level 3 (Second Level of Aggregation):** Department ID (40, 41, 42, 43, 44, 45, 46) and their corresponding sum of salaries.
- Level 4 (Third Level of Aggregation):** Department ID (40, 41, 42, 43, 44, 45, 46) and their corresponding sum of salaries.

	DEPARTMENT_ID	JOB_ID	MANAGER_ID	SUM(SALARY)
1	(null) SA_REP	149	7000	
2	(null) (null)	(null)	7000	
3	10 AD_ASST	101	4400	
4	10 (null)	(null)	4400	
5	20 MK_MAN	100	13000	
6	20 MK_REP	201	6000	
7	20 (null)	(null)	19000	
...				
40	100 FL_MGR	101	12000	
41	100 FL_ACCOUNT	108	39600	
42	100 (null)	(null)	51600	
43	110 AC_MGR	101	12000	
44	110 AC_ACCOUNT	205	8300	
45	110 (null)	(null)	20300	
46	(null) (null)	(null)	691400	

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Columnas Compuestas: Ejemplo

Consideré este ejemplo:

```
SELECT department_id, job_id,manager_id, SUM(salary)
      FROM employees
 GROUP BY ROLLUP( department_id,job_id, manager_id);
```

El resultado de esta consulta es el cálculo de los siguientes agrupamientos:

- (job_id, manager_id)
- (department_id, job_id, manager_id)
- (department_id)
- Suma total

Si sólo le interesan determinados grupos, no puede limitar el cálculo a dichos agrupamientos sin utilizar columnas compuestas. Con las columnas compuestas, esto es posible si las columnas JOB_ID y MANAGER_ID se tratan como una única unidad durante la acumulación. Las columnas entre paréntesis se tratan como una unidad al calcular ROLLUP y CUBE. Esto se ilustra en el ejemplo de la diapositiva. Al incluir las columnas JOB_ID y MANAGER_ID entre paréntesis, indica que el servidor de Oracle debe tratar JOB_ID y MANAGER_ID como una única unidad, es decir una columna compuesta.

Columnas Compuestas: Ejemplo (continuación)

El ejemplo de la diapositiva calcula los siguientes agrupamientos:

- (department_id, job_id, manager_id)
- (department_id)
- ()

El ejemplo de la diapositiva muestra los siguientes registros:

- Salario total para cada cargo y gestor (con la etiqueta 1)
- Salario total para cada departamento, cargo y gestor (con la etiqueta 2)
- Salario total para cada departamento (con la etiqueta 3)
- Suma total (con la etiqueta 4)

El ejemplo de la diapositiva también se puede escribir como:

```
SELECT department_id, job_id, manager_id, SUM(salary)
  FROM employees
 GROUP BY department_id, job_id, manager_id
UNION ALL
SELECT department_id, TO_CHAR(NULL), TO_NUMBER(NULL),
       SUM(salary)
  FROM employees
 GROUP BY department_id
UNION ALL
SELECT TO_NUMBER(NULL), TO_CHAR(NULL), TO_NUMBER(NULL),
       SUM(salary)
  FROM employees
 GROUP BY ( );
```

En ausencia de un optimizador que compruebe los bloques de consulta para generar el plan de ejecución, la consulta anterior necesitaría tres exploraciones de la tabla base, EMPLOYEES. Esto podría ser muy ineficaz. Por lo tanto, se recomienda el uso de columnas compuestas.

Agrupamientos Concatenados

- Los agrupamientos concatenados ofrecen una forma concisa de generar combinaciones útiles de agrupamientos.
- Para especificar juegos de agrupamientos concatenados, separe varios juegos de agrupamientos, las operaciones ROLLUP y CUBE con comas para que el servidor de Oracle las combine en una única cláusula GROUP BY.
- El resultado es un producto combinado de agrupamientos de cada GROUPING SET.

```
GROUP BY GROUPING SETS(a, b), GROUPING SETS(c, d)
```



Copyright © 2010, Oracle. Todos los derechos reservados.

Agrupamientos Concatenados

Los agrupamientos concatenados ofrecen una forma concisa de generar combinaciones útiles de agrupamientos. Los agrupamientos concatenados se especifican mediante listas de varios juegos de agrupamientos, CUBE y ROLLUP, y separados con comas. A continuación, se muestra un ejemplo de juegos de agrupamientos concatenados:

```
GROUP BY GROUPING SETS(a, b), GROUPING SETS(c, d)
```

Este ejemplo de SQL define los siguientes agrupamientos:

(a, c), (a, d), (b, c), (b, d)

La concatenación de juegos de agrupamientos es muy útil por estos motivos:

- **Facilidad de desarrollo de consultas:** no tiene que enumerar manualmente todos los agrupamientos.
- **Uso por parte de las aplicaciones:** el SQL generado por aplicaciones de procesamiento analítico en línea (OLAP) implica normalmente la concatenación de juegos de agrupamiento, con cada GROUPING SET que define los agrupamientos necesarios para una dimensión.

Agrupamientos Concatenados: Ejemplo

```
SELECT      department_id, job_id, manager_id,
            SUM(salary)
FROM        employees
GROUP BY    department_id,
            ROLLUP(job_id),
            CUBE(manager_id);
```

	DEPARTMENT_ID	JOB_ID	MANAGER_ID	SUM(SALARY)
1	1	(null) SA_REP	149	7000
	2	10 AD_ASST	101	4400
	3	20 MK_MAN	100	13000
	4	20 MK_REP	201	6000
		...		
1		90 AD_VP	100	34000
		90 AD_PRES	(null)	24000
		...		
		(null) SA_REP	(null)	7000
		10 AD_ASST	(null)	4400
		...		
2		110 (null)	101	12000
		110 (null)	205	8300
		110 (null)	(null)	20300
3				

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Agrupamientos Concatenados: Ejemplo

El ejemplo de la diapositiva genera los siguientes agrupamientos:

- (department_id, job_id,) (1)
- (department_id, manager_id) (2)
- (department_id) (3)

Se calcula el salario total de cada uno de estos grupos.

A continuación, se muestra un ejemplo de un agrupamiento concatenado.

```
SELECT department_id, job_id, manager_id, SUM(salary) totsal
FROM employees
WHERE department_id<60
GROUP BY GROUPING SETS(department_id),
GROUPING SETS (job_id, manager_id);
```

Resumen

En este apéndice, debe haber aprendido cómo utilizar:

- La operación ROLLUP para generar valores subtotales
- La operación CUBE para generar valores de matriz
- La función GROUPING para identificar los valores de fila creados por ROLLUP o CUBE
- Sintaxis GROUPING SETS para definir varios agrupamientos en la misma consulta
- La cláusula GROUP BY para combinar expresiones de diferentes formas:
 - Columnas compuestas
 - Juegos de agrupamientos concatenados



Copyright © 2010, Oracle. Todos los derechos reservados.

Resumen

- ROLLUP y CUBE son extensiones de la cláusula GROUP BY.
- ROLLUP se utiliza para mostrar los valores de subtotal y suma total.
- CUBE se utiliza para mostrar los valores de matriz.
- La función GROUPING permite determinar si una fila es un agregado generado por un operador CUBE o ROLLUP.
- Con la sintaxis GROUPING SETS, puede definir varios agrupamientos en la misma consulta. GROUP BY calcula todos los agrupamientos especificados y los combina a través de UNION ALL.
- En la cláusula GROUP BY, puede combinar expresiones de distintas formas:
 - Para especificar columnas compuestas, agrupe las columnas entre paréntesis para que el servidor de Oracle las trate como una unidad al calcular las operaciones ROLLUP o CUBE.
 - Para especificar juegos de agrupamientos concatenados, separe varios juegos de agrupamientos, las operaciones ROLLUP y CUBE con comas para que el servidor de Oracle las combine en una única cláusula GROUP BY. El resultado es un producto combinado de agrupamientos de cada juego de agrupamiento.

Recuperación Jerárquica

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

Al finalizar este apéndice, debería estar capacitado para lo siguiente:

- Interpretar el concepto de una consulta jerárquica
- Crear un informe con estructura de árbol
- Datos jerárquicos de formato
- Excluir ramas de la estructura de árbol

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

En este apéndice aprenderá a utilizar las consultas jerárquicas para crear informes con estructura de árbol.

Datos de Ejemplo de la Tabla EMPLOYEES

	EMPLOYEE_ID	LAST_NAME	JOB_ID	MANAGER_ID
1	100	King	AD_PRES	(null)
2	101	Kochhar	AD_VP	100
3	102	De Haan	AD_VP	100
4	103	Hunold	IT_PROG	102
5	104	Ernst	IT_PROG	103
6	107	Lorentz	IT_PROG	103

...

16	200	Whalen	AD_ASST	101
17	201	Hartstein	MK_MAN	100
18	202	Fay	MK_REP	201
19	205	Higgins	AC_MGR	101
20	206	Gietz	AC_ACCOUNT	205



Copyright © 2010, Oracle. Todos los derechos reservados.

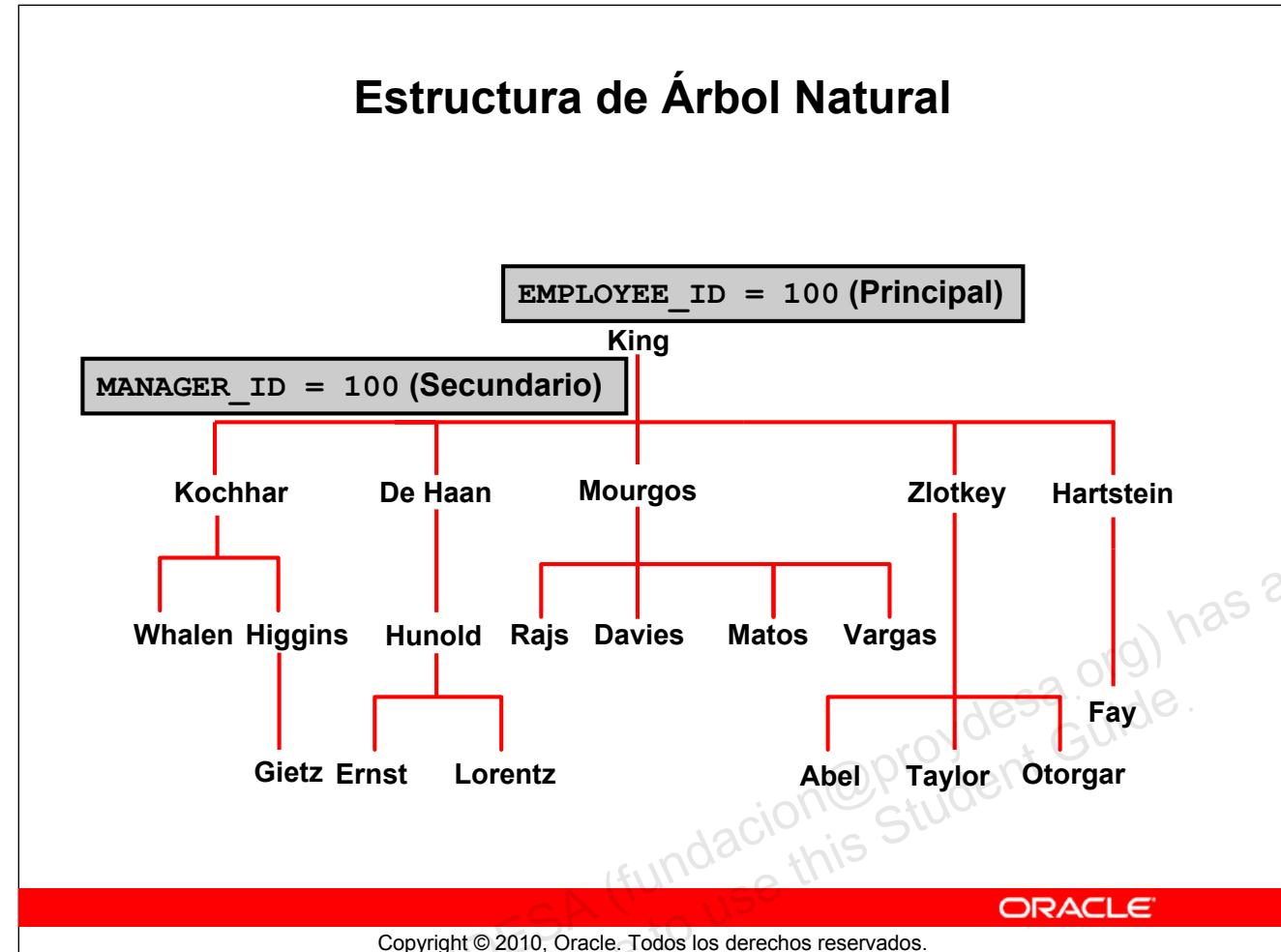
Datos de Ejemplo de la Tabla EMPLOYEES

Con las consultas jerárquicas, puede recuperar datos según una relación jerárquica natural entre las filas de una tabla. Una base de datos relacional no almacena los registros de forma jerárquica. Sin embargo, cuando existe una relación jerárquica entre las filas de una única tabla, un proceso denominado *recorrido por el árbol* permite construir la jerarquía. Una consulta jerárquica es un método de creación de informes, con las ramas de un árbol en un determinado orden.

Imagine un árbol familiar en el que los miembros más mayores de la familia se encuentran cerca de la base o el tronco del árbol y los más jóvenes representan las ramas del árbol. Las ramas pueden tener sus propias ramas y así sucesivamente.

Una consulta jerárquica es posible si existe una relación entre las filas de una tabla. Por ejemplo, en la diapositiva puede ver que Kochhar, De Haan y Hartstein dependen del MANAGER_ID 100, que es el EMPLOYEE_ID de King.

Nota: los árboles jerárquicos se utilizan en diferentes campos como la genealogía humana (árboles familiares), ganado (para crianza), gestión corporativa (jerarquías de gestión), fabricación (ensamblaje de productos), investigación evolutiva (desarrollo de especies) e investigación científica.



Consultas Jerárquicas

```
SELECT [LEVEL], column, expr...
FROM table
[WHERE condition(s)]
[START WITH condition(s)]
[CONNECT BY PRIOR condition(s)];
```

condition:

```
expr comparison_operator expr
```

Copyright © 2010, Oracle. Todos los derechos reservados.

Palabras Clave y Cláusulas

Las consultas jerárquicas se pueden identificar por la presencia de las cláusulas CONNECT BY y START WITH.

En la sintaxis:

SELECT	Es la cláusula SELECT estándar.
LEVEL	Para cada fila devuelta por una consulta jerárquica, la pseudocolumna LEVEL devuelve 1 para una fila raíz, 2 para un secundario de una raíz, etc.
FROM table	Especifica la tabla, vista o instantánea que contiene las columnas. Puede seleccionar sólo de una tabla.
WHERE	Restringe las filas devueltas por la consulta sin que afecte a otras filas de la jerarquía.
<i>condition</i>	Es una comparación con expresiones.
START WITH	Especifica las filas raíz de la jerarquía (dónde empezar). Esta cláusula es necesaria para una auténtica consulta jerárquica.
CONNECT BY	Especifica las columnas en las que existe la relación entre principal y secundario PRIOR. Esta cláusula es necesaria para una consulta jerárquica.

Recorrido por el Árbol

Punto de Inicio

- Especifica una condición que se debe cumplir.
- Acepta cualquier condición válida.

```
START WITH column1 = value
```

El uso de la tabla EMPLOYEES, comienza con el empleado cuyo apellido es Kochhar.

```
...START WITH last_name = 'Kochhar'
```

Copyright © 2010, Oracle. Todos los derechos reservados.

Recorrido por el Árbol

La fila o las filas que se utilizarán como raíz del árbol vienen determinadas por la cláusula START WITH. La cláusula START WITH puede contener cualquier condición válida.

Ejemplos

Con la tabla EMPLOYEES, se empieza con King, el presidente de la compañía.

```
... START WITH manager_id IS NULL
```

Con la tabla EMPLOYEES, se empieza por el empleado Kochhar. Una condición START WITH puede contener una subconsulta.

```
... START WITH employee_id = (SELECT employee_id
                                FROM   employees
                               WHERE  last_name = 'Kochhar')
```

Si se omite la cláusula START WITH, el recorrido por el árbol se inicia con todas las filas de la tabla como filas raíz.

Nota: las cláusulas CONNECT BY y START WITH no son SQL del estándar ANSI (American National Standards Institute).

Recorrido por el Árbol

```
CONNECT BY PRIOR column1 = column2
```

Recorrido de arriba abajo con la tabla EMPLOYEES.

```
... CONNECT BY PRIOR employee_id = manager_id
```

Dirección

Descendente → Columna1 = Clave Principal
 Columna2 = Clave Secundaria

Ascendente → Columna1 = Clave Secundaria
 Columna2 = Clave Principal

Copyright © 2010, Oracle. Todos los derechos reservados.

Recorrido por el Árbol (continuación)

La dirección de la consulta viene determinada por la colocación de la columna CONNECT BY PRIOR. Para la dirección descendente, el operador PRIOR hace referencia a la fila principal. Para la dirección ascendente, el operador PRIOR hace referencia a la fila secundaria. Para buscar las filas secundarias de una fila principal, el servidor de Oracle evalúa la expresión PRIOR para la fila principal y las demás expresiones para cada fila de la tabla. Las filas para las que la condición es verdadera son filas secundarias de la principal. El servidor de Oracle selecciona siempre filas secundarias mediante la evaluación de la condición CONNECT BY respecto a una fila principal actual.

Ejemplos

Recorrido de arriba abajo con la tabla EMPLOYEES. Defina una relación jerárquica en la que el valor EMPLOYEE_ID de la fila principal sea igual al valor MANAGER_ID de la fila secundaria:

```
... CONNECT BY PRIOR employee_id = manager_id
```

Recorrido de abajo arriba con la tabla EMPLOYEES:

```
... CONNECT BY PRIOR manager_id = employee_id
```

El operador PRIOR no se tiene que codificar necesariamente inmediatamente después de CONNECT BY. Por lo tanto, la siguiente cláusula CONNECT BY PRIOR proporciona el mismo resultado que la del ejemplo anterior:

```
... CONNECT BY employee_id = PRIOR manager_id
```

Nota: la cláusula CONNECT BY no puede contener una subconsulta.

Recorrido por el Árbol: De Abajo Arriba

```
SELECT employee_id, last_name, job_id, manager_id  
FROM employees  
START WITH employee_id = 101  
CONNECT BY PRIOR manager_id = employee_id ;
```

	EMPLOYEE_ID	LAST_NAME	JOB_ID	MANAGER_ID
1	101	Kochhar	AD_VP	100
2	100	King	AD_PRES	(null)



Copyright © 2010, Oracle. Todos los derechos reservados.

Recorrido por el Árbol: De Abajo Arriba

El ejemplo de la diapositiva muestra una lista de los gestores que empiezan por el empleado cuyo ID es 101.

Recorrido por el Árbol: De Arriba Abajo

```
SELECT last_name||' reports to '||  
PRIOR last_name "Walk Top Down"  
FROM employees  
START WITH last_name = 'King'  
CONNECT BY PRIOR employee_id = manager_id ;
```

Walk Top Down
1 King reports to
2 King reports to
3 Kochhar reports to King
4 Greenberg reports to Kochhar
5 Faviet reports to Greenberg

...

105 Grant reports to Zlotkey
106 Johnson reports to Zlotkey
107 Hartstein reports to King
108 Fay reports to Hartstein

Copyright © 2010, Oracle. Todos los derechos reservados.

Recorrido por el Árbol: De Arriba Abajo

El recorrido de arriba abajo muestra los nombres de los empleados y su gestor. Utilice el empleado King como punto de inicio. Imprima sólo una columna.

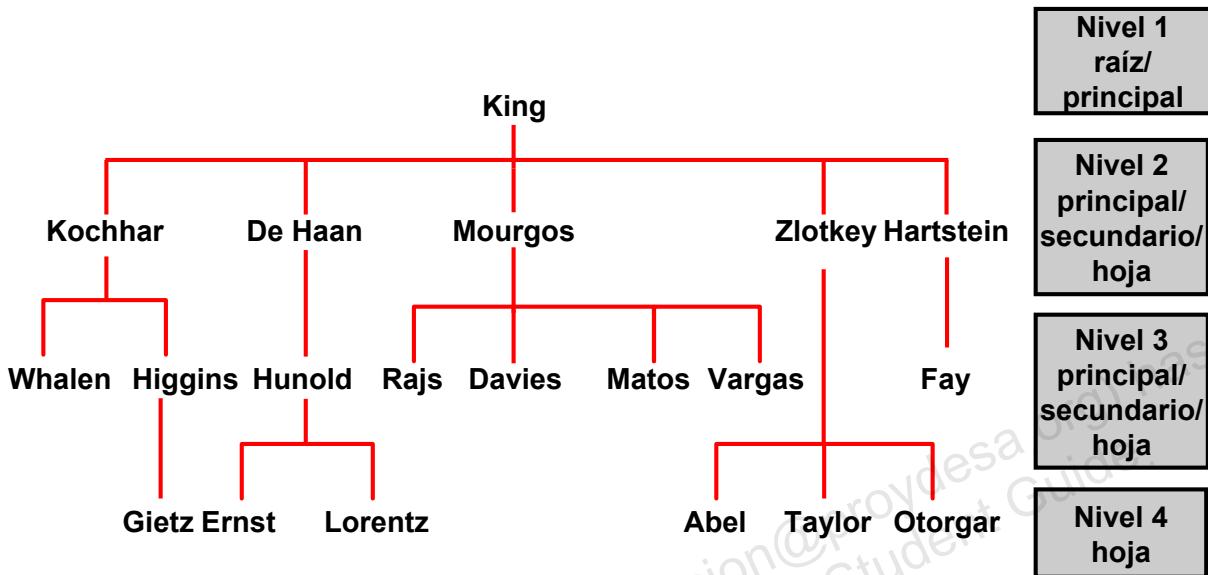
Ejemplo

En el siguiente ejemplo, los valores EMPLOYEE_ID se evalúan para la fila principal y los valores MANAGER_ID y SALARY para las filas secundarias. El operador PRIOR sólo se aplica al valor EMPLOYEE_ID.

```
... CONNECT BY PRIOR employee_id = manager_id  
          AND salary > 15000;
```

Para cualificarla como fila secundaria, una fila debe tener un valor MANAGER_ID igual al valor EMPLOYEE_ID de la fila principal y debe tener un valor SALARY superior a 15.000 dólares.

Clasificación de Filas con la Pseudocolumna LEVEL



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Clasificación de Filas con la Pseudocolumna LEVEL

Puede mostrar explícitamente la clasificación o el nivel de una fila en la jerarquía mediante la pseudocolumna LEVEL. Esto hará que el informe sea más legible. Las bifurcaciones en las que se escinden una o más ramas de una rama más grande se denominan nodos y el extremo de una rama se denomina hoja o nodo hoja. El gráfico de la diapositiva muestra los nodos del árbol invertido con sus valores LEVEL. Por ejemplo, el empleado Higgins es un principal y un secundario, mientras que el empleado Davies es un secundario y una hoja.

Pseudocolumna LEVEL

Value	Level for Top Down	Level for Bottom up
1	A root node	A root node
2	A child of a root node	The parent of a root node
3	A child of a child, and so on	A parent of a parent, and so on

En la diapositiva, King es la raíz o el principal (LEVEL = 1). Kochhar, De Haan, Mourgos, Zlotkey, Hartstein, Higgins y Hunold son secundarios y también principales (LEVEL = 2). Whalen, Rajas, Davies, Matos, Vargas, Gietz, Ernst, Lorentz, Abel, Taylor, Grant y Fay son secundarios y hojas (LEVEL = 3 y LEVEL = 4).

Nota: un *nodo raíz* es el nodo superior de un árbol invertido. Un *nodo secundario* es cualquier nodo que no sea raíz. Un *nodo principal* es cualquier nodo que tenga secundarios. Un *nodo hoja* es cualquier nodo sin secundarios. El número de niveles devueltos por una consulta jerárquica puede estar limitado por la memoria de usuario disponible.

Aplicación de Formato a Informes Jerárquicos con LEVEL y LPAD

Cree un informe que muestre los niveles de gestión de la compañía, desde el nivel superior y sangrando los siguientes niveles.

```
COLUMN org_chart FORMAT A12
SELECT LPAD(last_name, LENGTH(last_name)+(LEVEL*2)-2, '_')
      AS org_chart
  FROM employees
START WITH first_name='Steven' AND last_name='King'
CONNECT BY PRIOR employee_id=manager_id
```

Copyright © 2010, Oracle. Todos los derechos reservados.

Aplicación de Formato a Informes Jerárquicos con LEVEL y LPAD

A los nodos de un árbol se les asignan números de nivel desde la raíz. Utilice la función LPAD junto con la pseudocolumna LEVEL para mostrar un informe jerárquico como un árbol sangrado.

En el ejemplo de la diapositiva:

- LPAD(*char1*, *n* [, *char2*]) devuelve *char1*, con relleno a la izquierda hasta *n* caracteres con la secuencia de caracteres en *char2*. El argumento *n* es la longitud total del valor devuelto tal y como se muestra en la pantalla del terminal.
- LPAD(*last_name*, LENGTH(*last_name*) + (LEVEL*2) - 2, '_') define el formato de visualización.
- *char1* es el valor LAST_NAME, *n* la longitud total del valor devuelto, es la longitud de LAST_NAME + (LEVEL*2) - 2 y *char2* es '_'.

Es decir, indica a SQL que utilice LAST_NAME y que lo rellene a la izquierda con el carácter '_' hasta que la longitud de la cadena resultante sea igual al valor determinado por LENGTH(*last_name*) + (LEVEL*2) - 2.

Para King, LEVEL = 1. Por lo tanto, $(2 * 1) - 2 = 2 - 2 = 0$. De esta forma, King no se rellena con ningún carácter '_' y se muestra en la columna 1.

Para Kochhar, LEVEL = 2. Por lo tanto, $(2 * 2) - 2 = 4 - 2 = 2$. De esta forma, Kochhar se rellena con 2 caracteres '_' y se muestra sangrado.

El resto de registros de la tabla EMPLOYEES se muestran del mismo modo.

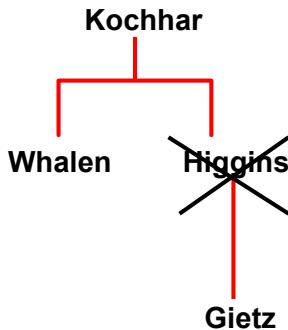
Aplicación de Formato a Informes Jerárquicos con LEVEL y LPAD (continuación)

ORG_CHART	
1	King
2	Kochhar
3	Greenberg
4	Faviet
5	Chen
6	Sciarra
7	Urman
8	Popp
9	Whalen
10	Mavris
11	Baer
12	Higgins
13	Gietz
14	De Haan
15	Hunold
16	Ernst
17	Austin

Eliminación de Ramas

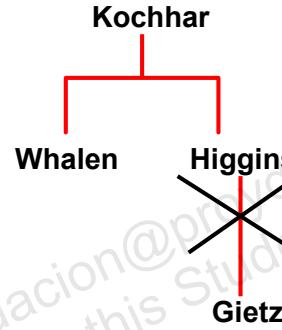
Utilice la Cláusula WHERE para eliminar un nodo.

```
WHERE last_name != 'Higgins'
```



Utilice la cláusula CONNECT BY para eliminar un nodo.

```
CONNECT BY PRIOR
employee_id = manager_id
AND last_name != 'Higgins'
```



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Eliminación de Ramas

Puede utilizar las cláusula WHERE y CONNECT BY para eliminar el árbol (es decir, para controlar los nodos y las filas que se deben mostrar). El predicado que se utiliza actúa como condición booleana.

Ejemplos

Desde la raíz, desplácese de arriba abajo y elimine el empleado Higgins del resultado, pero procese las filas secundarias.

```
SELECT department_id, employee_id, last_name, job_id, salary
FROM employees
WHERE last_name != 'Higgins'
START WITH manager_id IS NULL
CONNECT BY PRIOR employee_id = manager_id;
```

Desde la raíz, desplácese de arriba abajo y elimine el empleado Higgins y todas las filas secundarias.

```
SELECT department_id, employee_id, last_name, job_id, salary
FROM employees
START WITH manager_id IS NULL
CONNECT BY PRIOR employee_id = manager_id
AND last_name != 'Higgins';
```

Resumen

En este apéndice, debe haber aprendido que puede:

- Utilizar consultar jerárquicas para ver una relación jerárquica entre las filas de una tabla
- Especificar la dirección y el punto de inicio de la consulta
- Eliminar nodos o ramas mediante recorte



Copyright © 2010, Oracle. Todos los derechos reservados.

Resumen

Puede utilizar las consultas jerárquicas para recuperar datos según una relación jerárquica natural entre las filas de una tabla. La pseudocolumna LEVEL indica los niveles que ha descendido en un árbol jerárquico. Puede especificar la dirección de la consulta mediante la cláusula CONNECT BY PRIOR. Puede especificar el punto de inicio mediante la cláusula START WITH. Puede utilizar las cláusulas WHERE y CONNECT BY para eliminar ramas del árbol.

Escritura de Archivos de Comandos Avanzados

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

Al finalizar este apéndice, debería estar capacitado para lo siguiente:

- Describir el tipo de problemas que se solucionan al utilizar SQL para generar SQL
- Escribir un script que genere un script de sentencias `DROP TABLE`
- Escribir un script que genere un script de sentencias `INSERT INTO`



Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

En este apéndice, aprenderá a escribir un script SQL para generar un SQL script.

Uso de SQL para Generar SQL

- SQL se puede utilizar para generar scripts en SQL.
- El diccionario de datos:
 - Es una recopilación de tablas y vistas que contienen información sobre la base de datos.
 - Lo crea y mantiene el servidor de Oracle.



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Uso de SQL para Generar SQL

SQL puede ser una potente herramienta para generar otras sentencias SQL. En la mayoría de los casos, implica la escritura de un script. Puede utilizar SQL de SQL para:

- Evitar código repetitivo
- Acceder a información del diccionario de datos
- Borrar o volver a crear objetos de base de datos
- Generar predicados dinámicos que contengan parámetros de tiempo de ejecución

Los ejemplos utilizados en este apéndice implican la selección de información del diccionario de datos. El diccionario de datos es una recopilación de tablas y vistas que contienen información sobre la base de datos. El servidor de Oracle crea y mantiene esta recopilación. Todas las tablas del diccionario de datos pertenecen al usuario SYS. La información almacenada en el diccionario de datos incluye los nombres de los usuarios del servidor de Oracle, los privilegios otorgados a los usuarios, los nombres de los objetos de la base de datos, las restricciones de tabla e información de auditoría. Hay cuatro categorías de vistas del diccionario de datos. Cada categoría tiene un prefijo distinto que refleja su uso previsto.

Prefijo	Descripción
USER_	Contiene los detalles de los objetos propiedad del usuario
ALL_	Contiene los detalles de los objetos para los que el usuario tiene derechos de acceso, además de los objetos propiedad del usuario
DBA_	Contiene los detalles de los usuarios con privilegios de DBA para acceder a cualquier objeto de la base de datos
V\$ _	Almacena información sobre el bloqueo y el rendimiento del servidor de la base de datos; disponible sólo para el DBA

Creación de un Script Básico

```
SELECT 'CREATE TABLE ' || table_name ||
       '_test ' || 'AS SELECT * FROM '
      || table_name ||' WHERE 1=2;'
      AS "Create Table Script"
FROM user_tables;
```

Create Table Script
1 CREATE TABLE REGIONS_test AS SELECT * FROM REGIONS WHERE 1=2;
2 CREATE TABLE LOCATIONS_test AS SELECT * FROM LOCATIONS WHERE 1=2;
3 CREATE TABLE DEPARTMENTS_test AS SELECT * FROM DEPARTMENTS WHERE 1=2;
4 CREATE TABLE JOBS_test AS SELECT * FROM JOBS WHERE 1=2;
5 CREATE TABLE EMPLOYEES_test AS SELECT * FROM EMPLOYEES WHERE 1=2;
6 CREATE TABLE JOB_HISTORY_test AS SELECT * FROM JOB_HISTORY WHERE 1=2;

Copyright © 2010, Oracle. Todos los derechos reservados.

Script Básico

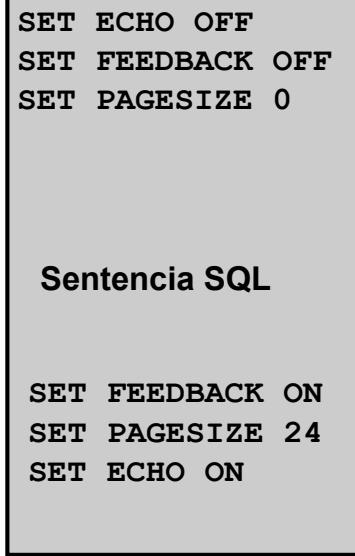
El ejemplo de la diapositiva genera un informe con las sentencias CREATE TABLE de cada tabla que posee. Cada sentencia CREATE TABLE generada en el informe incluye la sintaxis para crear una tabla utilizando el nombre de tabla con el sufijo _test y sólo con la estructura de la tabla correspondiente existente. El nombre de tabla antiguo se obtuvo de la columna TABLE_NAME de la vista USER_TABLES del diccionario de datos.

El siguiente paso es mejorar el informe para automatizar el proceso.

Nota: puede consultar las tablas del diccionario de datos para ver diferentes objetos de la base de datos que posee. Las vistas del diccionario de datos que se suelen utilizar incluyen:

- USER_TABLES: muestra la descripción de las tablas propiedad del usuario.
- USER_OBJECTS: muestra todos los objetos propiedad del usuario.
- USER_TAB_PRIVS_MADE: muestra todos los privilegios otorgados sobre los objetos propiedad del usuario.
- USER_COL_PRIVS_MADE: muestra todos los privilegios otorgados sobre las columnas de objetos propiedad del usuario.

Control del Entorno



Defina variables del sistema en los valores adecuados.

Vuelva a definir las variables del sistema en el valor por defecto.

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Control del Entorno

Para ejecutar las sentencias SQL que se generan, debe capturarlas en un archivo que se pueda ejecutar posteriormente. Debe prever también la limpieza de la salida que se genera y asegurarse de que suprime elementos como las cabeceras, los mensajes de comentarios, los títulos principales, etc. En SQL Developer, puede guardar estas sentencias en un script. Para guardar el contenido del cuadro Enter SQL Statement, haga clic en el icono Save o utilice la opción de menú **File > Save**. También puede hacer clic con el botón derecho del mouse en el cuadro Enter SQL Statement y seleccionar la opción Save File en el menú desplegable.

Nota: algunas de las sentencias SQL*Plus no están soportadas en la hoja de trabajo de SQL. Para obtener una lista completa de las sentencias SQL*Plus que están soportadas en la hoja de trabajo de SQL y las que no, consulte el tema titulado *SQL*Plus Statements Supported and Not Supported in SQL Worksheet*, sobre sentencias SQL*Plus soportadas y no soportadas en la hoja de trabajo de SQL, de la ayuda en pantalla de SQL Developer.

Imagen Completa

```
SET ECHO OFF
SET FEEDBACK OFF
SET PAGESIZE 0

SELECT 'DROP TABLE ' || object_name || ';' 
FROM user_objects
WHERE object_type = 'TABLE'
/

SET FEEDBACK ON
SET PAGESIZE 24
SET ECHO ON
```



Copyright © 2010, Oracle. Todos los derechos reservados.

Imagen Completa

La salida del comando de la diapositiva se guarda en un archivo denominado dropem.sql en SQL Developer. Para guardar la salida en un archivo en SQL Developer, debe utilizar la opción Save File del panel Script Output. El archivo dropem.sql contiene los siguientes datos. Ahora, puede iniciar este archivo desde SQL Developer. Para ello, busque el script, cárguelo y ejecútelo.

'DROPTABLE' OBJECT_NAME ';'
1 DROP TABLE REGIONS;
2 DROP TABLE COUNTRIES;
3 DROP TABLE LOCATIONS;
4 DROP TABLE DEPARTMENTS;
5 DROP TABLE JOBS;
6 DROP TABLE EMPLOYEES;
7 DROP TABLE JOB_HISTORY;
8 DROP TABLE JOB_GRADES;

Volcado del Contenido de una Tabla en un Archivo

```
SET HEADING OFF ECHO OFF FEEDBACK OFF
SET PAGESIZE 0

SELECT
  'INSERT INTO departments_test VALUES
   (' || department_id || ', "' || department_name || '
    ', '' || location_id || ''');'
  AS "Insert Statements Script"
FROM   departments
/

SET PAGESIZE 24
SET HEADING ON ECHO ON FEEDBACK ON
```



Copyright © 2010, Oracle. Todos los derechos reservados.

Volcado del Contenido de la Tabla en un Archivo

En ocasiones, es útil tener los valores de las filas de una tabla en un archivo de texto en formato de sentencia INSERT INTO VALUES. Este script se puede ejecutar para llenar la tabla en caso de que se haya borrado accidentalmente.

El ejemplo de la diapositiva genera sentencias INSERT para la tabla DEPARTMENTS_TEST, capturadas en el archivo data.sql mediante la opción Save File en SQL Developer.

El contenido del script data.sql es el siguiente:

```
INSERT INTO departments_test VALUES
  (10, 'Administration', 1700);
INSERT INTO departments_test VALUES
  (20, 'Marketing', 1800);
INSERT INTO departments_test VALUES
  (50, 'Shipping', 1500);
INSERT INTO departments_test VALUES
  (60, 'IT', 1400);
...
```

Volcado del Contenido de una Tabla en un Archivo

Origen	Resultado
' '' x '' '	'x'
' '' '	'
' '' department_name '' '	'Administration'
' '' , '' '	', '
' '') ; '	') ;

ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Generación de un Predicado Dinámico

```
COLUMN my_col NEW_VALUE dyn_where_clause

SELECT DECODE('&&deptno', null,
DECODE('&&hiredate', null, '',
'WHERE hire_date=TO_DATE(''||'||&&hiredate'||',''DD-MON-YYYY'))',
DECODE('&&hiredate', null,
'WHERE department_id = ' || '&&deptno',
'WHERE department_id = ' || '&&deptno' ||
' AND hire date = TO_DATE(''||'||&&hiredate'||',''DD-MON-YYYY'))')
AS my_col FROM dual;
```

```
SELECT last_name FROM employees &dyn_where_clause;
```

Copyright © 2010, Oracle. Todos los derechos reservados.

Generación de un Predicado Dinámico

El ejemplo de la diapositiva genera una sentencia SELECT que recupera datos de todos los empleados de un departamento contratados un determinado día. El script genera la cláusula WHERE dinámicamente.

Nota: después de que una variable de usuario esté colocada, deberá utilizar el comando `UNDEFINE` para suprimirla.

La primera sentencia SELECT le pedirá que introduzca el número de departamento. Si no introduce ningún número de departamento, la función DECODE trata el número de departamento como si fuese nulo y se solicita al usuario la fecha de contratación. Si no introduce ninguna fecha de contratación, la función DECODE trata la fecha de contratación como nula y la cláusula dinámica WHERE que se genera también es nula, lo que provoca que la segunda sentencia SELECT recupere todas las filas de la tabla EMPLOYEES.

Nota: la variable `NEW_V[ALUE]` especifica una variable para que contenga un valor de columna. Puede hacer referencia a la variable en los comandos `TTITLE`. Utilice `NEW_VALUE` para mostrar los valores de columna o la fecha en el título principal. Debe incluir la columna en un comando `BREAK` con la acción `SKIP PAGE`. El nombre de la variable no puede contener el signo de almohadilla (#). `NEW_VALUE` es útil para los informes maestros/detalles en los que hay un nuevo registro maestro para cada página.

Generación de un Predicado Dinámico (continuación)

Nota: aquí, la fecha de contratación se debe introducir con el formato DD-MON-YYYY.

La sentencia SELECT de la diapositiva se puede interpretar de la siguiente forma:

```

IF    (<<deptno>> is not entered)  THEN
    IF  (<<hiredate>> is not entered)  THEN
        return empty string
    ELSE
        return the string 'WHERE hire_date =
TO_DATE(''<<hiredate>>'', 'DD-MON-YYYY')'
    ELSE
        IF  (<<hiredate>> is not entered)  THEN
            return the string 'WHERE department_id = <<deptno>>
entered'
        ELSE
            return the string 'WHERE department_id = <<deptno>>
entered
                                AND hire_date =
TO_DATE(''<<hiredate>>'', 'DD-MON-YYYY')'
        END IF
    
```

La cadena devuelta se convierte en el valor de la variable DYN_WHERE_CLAUSE, que se utilizará en la segunda sentencia SELECT.

Nota: utilice SQL*Plus para estos ejemplos.

Cuando se ejecuta el primer ejemplo de la diapositiva, se solicita al usuario los valores DEPTNO y HIREDATE:

Enter value for deptno: 10

Enter value for hiredate: 17-SEP-1987

Se genera el siguiente valor para MY_COL:

MY_COL
WHERE department_id = 10 AND hire_date = TO_DATE('27-SEP-1987', 'DD-MON-YYYY')

Cuando se ejecuta el segundo ejemplo de la diapositiva, se genera la siguiente salida:

LAST_NAME

Whalen

Resumen

En este apéndice, debe haber aprendido lo siguiente:

- Puede escribir un script SQL para generar otro script SQL.
- Los scripts utilizan a menudo el diccionario de datos.
- Puede capturar la salida en un archivo.



Copyright © 2010, Oracle. Todos los derechos reservados.

Componentes Arquitectónicos de Oracle Database

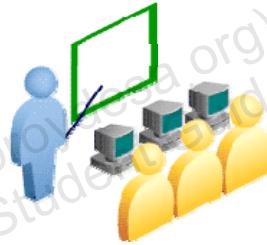
ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

Al finalizar este apéndice, debería estar capacitado para lo siguiente:

- Mostrar los principales componentes arquitectónicos de la base de datos
- Describir los procesos en segundo plano
- Explicar las estructuras de memoria
- Correlacionar estructuras de almacenamiento físico y lógico



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Objetivos

Este apéndice proporciona una visión general de la arquitectura de Oracle Database. Aprenderá sobre las estructuras lógicas y físicas así como los diferentes componentes de Oracle Database y sus funciones.

Arquitectura de Oracle Database: Visión General

El sistema de gestión de bases de datos relacionales (RDBMS) de Oracle proporciona un enfoque abierto, completo e integrado a la gestión de información.



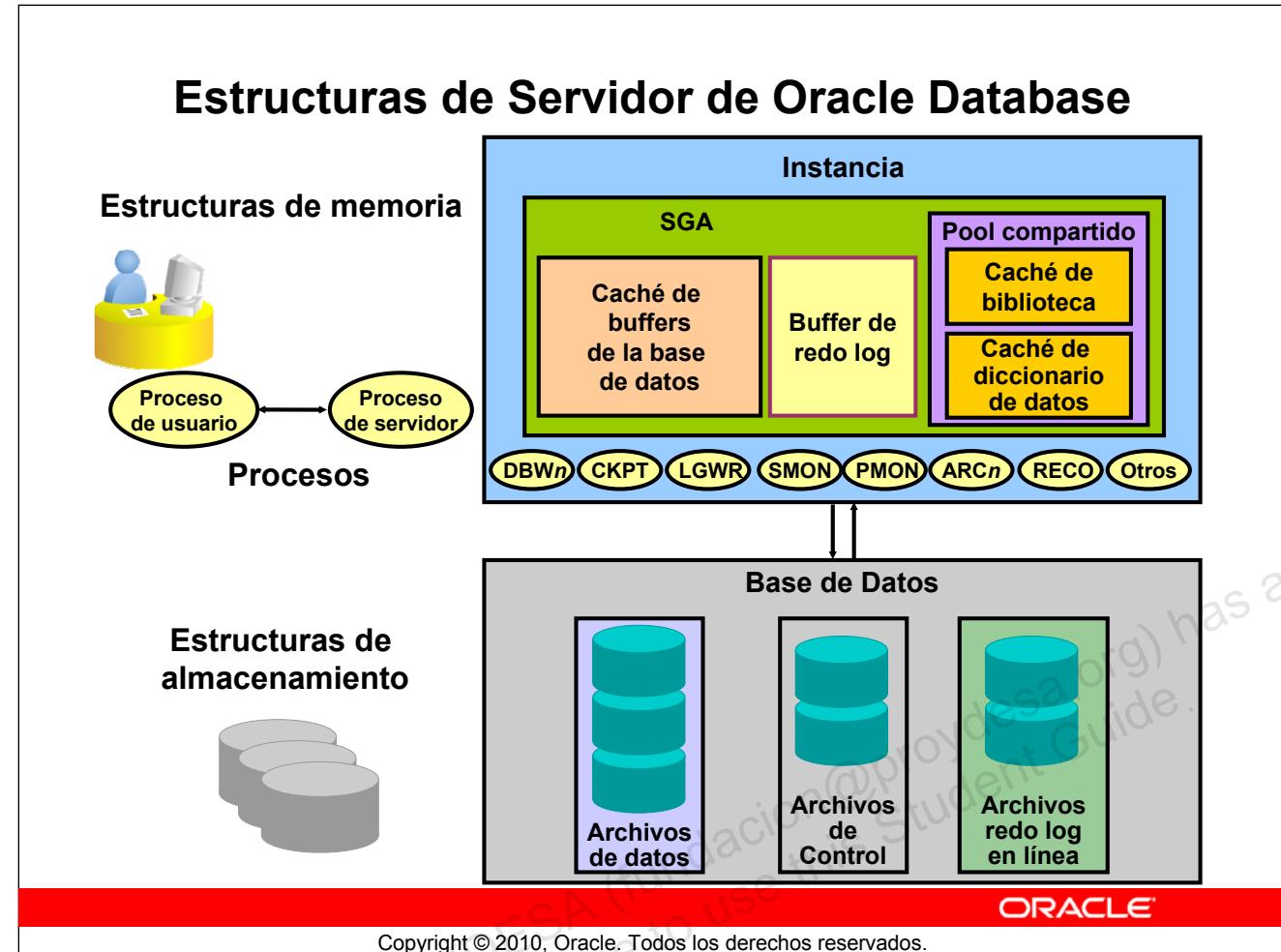
ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Arquitectura de Oracle Database: Visión General

Una base de datos es una recopilación de datos que se trata como una unidad. El objetivo general de una base de datos es almacenar y recuperar la información relacionada.

Oracle Database gestiona de manera fiable una gran cantidad de datos en un entorno de varios usuarios para que numerosos usuarios puedan acceder de manera simultánea a los mismos datos. Esto se debe realizar al mismo tiempo que se ofrece un alto rendimiento. Al mismo tiempo, impide el acceso no autorizado y proporciona soluciones eficaces para la recuperación ante fallos.



Estructuras de Servidor de Oracle Database

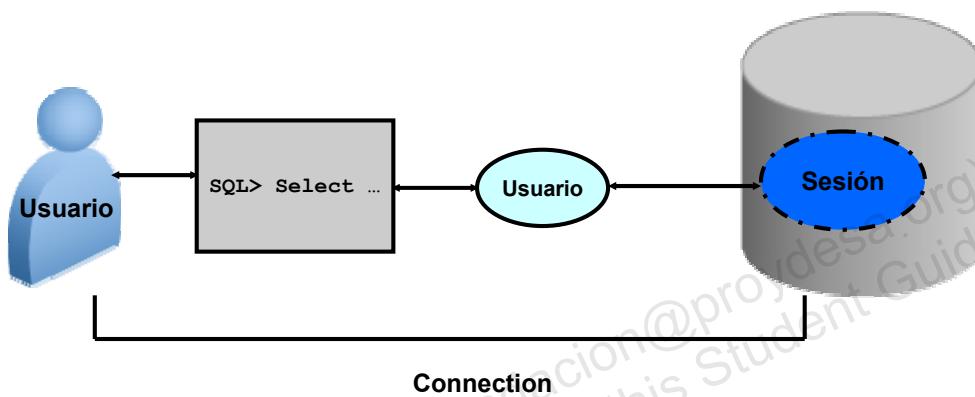
Oracle Database consta de dos componentes principales: la instancia y la base de datos.

- La instancia se compone del Área Global del Sistema (SGA), que es una recopilación de estructuras de memoria y procesos en segundo plano que realizan tareas en la base de datos. Cada vez que se inicia una instancia, se asigna el SGA y se inician los procesos en segundo plano.
- La base de datos consta de estructuras lógicas y físicas. Debido a que las estructuras lógicas y físicas están separadas, el almacenamiento físico de datos se puede gestionar sin que afecte al acceso de estructuras de almacenamiento lógicas. Las estructuras de almacenamiento físicas incluyen:
 - Los archivos de control donde se almacena la configuración de la base de datos
 - Los archivos redo log, que cuentan con información necesaria para la recuperación de la base de datos
 - Los archivos de datos donde se almacenan todos los datos

Una instancia de Oracle utiliza procesos y estructuras de memoria para gestionar y acceder a la base de datos. Todas las estructuras de memoria existen en la memoria principal de las computadoras que constituyen el servidor de la base de datos. Los procesos son trabajos que funcionan en la memoria de estas computadoras. Un proceso se define como “thread de control” o mecanismo de un sistema operativo que puede realizar una serie de pasos.

Conexión a la Base de Datos

- Conexión: vía de comunicación entre un proceso de usuario y una instancia de base de datos
- Sesión: conexión específica de un usuario a una instancia de base de datos mediante un proceso de usuario



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Conexión a la Base de Datos

Para acceder a la información en la base de datos, el usuario se debe conectar a la base de datos mediante una herramienta (como SQL*Plus). Una vez que el usuario ha establecido la conexión, se crea una sesión para el mismo. La conexión y la sesión guardan una estrecha relación con el proceso de usuario pero son muy diferentes en significado.

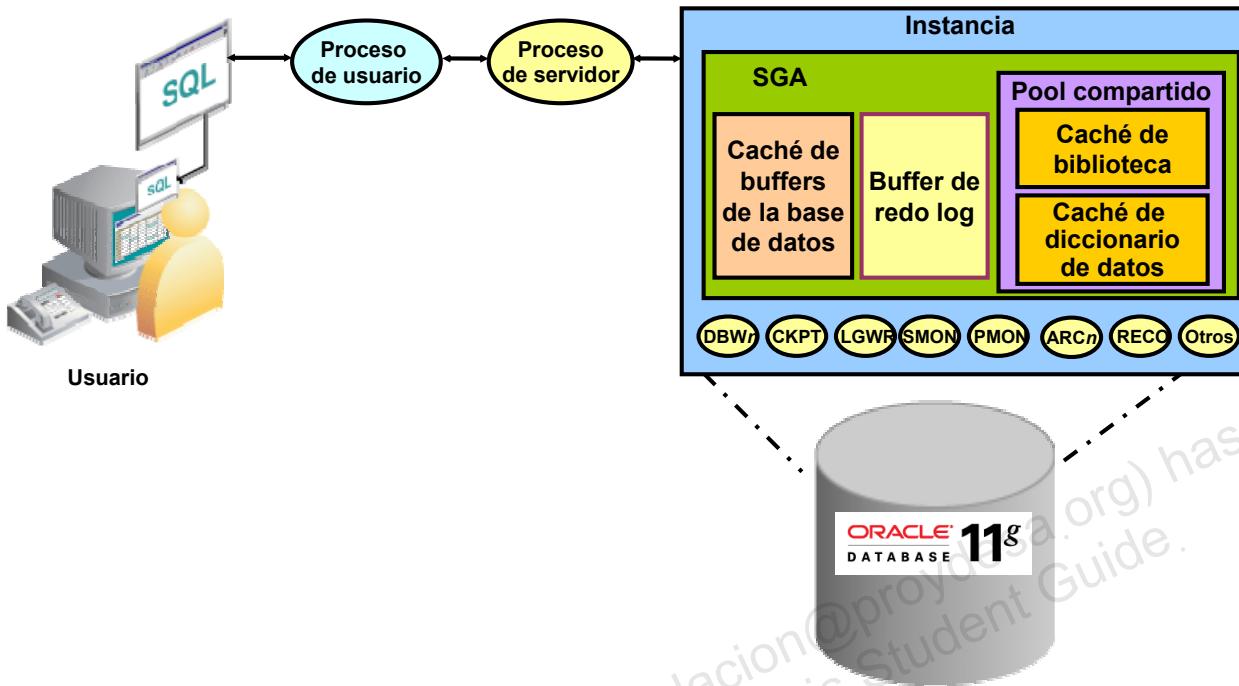
Una conexión es una vía de comunicación entre el proceso de usuario y una instancia de Oracle Database. Una vía de comunicación se establece mediante mecanismos de comunicación entre procesos o software de red disponibles (cuando diferentes computadoras ejecutan la aplicación de base de datos y Oracle Database, y se comunican a través de una red).

Una sesión representa el estado de una conexión del usuario actual a la instancia de base de datos. Por ejemplo, si un usuario inicia SQL*Plus, el usuario debe proporcionar un nombre de usuario y contraseña válidos y, a continuación, se establece una conexión para dicho usuario. Una sesión dura desde el momento en que se conecta el usuario hasta que se desconecta o sale de la aplicación de la base de datos.

En el caso de una conexión dedicada, un proceso dedicado permanente sirve a la sesión. En el caso de una conexión compartida, un proceso de servidor disponible sirve a la sesión seleccionada de un pool, mediante el nivel medio o arquitectura de servidor compartido de Oracle.

Se pueden crear y existir varias sesiones de forma simultánea para un único usuario de Oracle Database con el mismo nombre de usuario, pero a través de diferentes aplicaciones o varias llamadas de la misma aplicación.

Interacción con Oracle Database



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

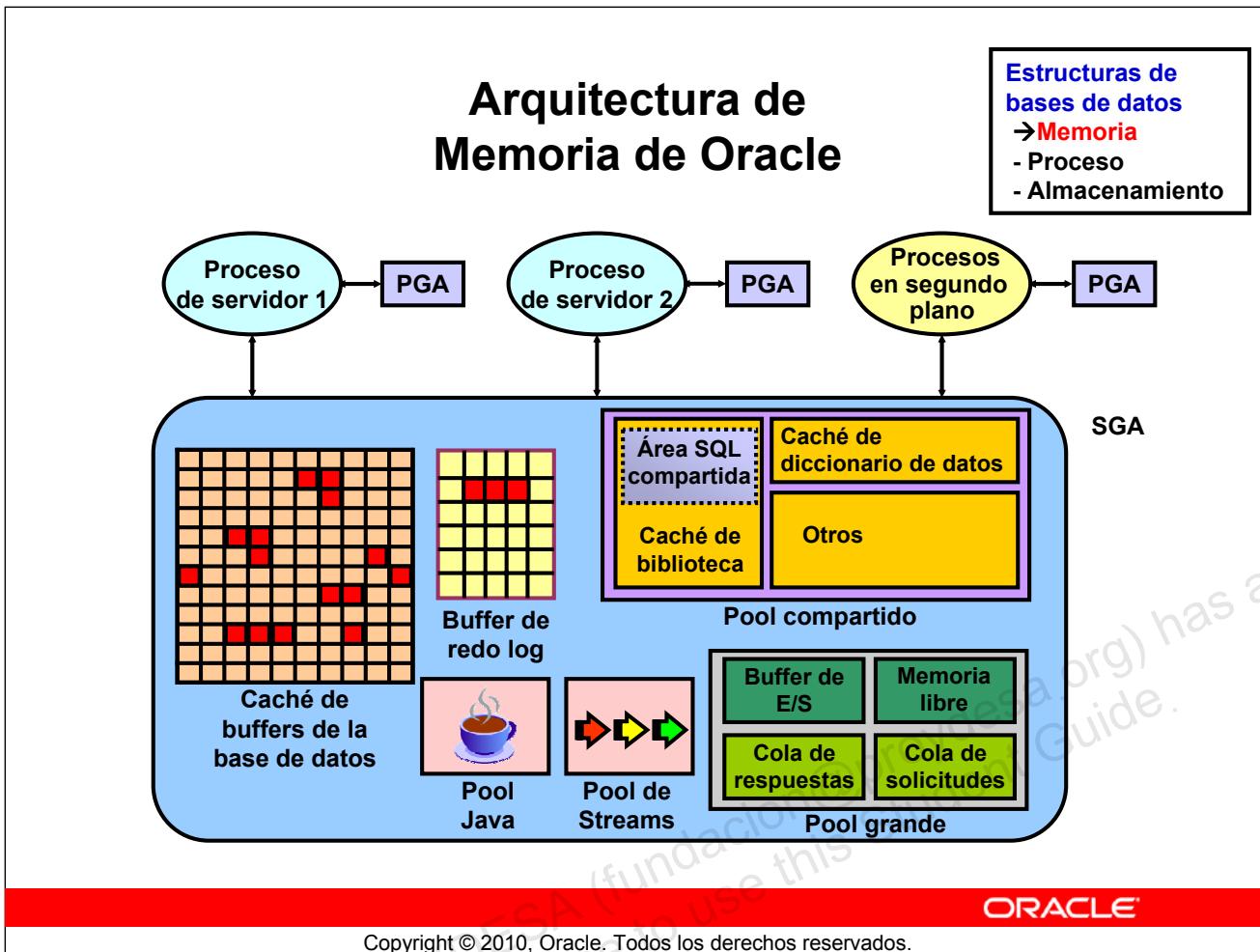
Interacción con Oracle Database

El siguiente ejemplo describe operaciones de Oracle Database al nivel más básico. Ilustra una configuración de Oracle Database en la que el usuario y el proceso de servidor asociado están en computadoras independientes, conectadas a través de una red.

1. Se ha iniciado una instancia en un nodo en el que se ha instalado Oracle Database, a menudo denominado host o servidor de base de datos.
2. Un usuario inicia una aplicación, lo que genera un proceso de usuario. La aplicación intenta establecer una conexión al servidor. (La conexión puede ser local, servidor de cliente o una conexión de tres niveles desde un nivel medio).
3. El servidor ejecuta un listener que tiene el manejador adecuado de Servicios de Red de Oracle. El servidor detecta la solicitud de conexión de la aplicación y crea a un proceso de servidor dedicado en nombre del proceso de usuario.
4. El usuario ejecuta una sentencia SQL de tipo DML y confirma la transacción. Por ejemplo, el usuario cambia la dirección de un cliente en una tabla y confirma el cambio.
5. El proceso de servidor recibe la sentencia y comprueba el pool compartido (un componente de SGA) de cualquier área SQL compartida que contenga una sentencia SQL similar. Si hay un área SQL compartida, el proceso de servidor comprueba los privilegios de acceso del usuario en los datos solicitados y el área SQL compartida existente se utiliza para procesar la sentencia. Si no, una nueva área SQL compartida se asignará para la sentencia, de manera que se pueda analizar y procesar.

Interacción con Oracle Database (continuación)

6. El proceso de servidor recupera los valores de datos necesarios, del archivo de datos reales (en el que se almacena la tabla) o aquellos almacenados en la caché en el SGA.
7. El proceso de servidor modifica los datos del SGA. Debido a que se ha confirmado la transacción, el proceso de escritura de log (LGWR) la registra inmediatamente en el archivo redo log. El proceso de escritores de base de datos (DBW n) escribe los bloques modificados de forma permanente en el disco, lo que lo hace eficaz.
8. Si la transacción se ha realizado de forma correcta, el proceso de servidor envía un mensaje en la red a la aplicación. Si no se ha realizado de forma correcta, se transmite un mensaje de error.
9. A lo largo de todo este procedimiento, los demás procesos en segundo plano se ejecutan, prestando atención a las condiciones que necesitan intervención. Además, el servidor de base de datos gestiona otras transacciones de usuarios y evita la contención entre transacciones que solicitan los mismos datos.



Estructuras de Memoria de Oracle (continuación)

SGA es un área de memoria que contiene datos e información de control para la instancia. SGA incluye las siguientes estructuras de datos:

- **Caché de buffers de la base de datos:** almacena en caché los bloques de datos que se recuperan de la base de datos.
- **Buffer de redo log:** almacena en caché la información de redo (utilizada para la recuperación de instancias) hasta que se pueda escribir en los archivos redo log físicos almacenados en el disco.
- **Pool compartido:** almacena en caché las distintas construcciones que se pueden compartir entre usuarios.
- **Pool grande:** es un área opcional que proporciona grandes asignaciones de memoria para determinados procesos grandes, como operaciones de recuperación y copia de seguridad de Oracle y procesos de entrada/salida (E/S) del servidor.
- **Pool Java:** se utiliza para todos los códigos y datos Java específicos de la sesión dentro de Java Virtual Machine (JVM).
- **Pool de Streams:** lo utiliza Oracle Streams para almacenar información necesaria mediante captura y aplicación.

Al iniciar la instancia con Enterprise Manager o SQL*Plus, aparece la cantidad de memoria asignada a SGA.

Con la infraestructura de SGA dinámica, el tamaño de la caché de buffers, el pool compartido, el pool grande, el pool Java y el pool de Streams de la base de datos cambian sin cerrar la instancia.

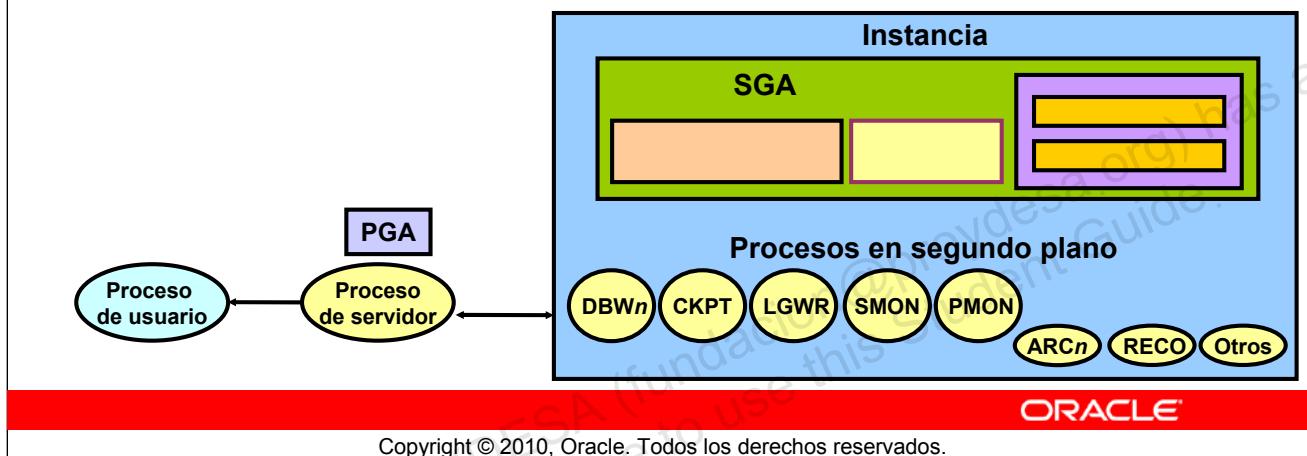
Oracle Database utiliza los parámetros de inicialización para crear y configurar las estructuras de memoria. Por ejemplo, el parámetro `SGA_TARGET` especifica el tamaño total de los componentes de SGA. Si define `SGA_TARGET` en 0, se desactivará la gestión automática de memoria compartida.

Arquitectura de Proceso

- Proceso de usuario:
 - Se inicia cuando un usuario de base de datos o proceso por lotes se conecta a Oracle Database
- Procesos de base de datos:
 - Proceso de servidor: conecta a la instancia de Oracle y se inicia cuando un usuario establece una sesión
 - Procesos en segundo plano: se inician al iniciar una instancia de Oracle

Estructuras de bases de datos

- Memoria
- **Proceso**
- Almacenamiento



Arquitectura de Proceso

Los procesos en un servidor de Oracle Database se pueden clasificar en dos grupos principales:

- Los procesos de usuario que ejecutan la aplicación o código de herramienta de Oracle
- Los procesos de Oracle Database que ejecutan el código del servidor de Oracle Database. Incluyen procesos de servidor y en segundo plano.

Cuando un usuario ejecuta un programa de aplicación en una herramienta como SQL*Plus, Oracle Database crea un *proceso de usuario* para ejecutar la aplicación de usuario. Oracle Database también crea un *proceso de servidor* para ejecutar los comandos ejecutados por el proceso de usuario. Además, el servidor de Oracle también crea un juego de *procesos en segundo plano* para una instancia que interactúan entre sí y con el sistema operativo para gestionar las estructuras de memoria, realizar una E/S asíncrona para escribir datos en disco y llevar a cabo otras tareas necesarias.

La estructura de proceso varía para las diferentes configuraciones de Oracle Database, según el sistema operativo y la selección de las opciones de Oracle Database. El código para los usuarios conectados se puede configurar como un servidor dedicado o compartido.

- Con un servidor dedicado, para cada usuario, un proceso de usuario ejecuta la aplicación de la base de datos que sirve un proceso de servidor dedicado que ejecuta el código de servidor de Oracle Database.
- Un servidor compartido elimina la necesidad de un proceso de servidor dedicado para cada conexión. Un distribuidor dirige varias solicitudes de sesión de red entrantes a un pool de procesos de servidor compartido. Un proceso de servidor compartido sirve a cualquier solicitud de cliente.

Arquitectura de Procesos (continuación)

Procesos de Servidor

Oracle Database crea procesos de servidor para manejar las solicitudes de los procesos de usuario conectados con una instancia. En algunas situaciones, cuando la aplicación y Oracle Database funcionan en la misma computadora, es posible combinar el proceso de usuario y el proceso de servidor correspondiente en un único proceso para reducir la sobrecarga del sistema. Sin embargo, cuando la aplicación y Oracle Database funcionan en diferentes computadoras, un proceso de usuario siempre se comunica con Oracle Database a través de un proceso de servidor independiente.

Los procesos del servidor creados en nombre de la aplicación de cada usuario pueden realizar una o más de las siguientes acciones:

- Analizar y ejecutar las sentencias SQL emitidas a través de la aplicación.
- Leer bloques de datos necesarios de archivos de datos en disco en buffers de la base de datos compartida del SGA, si los bloques no están ya en el SGA.
- Devolver los resultados para que la aplicación pueda procesar la información.

Procesos en Segundo Plano

Para maximizar el rendimiento y acomodar diferentes usuarios, un sistema de Oracle Database de varios procesos utiliza algunos procesos adicionales de Oracle Database denominados procesos en segundo plano. Una instancia de Oracle Database puede tener diferentes procesos en segundo plano.

Los siguientes procesos en segundo plano son necesarios para un inicio correcto de la instancia de base de datos:

- Escritor de base de datos (DBW n)
- Escritor de log (LGWR)
- Punto de control (CKPT)
- Supervisión del sistema (SMON)
- Supervisión de procesos (PMON)

Los siguientes procesos en segundo plano son algunos ejemplos de procesos en segundo plano opcionales que se pueden iniciar si es necesario:

- Proceso de recuperación (RECO)
- Cola de trabajos
- Archivador (ARC n)
- Supervisión de cola (QMN n)

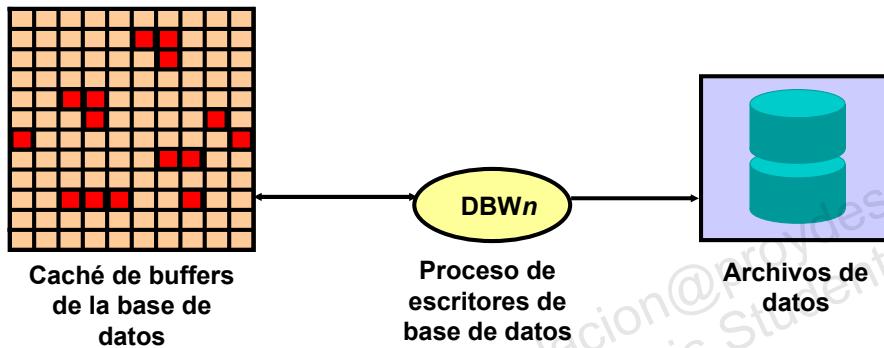
Se pueden encontrar otros procesos en segundo plano en configuraciones más avanzadas, como Real Application Clusters (RAC). Consulte la vista V\$BGPROCESS para obtener más información sobre los procesos en segundo plano.

En muchos sistemas operativos, los procesos en segundo plano se crean automáticamente al iniciar una instancia.

Proceso de Escritores de Base de Datos

Escribe los buffers modificados (sucios) en la caché de buffers de base de datos en el disco:

- De forma asíncrona al realizar otro procesamiento
- Periódicamente para avanzar el punto de control



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

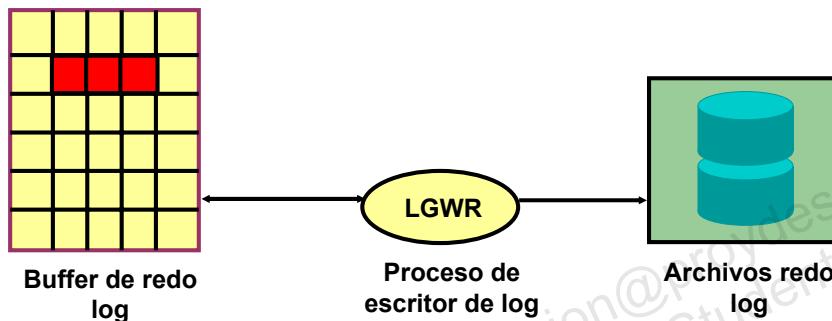
Proceso de Escritores de Base de Datos

El proceso de escritores de base de datos (DBW n) escribe el contenido de los buffers en los archivos de datos. Los procesos de DBW n son responsables de escribir los buffers modificados (sucios) de la caché de buffers de la base de datos en el disco. Aunque un proceso de escritores de base de datos (DBW0) es adecuado para la mayoría de los sistemas, puede configurar procesos adicionales (DBW1 a través de DBW9 y DBW a través de DBWj) para mejorar el rendimiento de escritura si el sistema modifica mucho los datos. Los procesos adicionales de DBW n no son útiles en los sistemas de uniprocesador.

Al modificar un buffer de la caché de buffers de la base de datos, se marca como "sucio" y se agrega a la lista LRUW de buffers sucios que se mantienen en un orden de número de cambio del sistema (SCN), coincidiendo así el orden de redo correspondiente a estos buffers escritos en los redo logs. Si el número de buffers disponibles en la caché de buffers está por debajo del umbral interno, puede resultar difícil que los procesos de servidor obtengan los buffers disponibles. DBW n escribe los buffers sucios en los archivos de datos en el orden en el que se han modificado siguiendo el orden de la lista LRUW.

Proceso de Escritor de Log

- Escribe el buffer de redo log en un archivo redo log del disco
- LGWR escribe:
 - Un proceso confirma una transacción
 - Cuando el buffer de redo log está lleno en un tercio
 - Antes de que un proceso DBW n escriba los buffers modificados en el disco



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Proceso de Escritor de Log

El proceso de escritor de log (LGWR) es responsable de la gestión de buffer de redo log mediante la escritura de entradas de buffer de redo log en un archivo redo log del disco. LGWR escribe todas las entradas que se han copiado en el buffer desde la última vez que se escribió.

El buffer de redo log es un buffer circular. Si LGWR escribe entradas de redo desde el buffer de redo log en un archivo redo log, los procesos de servidor pueden copiar las nuevas entradas en el buffer de redo log que se ha escrito en el disco. Normalmente, LGWR escribe lo bastante rápido para garantizar que siempre haya espacio disponible en el buffer para nuevas entradas, incluso cuando el acceso al archivo redo log es intenso.

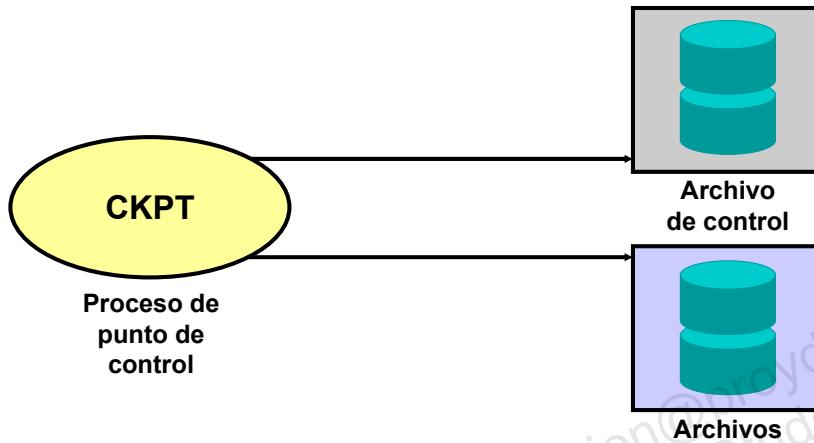
LGWR escribe una parte contigua del buffer en el disco. LGWR escribe:

- Cuando un proceso de usuario confirma una transacción
- Cuando el buffer de redo log está lleno en un tercio
- Antes de que un proceso DBW n escriba los buffers modificados en el disco

Proceso de Punto de Control

Registra la información de punto de control en:

- El archivo de control
- Cada cabecera de archivo de datos



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Proceso de Punto de Control

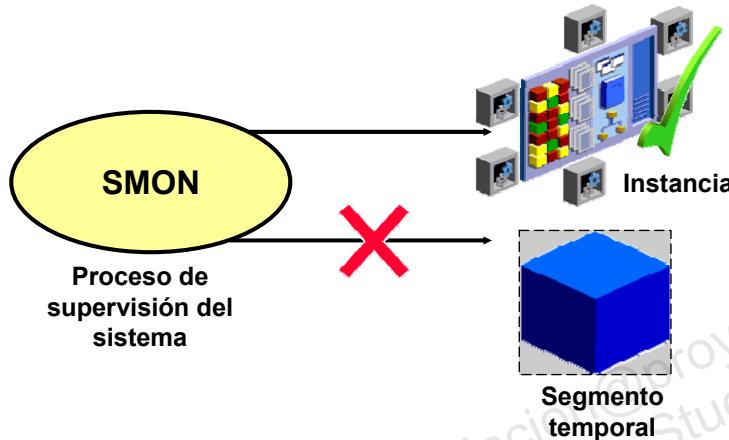
Un punto de control es una estructura de datos que define un SCN en el thread de redo de una base de datos. Los puntos de control se registran en el archivo de control y cada cabecera de archivo de datos, y son un elemento crucial en el proceso de recuperación.

Cuando se produce un punto de control, Oracle Database debe actualizar las cabeceras de todos los archivos de datos para registrar los detalles del punto de control. Se realiza mediante el proceso CKPT. El proceso CKPT no escribe bloques en el disco; DBW n siempre realiza dicho trabajo. Los SCN registrados en las cabeceras de los archivos garantizan que todos los cambios realizados en los bloques de la base de datos antes de que se hayan escrito SCN en el disco.

Los puntos de control de DBWR están estaticos que muestra el monitor SYSTEM_STATISTICS en Oracle Enterprise Manager indican el número de solicitudes de punto de control completadas.

Proceso de Supervisión del Sistema

- Realiza recuperaciones en el inicio de la instancia
- Limpia segmentos temporales no utilizados



ORACLE

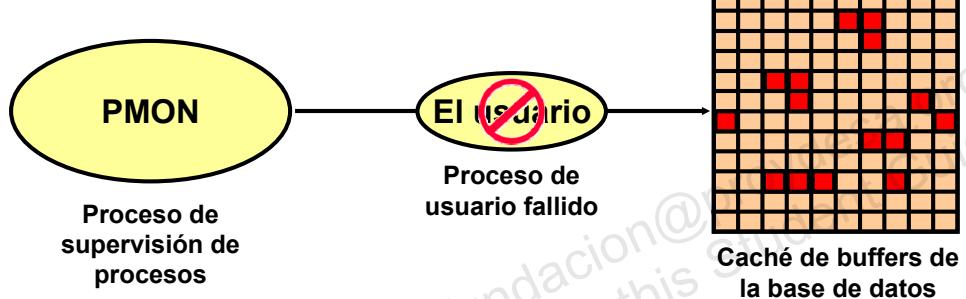
Copyright © 2010, Oracle. Todos los derechos reservados.

Proceso de Supervisión del Sistema

El proceso de supervisión del sistema (SMON) realiza operaciones de recuperación, si es necesario, al inicio de la instancia. SMON también es responsable de la limpieza de segmentos temporales que ya no se utilizan. Si se han omitido algunas transacciones terminadas durante la recuperación de la instancia debido a errores de fuera de línea o de lectura de archivos, SMON los recupera cuando el tablespace o el archivo se vuelve a estar en línea. SMON comprueba regularmente para ver si esta acción es necesaria. Otros procesos pueden llamar a SMON si detectan que es necesario.

Proceso de Supervisión de Procesos

- Realiza la recuperación de procesos cuando falla un proceso de usuario:
 - Limpia la caché de buffers de la base de datos
 - Libera los recursos utilizados por el proceso de usuario
- Supervisa sesiones para el timeout de inactividad de la sesión
- Registra de forma dinámica los servicios de base de datos con listeners



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Proceso de Supervisión de Procesos

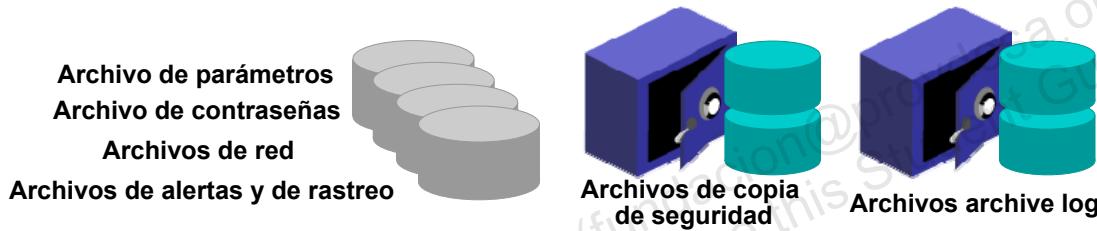
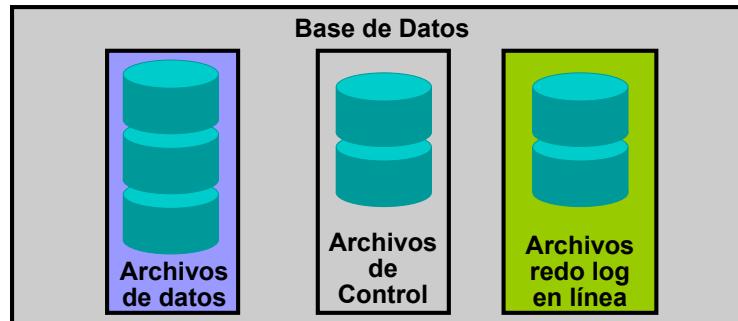
La supervisión de procesos (PMON) realiza recuperaciones de procesos cuando falla un proceso de usuario. PMON es responsable de limpiar toda la caché de buffers de la base de datos y de liberar recursos que estaba utilizando el proceso de usuario. Por ejemplo, restablece el estado de la tabla de transacción activa, libera bloqueos y elimina el ID de proceso de la lista de procesos activos.

PMON comprueba periódicamente el estados de procesos de servidor y de distribuidor y reinicia cualquier proceso que haya parado la ejecución (pero no que Oracle Database haya terminado intencionadamente). PMON también registra información sobre la instancia y los procesos de distribuidor en el listener de red.

Al igual que SMON, PMON comprueba regularmente para ver si es necesario y se puede llamar si otro proceso detecta que es necesario.

Arquitectura de Almacenamiento de Oracle Database

Estructuras de bases de datos
 - Memoria
 - Proceso
 → Almacenamiento



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Arquitectura de Almacenamiento de Oracle Database

Los archivos que constituyen una base de datos Oracle están organizados del siguiente modo:

- **Archivos de control:** Contienen datos sobre la propia base de datos (es decir, información sobre la estructura de la base de datos física). Estos archivos son críticos para la base de datos. Sin ellos, no se pueden abrir los archivos de datos para acceder a los datos que contiene la base de datos.
- **Archivos de datos:** contienen los datos de usuario o de aplicación de la base de datos, así como los metadatos y el diccionario de datos.
- **Archivos redo log en línea:** permiten la recuperación de instancias de la base de datos. Si el servidor de la base de datos falla y no pierde archivos de datos, la instancia puede recuperar la base de datos con la información de dichos archivos.

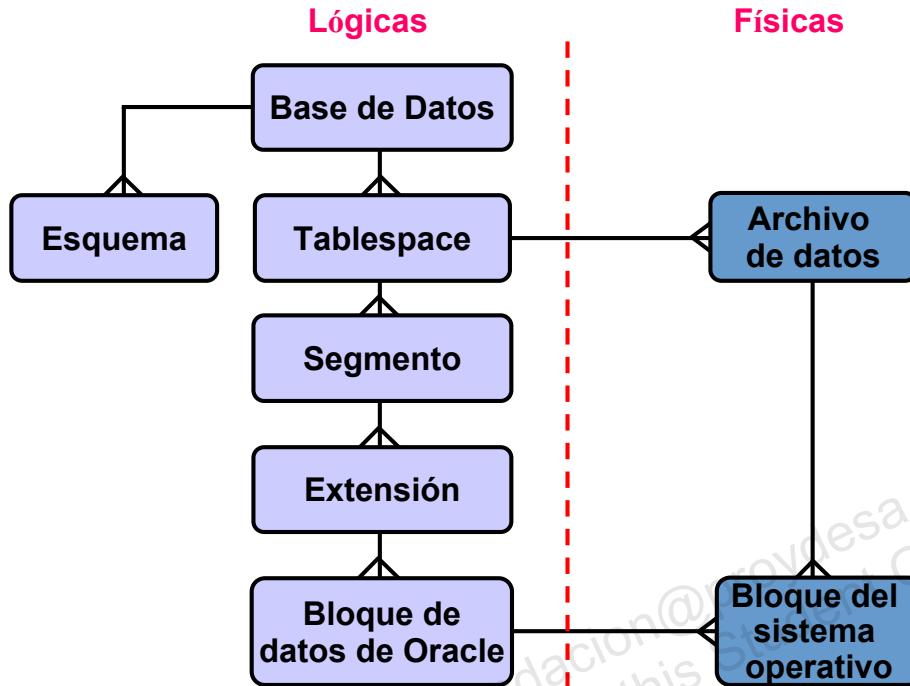
Los siguientes archivos adicionales son importantes para la correcta ejecución de la base de datos:

- **Archivos de copia de seguridad:** se utilizan para la recuperación de la base de datos. Normalmente, restaura un archivo de copia de seguridad cuando un fallo del medio físico o un error del usuario ha dañado o suprimido el archivo original.
- **Archivos archive log:** contienen un historial en curso de los cambios de datos (redo) generados por la instancia. Mediante estos archivos y una copia de seguridad de la base de datos, se puede recuperar un archivo de datos perdido. Es decir, los archive logs permiten la recuperación de archivos de datos restaurados.

Arquitectura de Almacenamiento de Oracle Database (continuación)

- **Archivo de parámetros:** se utiliza para definir el modo de configurar la instancia cuando se inicie.
- **Archivo de contraseñas:** permite a sysdba/sysoper/sysasm conectarse a la base de datos de forma remota y realizar tareas administrativas.
- **Archivos de red:** se utilizan para iniciar el listener de la base de datos y almacenar la información necesaria para las conexiones de usuario
- **Archivos de rastreo:** Cada proceso de servidor y en segundo plano puede escribir en un archivo de rastreo asociado. Cuando un proceso detecta un error interno, vuelve a informar sobre el error en su archivo de rastreo. Parte de la información escrita en un archivo de rastreo va destinada al administrador de la base de datos, mientras que otra información es para los Servicios de Soporte Oracle.
- **Archivos log de alertas:** son entradas de rastreo especiales. El log de alertas de una base de datos es un log cronológico de mensajes y errores. Cada instancia tiene un archivo log de alertas. Oracle recomienda que revise este log de alertas de forma periódica.

Estructuras de Bases de Datos Físicas y Lógicas



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Estructuras de Bases de Datos Físicas y Lógicas

Oracle Database tiene estructuras de almacenamiento lógicas y físicas.

Tablespaces

Una base de datos se divide en unidades lógicas de almacenamiento denominadas tablespaces, que agrupan estructuras lógicas relacionadas. Por ejemplo, los tablespaces suelen agrupar todos los objetos de una aplicación para simplificar algunas operaciones administrativas. Puede tener un tablespace para los datos de la aplicación y otro adicional para los índices de la aplicación.

Bases de Datos, Tablespaces y Archivos de Datos

La relación entre las bases de datos, los tablespaces y los archivos de datos se ilustra en la diapositiva. Cada base de datos está dividida de manera lógica en uno o más tablespaces. Se crean explícitamente uno o más archivos de datos para cada tablespace con el fin de almacenar físicamente los datos de todas las estructuras lógicas de un tablespace. Si se trata de un tablespace TEMPORARY, en lugar de un archivo de datos, el tablespace tendrá un archivo temporal.

Estructuras de Bases de Datos Físicas y Lógicas (continuación)

Esquemas

Un esquema es una recopilación de objetos de base de datos propiedad de un usuario de la base de datos. Los objetos de esquema son estructuras lógicas que hacen referencia directa a datos de la base de datos. Los objetos de esquema incluyen estructuras como, por ejemplo, tablas, vistas, secuencias, procedimientos almacenados, sinónimos, índices, clusters y enlaces de base de datos. En general, los objetos de esquema incluyen todo lo que la aplicación cree en la base de datos.

Bloques de Datos

Al nivel más detallado de granularidad, los datos de Oracle Database se almacenan en bloques de datos. Un bloque de datos corresponde a un número concreto de bytes de espacio de la base de datos física en el disco. Se especifica un tamaño del bloque de datos para cada tablespace cuando se crea. Una base de datos utiliza y asigna espacio libre de la base de datos en bloques de datos Oracle.

Extensões

Al nivel siguiente del espacio de la base de datos lógica se denomina extensión. Una extensión es un número específico de bloques de datos contiguos (obtenidos en una única asignación) que se utilizan para almacenar un tipo determinado de información.

Segmentos

Al nivel de almacenamiento de la base de datos lógica por encima de una extensión se le denomina segmento. Un segmento es un juego de extensiones asignadas para una determinada estructura lógica. Por ejemplo, los diferentes tipos de segmentos incluyen:

- **Segmentos de datos:** cada tabla no de cluster y no organizada por índices tiene un segmento de datos con la excepción de las tablas externas, tablas temporales globales y tablas particionadas donde cada tabla tiene uno o más segmentos. Todos los datos de la tabla se almacenan en las extensiones de su segmento de datos. Para una tabla particionada, cada partición tiene un segmento de datos. Cada cluster tiene un segmento de datos. Los datos de cada tabla del cluster se almacenan en el segmento de datos del cluster.
- **Segmentos de índice:** cada índice tiene un segmento de índice que almacena todos sus datos. Para un índice particionado, cada partición tiene un segmento de índice.
- **Segmentos de deshacer:** se crea un tablespace UNDO por instancia de base de datos que contiene numerosos segmentos de deshacer para almacenar temporalmente la información de *deshacer*. La información de un segmento de deshacer se utiliza para generar información de base de datos de lectura consistente y, durante la recuperación de la base de datos, para realizar una operación de rollback de las transacciones sin confirmar para los usuarios.
- **Segmentos temporales:** Oracle Database crea segmentos temporales cuando una sentencia SQL necesita un área de trabajo temporal para terminar la ejecución. Cuando la sentencia termina la ejecución, las extensiones del segmento temporal vuelven a la instancia para un uso futuro. Especifique un tablespace temporal por defecto para cada usuario o un tablespace temporal por defecto que se utilice en toda la base de datos.

Oracle Database asigna el espacio dinámicamente. Cuando las extensiones existentes de un segmento están completas, se agregan extensiones adicionales. Debido a que las extensiones se asignan según sea necesario, las extensiones de un segmento pueden o no ser contiguas en el disco.

Procesamiento de Sentencias SQL

- Conexión a una instancia mediante:
 - El proceso de usuario
 - El proceso de servidor
- Los componentes del servidor de Oracle que se utilizan dependen del tipo de sentencia SQL:
 - Consultas que devuelven filas.
 - Cambios del log de sentencias de lenguaje de manipulación de datos (DML)
 - La confirmación asegura la recuperación de la transacción.
- Algunos componentes del servidor de Oracle no participan en el procesamiento de sentencias SQL.



Copyright © 2010, Oracle. Todos los derechos reservados.

Procesamiento de Sentencias SQL

No todos los componentes de una instancia de Oracle se utilizan para procesar sentencias SQL. Los procesos de usuario y de servidor se utilizan para conectarse a una instancia de Oracle. Estos procesos no forman parte de la instancia de Oracle, pero son necesarios para procesar una sentencia SQL.

Algunos de los procesos en segundo plano, estructuras SGA, archivos de base de datos se utilizan para procesar sentencias SQL. Según el tipo de sentencia SQL, se utilizan diferentes componentes:

- Las consultas necesitan un procesamiento adicional para devolver filas al usuario.
- Las sentencias DML necesitan un procesamiento adicional para registrar los cambios realizados en los datos.
- La confirmación del procesamiento asegura que los datos modificados en una transacción se pueden recuperar.

Algunos de los procesos en segundo plano necesarios no participan directamente en el procesamiento de una sentencia SQL, pero se utilizan para mejorar el rendimiento y recuperar la base de datos. Por ejemplo, el proceso en segundo plano del proceso de archivado opcional, ARCn, se utiliza para garantizar que una base de datos producción se puede recuperar.

Procesamiento de Consultas

- Analizar:
 - Buscar una sentencia idéntica.
 - Comprobar la sintaxis, nombres de objetos y privilegios.
 - Bloquear los objetos utilizados durante el análisis.
 - Crear y almacenar el plan de ejecución.
- Ejecutar: identificar las filas seleccionadas.
- Recuperar: devolver las filas al proceso de usuario.



Copyright © 2010, Oracle. Todos los derechos reservados.

Procesamiento de Consultas

Las consultas son diferentes de otros tipos de sentencias SQL porque, si se realizan correctamente, devuelven datos como resultados. Otras sentencias simplemente devuelven resultados correctos o incorrectos, mientras que una consulta puede devolver una fila o miles de filas.

Hay tres etapas principales en el procesamiento de una consulta.

- Análisis
- Ejecución
- Recuperación

Durante la etapa de *análisis*, la sentencia SQL se transfiere del proceso de usuario al proceso de servidor y una representación analizada de la sentencia SQL se carga en un área SQL compartida.

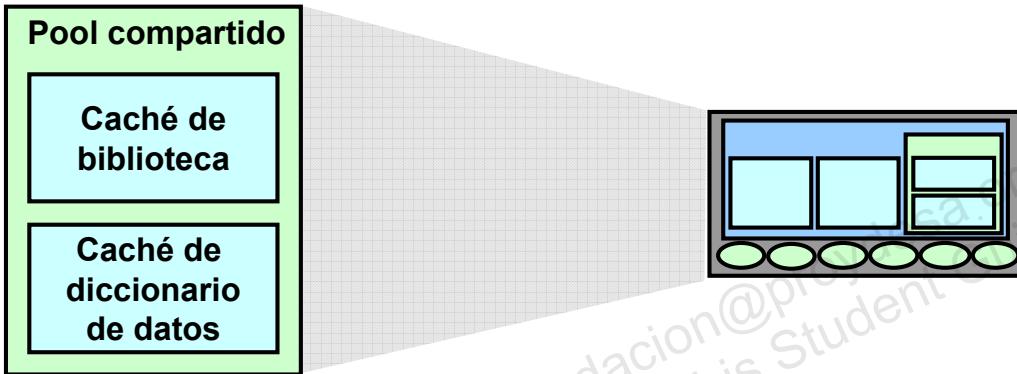
Durante el análisis, el proceso de servidor realiza las siguientes funciones:

- Busca una copia existente de la sentencia en el pool compartido
- Valida la sentencia SQL mediante la comprobación de la sintaxis
- Realiza búsquedas de diccionario de datos para validar las definiciones de tabla y columna

La etapa de ejecución ejecuta la sentencia mediante el mejor enfoque del optimizador y la etapa de recuperación recupera las filas para el usuario.

Pool Compartido

- La caché de biblioteca contiene texto de sentencias SQL, código analizado y plan de ejecución.
- La caché de diccionario de datos contiene privilegios y definiciones de tabla, columna y otro objeto.
- Al pool compartido se le asigna un tamaño mediante SHARED_POOL_SIZE.



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Pool Compartido

Durante la etapa de análisis, el proceso de servidor utiliza el área en el SGA conocida como el pool compartido para compilar la sentencia SQL. El pool compartido tiene dos componentes principales:

- Caché de biblioteca
- Caché de diccionario de datos

Caché de Biblioteca

La caché de biblioteca almacena información sobre las sentencias SQL utilizadas más recientemente en una estructura de memoria denominada área SQL compartida. El área SQL compartida contiene:

- El texto de la sentencia SQL
- El árbol de análisis, que es una versión compilada de la sentencia
- El plan de ejecución, con los pasos que se deben realizar al ejecutar la sentencia

El optimizador es la función en el servidor de Oracle que determina el plan de ejecución óptimo.

Si se vuelve a ejecutar una sentencia SQL y un área SQL compartida contiene el plan de ejecución para la sentencia, el proceso de servidor no necesita analizar la sentencia. La caché de biblioteca mejora el rendimiento de las aplicaciones que vuelven a utilizar sentencias SQL reduciendo el tiempo de análisis y los requisitos de memoria. Si no se vuelve a utilizar la sentencia SQL, se quedan obsoletos eventualmente en la caché de biblioteca.

Pool Compartido (continuación)

Caché de Diccionario de Datos

La caché de diccionario de datos, también conocida como caché de diccionario o caché de fila, es una recopilación de las definiciones utilizadas más recientemente en la base de datos. Incluye información sobre archivos, tablas, índices, columnas, usuarios, privilegios y otros objetos de base de datos.

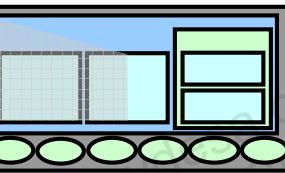
Durante la fase de análisis, el proceso de servidor para la información en la caché de diccionario para resolver los nombres de objeto especificados en la sentencia SQL y para validar los privilegios de acceso. Si necesario, el proceso de servidor inicia la carga de esta información del archivo de datos.

Ajuste de Tamaño del Pool Compartido

El tamaño del pool compartido se especifica mediante el parámetro de inicialización `SHARED_POOL_SIZE`.

Caché de buffers de la base de datos

- La caché de buffers de la base de datos almacena los bloques utilizados más recientemente.
- El tamaño del buffer se basa en DB_BLOCK_SIZE.
- El número de buffers se define mediante DB_BLOCK_BUFFERS.



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Caché de Buffers de la Base de Datos

Cuando se procesa una consulta, el proceso del servidor busca en la caché de buffers de la base de datos cualquier bloque que necesita. Si no encuentra el bloque en la caché de buffers de la base de datos, el proceso de servidor lee el bloque del archivo de datos y coloca una copia en esa caché de la base de datos. Puesto que es posible que solicitudes posteriores del mismo bloque encuentren el bloque en memoria, las solicitudes no requerirán lecturas físicas. El servidor de Oracle utiliza el algoritmo de uso menos reciente para dejar obsoletos los buffers a los que no se ha accedido últimamente y crear espacio para los nuevos bloques en la caché de buffers.

Ajuste de Tamaño de Caché de Buffers de la Base de Datos

El tamaño de cada buffer en la caché de buffers es igual al tamaño de un bloque de Oracle y se especifica mediante el parámetro DB_BLOCK_SIZE. El número de buffers es igual al valor del parámetro DB_BLOCK_BUFFERS.

Área Global de Programa (PGA)

- No se puede compartir
- Sólo puede escribir el proceso de servidor
- Contiene:
 - Área de ordenación
 - Información de la sesión
 - Estado del cursor
 - Espacio de pila



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

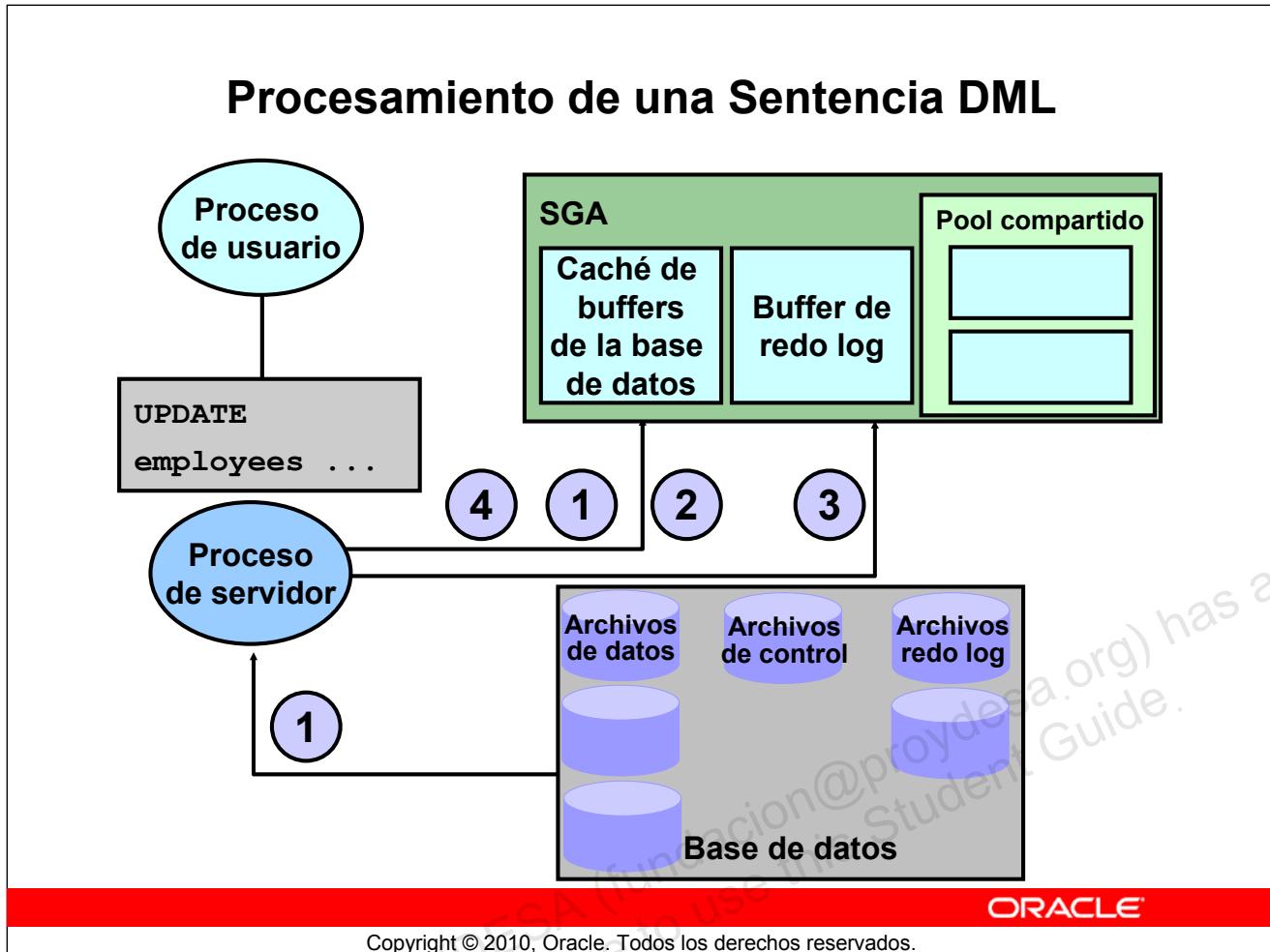
Área Global de Programa (PGA)

Un Área Global de Programa (PGA) es una región de la memoria que contiene datos e información de control para un proceso de servidor. Se trata de una memoria no compartida creada por Oracle cuando se inicia un proceso de servidor. El acceso es exclusivo para dicho proceso de servidor y se lee y escribe sólo mediante el servidor de Oracle que actúa en su nombre. La memoria PGA asignada por cada proceso de servidor relacionado con una instancia de Oracle que se denomina memoria PGA agregada asignada por la instancia.

En una configuración de servidor dedicado, el PGA del servidor incluye los siguientes componentes:

- **Área de ordenación:** se utiliza para cualquier ordenación que pueda ser necesaria para procesar la sentencia SQL
- **Información de la sesión:** incluye los privilegios de usuario y estadísticas de rendimiento de la sesión
- **Estado del cursor:** indica la etapa en el procesamiento de las sentencias SQL que se utilizan actualmente mediante la sesión
- **Espacio de pila:** contiene otras variables de sesión

El PGA se asigna al crear un proceso y se anula la asignación al terminar el proceso.



Procesamiento de una Sentencia DML

Una sentencia de lenguaje de manipulación de datos (DML) necesita sólo dos fases de procesamiento:

- El análisis es igual que la fase de análisis utilizada para procesar una consulta.
- La ejecución necesita procesamiento adicional para realizar cambios de datos.

Fase de Ejecución DML

Para ejecutar una sentencia DML:

- Si los datos y los bloques de rollback no están aún en la caché de buffers, el proceso de servidor las lee a partir de los archivos de datos en la caché de buffers
- El proceso de servidor bloquea las filas que se van a modificar
- En el buffer de redo log, el proceso de servidor registra los cambios que se van a realizar en el rollback y bloques de datos
- Los cambios de bloque de rollback registran los valores de los datos antes de su modificación. El bloque de rollback se utiliza para almacenar la "imagen previa" de los datos de manera que las sentencias DML puedan realizar un rollback en caso necesario.
- Los cambios de datos registran los nuevos valores de los datos

Procesamiento de una Sentencia DML (continuación)

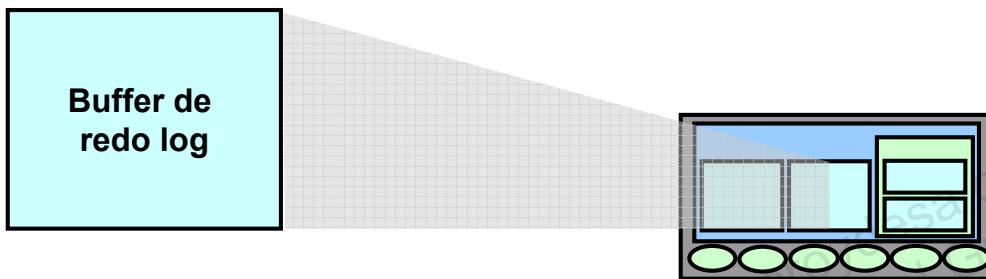
El proceso de servidor registra la "imagen anterior" en el bloque de rollback y actualiza el bloque de datos. Ambos de estos cambios se realizan en la caché de buffers de la base de datos. Todos los bloques modificados en la caché de buffers se marcan como buffers sucios (por lo que los buffers no son iguales que los de los bloques correspondientes del disco).

El procesamiento de un comando `DELETE` o `INSERT` utiliza pasos similares. La "imagen anterior" de `DELETE` contiene los valores de la columna en la fila suprimida y la "imagen anterior" de `INSERT` la información de ubicación de la fila.

Debido a que los cambios realizados en los bloques sólo se registran en las estructuras de memoria y no se escriben inmediatamente en el disco, si se produce un fallo de la computadora que provoque la pérdida del SGA también se pueden perder estos cambios.

Buffer de Redo Log

- El tamaño se define mediante `LOG_BUFFER`
- Registra los cambios realizados a través de la instancia
- Se utiliza de forma secuencial
- Es un buffer circular



Copyright © 2010, Oracle. Todos los derechos reservados.

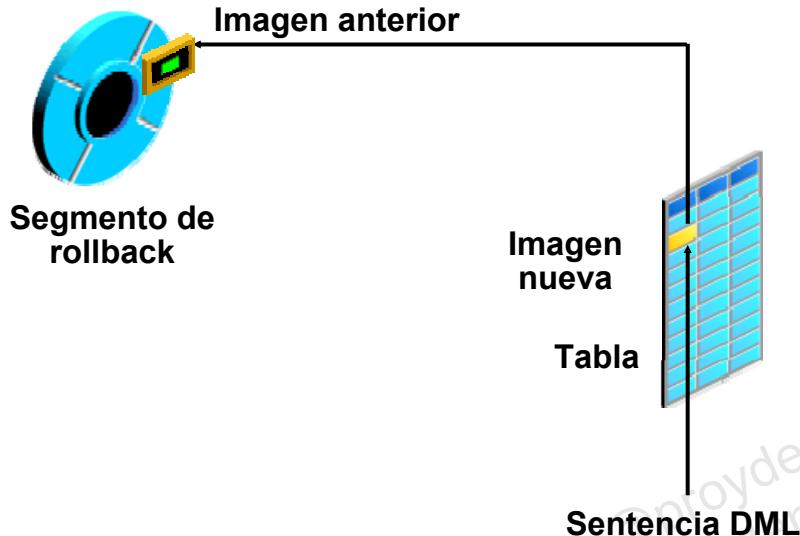
ORACLE

Buffer de Redo Log

El proceso de servidor registra la mayoría de los cambios realizados en los bloques de archivo de datos en el buffer de redo log, que forma parte del SGA. El buffer de redo log tiene las siguientes características:

- Su tamaño en bytes se define mediante el parámetro `LOG_BUFFER`.
- Registra el bloque que se ha cambiado, la ubicación del cambio y el nuevo valor en una entrada de redo. Una entrada de redo no hace distinción entre los tipos de bloque que han cambiado; sólo registra los bytes cambiado en el bloque.
- El buffer de redo log se utiliza de forma secuencial y los cambios realizados mediante una transacción se pueden intercalar con los cambios realizados por otras transacciones.
- Es un buffer circular que se vuelve a utilizar una vez lleno, pero sólo después de registrar todas las entradas de redo antiguas en los archivos de redo log.

Segmento de Rollback



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

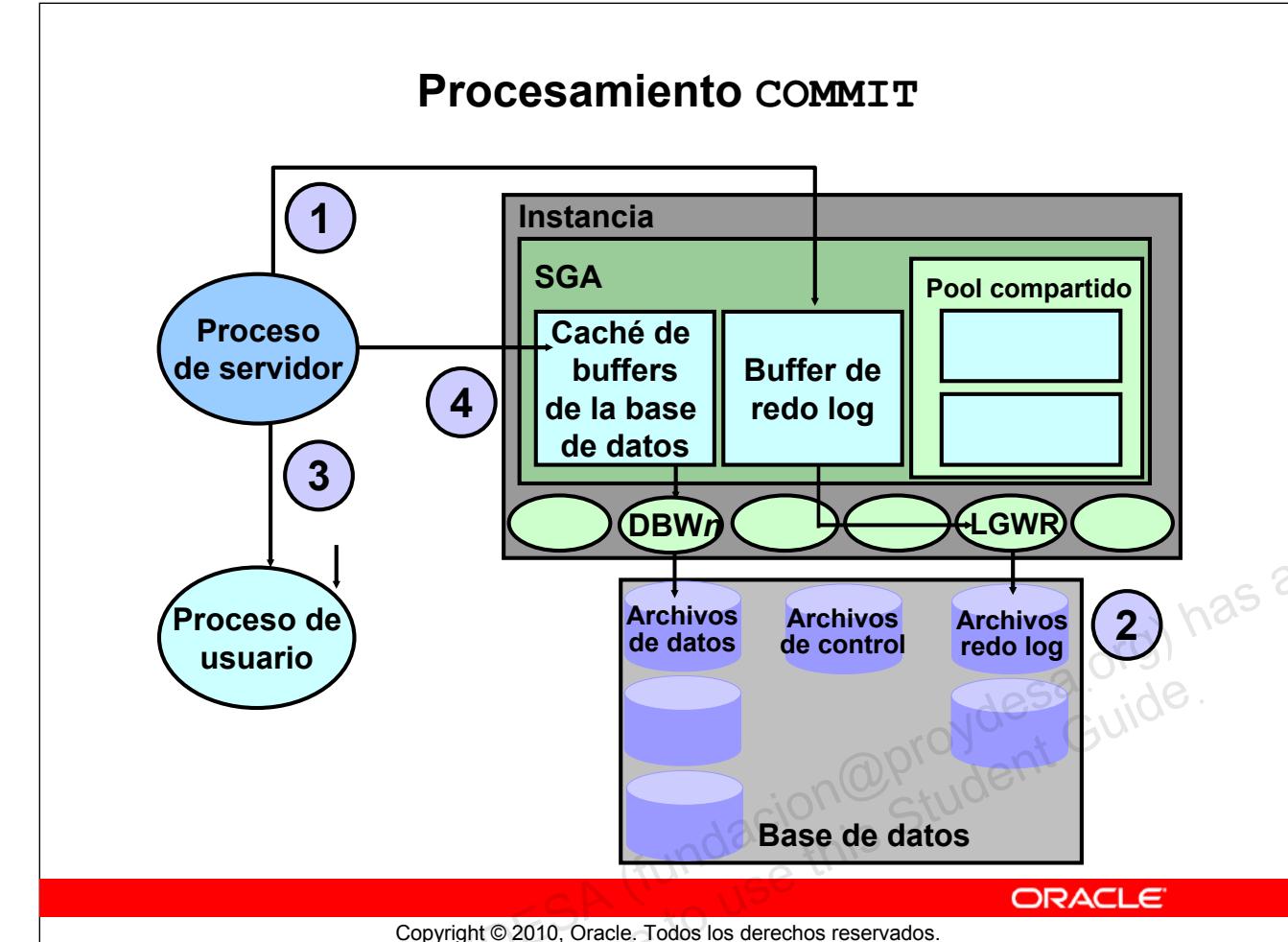
Segmento de Rollback

Antes de realizar un cambio, el proceso de servidor guarda el valor de datos antiguos en un segmento de rollback. Esta "imagen anterior" se utiliza para:

- Deshacer los cambios si se realiza un rollback de la transacción
- Proporcionar consistencia de lectura garantizando que otras transacciones no vean los cambios realizados no confirmados por la sentencia DML
- Recuperar la base de datos en estado consistente en caso de fallos

Los segmentos de rollback, como tablas e índices, existen en los archivos de datos y los bloqueos de rollback se incluyen en la caché de buffers de base de datos según sea necesario. DBA crea los segmentos de rollback.

Los cambios en los segmentos de rollback se registran en el buffer de redo log.



ORACLE

Copyright © 2010, Oracle. Todos los derechos reservados.

Procesamiento COMMIT

El servidor de Oracle utiliza un mecanismo COMMIT rápido que garantiza que los cambios confirmados se pueden recuperar en caso de fallo de la instancia.

Número de Cambio del Sistema (SCN)

Cuando se confirma una transacción, el servidor de Oracle asigna un SCN de confirmación a la transacción. El SCN se incrementa de forma monótona y es único dentro de la base de datos. El servidor de Oracle lo utiliza como registro de hora interno para sincronizar los datos y proporcionar consistencia de lectura cuando se recuperan datos de los archivos de datos. El uso del SCN permite al servidor de Oracle realizar comprobaciones de consistencia sin depender de la fecha y la hora del sistema operativo.

Pasos para Procesar COMMIT

Al ejecutar COMMIT, se realizan los siguientes pasos:

1. El proceso de servidor coloca un registro de confirmación, junto con el SCN, en el buffer de redo log.
2. LGWR realiza una escritura contigua de todas las entradas de buffers de redo log e incluye el registro de confirmación en los archivos redo log. Después de esto, el servidor de Oracle garantiza que no se perderán los cambios aunque se produzca un fallo de la instancia.

Procesamiento COMMIT (continuación)

3. Se informa a los usuarios que se ha completado la sentencia COMMIT.
4. El proceso de servidor registra información para indicar que la transacción se ha completado y los bloqueos de recursos se han liberado.

El vaciado de buffers sucios en el archivo de datos lo realiza DBW0 de forma independiente y se puede producir antes o después de la confirmación.

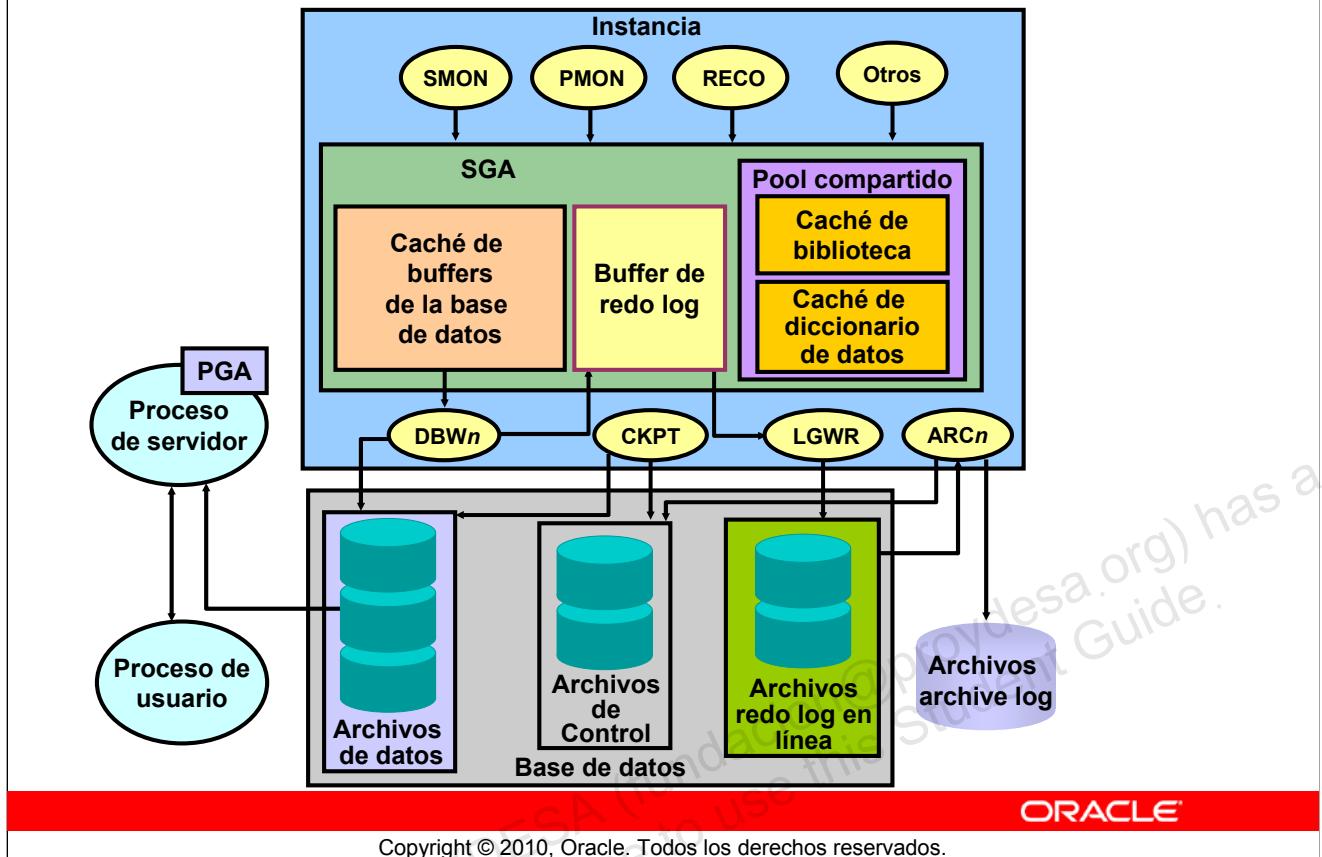
Ventajas del Mecanismo Rápido COMMIT

El mecanismo COMMIT rápido garantiza la recuperación de datos escribiendo los cambios en el buffer de redo log en lugar de en los archivos de datos. Tiene las siguientes ventajas:

- Las escrituras secuenciales en los archivos log son más rápidas que la escritura en diferentes bloques en el archivo de datos.
- Sólo la mínima información necesaria para registrar los cambios escritos en los archivos logs; la escritura en los archivos de datos requiere que se escriban los bloques de archivos de datos completos.
- Si varias transacciones solicitan confirmación al mismo tiempo, el redo log de piggybacks de instancia se registra en una única escritura.
- A menos que el buffer de redo log esté parcialmente lleno, sólo es necesaria una escritura síncrona por transacción. Si se producen piggybacks, puede haber menos de una escritura síncrona por transacción.
- Debido a que el buffer de redo log se puede vaciar antes de ejecutar COMMIT, el tamaño de la transacción no afecta a la cantidad de tiempo necesaria para una operación real de COMMIT.

Nota: el rollback de una transacción no dispara LGWR para escribir en el disco. El servidor de Oracle siempre realiza un rollback de los cambios no confirmados en la recuperación de fallos. Si existe un fallo después de un rollback, antes de registrar las entradas de rollback en el disco, la ausencia del registro de confirmación es suficiente para garantizar que se realiza un rollback de los cambios realizado por la transacción.

Resumen de Arquitectura de Oracle Database



Resumen de Arquitectura de Oracle Database

Una Oracle Database consta de una instancia y su base de datos asociada:

- Una instancia consta del SGA y los procesos en segundo plano
 - **SGA:** caché de buffers de base de datos, buffer de redo log, pool compartido, etc.
 - **Procesos en segundo plano:** SMON, PMON, DBW n , CKPT, LGWR, etc.
- Una base de datos consta de estructuras de almacenamiento:
 - **Lógicas:** tablespaces, esquemas, segmentos, extensiones y bloque de Oracle
 - **Físicas:** archivos de datos, archivos de control, archivos redo log

Si un usuario accede a Oracle Database a través de una aplicación, un proceso de servidor se comunica con la instancia en nombre del proceso de usuario.

Prácticas y soluciones adicionales

Tabla de Contenido

Prácticas Adicionales	3
Prácticas Adicionales	4
Prácticas Adicionales: Caso Práctico	10
Solución de Prácticas Adicionales	13
Solución de Prácticas Adicionales	14
Prácticas Adicionales: Soluciones del Caso Práctico	20

Prácticas Adicionales

Los siguientes ejercicios se pueden utilizar como práctica adicional después de tratar las sentencias de lenguaje de manipulación de datos (DML) y de lenguaje de definición de datos (DDL) en las lecciones tituladas "Gestión de Objetos de Esquema" y "Manipulación de Juegos de Datos Grandes".

Nota: ejecute los scripts `lab_ap_cre_special_sal.sql`, `lab_ap_cre_sal_history.sql` y `lab_ap_cre_mgr_history.sql` en la carpeta `labs` para crear las tablas `SPECIAL_SAL`, `SAL_HISTORY` y `MGR_HISTORY`.

Prácticas Adicionales

- El departamento de recursos humanos desea obtener una lista de empleados mal pagados, el historial de salarios de empleados y el historial de salarios de gestores según una encuesta de salario de la industria. Por lo que se le pide que realice lo siguiente:

Escriba una sentencia para realizar lo siguiente:

- Recupere los detalles, como ID de empleado, fecha de contratación, salario e ID de gestor, de aquellos empleados cuyo ID de empleado sea mayor o igual a 200 de la tabla EMPLOYEES.
- Si el salario es menor de 5.000 dólares, inserte los detalles como el ID de empleado y salario en la tabla SPECIAL_SAL.
- Inserte los detalles, como ID de empleado, fecha de contratación y salario, en la tabla SAL_HISTORY.
- Inserte los detalles como el ID de empleado, ID de gestor y salario en la tabla MGR_HISTORY.

- Consulte las tablas SPECIAL_SAL, SAL_HISTORY y MGR_HISTORY para ver los registros insertados.

SPECIAL_SAL

	EMPLOYEE_ID	SALARY
1	200	4400

SAL_HISTORY

	EMPLOYEE_ID	HIRE_DATE	SALARY
1	201	17-FEB-1996	13000
2	202	17-AUG-1997	6000
3	203	07-JUN-1994	6500
4	204	07-JUN-1994	10000
5	205	07-JUN-1994	12000
6	206	07-JUN-1994	8300

MGR_HISTORY

Prácticas Adicionales (continuación)

EMPLOYEE_ID	MANAGER_ID	SALARY
201	100	13000
202	201	6000
203	101	6500
204	101	10000
205	101	12000
206	205	8300

3. Nita, el DBA, necesita que cree una tabla, con una restricción de clave primaria, pero desea que el índice tenga un nombre diferente al nombre de la restricción. Cree la tabla LOCATIONS_NAMED_INDEX según el siguiente gráfico de instancias de tabla. Asigne un nombre al índice para la columna PRIMARY KEY como LOCATIONS_PK_IDX.

Column Name	Deptno	Dname
Primary Key	Yes	
Data Type	Number	VARCHAR2
Length	4	30

4. Consulte la tabla USER_INDEXES para mostrar INDEX_NAME para la tabla LOCATIONS_NAMED_INDEX.

INDEX_NAME	TABLE_NAME
LOCATIONS_PK_IDX	LOCATIONS_NAMED_INDEX

Prácticas Adicionales (continuación)

Los siguientes ejercicios se pueden utilizar como práctica adicional después de tratar las funciones datetime.

Trabaja para una compañía global y el nuevo vicepresidente de operaciones desea conocer las diferentes zonas horarias de todas las sucursales de la compañía. El nuevo vicepresidente ha solicitado la siguiente información:

5. Modifique la sesión para definir NLS_DATE_FORMAT en DD-MON-YYYY
HH24:MI:SS.

6.

- a. Escriba consultas para mostrar los desplazamientos de zona horaria (TZ_OFFSET) de las siguientes zonas horarias:
– Australia/Sydney

<code>TZ_OFFSET('AUSTRALIA/SYDNEY')</code>
1 +10:00

- Chile/Isla de Pascua

<code>TZ_OFFSET('CHILE/EASTERISLAND')</code>
1 -06:00

- b. Modifique la sesión para definir el valor del parámetro TIME_ZONE en el desplazamiento de zona horaria de Australia/Sídney.
c. Muestre SYSDATE, CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP para esta sesión.

Nota: la salida puede ser diferente según la fecha de ejecución del comando.

SYSDATE	CURRENT_DATE	CURRENT_TIMESTAMP	LOCALTIMESTAMP
1 02-JUL-2009 17:11:46	02-JUL-2009 20:11:46	02-JUL-09 08.11.46.000000000 PM +10:00	02-JUL-09 08.11.46.000000000 PM

- d. Modifique la sesión para definir el valor del parámetro TIME_ZONE en el desplazamiento de zona horaria de Chile/Isla de Pascua.

Nota: los resultados de la pregunta anterior se basan en una fecha diferente y, en algunos casos, no coincidirán con los resultados reales que obtienen los estudiantes. Además, el desplazamiento de zona horaria de distintos países puede variar según la hora del horario de verano.

- e. Muestre SYSDATE, CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP para esta sesión.

Nota: la salida puede ser diferente según la fecha de ejecución del comando.

SYSDATE	CURRENT_DATE	CURRENT_TIMESTAMP	LOCALTIMESTAMP
1 02-JUL-2009 17:12:33	02-JUL-2009 04:12:33	02-JUL-09 04.12.33.000000000 AM -06:00	02-JUL-09 04.12.33.000000000 AM

- f. Modifique la sesión para definir NLS_DATE_FORMAT en DD-MON-YYYY.

Prácticas Adicionales (continuación)

Nota

- Observe que en la pregunta anterior CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP son sensibles a mayúsculas/minúsculas en la zona horaria de la sesión. Observe que SYSDATE no es sensible a mayúsculas/minúsculas en la zona horaria de la sesión.
 - Los resultados de la pregunta anterior se basan en una fecha diferente y, en algunos casos, no coincidirán con los resultados reales que obtienen los estudiantes. Además, el desplazamiento de zona horaria de distintos países puede variar según la hora del horario de verano.
7. El departamento de recursos humanos desea una lista de empleados activos para la revisión en enero, por lo que se le ha solicitado que realice lo siguiente:
 Escriba una consulta para mostrar los apellidos, el mes de la fecha de contratación y la fecha de contracción de aquellos empleados a los que se les ha contratado en el mes de enero, sea cual sea el año de la contratación.

	LAST_NAME	EXTRACT(MONTHFROMHIRE_DATE)	HIRE_DATE
1	Grant	1	13-JAN-2000
2	De Haan	1	13-JAN-1993
3	Hunold	1	03-JAN-1990
4	Landry	1	14-JAN-1999
5	Davies	1	29-JAN-1997
6	Partners	1	05-JAN-1997
7	Zlotkey	1	29-JAN-2000
8	Tucker	1	30-JAN-1997
9	King	1	30-JAN-1996
10	Marvins	1	24-JAN-2000
11	Fox	1	24-JAN-1998
12	Johnson	1	04-JAN-2000
13	Taylor	1	24-JAN-1998
14	Sarchand	1	27-JAN-1996

Prácticas Adicionales (continuación)

Estos ejercicios se pueden utilizar como práctica adicional después de tratar las consultas avanzadas.

8. El presidente necesita un informe sobre los tres asalariados principales de la compañía para el reparto de beneficios. Es el responsable de proporcionar al presidente una lista. Escriba una consulta para mostrar los tres asalariados principales en la tabla EMPLOYEES. Muestre sus apellidos y salarios.

	LAST_NAME	SALARY
1	King	24000
2	Kochhar	17000
3	De Haan	17000

9. La prestación del estado de California ha cambiado según una ordenanza local. De manera que el representante de la prestación ha solicitado que compile una lista de personas afectadas.

Escriba una consulta para mostrar el ID de empleado y los apellidos de los empleados que trabajan en el estado de California.

Indicación: utilice subconsultas escalares.

	EMPLOYEE_ID	LAST_NAME
1	198	O'Connell
2	199	Grant
3	120	Weiss
4	121	Fripp
5	122	Kaufling
6	123	Vollman
7	124	Mourgos
8	125	Nayer
9	126	Mikkilineni
10	127	Landry
11	128	Markle

10. Nita, el DBA, desea eliminar la información antigua de la base de datos. Los registros de empleados antiguos es una de las cosas que cree que no es necesaria. Le pide que realice lo siguiente:

Escriba una consulta para suprimir la fila más antigua JOB_HISTORY de un empleado consultando la tabla JOB_HISTORY para MIN(START_DATE) del empleado. Suprima los registros *sólo* de aquellos empleados que hayan tenido al menos dos cargos.

Indicación: utilice un comando correlacionado DELETE.

Prácticas Adicionales (continuación)

11. El vicepresidente de recursos humanos necesita los registros completos de empleados para el discurso del banquete de reconocimiento de empleados anual. El vicepresidente realiza una llamada rápida para que no siga las órdenes del DBA.

Realice un rollback de la transacción.

12. La lentitud del crecimiento económico está obligando a tomar medidas de gestión para la reducción de costes. El presidente desea revisar los cargos mejor pagados de la compañía. Es responsable de proporcionar al presidente una lista basada en las siguientes especificaciones:

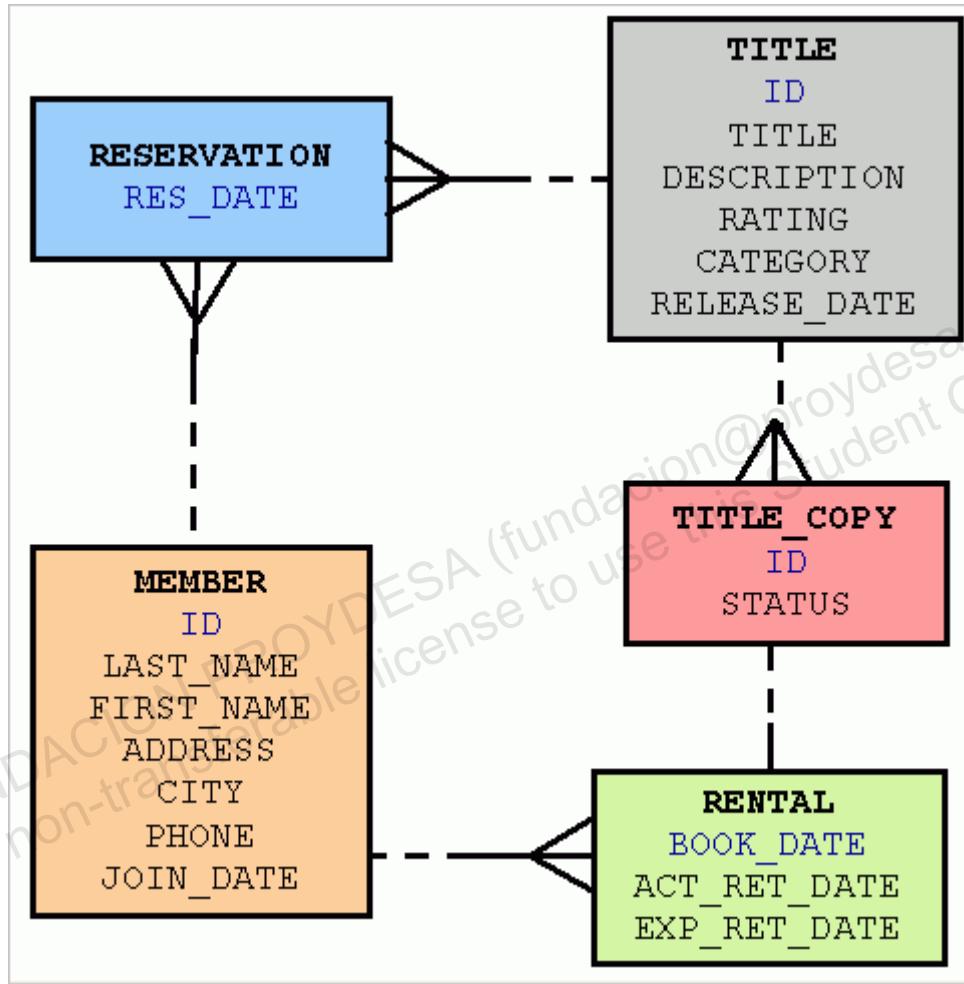
Escriba una consulta para mostrar los ID de cargo de aquellos cargos cuyo salario máximo esté por encima de la mitad del salario máximo en toda la compañía. Utilice la cláusula WITH para escribir esta consulta. Asigne el nombre MAX_SAL_CALC a la consulta.

JOB_TITLE	JOB_TOTAL
1 President	24000
2 Administration Vice President	17000
3 Sales Manager	14000
4 Marketing Manager	13000

Prácticas Adicionales: Caso Práctico

En el caso práctico para el curso *Conceptos Fundamentales de SQL I*, ha creado un juego de tablas de base de datos para una aplicación de vídeo. Además, ha insertado, actualizado y suprimido los registros de una base de datos de almacén de un videoclub y generado un informe.

A continuación, se muestra un diagrama de las tablas y columnas creadas para la aplicación de vídeo:



Nota: en primer lugar, ejecute el script `dropvid.sql` en la carpeta `labs` para borrar las tablas si ya existen. A continuación, ejecute el script `buildvid.sql` en la carpeta `labs` para crear y llenar las tablas.

1. Verifique que las tablas se han creado correctamente ejecutando un informe para mostrar la lista de tablas y sus definiciones de columna.

Prácticas Adicionales: Caso Práctico (continuación)

TABLE_NAME	COLUMN_NAME	DATA_TYPE	NULLABLE
1 MEMBER	MEMBER_ID	NUMBER	N
2 MEMBER	LAST_NAME	VARCHAR2	N
3 MEMBER	FIRST_NAME	VARCHAR2	Y
4 MEMBER	ADDRESS	VARCHAR2	Y
5 MEMBER	CITY	VARCHAR2	Y
6 MEMBER	PHONE	VARCHAR2	Y
7 MEMBER	JOIN_DATE	DATE	N
8 RENTAL	BOOK_DATE	DATE	N
9 RENTAL	COPY_ID	NUMBER	N
10 RENTAL	MEMBER_ID	NUMBER	N
11 RENTAL	TITLE_ID	NUMBER	N
12 RENTAL	ACT_RET_DATE	DATE	Y
13 RENTAL	EXP_RET_DATE	DATE	Y
14 RESERVATION	RES_DATE	DATE	N
15 RESERVATION	MEMBER_ID	NUMBER	N
16 RESERVATION	TITLE_ID	NUMBER	N

2. Verifique la existencia de las secuencias MEMBER_ID_SEQ y TITLE_ID_SEQ en el diccionario de datos.

SEQUENCE_NAME
1 DEPARTMENTS_SEQ
2 EMPLOYEES_SEQ
3 LOCATIONS_SEQ
4 MEMBER_ID_SEQ
5 TITLE_ID_SEQ

3. Desea crear algunos usuarios con acceso sólo a sus propios alquileres. Cree un usuario denominado Carmen y otórguele el privilegio para seleccionar de la tabla RENTAL.

Nota: asegúrese de anteponer al nombre de usuario su cuenta de base de datos.

Por ejemplo, si es el usuario oraxx, cree un usuario denominado oraxx_Carmen.

4. Agregue una columna de precio (número 4,2) a la tabla TITLE para almacenar cuánto cuesta alquilar el título.
5. Agregue una tabla CATEGORY para almacenar CATEGORY_ID y CATEGORY_DESCRIPTION. La tabla tiene una clave ajena con la columna CATEGORY en la tabla TITLE.
6. Seleccione todas las tablas del diccionario de datos.
7. Ya no es necesario almacenar las reservas. Puede borrar la tabla.

Prácticas Adicionales: Caso Práctico (continuación)

8. Cree una tabla RENTAL_HISTORY para almacenar los detalles de un alquiler por miembro durante los últimos seis meses. (**Indicación:** puede copiar la tabla RENTAL).
9. Muestre una lista de los 10 principales títulos alquilados en el último mes agrupados por categoría.

CATEGORY	TITLE
1 ACTION	Soda Gang
2 CHILD	Willie and Christmas Too
3 COMEDY	My Day Off
4 SCIFI	Alien Again
5 SCIFI	Interstellar Wars

10. Desea calcular el recargo (precio del título/día) si el miembro devuelve el video seis días más tarde.

TITLE	MEMBER_ID	PRICE	LATEFEE
1 Alien Again	101	(null)	(null)
2 My Day Off	102	(null)	(null)
3 Interstellar Wars	101	(null)	(null)

11. Muestre una lista de los miembros que han alquilado dos o más veces.

MEMBER_ID	LAST_NAME	FIRST_NAME
1	101 Velasquez	Carmen

12. Muestre una lista de los títulos con estado alquilado.

TITLE
1 Alien Again
2 My Day Off
3 Interstellar Wars

13. Muestre una lista de miembros con "99" en sus números de teléfono.

POSITION	MEMBER_ID	LAST_NAME	FIRST_NAME
1	1	101 Velasquez	Carmen
2	1	106 Urguhart	Molly
3	1	109 Catchpole	Antoinette

Solución de Prácticas Adicionales

Los siguientes ejercicios se pueden utilizar como práctica adicional después de tratar las sentencias de lenguaje de manipulación de datos (DML) y de lenguaje de definición de datos (DDL) en las lecciones tituladas "Gestión de Objetos de Esquema" y "Manipulación de Juegos de Datos Grandes".

Nota: ejecute los scripts lab_ap_cre_special_sal.sql, lab_ap_cre_sal_history.sql y lab_ap_cre_mgr_history.sql en la carpeta labs para crear las tablas SPECIAL_SAL, SAL_HISTORY y MGR_HISTORY.

Solución de Prácticas Adicionales

- El departamento de recursos humanos desea obtener una lista de empleados mal pagados, el historial de salarios de empleados y el historial de salarios de gestores según una encuesta de salario de industria. Por lo que se le pide que realice lo siguiente:

Escriba una sentencia para realizar lo siguiente:

- Recupere los detalles, como ID de empleado, fecha de contratación, salario e ID de gestor, de aquellos empleados cuyo ID de empleado sea mayor o igual a 200 de la tabla EMPLOYEES.
- Si el salario es menor de 5.000 dólares, inserte los detalles, como ID de empleado y salario, en la tabla SPECIAL_SAL.
- Inserte los detalles, como ID de empleado, fecha de contratación y salario, en la tabla SAL_HISTORY.
- Inserte los detalles, como ID de empleado, ID de gestor y salario, en la tabla MGR_HISTORY.

```
INSERT ALL
WHEN SAL < 5000 THEN
  INTO special_sal VALUES (EMPID, SAL)
ELSE
  INTO sal_history VALUES(EMPID, HIREDATE, SAL)
  INTO mgr_history VALUES(EMPID, MGR, SAL)
SELECT employee_id EMPID, hire_date HIREDATE,
       salary SAL, manager_id MGR
FROM employees
WHERE employee_id >=200;
```

- Consulte las tablas SPECIAL_SAL, SAL_HISTORY y MGR_HISTORY para ver los registros insertados.

```
SELECT * FROM special_sal;
SELECT * FROM sal_history;
SELECT * FROM mgr_history;
```

- Nita, el DBA, necesita que cree una tabla, con una restricción de clave primaria, pero desea que el índice tenga un nombre diferente al nombre de la restricción. Cree la tabla LOCATIONS_NAMED_INDEX según el siguiente gráfico de instancias de tabla. Asigne un nombre al índice para la columna PRIMARY KEY, como LOCATIONS_PK_IDX.

Nombre de la Columna	Deptno	Dname
Clave Primaria	Yes	
Tipo de Dato	Number	VARCHAR2
Longitud	4	30

Solución de las Prácticas Adicionales (continuación)

```
CREATE TABLE LOCATIONS_NAMED_INDEX
(location_id NUMBER(4) PRIMARY KEY USING INDEX
(CREATE INDEX locations_pk_idx ON
LOCATIONS_NAMED_INDEX(location_id)),
location_name VARCHAR2(20));
```

4. Consulte la tabla USER_INDEXES para mostrar INDEX_NAME para la tabla LOCATIONS_NAMED_INDEX.

```
SELECT INDEX_NAME, TABLE_NAME
FROM USER_INDEXES
WHERE TABLE_NAME = 'LOCATIONS_NAMED_INDEX';
```

Solución de las Prácticas Adicionales (continuación)

Los siguientes ejercicios se pueden utilizar como práctica adicional después de tratar las funciones datetime.

Trabaja para una compañía global y el nuevo vicepresidente de operaciones desea conocer las diferentes zonas horarias de todas las sucursales de la compañía. El nuevo vicepresidente ha solicitado la siguiente información:

5. Modifique la sesión para definir NLS_DATE_FORMAT en DD-MON-YYYY HH24:MI:SS.

```
ALTER SESSION
SET NLS_DATE_FORMAT = 'DD-MON-YYYY HH24:MI:SS';
```

6.
 - a. Escriba consultas para mostrar los desplazamientos de zona horaria (TZ_OFFSET) de las siguientes zonas horarias:
 - Australia/Sídney
 - Chile/Isla de Pascua

```
SELECT TZ_OFFSET ('Australia/Sydney') from dual;
```

– Chile/Isla de Pascua

```
SELECT TZ_OFFSET ('Chile/EasterIsland') from dual;
```

- b. Modifique la sesión para definir el valor del parámetro TIME_ZONE en el desplazamiento de zona horaria de Australia/Sídney.

```
ALTER SESSION SET TIME_ZONE = '+10:00';
```

- c. Muestre SYSDATE, CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP para esta sesión.

Nota: la salida puede ser diferente según la fecha de ejecución del comando.

```
SELECT SYSDATE, CURRENT_DATE, CURRENT_TIMESTAMP,
LOCALTIMESTAMP FROM DUAL;
```

- d. Modifique la sesión para definir el valor del parámetro TIME_ZONE en el desplazamiento de zona horaria de Chile/Isla de Pascua.

Nota: los resultados de la pregunta anterior se basan en una fecha diferente y, en algunos casos, no coincidirán con los resultados reales que obtienen los estudiantes. Además, el desplazamiento de zona horaria de distintos países puede variar según la hora del horario de verano.

```
ALTER SESSION SET TIME_ZONE = '-06:00';
```

- e. Muestre SYSDATE, CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP para esta sesión.

Solución de las Prácticas Adicionales (continuación)

Nota: la salida puede ser diferente según la fecha de ejecución del comando.

```
SELECT SYSDATE, CURRENT_DATE, CURRENT_TIMESTAMP,
LOCALTIMESTAMP FROM DUAL;
```

- f. Modifique la sesión para definir NLS_DATE_FORMAT en DD-MON-YYYY.

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MON-YYYY';
```

Nota

- Observe que en la pregunta anterior CURRENT_DATE, CURRENT_TIMESTAMP y LOCALTIMESTAMP son sensibles a mayúsculas/minúsculas en la zona horaria de la sesión. Observe que SYSDATE no es sensible a mayúsculas/minúsculas en la zona horaria de la sesión.
- Los resultados de la pregunta anterior se basan en una fecha diferente y, en algunos casos, no coincidirán con los resultados reales que obtienen los estudiantes. Además, el desplazamiento de zona horaria de distintos países puede variar según la hora del horario de verano.

7. El departamento de recursos humanos desea una lista de empleados activos para la revisión en enero, por lo que se le ha solicitado que realice lo siguiente:

Escriba una consulta para mostrar los apellidos, mes de la fecha de contratación y fecha de contracción de aquellos empleados a los que se les ha contratado en el mes de enero, sea cual sea el año de la contratación.

```
SELECT last_name, EXTRACT (MONTH FROM HIRE_DATE), HIRE_DATE
FROM employees
WHERE EXTRACT (MONTH FROM HIRE_DATE) = 1;
```

Estos ejercicios se pueden utilizar como práctica adicional después de tratar las consultas avanzadas.

Nota: si ha convertido la columna HIRE_DATE a TIMESTAMP mediante code_05_12_sb.sql, la visualización de la columna HIRE_DATE será 13-JAN-00 12.00.00.000000000 AM

8. El presidente necesita un informe sobre los tres asalariados principales de la compañía para el reparto de beneficios. Es el responsable de proporcionar al presidente una lista.

Escriba una consulta para mostrar los tres asalariados principales en la tabla EMPLOYEES. Muestre sus apellidos y salarios.

```
SELECT last_name, salary
  FROM employees e
 WHERE 3 > (SELECT COUNT (*)
      FROM employees
     WHERE e.salary < salary);
```

Solución de las Prácticas Adicionales (continuación)

9. La prestación del estado de California ha cambiado según una ordenanza local, de manera que el representante de la prestación ha solicitado que compile una lista de personas afectadas. Escriba una consulta para mostrar el ID de empleado y los apellidos de los empleados que trabajan en el estado de California.

Indicación: utilice subconsultas escalares.

```
SELECT employee_id, last_name
  FROM employees e
 WHERE ( (SELECT location_id
            FROM departments d
           WHERE e.department_id = department_id )
        IN  (SELECT location_id
              FROM locations l
             WHERE state_province = 'California'));
```

10. Nita, el DBA, desea eliminar la información antigua de la base de datos. Los registros de empleados antiguos es una de las cosas que cree que no es necesaria. Le pide que realice lo siguiente:

Escriba una consulta para suprimir la fila más antigua JOB_HISTORY de un empleado consultando la tabla JOB_HISTORY para MIN(START_DATE) del empleado. Suprime los registros *sólo* de aquellos empleados que hayan tenido al menos dos cargos.

Indicación: utilice un comando correlacionado DELETE.

Solución de las Prácticas Adicionales (continuación)

```

DELETE FROM job_history JH
WHERE employee_id =
      (SELECT employee_id
       FROM employees E
       WHERE JH.employee_id = E.employee_id
       AND start_date = (SELECT MIN(start_date)
                          FROM job_history JH
                          WHERE JH.employee_id =
                                E.employee_id)
       AND 3 > (SELECT COUNT(*)
                  FROM job_history JH
                  WHERE JH.employee_id =
                        E.employee_id)
       GROUP BY EMPLOYEE_ID
       HAVING COUNT(*) >= 2));
  
```

11. El vicepresidente de recursos humanos necesita los registros completos de empleados para el discurso del banquete de reconocimiento de empleados anual. El vicepresidente realiza una llamada rápida para que no siga las órdenes del DBA.
- Realice un rollback de la transacción.

```
ROLLBACK;
```

12. La lentitud del crecimiento económico está obligando a tomar medidas de gestión para la reducción de costes. El presidente desea revisar los cargos mejor pagados de la compañía. Es responsable de proporcionar al presidente una lista basada en las siguientes especificaciones:

Escriba una consulta para mostrar los ID de cargo de aquellos cargos cuyo salario máximo esté por encima de la mitad del salario máximo en toda la compañía.

Utilice la cláusula WITH para escribir esta consulta. Asigne el nombre MAX_SAL_CALC a la consulta.

```

WITH
MAX_SAL_CALC AS (SELECT job_title, MAX(salary) AS
job_total
FROM employees, jobs
WHERE employees.job_id = jobs.job_id
GROUP BY job_title)
SELECT job_title, job_total
FROM MAX_SAL_CALC
WHERE job_total > (SELECT MAX(job_total) * 1/2
FROM MAX_SAL_CALC)
ORDER BY job_total DESC;
  
```

Prácticas Adicionales: Soluciones del Caso Práctico

En primer lugar, ejecute el script dropvid.sql en la carpeta labs para borrar las tablas si ya existen. A continuación, ejecute el script buildvid.sql en la carpeta labs para crear y llenar las tablas.

- Verifique que las tablas se han creado correctamente ejecutando un informe para mostrar la lista de tablas y sus definiciones de columna.

```
SELECT table_name,column_name,data_type,nullable
FROM user_tab_columns
WHERE table_name
IN ('MEMBER','TITLE','TITLE_COPY','RENTAL','RESERVATION');
```

- Verifique la existencia de las secuencias MEMBER_ID_SEQ y TITLE_ID_SEQ en el diccionario de datos.

```
SELECT sequence_name FROM user_sequences;
```

- Desea crear algunos usuarios con acceso sólo a sus propios alquileres. Cree un usuario denominado Carmen y otórguele el privilegio para seleccionar de la tabla RENTAL.

Nota: asegúrese de anteponer al nombre de usuario su cuenta de base de datos. Por ejemplo, si es el usuario oraxx, cree un usuario denominado oraxx_Carmen.

```
CREATE USER oraxx_carmen IDENTIFIED BY oracle ;
GRANT select ON rental TO oraxx_carmen;
```

- Agregue una columna de precio (número 4,2) a la tabla TITLE para almacenar cuánto cuesta alquilar el título.

```
ALTER TABLE title ADD(price NUMBER(6))
```

- Agregue una tabla CATEGORY para almacenar CATEGORY_ID y CATEGORY_DESCRIPTION. La tabla tiene una clave ajena con la columna CATEGORY en la tabla TITLE.

```
CREATE TABLE CATEGORY
(
    "CATEGORY_ID" NUMBER(6,0) NOT NULL ENABLE,
    "CATEGORY_DESCRIPTION" VARCHAR2(4000 BYTE),
    CONSTRAINT "CATEGORY_PK" PRIMARY KEY ("CATEGORY_ID"))
```

- Seleccione todas las tablas del diccionario de datos.

```
SELECT table_name FROM user_tables order by table_name;
```

Prácticas Adicionales: Soluciones del Caso Práctico (continuación)

7. Ya no es necesario almacenar las reservas. Puede borrar la tabla.

```
DROP TABLE reservation cascade constraints;
```

8. Cree una tabla RENTAL_HISTORY para almacenar los detalles de un alquiler por miembro durante los últimos seis meses. (**Indicación:** puede copiar la tabla RENTAL).

```
CREATE TABLE rental_history as select * from rental where '1' = '1'
```

9. Muestre una lista de los 10 principales títulos alquilados en el último mes agrupados por categoría.

```
SELECT t.CATEGORY, t.TITLE
FROM TITLE t, RENTAL r
WHERE t.TITLE_ID = r.TITLE_ID AND
      r.BOOK_DATE > (SYSDATE - 30) AND
      rownum < 10
order by category;
```

10. Desea calcular el recargo (precio del título/día) si el miembro devuelve el vídeo seis días más tarde.

```
SELECT t.title, m.member_id, t.price, (t.price*6) latefee
FROM title t, member m, rental r
WHERE t.title_id = r.title_id AND
      m.member_id = r.member_id AND
      r.act_ret_date is null;
```

11. Muestre una lista de los miembros que han alquilado dos o más veces.

```
SELECT member_id, last_name, first_name FROM member m
where 2 <= (select count(*) from rental_history where
member_id = m.member_id);
```

12. Muestre una lista de los títulos con el estado de alquilado.

```
SELECT t.title
FROM title t
JOIN (select title_id, status from title_copy) b
ON t.title_id = b.title_id AND b.status = 'RENTED';
```

13. Muestre una lista de miembros con "99" en sus números de teléfono.

```
SELECT REGEXP_COUNT(phone,'99',1,'i') position, member_id,
last_name, first_name
FROM member
WHERE REGEXP_COUNT(phone,'99',1,'i') > 0;
```

