



Capacitación ORACLE

Lenguaje SQL y Administración Básica

Gestión de Datos

Docentes: Ing. Juan Zaffaroni
Ing. María Cristina Chahin



Capacitación ORACLE. Lenguaje SQL y Administración Básica

AGENDA GENERAL MÓDULO 3: SQL Avanzado

❖ Día 10

- 6- Diccionario de Datos
 - Vistas
 - Generación de Scripts
- 7- Flashback queries

❖ Día 11

- 8- Gestión de Objetos
 - Tipos de Dato
 - DDL – Tablas
 - DDL – Vistas

❖ Día 12

- 8- Gestión de Objetos (cont.)
 - DDL – Restricciones

Capacitación ORACLE. Lenguaje SQL y Administración Básica

AGENDA GENERAL MÓDULO 3: SQL Avanzado (cont.)

- ❖ Día 13
 - 8- Gestión de Objetos (cont.)
 - DDL – Índices
- ❖ Día 14
 - 8- Gestión de Objetos (cont.)
 - DDL – Secuencias
 - DDL – DBLinks
 - DDL – Sinónimos
- ❖ Día 15
 - 8- Gestión de Objetos (cont.)
 - DDL – Vistas Materializadas (Snapshots)
- ❖ Día 16
 - Evaluación Módulo 3

6- Diccionario de Datos

- ✓Vistas del Diccionario de Datos
- ✓Generación de Scripts usando el Diccionario de Datos

Introducción

- Oracle mantiene una serie de tablas y vistas propias, que contienen información sobre otras estructuras de la base de datos (metadatos) y sobre la base en general.
- En este capítulo se explican las categorías de vistas del diccionario de datos, se detallan las más comúnmente utilizadas, y se muestra una forma de realizar scripts SQL ayudándose con estas vistas.

Características

- Conjunto de tablas y vistas mantenido exclusivamente por el servidor Oracle.
- Propiedad del usuario SYS.
- Hay 4 categorías, diferenciadas por su prefijo, según el uso que se les da:
 - *USER_*: Información sobre los objetos que son propiedad del usuario.
 - *ALL_*: Información sobre todos los objetos accesibles para el usuario.
 - *DBA_*: Información sobre todos los objetos de la base. Vistas restringidas a usuarios con rol DBA.
 - *V\$*: Vistas dinámicas de rendimiento, sobre el estado de la base.

6- Diccionario de Datos: Vistas del Diccionario de Datos.

Objetivo

- Recuperar información sobre objetos de la Base de Datos, para distintos perfiles de acceso.
- Recuperar información sobre el rendimiento de la Base de Datos.

Ejemplo

- Obtener los nombres de todos los índices que son propiedad del usuario.
- Buscar el nombre de todas las tablas que contengan columnas en las que aparezca la cadena "ID" en el nombre, entre todas las accesibles al usuario.
- Consultar todos los roles creados en la base.
- Obtener información de las sesiones actuales de la base.

6- Diccionario de Datos: Vistas del Diccionario de Datos.

Vistas más usadas

Diccionario

DICTIONARY: Descripciones de tablas y vistas del diccionario.

DICT_COLUMNS: Descripciones de columnas de tablas y vistas del diccionario.

Objetos más comunes

ALL_TABLES: Tablas.

ALL_TAB_COLUMNS: Columnas de tablas.

ALL_INDEXES: Índices.

ALL_IND_COLUMNS: Columnas de los índices de todas las tablas.

ALL_IND_EXPRESSIONS: Expresiones de índices basados en función de las tablas.

ALL_OBJECTS: Todos los objetos.

ALL_SEQUENCES: Secuencias.

ALL_SYNONYMS: Sinónimos.

ALL_VIEWS: Vistas.

Vistas más usadas (cont.)

Otros objetos

ALL_CONSTRAINTS: Definiciones de restricciones sobre tablas.

ALL_CONS_COLUMNS: Columnas especificadas en restricciones.

ALL_DB_LINKS: Enlaces de Base de Datos.

ALL_DIRECTORIES: Directorios

ALL_MVIEWS: Vistas materializadas.

ALL_BASE_TABLE_MVIEWS: Tablas base de vistas materializadas.

ALL_MVIEW_LOGS: Logs de vistas materializadas.

ALL_TRIGGERS: Triggers (del owner o sobre tablas del owner).

Particiones

ALL_TAB_PARTITIONS: Particiones de tablas.

ALL_IND_PARTITIONS: Particiones de índices.

ALL_PART_KEY_COLUMNS: Columnas que son clave de particionamiento de los objetos.

Vistas más usadas (cont.)

Privilegios

ALL_USERS: Usuarios de la base de datos.

ALL_COL_PRIVS: Grants sobre columnas para las cuales el usuario actual es owner, emisor o receptor del permiso. (Relacionadas: **ALL_COL_PRIVS_MADE** y

ALL_COL_PRIVS_RECD) (*)

ALL_TAB_PRIVS: Grants para los cuales el usuario actual es owner, emisor o receptor del permiso, o el receptor es un rol habilitado o PUBLIC.

(Relacionadas: **ALL_TAB_PRIVS_MADE** y **ALL_TAB_PRIVS_RECD**) (*)

DBA_ROLE_PRIVS: Roles concedidos a todos los usuarios y roles en la bd.

DBA_ROLES: Roles existentes en la bd.

DBA_SYS_PRIVS: Privilegios de sistema concedidos a usrs y roles.

ROLE_ROLE_PRIVS: Roles concedidos a otros roles.

ROLE_SYS_PRIVS: Privilegios de sistema concedidos a roles.

ROLE_TAB_PRIVS: Privilegios de tablas concedidos a roles.

SESSION_PRIVS: Privilegios disponibles para el usuario actual.

SESSION_ROLES: Role habilitados para el usuario actual.

(*) Tener en cuenta las variantes de estas vistas con las de igual nombre y prefijo USER_ o DBA_.

Vistas más usadas (cont.)

Base de Datos

DBA_DATA_FILES: Archivos de la base de datos.

DBA_DDL_LOCKS: Bloqueos de DDL actuales en la base de datos.

DBA_EXTENTS: Extensiones asociadas a los segmentos de todos los tablespaces de la bd.

DBA_ROLLBACK_SEGS: Segmentos de rollback.

DBA_SEGMENTS: Almacenamiento alocado para los segmentos de la base.

DBA_TABLESPACES: Tablespaces en la base de datos.

GLOBAL_NAME: Nombre global de la base actual.

Aplicación

- **SELECT index_name**
FROM USER_INDEXES;
- **SELECT table_name**
FROM ALL_TAB_COLUMNS
WHERE column_name like '%ID%';
- **SELECT role_name**
FROM DBA_ROLES;
- **SELECT ***
FROM V\$SESSION;

Objetivo

- Utilizar el [diccionario de datos](#) para:
 - Generar un archivo de comandos que realice operaciones DDL.
- Utilizar los metadatos de la base para:
 - Generar un archivo de comandos que realice operaciones DML.

Ejemplo

- Crear un archivo de comandos que cree copias de todas las tablas de su usuario, con igual nombre y sufijo “_copy”.
- Crear un archivo de comandos que llene la tabla departments con sus datos actuales.

Aplicación

- ```
SELECT 'CREATE TABLE ' || table_name || '_copy' |
 ' AS SELECT * FROM ' || table_name | |
 ' WHERE 1 = 2; '
FROM USER_TABLES;
```
- ```
SELECT 'INSERT INTO DEPARTMENTS VALUES ('  
      || department_id || ', ' || department_name || ', '  
      || location_id || ');'  
FROM   departments;
```

Resumen

- El DICCIONARIO DE DATOS de la Base de Datos Oracle contiene:

- Información sobre todos los **objetos** contenidos en la base de datos.
- Información sobre **usuarios, permisos y roles** creados en la base de datos.
- Vistas de información **estática** de la **base de datos**, clasificadas según el **nivel de acceso** de los usuarios.
- Información **dinámica** sobre el funcionamiento de la **base de datos**.

7- Flashback Queries

- ✓ Formas de consulta con consistencia de lectura

Introducción

- El uso de la **CONSISTENCIA DE LECTURA** garantiza que un conjunto de datos visto por una sentencia es consistente respecto de un punto en el tiempo y no cambia durante la ejecución.
- Flashback queries explica algunos mecanismos que permiten ejecutar sentencias consistentes a un punto dado en el tiempo, mediante:
 - Uso de transacciones de sólo lectura.
 - Consultas especificadas en un momento pasado en el tiempo.
 - Llamada al paquete DBMS_FLASHBACK, para alterar la fecha de la base de datos.

Objetivo

- Recuperar información de la Base de Datos que no se vea afectada por otras sentencias [una vez iniciadas las operaciones](#).
- [Recuperar información](#) de la Base de Datos tal cual se encontraba en un [momento específico anterior](#) en el tiempo.
- [Realizar operaciones](#) en la Base de Datos simulando un [momento específico anterior](#) en el tiempo.

Ejemplo

- Obtener la información de las tablas departments y employees, asegurando que se ve la información existente al inicio de la primera consulta.
- Obtener la información de la tabla locations, como estaba hace 2 horas.
- Configurar la base para que se ubique en el tiempo 1 día atrás. Luego, consultar la tabla jobs. Volver a restaurar el tiempo de la base.

7- Flashback Queries

Sintaxis

Transacción de sólo lectura

```
SET TRANSACTION READ ONLY;
```

...

```
SELECT ...;
```

La consistencia se inicia con el "set transaction". No se ven datos pasados.

Consulta en un momento anterior en el tiempo

```
SELECT ...
```

```
FROM ...
```

```
AS OF TIMESTAMP to_timestamp(...)
```

```
WHERE ...;
```

La consistencia se logra utilizando el mismo timestamp en las consultas.
Requiere privilegio de FLASHBACK sobre las tablas a consultar.

Seteo de la base a un momento anterior en el tiempo

```
EXECUTE DBMS_FLASHBACK.ENABLE_AT_TIME(to_date(...));
```

```
SELECT ...;
```

```
EXECUTE DBMS_FLASHBACK.DISABLE;
```

Las consultas no requieren referencia a fecha pasada (se ejecutan entre las llamadas al paquete).
Requiere permisos de ejecución sobre el paquete DBMS_FLASHBACK.

7- Flashback Queries

Aplicación

- SET TRANSACTION READ ONLY;
SELECT * FROM departments;
SELECT * FROM employees;
- SELECT *
FROM locations
AS OF TIMESTAMP
to_timestamp('20100901110000', 'yyyymmddhh24miss');
- EXECUTE DBMS_FLASHBACK.ENABLE_AT_TIME
to_date(sysdate - 1);
SELECT *
FROM jobs;
EXECUTE DBMS_FLASHBACK.DISABLE;

7- Flashback Queries

Resumen

- Mediante los FLASHBACK QUERIES podemos obtener información utilizando las características de consistencia de lectura de Oracle para:

- Recuperar *datos perdidos* o *deshacer cambios* no deseados (incluso ya *validados*).
- *Comparar* datos *actuales* contra los que había en un *momento anterior* en el tiempo.
- Permitir que ciertas aplicaciones puedan trabajar con *versiones pasadas de los datos*.

8- Gestión de Objetos

- ✓Tipos de Dato
- ✓Tablas
 - Creación. Modificación. Borrado. Truncamiento.
- ✓Vistas
 - Concepto
 - Creación. Borrado.
- ✓Restricciones
 - Tipos. Usos.
 - Creación. Borrado. Activación / Desactivación.
- ✓Índices
 - Concepto. Tipos
 - Creación. Uso. Borrado.

8- Gestión de Objetos (cont.)

- ✓ Secuencias
 - Concepto
 - Creación. Uso. Borrado.
- ✓ DB Links
 - Concepto.
 - Creación. Borrado
- ✓ Sinónimos
 - Concepto
 - Creación. Borrado.
- ✓ Vistas materializadas
 - Concepto.
 - Creación. Modificación. Borrado.

8- Gestión de Objetos

Introducción

- Las sentencias DDL permiten modificar las estructuras contenidas en la Base de Datos.
- Gestión de Objetos explica las operaciones necesarias para la gestión de:
 - Tablas
 - Creación / Modificación / Borrado / Truncamiento
 - Vistas
 - Creación / Borrado
 - Restricciones
 - Creación / Borrado / Activación / Desactivación
 - Índices
 - Tipos / Creación / Uso / Borrado
 - Secuencias
 - Creación / Uso / Borrado
 - DB Links
 - Creación / Borrado
 - Sinónimos
 - Creación / Borrado
 - Vistas materializadas
 - Creación / Modificación / Borrado

8- Gestión de Objetos: Tipos de Dato

Tipos de Dato

VARCHAR2(< tamaño >) : Dato de caracteres de longitud variable. Se indica el máximo.

CHAR(< tamaño >) : Dato de caracteres de longitud fija. Se indica el tamaño.

NUMBER(p, s) : Dato numérico de longitud variable. Se indica la precisión p (nro. total de dígitos) y la escala s (nro. de dígitos a la derecha de la coma).

DATE : Valores de fecha y hora hasta segundos.

LONG : Dato de caracteres de longitud variable hasta 2 Gb.

CLOB : Dato de caracteres de longitud variable hasta 4 Gb.

RAW y LONG RAW : Dato raw binario.

BLOB : Dato binario de hasta 4 Gb.

BFILE : Dato binario almacenado en un archivo externo hasta 4 Gb.

ROWID : Sistema numérico de base 64 que representa la dirección única de una fila en su tabla.

TIMESTAMP : Fecha y hora con segundos fraccionarios.

INTERVAL YEAR TO MONTH : Intervalo de años y meses.

INTERVAL DAY TO SECOND : Intervalo de días a segundos.

TIMESTAMP WITH TIME ZONE : Variante de **TIMESTAMP** con cambio en la zona horaria.

Capacitación Oracle SQL

25

8- Gestión de Objetos: Tablas - Creación

Objetivo

- [Crear nuevas tablas](#) para el almacenamiento de datos mediante la especificación de las columnas.
- [Crear nuevas tablas](#) para el almacenamiento de datos, copiando la estructura y filas de una tabla existente.

Ejemplo

- Crear una nueva tabla EMP con las columnas: id (number (4)), empleado (varchar2(40)).
- Crear una nueva tabla EMP2 que contenga el employee_id y last_name de la tabla EMPLOYEES.

Capacitación Oracle SQL

26

Sintaxis

Nomenclatura de tablas: de 1 a 30 caracteres, comenzando por letra, conteniendo a-z A-Z 0-9 - \$ y #, no usando palabras reservadas ni nombres de objetos del mismo esquema.

```
CREATE TABLE [esquema.]tabla  
    (columna tipo_de_dato [DEFAULT expresion] [, ...]);
```

Crea una tabla con la estructura especificada.

```
CREATE TABLE [esquema.]tabla  
    [(columna [, columna...])]  
AS subconsulta;
```

Crea una tabla basada en la subconsulta, insertándole las filas devueltas.

```
DESC[RIBE] tabla;
```

Aplicación

➤ **CREATE TABLE emp**
(id number(4), empleado varchar2(40));

➤ **CREATE TABLE emp2**
AS SELECT employee_id, last_name FROM employees;

Objetivo

- Agregar, modificar y borrar columnas de una tabla existente.

Ejemplo

- Agregar en la tabla EMPLOYEES una nueva columna "Ciudad" de tipo varchar2(30).
- Modificar la columna que se acaba de agregar, para que tenga el tipo varchar2(35).
- Eliminar la nueva columna.

Confirmar después de cada caso la estructura de la tabla.

Sintaxis

```
ALTER TABLE tabla
ADD          (columna tipo_de_dato [DEFAULT expresion]
              [, columna tipo_de_dato] ...);
```

Agrega una o más columnas, al final de la tabla.

```
ALTER TABLE tabla
MODIFY       (columna tipo_de_dato [DEFAULT expresion]
              [, columna tipo_de_dato] ...);
```

Modifica tipo de dato, tamaño, o valor por defecto de un o más columnas.

```
ALTER TABLE tabla
DROP         {(columna [, columna...])| COLUMN columna};
```

Elimina una columna (sólo una a la vez) de la tabla.

Aplicación

- ALTER TABLE employees
ADD (ciudad varchar2(30));
- DESC employees;
- ALTER TABLE employees
MODIFY (ciudad varchar2(35));
- DESC employees;
- ALTER TABLE employees
DROP (ciudad);
- DESC employees;

Objetivo

- [Borrar una tabla](#)
 - Eliminar todos los datos y la estructura de una tabla.
- [Renombrar una tabla](#)
 - Cambiar el nombre de una tabla existente.
- [Truncar una tabla](#)
 - Eliminar las filas de una tabla y liberar su espacio ocupado.

Ejemplo

- Truncar la tabla EMP2.
- Renombrar la tabla EMP2 a EMP3.
- Eliminar la tabla EMP3.

Confirmar después de cada caso la estructura, contenido y existencia de la tabla.

Sintaxis

DROP TABLE tabla;

Suprime la tabla, borrando todos sus datos, en forma permanente.

RENAME nombre_viejo TO nombre_nuevo;

Renombra la tabla.

TRUNCATE TABLE tabla;

Elimina todas las filas de la tabla, en forma permanente.

Aplicación

- **TRUNCATE TABLE** emp2;
- **SELECT * FROM** emp2;
- **RENAME** emp2 TO emp3;
- **DESC** emp2;
- **DESC** emp3;
- **DROP TABLE** emp3;
- **DESC** emp3;

Objetivo

- Utilizar vistas para [crear subconjuntos](#) de información basado en tablas, y restringir el acceso a los datos.
- [Crear y modificar vistas.](#)
- [Eliminar vistas.](#)

Ejemplo

- Crear la vista EMP_SAL que contenga códigos de empleado y sus salarios. Asegurarse que sólo sea utilizada para hacer consultas sobre ella.
- Verificar la creación seleccionando datos de la vista.
- Eliminar la vista.

Sintaxis

```
CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW vista
    [(alias [, alias]...)]
AS subconsulta
[WITH CHECK OPTION [CONSTRAINT restriccion]]
[WITH READ ONLY [CONSTRAINT restriccion]];

DROP VIEW vista;
```

Aplicación

- **CREATE OR REPLACE VIEW emp_sal**
AS (SELECT employee_id, salary
FROM employees)
WITH READ ONLY;
- **SELECT * FROM emp_sal;**
- **DROP VIEW emp_sal;**

Objetivo

- [Agregar o eliminar restricciones](#) a los datos de las tablas.
- Diferenciar los *tipos de restricción*:
 - NOT NULL
 - UNIQUE
 - PRIMARY KEY
 - FOREIGN KEY
 - CHECK

Ejemplo

- Crear una nueva tabla EMP con las columnas: id (number (4)), empleado (varchar2(40)). La columna id debe ser la clave primaria, y empleado no debe aceptar nulos, y además es foreign key de la columna employee_id de employees.
- Eliminar de la tabla EMP la clave primaria.

Sintaxis

```
CREATE TABLE [esquema.]tabla
    (columna tipo_de_dato [DEFAULT expresion]
    [restriccion_a_nivel_columna],
    ...
    [restriccion_a_nivel_tabla][,...]);
```

Creación de restricción al crear la tabla.

```
ALTER TABLE [esquema.]tabla
ADD          [CONSTRAINT nombre_restriccion] tipo (columna);
```

Agregado de restricción sobre tabla existente.

```
ALTER TABLE [esquema.]tabla
DROP          PRIMARY KEY | UNIQUE (columna) |
              CONSTRAINT nombre_restriccion [CASCADE];
```

Borrado de restricción existente.

Sintaxis (cont.)

[restriccion_a_nivel_columna]:
columna [CONSTRAINT nombre_restriccion] tipo_restriccion
columna NOT NULL
columna UNIQUE
columna PRIMARY KEY
columna REFERENCES tabla(columna)
columna CHECK (condición)

[restriccion_a_nivel_tabla]:
columna,...
[CONSTRAINT nombre_restriccion] tipo_restriccion (columna,...),
UNIQUE (columna)
PRIMARY KEY (columna [, columna...])
FOREIGN KEY (columna) REFERENCES tabla(columna)
CHECK (condición)

Aplicación

- **CREATE TABLE emp**
(id number(4) PRIMARY KEY,
empleado varchar2(40) NOT NULL,
FOREIGN KEY empleado REFERENCES employees(employee_id));
- **ALTER TABLE emp**
DROP PRIMARY KEY;

Objetivo

- [Activar o desactivar restricciones](#) de integridad.

Sintaxis

```
ALTER TABLE [esquema.]tabla  
ENABLE | DISABLE  
CONSTRAINT nombre_restriccion [CASCADE];
```

8- Gestión de Objetos: Índices

Objetivo

- Comprender el concepto de [índice](#) y cuándo son necesarios.
- [Crear nuevos índices](#) que optimicen el acceso del motor a los datos.
- [Eliminar índices](#) existentes.

Ejemplo

- Crear un índice único sobre las columnas last_name + first_name de la tabla EMPLOYEES.
- Crear un índice único sobre upper(last_name) de la tabla EMPLOYEES.
¿Es redundante este índice con el anterior?
- Reconstruir el primer índice.
- Borrar el segundo índice.

Capacitación Oracle SQL

43

8- Gestión de Objetos: Índices

Consideraciones sobre índices

- Los índices [aceleran las búsquedas](#) sobre todo si la columna contiene un rango amplio de valores.
- Los nulos no se guardan en el índice, por lo tanto, la búsqueda de valores [no nulos](#) donde hay muchos nulos, resulta [eficiente](#) con índices.
- Se deben indexar las columnas teniendo en cuenta la forma de [acceso a los datos](#) desde las consultas, en cuanto a las [combinaciones de columnas](#) y [joins](#) en las condiciones.
- Las operaciones [DML](#) se ven [afectadas](#) cuantos más índices haya, porque deben actualizarlos.
- El índice debería ayudar a recuperar [hasta un 2-4%](#) de la tabla.
- Se deben tener en cuenta al indexar, las [expresiones o funciones](#) que se le apliquen a los datos desde las consultas.
- Clasificaciones de índice:
 - [Unicidad](#): Único o No único.
 - [Composición](#): Simple o Compuesto
 - [Tipo](#): De columna o de Función; de árbol B o Bitmap

Capacitación Oracle SQL

44

Sintaxis

```
CREATE [UNIQUE|BITMAP] INDEX indice  
      ON tabla(columna [, columna ...]);
```

Crea explícitamente el índice como objeto de esquema independiente de la tabla, para acelerar la recuperación de sus filas.

Se crean índices únicos automáticamente al definir restricciones UNIQUE o PRIMARY KEY.

```
CREATE [UNIQUE] INDEX indice  
      ON tabla(expresion);
```

Crea un índice de función para la expresión entre paréntesis.

```
DROP INDEX indice;
```

Elimina el índice. Requiere ser el propietario o tener DROP ANY INDEX.

```
ALTER INDEX indice REBUILD;
```

Reconstruye el árbol del índice.

Aplicación

- ```
CREATE UNIQUE INDEX ln_fn_emp
 ON employees (last_name, first_name);
```
- ```
CREATE UNIQUE INDEX up_ln_emp  
  ON employees(upper(last_name));
```
- ```
ALTER INDEX ln_fm_emp REBUILD;
```
- ```
DROP INDEX up_ln_emp;
```

8- Gestión de Objetos: Secuencias

Objetivo

- Utilizar secuencias para [generar números secuenciales](#) únicos para claves primarias.
- [Crear](#) secuencias.
- [Utilizar](#) secuencias.
- [Eliminar](#) secuencias.

Ejemplo

- Crear la secuencia DEP_SEQ para utilizar valores de clave primaria para DEPARTMENTS. Debe comenzar en 200, incrementarse de a 1, y no ciclar. El tope máximo es 9999.
- Insertar en DEPARTMENTS utilizando la secuencia como identificador.
- Validar el último valor asignado, consultando la secuencia.
- Borrar la secuencia.

Capacitación Oracle SQL

47

8- Gestión de Objetos: Secuencias

Sintaxis

```
CREATE SEQUENCE secuencia
  [INCREMENT BY n]
  [START WITH n]
  [{MAXVALUE n | NOMAXVALUE}]
  [{MINVALUE n | NOMINVALUE}]
  [{CYCLE | NOCYCLE}]
  [{CACHE n | NOCACHE}];
```

```
DROP SEQUENCE secuencia;
```

La pseudocolumna NEXTVAL devuelve el siguiente valor disponible de la secuencia.

La pseudocolumna CURRVAL devuelve el valor actual de la secuencia.

Capacitación Oracle SQL

48

Aplicación

- **CREATE SEQUENCE** dep_seq
INCREMENT BY 1
START WITH 200
MAXVALUE 9999
NOCYCLE;
- **INSERT INTO** departments (dep_seq.nextval, 'Nuevo');
- **SELECT** dep_seq.currval FROM dual;
- **DROP SEQUENCE** dep_seq;

Objetivo

- Crear o eliminar enlaces de base de datos (DBLinks), para establecer una conexión lógica entre dos bases de datos físicas.

Ejemplo

- Crear un DBLink público llamado “base1” para conectarse a la base “base1.world” utilizando el usuario “admin” y password “passadm”.
- Crear un DBLink privado llamado “base2” para conectarse a la base “base2.world” utilizando el usuario conectado.
- Eliminar los DBLinks.

Sintaxis

```
CREATE [PUBLIC] DATABASE LINK dblink  
[CONNECT TO usuario IDENTIFIED BY password]  
USING 'nombre_servicio';
```

Define una comunicación unidireccional desde una BD Oracle a otro servidor de BD.

Si el enlace es privado, sólo el usuario que lo creó puede utilizarlo.

El enlace puede crearse para que lo use el usuario conectado (en la base remota), un usuario fijo o el usuario actual (global).

```
DROP [PUBLIC] DATABASE LINK dblink;
```

Aplicación

- ```
CREATE PUBLIC DATABASE LINK base1
CONNECT TO admin IDENTIFIED BY passadm
USING 'base1.world';
```
- ```
CREATE DATABASE LINK base2  
USING 'base2.world';
```
- ```
DROP PUBLIC DATABASE LINK base1;
```
- ```
DROP DATABASE LINK base2;
```

8- Gestión de Objetos: Sinónimos

Objetivo

- Utilizar sinónimos para [simplificar la referencia](#) a objetos de otros esquemas o de otras bases de datos.
- [Crear sinónimos](#).
- [Eliminar sinónimos](#).

Ejemplo

- Crear el sinónimo EMP_USR_A que apunte a la tabla EMPLOYEES del usuario USERA. Si ya existe un sinónimo con ese nombre, debe reemplazarse.
- Crear el sinónimo público JOBS que apunte a la tabla JOBS del usuario ADMIN en la base remota BASE2.
- Eliminar ambos sinónimos.

Capacitación Oracle SQL

53

8- Gestión de Objetos: Sinónimos

Sintaxis

```
CREATE [OR REPLACE] [PUBLIC] SYNONYM sinonimo  
FOR objeto;
```

```
DROP [PUBLIC] SYNONYM sinonimo;
```

Capacitación Oracle SQL

54

Aplicación

- CREATE OR REPLACE SYNONYM emp_usr_a FOR usera.employees;
- CREATE PUBLIC SYNONYM jobs FOR admin.jobs@base2;
- DROP SYNONYM emp_usr_a;
- DROP PUBLIC SYNONYM jobs;

Objetivo

- Utilizar vistas materializadas (snapshots) para [replicar](#) información en la BD.
- [Modificar](#) vistas materializadas.
- [Eliminar](#) vistas materializadas.

Ejemplo

- Crear el log de vista materializada para una réplica de la tabla employees, en base a los cambios de la clave primaria.
- Crear la vista materializada s_employees, como réplica de la tabla employees, para refresco diario de tipo fast.
- Modificar el refresco de s_employees para que sea a partir de la siguiente hora.
- Eliminar s_employees.

Sintaxis

```
CREATE MATERIALIZED VIEW vista_materializada
[ON PREBUILT TABLE]
{NEVER REFRESH | [REFRESH {FAST|COMPLETE|FORCE}
  [ON {DEMAND|COMMIT}]]}
[START WITH fecha] [NEXT fecha]
AS subconsulta;
```

Para el refresco de tipo FAST, debe existir el correspondiente MATERIALIZED VIEW LOG con cláusula "including new values".

```
ALTER MATERIALIZED VIEW vista_materializada
REFRESH {FAST|COMPLETE|FORCE}
  [ON {DEMAND|COMMIT}]
[START WITH fecha] [NEXT fecha];

CREATE MATERIALIZED VIEW LOG ON tabla
[WITH {OBJECT ID|PRIMARY KEY|ROWID|SEQUENCE}
  (columna [, columna...])]
[ {INCLUDING|EXCLUDING} NEW VALUES];

DROP MATERIALIZED VIEW vista_materializada;
```

Capacitación Oracle SQL

57

Aplicación

- CREATE MATERIALIZED VIEW LOG ON employees dep_seq
WITH PRIMARY KEY
INCLUDING NEW VALUES;
- CREATE MATERIALIZED VIEW s_employees
REFRESH FAST
NEXT sysdate + 1
AS select * from employees;
- ALTER MATERIALIZED VIEW s_employees
REFRESH FAST
NEXT sysdate + 1/24;
- DROP MATERIALIZED VIEW s_employees;

Capacitación Oracle SQL

58

Resumen

- Mediante las sentencias DDL podemos modificar las estructuras contenidas en la Base de Datos.

- **Tablas:** crear nuevas, agregarles, modificarles, o eliminarles columnas, eliminar existentes, renombrarlas, truncarlas.
- **Vistas:** crear nuevas o modificarlas, eliminarlas.
- **Restricciones:** crear nuevas en la creación de tablas, agregar en tablas existentes, borrarlas, activarlas o desactivarlas.
- **Índices:** crear nuevos, borrarlos.
- **Secuencias:** crear nuevas, utilizarlas para la obtención de números secuenciales, eliminarlas.
- **DBLinks:** crear nuevos, borrarlos.
- **Sinónimos:** crear nuevos, borrarlos.
- **Vistas materializadas:** crear nuevas, modificar, borrarlas.

Capacitación ORACLE. Lenguaje SQL y Administración Básica

EVALUACIÓN MÓDULO 3

