CREACIÓN Y USO DE VISTAS

7.5 Creación y uso de vistas

A veces, para obtener datos de varias tablas hemos de construir una sentencia SELECT compleja y, si en otro momento necesitamos realizar esa misma consulta, tenemos que construir de nuevo la sentencia SELECT. Sería muy cómodo obtener los datos de una consulta compleja con una simple sentencia SELECT.

Pues bien, las vistas solucionan este problema: mediante una consulta simple de una vista cabe la posibilidad de obtener datos de una consulta compleja. Una **vista** es una tabla lógica que permite acceder a la información de una o de varias tablas. No contiene información por sí misma, sino que su información está basada en la que contienen otras tablas, llamadas *tablas base*, y siempre refleja los datos de estas tablas; es, simplemente, una sentencia SQL.

Si se suprime una tabla, la vista asociada se invalida. Las vistas tienen la misma estructura que una tabla: filas y columnas, y se tratan de forma semejante a una tabla. El formato para crear una vista es:

```
CREATE [OR REPLACE] VIEW Nombrevista [(columna [,columna])]
AS consulta
[WITH {CHECK OPTION | READ ONLY} CONSTRAINT nombrerestric-
                                                    ción];
```

Nombrevista es el nombre que damos a la vista.

[(*columna* [,*columna*])] son los nombres de columnas que va a contener la vista. Si no se ponen, se asumen los nombres de columna devueltos por la consulta.

AS *consulta* determina las columnas y las tablas que aparecerán en la vista.

[OR REPLACE] crea de nuevo la vista si ya existía.

[**WITH CHECK OPTION**] es la opción de comprobación para una vista. Si se especifica, SQL comprueba automáticamente cada operación INSERT y UPDATE sobre la vista para asegurarse que las filas resultantes satisfagan el criterio de búsqueda de la definición de la vista.

Si la fila insertada o modificada no satisface la condición de creación de la vista, la sentencia INSERT o UPDATE falla y no se realiza la operación.

[**WITH READ ONLY**] especifica que sólo se puede hacer SELECT de las filas de la vista.

[**CONSTRAINT** *nombrerestriccion*] especifica el nombre de la restricción WITH CHECK OPTION o WITH READ ONLY. Es opcional.

A. Creación y uso de vistas sencillas

Las vistas más sencillas son las que acceden a una única tabla. Por ejemplo, creamos la vista DEP30 que contiene el APELLIDO, el OFICIO y el SALARIO de los empleados de la tabla EMPLE del departamento 30:

```
CREATE VIEW DEP30
AS SELECT APELLIDO, OFICIO, SALARIO  FROM EMPLE
   WHERE DEPT_NO=30;
```

Ahora la vista creada se puede usar como si de una tabla se tratase. Se puede consultar, se pueden borrar filas, actualizar filas siempre y cuando las columnas a actualizar no sean expresiones (funciones de grupo o referencias a pseudocolumnas); y se puede insertar siempre y cuando todas las columnas obligatorias de la tabla asociada estén presentes en la vista.

También podríamos haberla creado dando nombre a las columnas, por ejemplo, APE, OFI y SAL:

```
CREATE OR REPLACE VIEW DEP30 (APE, OFI, SAL)
AS SELECT APELLIDO, OFICIO, SALARIO FROM EMPLE
   WHERE DEPT_NO = 30;
```

Para consultar las vistas creadas se dispone de la vista **USER_VIEWS** y **ALL_VIEWS**. Así, para visualizar los nombres de vistas con sus textos, tenemos:

```
SQL> SELECT VIEW_NAME, TEXT FROM USER_VIEWS;
```

Actividades propuestas

8 **Crea una vista que contenga los datos de los empleados del departamento 10 con salario > 1200. Después realiza operaciones INSERT, UPDATE y DELETE sobre la vista.**

B. Creación y uso de vistas con WITH CHECK OPTION y READ ONLY

Se puede crear una vista de forma que todas las operaciones INSERT, UPDATE y DELETE que se hagan sobre ella satisfagan la condición por la que se creó. Por ejemplo, creo una vista que contiene todos los datos de los empleados del departamento 20:

```
CREATE OR REPLACE  VIEW DEP20
AS SELECT * FROM EMPLE WHERE DEPT_NO = 20;
```

Ahora inserto en la vista un empleado del departamento 30:

```
INSERT INTO DEP20 VALUES (3333, 'PEREZ', 'EMPLEADO',7902,
SYSDATE, 1300, NULL, 30);
```

La inserción se realizará correctamente.

Ahora creamos la vista con WITH CHECK OPTION:

```
CREATE OR REPLACE VIEW DEP20
AS SELECT * FROM EMPLE WHERE DEPT_NO = 20 WITH CHECK
OPTION;
```

Intentamos insertar una fila en la vista con el departamento 30; en este caso se producirá un error:

```
ERROR en línea 1:
ORA-01402: violación de la cláusula WHERE en la vista
WITH CHECK OPTION
```

La opción **WITH READ ONLY** sólo nos permitirá hacer SELECT en la vista:

```
CREATE OR REPLACE VIEW DEP30
AS SELECT * FROM EMPLE WHERE DEPT_NO=30 WITH READ ONLY;
```

Cualquier operación INSERT, UPDATE o DELETE sobre la vista DEP30 fallará.

Actividades propuestas

9 **Prueba a realizar operaciones INSERT, UPDATE o DELETE sobre la vista DEP30.**

C. Creación y uso de vistas complejas

Las vistas complejas contienen consultas que se definen sobre más de una tabla, agrupan filas usando las cláusulas GROUP BY o DISTINCT, y contienen llamadas a funciones. Se pueden crear vistas usando funciones, expresiones en columnas y consultas avanzadas, pero únicamente se podrán consultar estas vistas.

Caso práctico

17 A partir de las tablas EMPLE y DEPART creamos una vista que contenga las columnas EMP_NO, APELLIDO, DEPT_NO y DNOMBRE :

```
CREATE VIEW EMP_DEPT (EMP_NO, APELLIDO, DEPT_NO, DNOMBRE) AS
   SELECT EMP_NO,APELLIDO, EMPLE.DEPT_NO, DNOMBRE
   FROM EMPLE, DEPART WHERE EMPLE.DEPT_NO = DEPART.DEPT_NO;
```

Insertamos una fila en la vista: `INSERT INTO EMP_DEPT VALUES(2222,'SUELA', 20, 'INVESTIGACION');`
Pero se produce un error debido a que la vista se creó a partir de dos tablas.

```
ERROR en línea 1:
ORA-01776: no se puede modificar más de una tabla base a través de una vista de unión
```

Los borrados y las modificaciones también producirán errores.

Creamos una vista llamada PAGOS a partir de las filas de la tabla EMPLE, cuyo departamento sea el 10. Las columnas de la vista se llamarán NOMBRE, SAL_MES, SAL_AN y DEPT_NO. El NOMBRE es la columna APELLIDO, a la que aplicamos la función INITCAP( ). SAL_MES es el SALARIO. SAL_AN es el SALARIO*12:

```
CREATE VIEW PAGOS (NOMBRE, SAL_MES, SAL_AN, DEPT_NO)
AS SELECT INITCAP(APELLIDO), SALARIO, SALARIO*12, DEPT_NO FROM EMPLE WHERE DEPT_NO = 10;
```

Podemos modificar filas siempre y cuando la columna que se va a modificar no sea la columna expresada en forma de cálculo (SAL_AN) o la que fue creada mediante la función INITCAP(): `UPDATE PAGOS SET SAL_MES = 5000 WHERE NOMBRE = 'Muñoz';`

Actividades propuestas

10 Crea la vista VMEDIA a partir de las tablas EMPLE y DEPART. Esta vista contendrá por cada departamento el número de departamento, el nombre, la media de salario y el máximo salario. Prueba hacer inserciones modificaciones y borrados en la vista.

D. Borrado de vistas

Es posible borrar las vistas con la orden DROP VIEW, cuyo formato es:

```
DROP VIEW nombrevista;
```

Por ejemplo, borramos la vista PAGOS: `DROP VIEW PAGOS;`

7.6 Creación de sinónimos

Cuando tenemos acceso a las tablas de otro usuario y deseamos consultarlas es preciso teclear el nombre del usuario propietario antes del nombre de la tabla. Es decir, si DIEGO tiene acceso a la tabla DEPART de PEDRO y la quiere consultar, tendrá que teclear la siguiente orden para poder hacerlo: `SELECT * FROM PEDRO.DEPART;`

Mediante el uso de sinónimos, DIEGO puede crear un sinónimo para referirse a la tabla de PEDRO sin necesidad de incluir su nombre: `SELECT * FROM TABLAPEDRO;`

Un **sinónimo** es un nuevo nombre que se puede dar a una tabla o vista. Con los sinónimos se pueden utilizar dos nombres diferentes para referirse a un mismo objeto. Resultan interesantes cuando se tiene acceso a tablas de otros usuarios; se pueden crear sinónimos para hacer referencia a esas tablas y, así, no hay que escribir el nombre de usuario propietario de la tabla delante de la tabla a la que tenemos acceso cada vez que deseemos consultarla.

El formato para crear sinónimos es el siguiente:

```
CREATE [PUBLIC] SYNONYM nombresinónimo FOR [usuario.]Nombretabla;
```

`PUBLIC` hace que el sinónimo esté disponible para todos los usuarios.

Caso práctico

18 **Creamos el sinónimo DEPARTAMENTOS asociado a la tabla DEPART:** `CREATE SYNONYM DEPARTAMENTOS FOR DEPART;` Ahora podemos acceder a la tabla DEPART mediante su nombre o usando el sinónimo:

```
SELECT * FROM DEPARTAMENTOS;        SELECT * FROM DEPART;
```

DIEGO puede acceder a la tabla DEPART de PEDRO y crea un sinónimo llamado DEPART:

```
CREATE SYNONYM DEPART FOR PEDRO.DEPART;
```

Diego puede utilizar el sinónimo DEPART para consultar la tabla DEPART de PEDRO: `SELECT * FROM DEPART;`

Como DIEGO ha nombrado al sinónimo igual que el nombre que tiene la tabla de PEDRO (DEPART), podrá hacer uso de las aplicaciones que PEDRO ha desarrollado sobre esa tabla.

Por otra parte, existen sinónimos públicos a los que puede hacer referencia cualquier usuario. Sólo el administrador de la base de datos (DBA) y los usuarios con privilegio CREATE PUBLIC SYNONYM pueden crear este tipo de sinónimos. Por ejemplo, un usuario que es DBA crea un sinónimo público para su tabla DEPART; llama DEP al sinónimo:

```
CREATE PUBLIC SYNONYM DEP FOR DEPART;
```

Para que todos los usuarios puedan usar la tabla DEPART y su sinónimo han de tener permiso. Se da permiso a todos los usuarios para hacer SELECT en la tabla DEPART con la orden GRANT: **GRANT SELECT** ON DEPART TO **PUBLIC**; ahora todos los usuarios pueden hacer SELECT del sinónimo público creado para la tabla DEPART: SELECT * FROM **DEP**;

A. Borrado de sinónimos

Del mismo modo que se crean sinónimos, se pueden borrar, con la orden DROP SYNONYM, cuyo formato es:

DROP [PUBLIC] **SYNONYM** [*usuario.*]*sinónimo*;

sinónimo es el nombre de sinónimo que se va a suprimir. Únicamente los DBA y los usuarios con el privilegio DROP PUBLIC SYNONYM pueden suprimir sinónimos PUBLIC. Igualmente, sólo los DBA y los usuarios con el privilegio DROP ANY SYNONYM pueden borrar los sinónimos de otros usuarios. Por ejemplo, borramos el sinónimo DEPARTAMENTOS:

DROP SYNONYM DEPARTAMENTOS;

Borramos el sinónimo público DEP: **DROP PUBLIC SYNONYM** DEP;

Por otro lado, la vista USER_SYNONYMS permite ver los sinónimos que son propiedad del usuario. Para ver los sinónimos creados por el usuario sobre sus objetos: SELECT SYNONYM_NAME, TABLE_NAME FROM **USER_SYNONYMS**;

7.7 Cambios de nombre

RENAME es una orden SQL que cambia el nombre de una tabla, vista o sinónimo. El nuevo nombre no puede ser una palabra reservada ni el nombre de un objeto que tenga creado el usuario. El formato es éste:

RENAME *nombreanterior* **TO** *nombrenuevo*;

Las restricciones de integridad, los índices y los permisos dados al objeto se transfieren automáticamente al nuevo objeto. Oracle invalida todos los objetos que dependen del objeto renombrado, como las vistas, los sinónimos y los procedimientos almacenados que hacen referencia a la tabla renombrada. No se puede usar esta orden para renombrar sinónimos públicos ni para renombrar columnas de una tabla. Las columnas de una tabla se renombran mediante la orden CREATE TABLE AS.

Conceptos básicos

A continuación se muestra un resumen sobre la orden CREATE TABLE. El formato más básico es el siguiente:

```
CREATE  TABLE  Nombretabla
(
  Columna1   Tipo_dato  [NOT NULL],
  Columna2   Tipo_dato  [NOT NULL],
  ...
) [TABLESPACE espacio_de_tabla];
```

Restricciones en la orden CREATE TABLE:

- Restricción de un solo campo:

```
CONSTRAINT nombrerestricción (
    [NOT] NULL | (PRIMARY KEY | UNIQUE) |
    REFERENCES Nombretabla [(columna[, columna…])]  [ON DELETE CASCADE] |
    CHECK (condición)
)
```

- Restricción de múltiples campos:

```
CONSTRAINT nombrerestricción (
     PRIMARY KEY (columna[, columna …]) |
     UNIQUE (columna[, columna …]) |
     FOREIGN KEY (columna[, columna …])
     REFERENCES Nombretabla [(columna[, columna …])] [ON DELETE CASCADE] |
     CHECK (condición)
}
```

VISTAS DEL DICCIONARIO DE DATOS:

Información de tablas y otros objetos	USER_TABLES, USER_OBJECTS, USER_CATALOG
Información de restricciones	USER_CONSTRAINTS, ALL_CONSTRAINTS, DBA_CONSTRAINTS USER_CONS_COLUMNS, ALL_CONS_COLUMNS, DBA_CONS_COLUMNS
Información sobre vistas	USER_VIEWS, ALL_VIEWS
Información sobre sinónimos	USER_SYNONYMS, ALL_SYNONYMS