VISTAS

BASE DE DATOS II – CISC - UG

Vistas

Una vista no es más que una consulta almacenada a fin de utilizarla tantas veces como se desee.

Una vista no contiene datos en sí misma; es como una ventana a través de la cual se pueden ver o cambiar los datos de las tablas.

Las tablas sobre las cuales se basa una vista se llaman tablas base.

Una vista suele llamarse también tabla virtual porque los resultados que retorna y la manera de referenciarlas es la misma que para una tabla.

Se almacenan en el Diccionario de Datos, USER_VIEWS.

Vistas - Aplicaciones

Realizar consultas complejas más fácilmente, ya que permiten dividir la consulta en varias subconsultas (cada una de las cuales es más sencilla que la original)

Proporcionar tablas con datos completos, resultados formatear o realizar cálculos sobre los datos originales

Proporcionar formas personalizadas y más entendibles de los datos

Ocultar el almacenamiento intrínseco de la base de datos y conseguir una mayor independencia de los datos respecto al resto de elementos de la base de datos.

Restringir el acceso a los datos originales

Ser utilizadas como cursores de datos en los lenguajes procedimentales (como PL/SQL)

Vistas - Tipos

Simples.

- Formada por una sola tabla y no contienen funciones de agrupación.
- Ventaja: Permiten siempre realizar operaciones DML sobre ellas.
- Características:
  - Sólo pueden referirse a una tabla de la base de datos.
  - No pueden contener funciones.
  - No pueden contener ninguna cláusula de agrupación
  - Admiten realizar sobre ellas operaciones DML.

Complejas.

- Obtienen datos de varias tablas, pueden utilizar funciones de agrupación y de cualquier otro tipo. No siempre permiten operaciones DML.

Vistas – Sintaxis creación y eliminación

```
CREATE [OR REPLACE] [FORCE | NOFORCE]
VIEW nombre_vista  AS subconsulta
[WITH CHECK OPTION [CONSTRAINT nombre_constraint]]
[WITH READ ONLY];
```

- **FORCE**: Crea la vista sin importar que la tabla base exista o no.
- **WITH CHECK OPTION**: Especifica que solamente las filas accesibles a la vista pueden ser insertadas o actualizadas.
- **CONSTRAINT**: Nombre asignado a la restricción CHECK OPTION.
- **WITH READ ONLY**: Asegura que ninguna operación DML pueda realizarse sobre esta vista.

```
DROP VIEW [schema.] tabla;
```

- Al borrar una vista no perderá los datos, porque la vista está basada en tablas subyacentes de la B.D.
- Únicamente el creador o un usuario con el privilegio DROP ANY VIEW puede eliminar una vista.

Vistas – Ejemplos de creación

Creación de la vista "vListaProveedores" a partir de una consulta a la tabla "suppliers", la consulta recupera todas las columnas de la tabla y el resultado aparece ordenado alfabéticamente de acuerdo con el contenido de la columna "companyName"

```
create or replace view vListaProveedores as
select supplierId, companyname, contactname, cityId, phone
from suppliers
order by companyName;
```

Vistas – Ejemplos de creación

Creación de la vista "vListaProveedores y alias para cada una de las columnas" a partir de una consulta a la tabla "suppliers", la consulta recupera todas las columnas de la tabla y el resultado aparece ordenado alfabéticamente de acuerdo con el contenido de la columna "companyName"

```
create or replace view vListaProveedores
(proveedorId, nombreCompania, nombreContacto, ciudadId, telefono) as
select supplierId, companyname, contactname, cityId, phone
from suppliers
order by companyName;
```

Vistas – Ejemplos de creación

Creación de la vista "**vProductosOrdenados**" a partir de las tablas "products, orders, ordersdetail", la consulta recupera la cantidad de productos ordenados y su valor monetario.

```
create or replace view vProductosOrdenados as
select pro.productname, count(det.quantity) CantidadOrdenada,
      sum(det.quantity * det.unitprice) Total
from products pro
inner join ordersdetail det on pro.productid = det.productid
inner join orders ord on det.orderid = ord.orderid
group by pro.productname
order by CantidadOrdenada asc;
```

Vistas – Ejemplos de eliminación

Eliminación de la vista "vListaProveedores"

```
Drop view vListaProveedores;
```

SECUENCIAS

BASE DE DATOS II
CISC - UG

Secuencias

Una secuencia se emplea para generar valores enteros secuenciales únicos y asignárselos a campos numéricos.

Se utilizan generalmente para las claves primarias de las tablas garantizando que sus valores no se repitan.

Secuencias Sintaxis Creación

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

- La cláusula "start with" indica el valor desde el cual comenzará la generación de números secuenciales.
  - Si no se especifica, se inicia con el valor que indique "minvalue".
- La cláusula "increment by" especifica el incremento; debe ser un valor numérico entero positivo o negativo diferente de 0.
  - Si no se indica, por defecto es 1.
- "maxvalue" define el valor máximo para la secuencia.
  - Si se omite, por defecto es 9999999999999999999999.
- "minvalue" establece el valor mínimo de la secuencia.
  - Si se omite será 1.
- La cláusula "cycle" indica que, cuando la secuencia llegue a máximo valor (valor de "maxvalue") se reinicie, comenzando con el mínimo valor ("minvalue") nuevamente.
  - Si se omite, por defecto la secuencia se crea "nocycle".

Si no se especifica ninguna cláusula, excepto el nombre de la secuencia, por defecto, comenzará en 1, se incrementará en 1, el mínimo valor será 1, el máximo será 9999999999999999999999999 y "nocycle".

Secuencias Sintaxis modificación y eliminación

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

```
DROP SEQUENCE secuencia;
```

Secuencias Ejemplos

```
create sequence sCodigoLibros
start with 1 increment by 1
maxvalue 99999 minvalue 1;
```

```
drop sequence sCodigoLibros;
```

Secuencias Ejemplos

```
create sequence sNumeroSocios
increment by 5
cycle;
```

```
Alter sequence sNumeroSocios
increment by 5 minvalue 1
maxvalue 30 cycle;
```

```
drop sequence sNumeroSocios;
```

Secuencias – Pseudocolumnas

- Después de crear una secuencia, se puede acceder a sus valores con sentencias SQL invocando el nombre de las siguientes pseudocolumnas:
  - CURRVAL, devuelve el valor actual de la secuencia
  - NEXTVAL, incrementa la secuencia y devuelve el nuevo valor.

  - **select sCodigoLibros.nextval from dual;**
  - **select sCodigoLibros.currval from dual;**
  - **select sCodigoLibros.currval from dual;**

  - **insert into libros values (sCodigoLibros.nextval, 'El aleph', 'Borges','Emece');**

Secuencias – Consideraciones

La primera vez que una consulta llama a una secuencia, se devuelve un valor predeterminado.

Cuando se genera un número de secuencia, la secuencia se incrementa, independientemente de la transacción confirmada o retrotraída.

Si dos usuarios incrementan simultáneamente la misma secuencia, entonces los números de secuencia que cada usuario adquiere pueden tener huecos, ya que el otro usuario está generando números de secuencia.

ÍNDICES

BASE DE DATOS II – CISC - UG

Índices

Los índices son objetos asociados a columnas de tablas que sirven para acelerar las operaciones de consulta y ordenación.

Se almacenan aparte de la tabla a la que hace referencia, lo que permite crearles y borrarles de forma independiente respecto a la tabla.

Cada vez que se añade una nueva fila, los índices involucrados se actualizan a fin de que su información esté al día. De ahí que cuantos más índices haya, más le cuesta a Oracle añadir nuevos datos.

Indices

Se aconseja crear índices en campos que:

- Contengan una gran cantidad de valores
- Sean parte habitual de cláusulas **WHERE**, **GROUP BY** u **ORDER BY**
- Sean parte de listados de consultas de grandes tablas sobre las que casi siempre se muestran como mucho un 4% de su contenido.

No se aconseja en campos que:

- Pertenezcan a tablas pequeñas
- No se usen a menudo en las consultas
- Pertenecen a tablas cuyas consultas muestran, habitualmente, menos de un 4% del total de registros
- Pertenecen a tablas que se actualizan frecuentemente
- Se utilizan en expresiones

Índices – Sintaxis Creación - Eliminación

```
CREATE INDEX nombre
ON tabla (columna1 [,columna2...])
```

```
DROP  INDEX  NOMBRE_INDICE;
```

Índices – Ejemplos

```
CREATE INDEX iClientes_NombreCompleto
ON Clientes (apellidoPat, apellidoMat, nombre);
```

```
DROP  INDEX iClientes_NombreCompleto;
```

SINÓNIMOS

BASE DE DATOS II CISC - UG

Sinónimos

Un sinónimo es un nombre alternativo para objetos como: tablas, vistas, secuencias, procedimientos almacenados y otros objetos de base de datos.

Por lo general, se usan sinónimos cuando se desea otorgar acceso a un objeto desde otro esquema y no se quiere que los usuarios tengan que preocuparse por saber qué esquema es el propietario del objeto.

Recuerde:

- Se deben otorgar privilegios apropiados a un usuario antes de que el usuario pueda usar el sinónimo.
- Ud. puede referirse a sinónimos en las siguientes declaraciones DML: select, insert, update, delete, explain plan.

Sinónimos – creación

CREATE [OR REPLACE] [PUBLIC] SYNONYM [schema .] synonym_name
FOR [schema .] object_name [@ dblink];

OR REPLACE. - Allows you to recreate the synonym (if it already exists) without having to issue a DROP synonym command.

PUBLIC.- It means that the synonym is a public synonym and is accessible to all users. Remember though that the user must first have the appropriate privileges to the object to use the synonym.

- To create a private synonym in your own schema, you must have the CREATE SYNONYM system privilege.
- To create a private synonym in another user's schema, you must have the CREATE ANY SYNONYM system privilege.
- To create a PUBLIC synonym, you must have the CREATE PUBLIC SYNONYM system privilege.

Schema.- The appropriate schema. If this phrase is omitted, Oracle assumes that you are referring to your own schema.

object_name.- The name of the object for which you are creating the synonym. It can be one of the following:

- Table, view, sequence, stored procedure, function, package, java class schema object, user-defined object

Sinónimos – eliminación

DROP [PUBLIC] SYNONYM [schema .] synonym_name [force];

PUBLIC.- Allows you to drop a public synonym. If you have specified PUBLIC, then you don't specify a schema.

Force.- It will force Oracle to drop the synonym even if it has dependencies. It is probably not a good idea to use force as it can cause invalidation of Oracle objects.

Sinónimos - Ejemplos

```
CREATE PUBLIC SYNONYM suppliers
FOR app.suppliers;
```

```
CREATE SYNONYM offices
    FOR hr.locations;
```

```
CREATE PUBLIC SYNONYM customers FOR oe.customers;
```

```
SELECT COUNT(*) FROM customers;
```

```
DROP PUBLIC SYNONYM suppliers;
```