Universidad de los Andes
Departamento de Ingeniería de Sistemas y Computación
isis2304 – Sistemas Transaccionales
Junio de 2010

Guía básica de comandos SQL

A continuación se presenta una descripción breve de algunos de los comando ofrecidos por ORACLE en su herramienta SQL, para la manipulación y definición de objetos en la base de datos.

Mayor información en:

Manuales de ORACLE para SQL (comandos, funciones, etc.)

Oracle10g SQL Reference PDF

Comandos para la definición de objetos (DDL):

Como su nombre lo índica, estos comandos, permiten definir objetos esto significa: crear, borrar o alterar los ya existentes.

Creación de una tabla:

```
CREATE TABLE nom_tabla (nom_col1 tipo_dato1 [NOT NULL],nom_col2
tipo_dato2, … );
```

Borrar una tabla:

```
DROP TABLE nom_tabla [cascade constraint];
```

Creación de una Vista:

Son otros elementos importantes y muy útiles para realizar consultas, la sintaxis básica para crear una vista es:

```
CREATE VIEW nom_vista AS SELECT nom_col1 …FROM nom_tabla1 …;
```

Creación de una secuencia:

```
CREATE SEQUENCE nom_secuencia [INCREMENT BY integer] [START WITH
integer] [MAXVALUE integer | NOMAXVALUE] [MINVALUE integer |
NOMINVALUE] [CYCLE | NOCYCLE] [CACHE integer | NOCACHE] [ORDER |
NOORDER]
```

Creación de un *Constraint*:

Los *constraints* puede ser creados en el momento en el que se crea la tabla (Caso a.) o después de su creación. Estos pueden ser:

- Llaves Primarias
- Llaves Foráneas
- Condiciones de chequeo
- Evitar registros duplicados

Llave Primaria:

La columna 1 corresponde a la llave primaria, este método sólo se utiliza cuando la llave primaria está compuesta por una sola columna.

```
CREATE TABLE nom_tabla (nom_col1 tipo_dato1 primary key,nom_col2
tipo_dato2 , … );
```

La columna 1 corresponde a la llave primaria y se le desea dar un nombre al *constraint.*

```
CREATE TABLE nom_tabla (nom_col1 tipo_dato1 CONSTRAINT nom_primary_key
PRIMARY KEY,nom_col2 tipo_dato2, … );
```

La llave primaria está compuesta por dos columnas -columna 1 y 2.

```
CREATE TABLE nom_tabla (nom_col1 tipo_dato1, nom_col2 tipo_dato2 , …
CONSTRAINT nom_primary_key PRIMARY KEY (nom_col1, nom_col2));
```

La llave primaria está compuesta por dos columnas -columna 1 y 2, y se va a crear posterior a la creación de la tabla.

```
ALTER TABLE nom_tabla ADD CONSTRAINT nom_primary_key PRIMARY KEY (
nom_col1, nom_col2);
```

Para habilitar un *Constraint* ya creado, usted puede dar el siguiente comando:

```
ALTER TABLE nom_tabla ENABLE CONSTRAINT nom_constraint ;
```

Para deshabilitar ya creado, usted puede dar el siguiente comando:

```
ALTER TABLE nom_tabla DISABLE CONSTRAINT nom_constraint ;
```

Llave Foránea:

La columna 1 corresponde a una llave foránea.

```
CREATE TABLE nom_tabla (nom_col1 tipo_dato1 REFERENCES
nom_tabla_referencia,nom_col2 tipo_dato2, … );
```

Las columnas 1 y 2 forman una llave foránea y se le quiere asociar un nombre al *constraint.*

```
CREATE TABLE nom_tabla (nom_col1 tipo_dato1,nom_col2 tipo_dato2,
…,CONSTRAINT nom_llave_foránea FOREIGN KEY (nom_col1, nom_col2)
REFERENCES nom_tabla_referencia (col1_tabla_referencia,
col2_tabla_referencia);
```

La llave foránea está compuesta por dos columnas -columna 1 y 2-, y se va a crear posterior a la creación de la tabla.

```
ALTER TABLE nom_tabla ADD CONSTRAINT nom_llave_foránea FOREIGN KEY
(nom_col1, nom_col2) REFERENCES nom_tabla_referencia
(col1_tabla_referencia, col2_tabla_referencia);
```

***Constraints* de Chequeo:**

La columnas 1 debe ser mayor que cero.

```
CREATE TABLE nom_tabla ( nom_col1 tipo_dato1 CHECK
(nom_col1>0),nom_col2 tipo_dato2 , … );
```

Valores no duplicados:

```
CREATE TABLE nom_tabla (nom_col1 tipo_dato1 UNIQUE,nom_col2
tipo_dato2, … );
```

Adición de una columna:

Cuando se desea adicionar una columna a una tabla ya existente: ALTER TABLE nom_tabla ADD (nom_col1 tipo_dato1, nom_col2 tipo_dato2, …);

Cambiar el tipo de dato de una columna en una tabla ya existente

```
ALTER TABLE nom_tabla MODIFY (nom_col1 nuevo_tipo_dato1, …);
```

Comandos para la manipulación de objetos (DML):

Como su nombre lo índica, este es un lenguaje que permite manipular los objetos, esto significa: consultar, insertar, borrar y actualizar.

Consultar información:

```
SELECT nombre_columnas FROM nombre_tablas [WHERE condición | GROUP BY
nombre_columnas_grupo HAVING condición] ORDER BY
nombre_columnas_que_definen_el_orden
```

Consultar el valor de una secuencia

Su siguiente valor “nextval” y su valor actual luego de utilizar nextval “currval”

```
SELECT nombre_secuencia.nextval FROM dual;
```

Consultar información en forma recursiva:

```
SELECT LPAD('',2*(level-1)) || nom_col1, nom_col2 FROM nombre_tabla
[WHERE condición ] [START WITH condicion] CONNECT BY condición
```

NOTA:

- Recuerde que si está haciendo una consulta recursiva, no puede tener un *join*, ni hacer la consulta sobre una vista que sea el resultado de un *join*.
- LEVEL: Es una columna que indica el nivel de recursión en el cual se encuentra un registro dado.

Insertar información:

```
INSERT INTO nombre_tabla [(nombre_columnas)] VALUES (val_col1,
val_col2, …);
```

Borrar información:

```
DELETE nombre_tabla [WHERE condición];
```

Actualizar información:

```
UPDATE nombre_tabla SET col1 = nuevo_valor1, col2 = nuevo_valor2 WHERE
condicion]
```