RESUMEN CUADERNO DE PRÁCTICAS SQL

Tipos de datos

TIPOS	Descripción
INT/ INTEGER / NUMERIC	Valor entero
NUMERIC(n,m)	Números de hasta 18 dígitos. N: total dígitos admitidos M: n.º de decimales
DECIMAL(n,m)	Igual que NUMERIC(n,m)
REAL / FLOAT	Número de coma flotante
CHAR(n)	Almacena de 1 a 255 caracteres. Fijo
VARCHAR(n)	Datos de cadena de tamaño variable
VARCHAR2(n)	0
DATE	Almacena fechas con día, mes y año
DATETIME	Almacena fechas con fecha y hora
BIT	Se aplica a la lógica booleana
BOOLEAN	False si es 0 True si es distinto de 0
LONG	Cadena de caracteres de longitud variable de hasta 2 gigabytes

Creación de tablas

Para crear una tabla o relación se utiliza el comando:

```
CREATE TABLE nombre_tabla(
        columna1  tipo_datos1
        columna2  tipo_datos2
        …
);
```

Donde pone columna se refiere a cada atributo de la tabla. Ejemplo (hay cosas que por ahora no entenderemos):

```
CREATE TABLE proveedor(
        codpro VARCHAR2(3) CONSTRAINT cod_pro NOT NULL CONSTRAINT cod_prim
PRIMARY KEY,
        nompro VARCHAR2(30) CONSTRAINT nompro_no_nulo NOT NULL,
```

```
        status NUMBER CONSTRAINT status_entre_1_y_10 CHECK(status>=1 and status<=10),
        ciudad VARCHAR2(15)
);
```

Otro ejemplo:

```
CREATE TABLE pieza(
        codpie  VARCHAR2(3) CONSTRAINT codpie_clave_primaria PRIMARE KEY,
        nompie  VARCHAR2(10) CONSTRAINT nompie_no_nulo NOT NULL,
        color  VARCHAR2(10),
        peso NUMBER(5,2) CONSTRAINT peso_entre_0_y_100 CHECK(peso>0 and peso
<=100),
        ciudad  VARCHAR2(15)

);
```

Básicamente al crear las tablas estamos dándole un nombre a cada columna, y especificando el tipo de dato que se puede almacenar en esas columnas junto con las restricciones que tienen que tener estos atributos, como por ejemplo que sea clave primaria (PRIMARE KEY), que sea no nulo (NOT NULL)… Cabe destacar que CONSTRAINT nos permite “ponerle” un nombre a la restricción que le estamos dando a ese atributo, de ahí que después de CONSTRAINT pongamos un nombre que permita describir claramente qué restricción le estamos dando a nuestra columna.

Eliminación de tablas

```
DROP TABLE nombre_tabla;
```

Restricciones o constraints

- NOT NULL: En la columan siempre tiene que haber un valor.

- UNIQUE: La columa tiene un valor diferente en cada fila.

- PRIMARY KEY: Clave primaria. La columna debe ser UNIQUE y NOT NULL.

- FOREIGN KEY: Columna que contiene la clave primaria de otra tabla. Esta columna solo puede tener valores que existan como clave primaria en otra tabla.

-REFERENCES: Clave externa. REFERENCES nombre_tabla(columna)

-CHECK: Limita el rango de valores que puede tener una columna. Se peuden definir varias restricciones CHECK en una tabla.

-DEFAULT: Valor por defecto para la columna. Si no se especifica nada al insertar una fila se le asigna ese valor a la columna afectada por DEFAULT.

Modificación de tablas

La sentencia básica para iniciar cualquier modificación es : ALTER TABLE nombre_tabla modificador. Dependiendo del modificador que añadamos después podremos realizar una modificación u otra.

- Para añadir un nuevo atributo o columna a la tabla:

ALTER TABLE nombre_tabla ADD nombre_columna tipo_dato

-Para borrar un atributo o columna de la tabla:
ALTER TABLE nombre_tabla DROP COLUMN nombre_columna

-Para modificar el contenido de una columna o atributo (Cambiar el tipo de dato…)
ALTER TABLE nombre_tabla ALTER COLUMN nombre_columna tipo_dato

Inserción de tuplas en las tablas

Nos permite introducir tuplas en las tablas
INSERT INTO nombre_tabla VALUES (valor1, valor2, valor3);

INSERT INTO nombre_tabla (columna1, columna2…) VALUES (valor1, valor2…) Esta permite darle el valor solo a columnas específicas y el número de columnas debe coincidir con el número de valores proporcionados.

Eliminar datos de una tabla

Elimina los datos de una tabla. Podemos combinar con WHERE para eliminar una serie de filas, o si lo omitimos elimina todas las filas de la tabla:
DELETE nombre_tabla WHERE condicion/es;

Mostrar el contenido de una tabla

Select from

Selecciona columnas específicas de una tabla

SELECT columna1, columna2… FROM nombre_tabla;
SELECT * FROM nombre_tabla nos permite mostrar toda la tabla

SELECT table_name FROM user_tables; nos permite mostrar todas las tablas creadas hasta el momento.

Describe

describe nombre_tabla Permite mostrar todo el contenido de la tabla

Where

Especifica una condición al seleccionar unos datos. Si se cumple esta condición se devuelven los valores que la cumplen.

SELECT columna1, columna2… FROM nombre_tabla WHERE condicion1 and condicion2…

Distinct

Solo se tienen en cuenta los valores distintos, es decir, se eliminan las tuplas repetidas
SELECT DISTINCT columna1, columna2… FROM nombre_tabla WHERE condicion;

Order By

Ordena los datos; por defecto en orden ascendente

SELECT columna1, columna2… FROM nombre_tabla ORDER BY columna ASC/DESC

In

Consiste en obtener mediante una subconsulta los elementos de dicho conjunto
SELECT codpie FROM ventas WHERE codpro IN (SELECT codpro FROM proveedor WHERE ciudad = 'Londres');

Exists

Este operador devuelve verdadero cuando existe alguna tupla en la relación sobre la que se aplica. El operador EXISTS puede interpretarse también como de comprobación de conjunto no vacío.
SELECT codpro, nompro FROM proveedor WHERE EXISTS (SELECT * FROM ventas WHERE ventas.codpro = proveedor.codpro AND ventas.codpie='P1');

Null

Valor nulo que no tiene por qué ser 0. Para comprobar que un valor es NULL:
-IS NULL
-IS NOT NULL

Update

Actualiza los datos de una tabla. Si no ponemos WHERE se actualizan todos los registros:

UPDATE nombre_tabla SET columna1=valor1, columna2=valor2… WHERE condicion/es

Funciones MIN() y MAX()

-MIN() devuelve el valor más pequeño de la columna seleccionada
SELECT MIN(columna) FROM nombre_tabla WHERE condicion/es

-MAX() devuelve el valor más grande de la columna seleccionada
SELECT MAX(columna) FROM nombre_tabla WHERE condicion/es

Count

Devuelve el n.º de filas de la consulta. Los valores NULL se ignoran.

SELECT COUNT(columna) FROM nombre_tabla WHERE condicion/es

SELECT COUNT(*) FROM nombre_tabla > Devuelve el número de filas de una tabla

Función AVG()

Devuelve el valor promedio de una columna específica (numérica)

SELECT AVG(columna) FROM nombre_tabla WHERE condicion/es

Función SUM()

Devuelve la suma total de una columna (numérica)
SELECT SUM(columna) FROM nombre_tabla WHERE condicion/es

Alias

Dan un nombre temporal a una tabla o columna

SELECT columna AS columna_alias FROM nombre_tabla > Para columnas
SELECT columna FROM nombre_tabla AS tabla_alias > Para tablas

Se utilizan para cuando tenemos más de una tabla involucrada en la consulta, se utilizan funciones en la consulta, las columnas tienen nombres grandes o poco legibles, o dos o más columnas se combinan juntas

Join

Combina filas de dos o más tablas, según una columna relacionada entre ellas.

-INNER JOIN: hace coincidir filas de la primera tabla con las de la segunda que tienen la misma clave (definida con ON) para crear un resultado con las columnas combinadas de ambas tablas

SELECT titulo, rating FROM peliculas JOIN calificacion ON peliculas.id=calificacion.pelicula_id ORDER BY rating ASC

Union

Combina el conjunto de resultados de dos o más SELECT. Las sentencias SELECT deben tener el mismo número columnas, mismo tipo de dato y en el mismo orden.

SELECT columna1 FROM tabla1…
UNION
SELECT columna2 FROM tabla2….

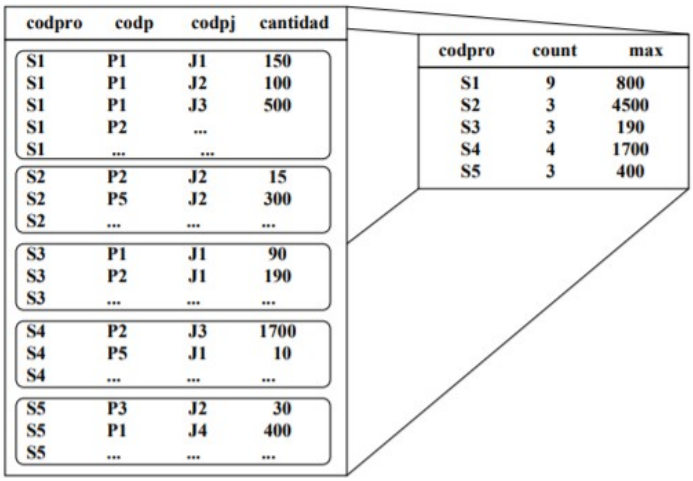
Interect

Group by

Junta las filas de resultados que cooinciden en el valor de alguna columna seleccionada.

SELECT columna1 FROM tabla1 WHERE condicion GROUP BY columna

```
SQL> SELECT codpro, COUNT(*), MAX(cantidad)
     FROM ventas
     GROUP BY codpro;
```



codpro	codp	codpj	cantidad
S1	P1	J1	150
S1	P1	J2	100
S1	P1	J3	500
S1	P2	...	
S1	...	...	
S2	P2	J2	15
S2	P5	J2	300
S2	...	...	...
S3	P1	J1	90
S3	P2	J1	190
S3	...	...	...
S4	P2	J3	1700
S4	P5	J1	10
S4	...	...	...
S5	P3	J2	30
S5	P1	J4	400
S5	...	...	...

codpro	count	max
S1	9	800
S2	3	4500
S3	3	190
S4	4	1700
S5	3	400

Having

Funciona igual que WHERE pero con funciones (SUM, MAX, MIN, AVG..)

SELECT columna1, SUM(columna2) FROM tabla WHERE coindicion ORDER BY columna HAVING SUM(columna2)< n

Index

Permite buscar rápidamente datos. Si una columna es índice de una tabla, al buscar por valor de esa columna iremos directamente a la fila correspondiente.

-Admitiendo valores duplicados
CREATE INDEX nombre_indice ON nombre_tabla(nombre_columna);

-Sin admitir valores duplicados
CREATE UNIQUE INDEX nombre_indice ON nombre_tabla(nombre_columna);

View

Tabla virtual basada en un conjunto de resultados de una declaración SQL. Muestran siempre datos reales de una o varias tablas.

-Crear vista:
CREATE VIEW nombre_vista AS SELECT nombre_columna/s FROM nombre_tabla WHERE condición

-Eliminar vista:

DROP VIEW nombre_vista