SQL (Structured Query Language)- DML

Data Manipulation Language:	SELECT, UPDATE, INSERT, DELETE
Data Definition Language:	CREATE: TABLE, INDEX, VIEW DROP: TABLE, INDEX, VIEW ALTER TABLE
Control Commands:	LOCK COMMIT / ROLLBACK UPDATE STATISTICS
Authorization Commands:	GRANT, REVOKE

Este lenguaje ofrece **cuatro proposiciones de DML** (Lenguaje de manejo de datos)

- SELECT (seleccionar)
- UPDATE (actualizar)
- DELETE (eliminar)
- INSERT (insertar)

Comando SELECT

Sintaxis:

SELECT {DISTINCT} elemento(s)
FROM tabla(s)
{WHERE condición}
{GROUP BY campo(s)}
{HAVING condición}
{ORDER BY campo(s)} ;

- Puede haber diferencias según el DBMS particular
- No confundir SELECT con la selección del álgebra relacional

Consultas Simples:

```
SELECT S#, SITUACION
FROM S
WHERE CIUDAD = "París" ;
```

ó

```
SELECT S.S#, S.SITUACION FROM S WHERE S.CIUDAD = "París" ;
```

S#	SITUACION
S2	10
S3	30

Recuperación simple

1. Obtener los números de parte de todas las partes suministradas

SELECT P# FROM SP ;

P#
P1
P2
P3
P4
P5
P6
P1
P2
P2
P2
P4
P5

SELECT **DISTINCT** P# FROM SP ;

P#
P1
P2
P3
P4
P5
P6

Recuperación de valores calculados

2. Obtener, para todas las partes, el nro. de parte y su peso en gramos

SELECT P.P#, 'Peso en gramos = ' , P.PESO * 454 FROM P ;

P#		
P1	Peso en gramos =	5448
P2	Peso en gramos =	7718
P3	Peso en gramos =	7718
P4	Peso en gramos =	6356
P5	Peso en gramos =	5448
P6	Peso en gramos =	8626

3. Obtener los datos de todos los proveedores

```
SELECT  *  FROM  S ;
```

(El resultado es toda la tabla)

Recuperación calificada

4. Obtener los números de proveedores radicados en París cuya situación sea mayor que 20.

```
SELECT  S#  FROM  S
WHERE  CIUDAD = París  AND Situación > 20 ;
```

S#
S3

Recuperación con ordenamiento

5. Obtener números de proveedor y situación de los proveedores radicados en París, en orden descendente por situación.

```
SELECT  S#,  SITUACION   FROM  S
WHERE  CIUDAD = "París"
ORDER BY  SITUACION  DESC ;
```

S#	SITUACION
S3	30
S2	10

Consultas de reunión:

6. Obtener todas las combinaciones de información de proveedores y partes tales que el proveedor y la parte en cuestión estén situados en la misma ciudad, pero omitiendo a los proveedores cuya situación sea 20.

```
SELECT  S.* , P.*   FROM  S,  P
WHERE  S.CIUDAD = P.CIUDAD
AND S.SITUACION <> 20 ;
```

S#	SNOMBRE	SITUACION	S.CIUDAD	P#	PNOMBRE	COLOR	PESO	P.CIUDAD
S2	Jaimes	10	París	P2	Perno	Verde	17	París
S2	Jaimes	10	París	P5	Leva	Azul	12	París
S3	Bernal	30	París	P2	Perno	Verde	17	París
S3	Bernal	30	París	P5	Leva	Azul	12	París

Recuperación de campos específicos de una reunión:

7. Obtener todas las combinaciones de número de proveedor/número de parte tales que el proveedor y la parte en cuestión estén cosituados.

```
SELECT  S.S# , P.P#  FROM  S,  P
WHERE  S.CIUDAD = P.CIUDAD ;
```

S#	P#
S1	P1
S1	P4
S1	P6
S2	P2
S2	P5
S3	P2
S3	P5
S4	P1
S4	P4
S4	P6

Funciones de agregados: **COUNT, SUM, AVG, MAX, MIN**

- Actúan sobre una columna

8. Obtener el número total de proveedores

```
SELECT COUNT (*)   FROM  S ;
```
Resultado: 5

9. Obtener el número de proveedores que suministran partes en la actualidad

```
SELECT COUNT (DISTINCT S#)  FROM  SP ;
```
Resultado: 4

Función de agregados en el SELECT con una condición:

10. Obtener el número de envíos de la parte P2

```
SELECT  COUNT (*)  FROM SP  WHERE  P# = "P2" ;
```
Resultado: 4

11. Obtener la cantidad total suministrada de la parte P2

SELECT SUM(CANT) FROM SP WHERE P# = 'P2'; Resultado: 1000

Empleo de GROUP BY:

12. Calcular la cantidad total suministrada de cada parte

SELECT P# , SUM(CANT) FROM SP GROUP BY P# ;

P#	
P1	600
P2	1000
P3	400
P4	500
P5	500
P6	100

Empleo de HAVING (con)

13. Obtener los números de todas las partes suministradas por más de un proveedor

SELECT P# FROM SP

GROUP BY P# HAVING COUNT(*) > 1 ;

P#
P1
P2
P4
P5

- HAVING es a los grupos lo que WHERE es a las filas.
- Las funciones de agregados van en la lista de campos. Si se hacen sobre los grupos se ponen las funciones de agregados en el having.

OTRAS CARACTERÍSTICAS

Recuperación de datos con LIKE (como)

14. Obtener todas las partes cuyos nombres comiencen con la letra B

SELECT P.* FROM P

WHERE P.PNOMBRE LIKE 'B%' ;

P#	PNOMBRE	COLOR	PESO	CIUDAD
P3	Birlo	Azul	17	Roma
P4	Birlo	Rojo	14	Londres

NOTAS:

Para detectar la ausencia o presencia de nulos: nombre columna IS {NOT} NULL

```
SELECT   S#   FROM  S
WHERE SITUACION IS NULL ;
```

Resultado: S# = S5

Para buscar en un conjunto de valores: nombre columna IN (conjunto valores)

<u>Recuperación de datos con subconsulta:</u>

15. Obtener los nombres de los proveedores que suministran la parte P2

```
SELECT  SNOMBRE  FROM  S
WHERE  S#  IN  (SELECT  S#  FROM SP  WHERE  P# = "P2" ) ;
```

SNOMBRE
Salazar
Jaimes
Bernal
Corona

equivale a:

```
SELECT SNOMBRE  FROM  S
WHERE S#  IN ('S1','S2','S3','S4') ;
```

también podría expresarse como una consulta de reunión:

```
SELECT  S.NOMBRE  FROM  S, SP
WHERE   S.S# =  SP.S#   AND   SP.P# = 'P2' ;
```

16. Obtener los nombres de los proveedores que suministren por lo menos una parte roja

```
SELECT  SNOMBRE  FROM  S
WHERE  S#   IN
        (SELECT  S#  FROM  SP
         WHERE  P#  IN
                (SELECT  P#  FROM  P
                 WHERE  COLOR = 'Rojo' )) ;
```

SNOMBRE
Salazar
Jaimes
Corona

Función de agregados en una subconsulta:

17. Obtener los números de los proveedores cuya situación sea menor que el valor máximo actual de situación en la tabla S

```
SELECT  S#  FROM  S
WHERE  SITUACION  <  (SELECT   MAX (SITUACION) FROM  S) ;
```

S#
S1
S2
S4

Consulta con EXISTS (existe)

18. Obtener los nombres de los proveedores que suministran la parte P2

```
SELECT SNOMBRE  FROM  S
WHERE  EXISTS  (SELECT  *  FROM  SP WHERE  S# = S.S#  AND  P# = 'P2')
```

- **EXITS (subconsulta)** da **verdadero** si el resultado de evaluar la subconsulta **no es el conjunto vacío**
- Esta consulta se podría haber resuelto utilizando **IN**

Consulta NOT EXISTS

19. Obtener los nombres de los proveedores que no suministran la parte P2 (lo inverso del caso anterior)

```
SELECT  SNOMBRE  FROM  S
WHERE NOT EXISTS  ( SELECT  *  FROM  SP  WHERE  S# = S.S#  AND P# = 'P2');
```

Resultado:

SNOMBRE
Aldana

- Equivale a enunciar: “seleccionar nombres de proveedores tales que no exista un envio que relacione a esos proveedores con la parte P2”. Y resolverla con un IN

```
SELECT  SNOMBRE  FROM  S
WHERE   S#  NOT IN  ( SELECT  S  FROM  SP  P# = ‘P2’);
```

20. Obtener los nombres de los proveedores que suministren todas las partes (Obtener el nombre de los proveedores tales que no exista una parte que no suministren)

```
SELECT SNOMBRE FROM  S
WHERE  NOT EXISTS
      (SELECT  * FROM  P  WHERE NOT EXISTS
      (SELECT  * FROM  SP  WHERE  S# = S.S#  AND  P = P.P# )) ;
```

Resultado:

SNOMBRE
Salazar

<u>Consulta con UNION</u>

21. Obtener los números de las partes que pesen más de 16 libras o sean suministradas por el proveedor S2 (o las dos cosas)

```
SELECT  P#  FROM  P  WHERE  PESO  >  16
UNION
SELECT  P#  FROM   SP  WHERE  S# = ‘S2’ ;
```

P1
P2
P3
P6

<u>Consulta con MINUS</u>

Operaciones de actualización

UPDATE (ACTUALIZAR)

UPDATE tabla SET campo = expresión [, campo = expresión]...
[WHERE condición] ;

- Sin WHERE se modifican todas las filas

1- Modificación de un solo registro

- Cambiar a color amarillo el color de la parte P1, aumentar su peso en 5 e indicar que su ciudad es desconocida

```
UPDATE P
SET COLOR = 'Amarillo', PESO = PESO + 5, CIUDAD = NULL
WHERE P# = 'P1' ;
```

2- Modificación de varios registros

- Duplicar la situación de todos los proveedores de Londres.

```
UPDATE S
SET SITUACION = 2 * SITUACION
WHERE CIUDAD = 'Londres' ;
```

3- Modificación con subconsulta

- Poner en cero la cantidad enviada por todos los proveedores de Londres

```
UPDATE SP
SET CANT = 0
WHERE S# IN (SELECT S# FROM S WHERE CIUDAD ='Londres') ;
```

4- Modificación de varios tablas

- Cambiar el número de proveedor S2 a S9.

```
UPDATE S SET S# = 'S9'
WHERE S# = 'S2' ;

UPDATE SP SET S# = 'S9'
WHERE S# = 'S2' ;
```

Si se la altera en forma incorrecta **se puede violar la integridad de datos** (int. referencial)

DELETE (ELIMINAR)

DELETE FROM tabla [WHERE condición] ;

- Elimina todos los registros de una tabla que satisfagan una condición

1- Eliminación de un solo registro

- Eliminar el proveedor S5

```
DELETE  FROM  S
WHERE   S# = 'S5' ;
```

2- Eliminación de varios registros

- Eliminar todos los envíos cuya cantidad sea mayor que 300

```
DELETE  FROM  SP
WHERE CANT  > 300 ;
```

- Eliminar todos los envíos

```
DELETE  FROM  SP
```

- SP existe pero vacía. La estructura se borra con DROP.

3- Eliminación con subconsulta

- Eliminar todos los envíos de los proveedores situados en Londres

```
DELETE  FROM  SP
WHERE   S#  IN   (SELECT S#  FROM  S  WHERE  CIUDAD = 'Londres') ;
```

INSERT (INSERTAR)

INSERT INTO tabla [(campo1 [, campo2, ..])]
VALUES (valor1 [, valor2,...]) ;

o bien **INSERT INTO tabla [(campo1 [, campo2, ..])]**
subconsulta ;

1- Inserción de un solo registro

- Añadir: parte P7 (ciudad Atenas, peso 24, nombre y color desconocidos) a la tabla P

```
INSERT
INTO P ( P#, CIUDAD, PESO)
VALUES ('P7','Atenas',24) ;
```

- Suponemos que NOMBRE y COLOR no se definieron como NOT NULL

2- Inserción de un solo registro omitiendo nombres de campos

- Añadir: parte P8 (nombre: cadena, color: rosa, peso: 14, ciudad: Niza) a la tabla P

```
INSERT INTO P
VALUES ('P8','Cadena', 'Rosa', 14, 'Niza') ;
```

3- Inserción de varios registros

- Para cada parte suministrada obtener el número de parte y la cantidad total suministrada y guardar el resultado en la BD Temp.

```
CREATE TABLE TEMP
        ( P# CHAR(6) NOT NULL, CANTTOTAL INTEGER NOT NULL,
        PRIMARY KEY ( P# ) );
CREATE INDEX XT ON TEMP ( P#);
INSERT INTO TEMP ( P#, CANTTOTAL)
        SELECT P#, SUM (CANT) FROM SP
        GROUP BY P# ;
```

- Añadir a los envíos las combinaciones de partes, proveedores y proyectos que no existan.

```
INSERT INTO SPJ (S#, P#, J#)
        SELECT  S#,  P#,  J#   FROM  S, P, J
                WHERE   S#  NOT IN (SELECT DISTINCT S#  FROM SPJ)
                AND  P#  NOT IN (SELECT DISTINCT P#  FROM SPJ)
                AND  J#  NOT IN (SELECT DISTINCT J#  FROM SPJ)
```