

Laboratorio de Bases de Datos I

Fase 3: Modelado Lógico

Capítulo 2: SQL - Lenguaje de Consultas Estructurado

Sergio Caro Álvaro
Ciencias de la Computación
Curso Académico 2017/2018

Índice

- Introducción
- Consultar datos
- Sentencias anidadadas
- Unir tablas
- Modificar datos
- Insertar datos
- Borrar datos

➤ **NOTA:** recomendable acudir a <https://www.w3schools.com/sql/>

Introducción

Conceptos de SQL

SQL - Introducción

- Un lenguaje de consulta estructurado (SQL) es un lenguaje de bases de datos normalizado, utilizado por diferentes motores de bases de datos para realizar determinadas operaciones sobre los datos o sobre la estructura de los mismos.
 - **SELECT** * **FROM** Table;
- Permite acceso y manipulación de bases de datos.
- Es un estándar ANSI.
 - Aún así, cada SGBD posee su propia implementación.
 - Están obligados a soportar los comandos básicos (SELECT, UPDATE, DELETE, INSERT, WHERE).

SQL - Introducción

- `SELECT` * `FROM` Table;
- En SQL, las palabras reservadas no son “*case sensitive*”, por lo que “SELECT”, “select”, “SelEcT” y “seLECT” son lo mismo para el SGBD.
- ¿Acabar sentencias con “;”?
 - Depende del SGDB (por lo general no es necesario)

SQL - Introducción

- Aunque el intérprete de todo SGBD es *inmune* al formato en el que se codifican las *queries* (consultas), existen una serie de convenciones a la hora de codificar que ayudan a la uniformidad y legibilidad del código, facilita el mantenimiento y elimina la dependencia del desarrollador.
- En BlackBoard hay una pequeña guía (al final del documento):
 - *[Laboratorio] / [Sesión 8, 9, 10 y 11] / [arrancando postgresSQL]*
 - En resumen, formatear siguiendo normas de sangrado (tabulador o varios espacios), alineación, en bloques, etc. Así como usar mayúsculas para las palabras clave.

SQL - Introducción

- Los comentarios en SQL se escriben con “--” y afectan al resto de la línea (aunque se pongan palabras reservadas, serán ignoradas):

--Select all:

```
SELECT * FROM tabla;
```

Consultar Datos

SELECT

Consultar datos. SELECT básico

SELECT ... FROM ... WHERE ...

Borrar
duplicados

Mostrar todas
las columnas

Nombre de las
columnas a mostrar

Nombre de la tabla

SELECT [DISTINCT] [* | columna(s)]

FROM tabla [AS nombre]

WHERE condition1 [AND|OR] [NOT] condition2

Se puede renombrar para trabajar con un nombre más cómodo (solo visual, no renombra la DB) (se puede usar también en las columnas del select)

Las condiciones
son opcionales

Conectores de
condiciones

Que no cumpla la condición
(aplicable a cualquiera)

Ejemplo

```
SELECT *  
FROM Alumno
```

ID	Nombre	Ciudad	Nota	Creditos
1	Hector	Guadalajara	3.5	6
2	Pablo	Meco	5	12
3	Ana	Alcalá	5.5	20
4	Lara	Guadalajara	8	24
5	Marina	San Fernando	9	80
6	Antonio	Alcalá	2.5	9
7	Pablo	Guadalajara	7.5	19

```
SELECT Nombre, Ciudad  
FROM Alumno
```

Nombre	Ciudad
Hector	Guadalajara
Pablo	Meco
Ana	Alcalá
Lara	Guadalajara
Marina	San Fernando
Antonio	Alcalá
Pablo	Guadalajara

WHERE - Ejemplos

```
SELECT *  
FROM Alumno  
WHERE créditos > 14
```

ID	Nombre	Ciudad	Nota	Creditos
3	Ana	Alcalá	5.5	20
4	Lara	Guadalajara	8	24
5	Marina	San Fernando	9	80
7	Pablo	Guadalajara	7.5	19

Carácter comodín de búsqueda en cadenas de texto:

% - 0, 1 o n caracteres

_ - 1 carácter

Según el SGBD podemos usar expresiones regulares.

```
SELECT *  
FROM Alumno  
WHERE nombre LIKE '%ar%'
```

ID	Nombre	Ciudad	Nota	Creditos
4	Lara	Guadalajara	8	24
5	Marina	San Fernando	9	80

Ejemplos

```
SELECT nota, count(*) AS nAlum  
FROM Alumnos  
GROUP BY nota  
ORDER BY nAlum DESC
```

Nota: funciones propias de SQL son: COUNT(), AVG(), SUM(), MIN() y MAX()

Nota	nAlum
2.5	1
3.5	1
5	1
5.5	1
7.5	1
8	1
9	1

```
SELECT * FROM Alumno  
WHERE nota > 5
```

```
AND créditos
```

```
BETWEEN 12 AND 40
```

ID	Nombre	Ciudad	Nota	Creditos
3	Ana	Alcalá	5.5	20
4	Lara	Guadalajara	8	24
7	Pablo	Guadalajara	7.5	19

GROUPBY - Ejemplo

```
SELECT nota, count(*) AS nAlum  
FROM Alumnos  
GROUP BY nota  
HAVING nota > 5  
ORDER BY nAlum DESC
```

Nota	nAlum
5.5	1
7.5	1
8	1
9	1

NOTA: HAVING solo se usa con GROUP BY. Se usa como filtro de la agregación, ya que al agrupar no se puede usar un WHERE.

Limitar resultados - Ejemplo

```
SELECT *  
FROM Alumno  
WHERE créditos > 14  
LIMIT 2
```

Para mostrar un número limitado de filas.

NOTA: hay SGBD que, en vez de LIMIT, usan TOP. Destacar que top se usa junto a la clausula SELECT:

SELECT TOP numero [PERCENT] columna(s)

ID	Nombre	Ciudad	Nota	Creditos
3	Ana	Alcalá	5.5	20
4	Lara	Guadalajara	8	24

Comparar NULL

- **NOTA:** en SQL no es posible comparar elementos que sean nulos (NULL) con los operadores típicos de comparación usados hasta ahora (<, >, =, <>).
- Se debe utilizar la siguiente sintaxis: “condition **IS [NOT] NULL**”

Sentencias Anidadas

Subconsultas

Consultas Anidadas

- Una **subconsulta** es una consulta anidada en una instrucción SELECT, INSERT, UPDATE o DELETE, o también puede estar en otra subconsulta.
 - Es decir, distinguimos entre consulta externa y consulta interna

```
SELECT *  
FROM cliente  
WHERE país IN  
(  
    SELECT país  
    FROM distribuidor  
)
```

Consultas Anidadas

- Teóricamente, podemos tener hasta 32 niveles de subconsultas. En la práctica, el planificador del SGBD determinará si se puede o no realizar la consulta completa.
- **NOTA:** las subconsultas, siempre entre paréntesis.
- Típicamente, podemos encontrar subconsultas en:
 - **WHERE** expresion [**NOT**] **IN** (subconsulta)
 - **WHERE** expresion [< | > | = | <>] [**ANY**|**ALL**] (subconsulta)
 - **WHERE** [**NOT**] **EXISTS** (subconsulta)

Unir Tablas

JOINS

Joins

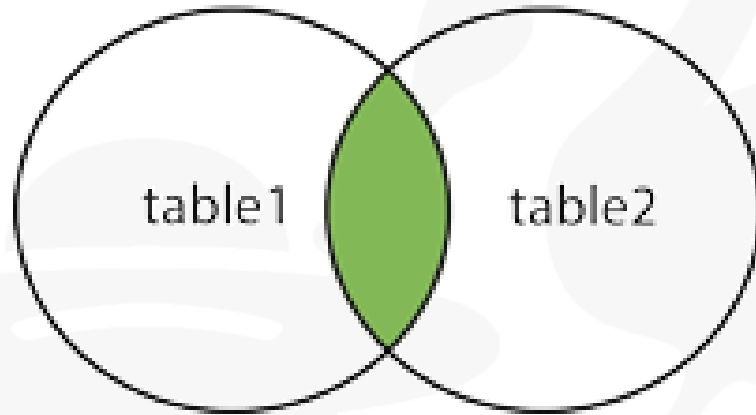
- La cláusula JOIN permite combinar filas de dos o mas tablas, siempre y cuando estén relacionadas por una columna entre sí.
 - Típicamente, sobre las PKs y las FKs

```
SELECT t1.columna(s), t2.columna(s)
FROM tabla1 AS t1
      [INNER | LEFT | RIGHT | FULL] JOIN tabla2 As t2
      ON t1.id = t2.id
```

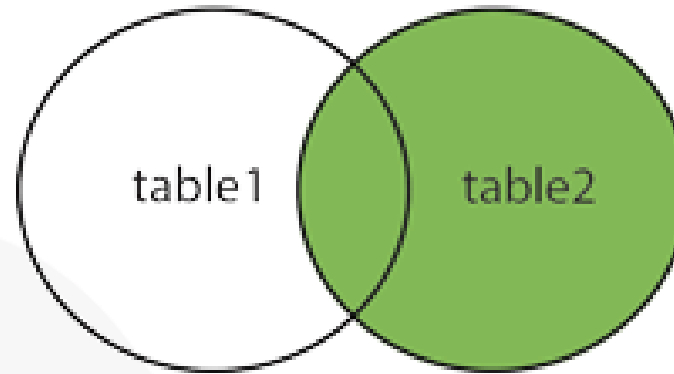
← Columnas relacionadas.

Joins

INNER JOIN

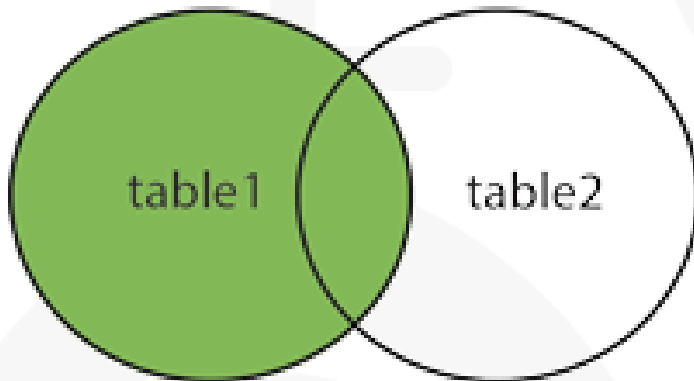


RIGHT JOIN

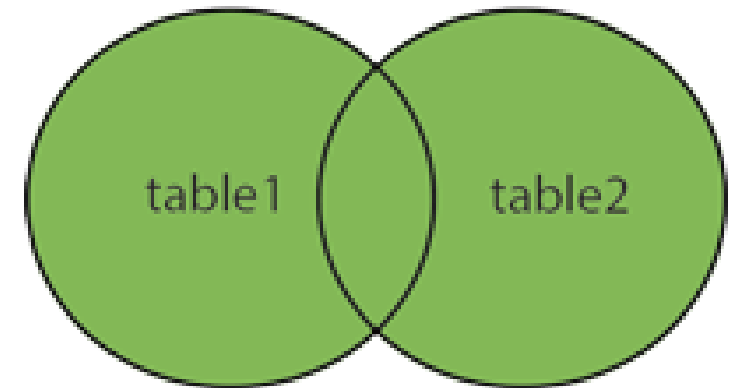


NOTA: se pueden obtener otras alternativas de resultado realizando indicaciones en el WHERE de la consulta.

LEFT JOIN



FULL OUTER JOIN



OBS: estos joins pueden producir columnas NULL, puesto que puede no haber asociación entre instancias de las tablas.

Producto Cartesiano

```
SELECT t1.columna(s), t2.columna(s)
FROM tabla1 AS t1, tabla2 AS t2
```

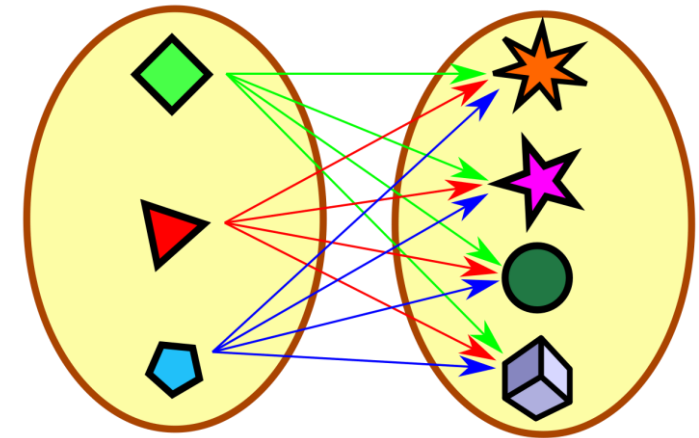
1	A
2	B
3	C

 \times

A
B
C

 $=$

1	A
1	B
1	C
2	A
2	B
2	C
3	A
3	B
3	C



© Wikipedia

Modificar Datos

UPDATE


Modificar datos

UPDATE ... SET ... WHERE ...

UPDATE tabla

SET columna(a) = nuevoValor

WHERE condición



¿Qué filas actualizamos?
NOTA: puede usarse una
subconsulta.

Ejemplo

UPDATE Alumnos

SET ciudad='Mostoles'

WHERE ciudad LIKE 'San Fernando'

ID	Nombre	Ciudad	Nota	Creditos
1	Hector	Guadalajara	3.5	6
2	Pablo	Meco	5	12
3	Ana	Alcalá	5.5	20
4	Lara	Guadalajara	8	24
5	Marina	Mostoles	9	80
6	Antonio	Alcalá	2.5	9
7	Pablo	Guadalajara	7.5	19

Insertar Datos

INSERT

Insertar datos

INSERT INTO ... (...,...) VALUES (...,...)

INSERT INTO ... SET ...

INSERT INTO tabla (columna(s))
VALUES (valor(es))

INSERT INTO tabla
SET columna(s) = valor(es)

Este formato suele usarse
en MySQL

Raramente se usa el WHERE aquí, puesto
que se considera que no hay elementos en
la tabla. Sin embargo, sí se usa si basamos
la inserción en el resultado de un SELECT.

Ejemplo

- Si se insertan todas las columnas de la tabla:

```
INSERT INTO Alumno VALUES (1, 'Hector', 'Guadalajara', 3.5, 6);
```

- En caso de no introducir datos en todas las columnas:

```
INSERT INTO Alumnos (ID, nombre, ciudad, nota, creditos)  
VALUES (4, 'Lara', 'Guadalajara', 8.0, 24);
```

```
INSERT INTO Alumnos (ID, nombre, nota)  
VALUES (4, 'Lara', 8.0);
```

Ejemplo

- Se pueden insertar varias tuplas en la misma sentencia:

INSERT INTO Alumno

VALUES (1, 'Hector', 'Guadalajara', 3.5, 6),
 (2, 'Tomas', 'Madrid', 3.5, 6),
 (3, 'Ana', 'Mostoles', 3.5, 6);

Borrar Datos

DELETE

Borrar datos

DELETE FROM ... WHERE ...

DELETE FROM table
WHERE condition

OJO: si no se pone el WHERE,
se borran todas las tuplas de
la tabla indicada.



Ejemplo

DELETE FROM Alumnos WHERE id = 4

ID	Nombre	Ciudad	Nota	Creditos
1	Hector	Guadalajara	3.5	6
2	Pablo	Meco	5	12
3	Ana	Alcalá	5.5	20
4	Lara	Guadalajara	8	24
5	Marina	Mostoles	9	80
6	Antonio	Alcalá	2.5	9
7	Pablo	Guadalajara	7.5	19