{Introducción a SQL}

[Clase 2]

Clase 2.

Contenidos principales:

- Breve repaso de los temas de la clase 1.
- Tipos de datos y valores nulos.
- Concepto de **JOIN**.
- **INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN**
- **UNION, UNION ALL.**

Luego de esta clase serás capaz de:

- Identificar los tipos de variables y trabajar con valores ausentes o nulos.
- Combinar dos o más tablas en una sola.
- Generar uniones entre tablas.

Repaso clase 1.

Vimos cómo **seleccionar las columnas** de una tabla, como realizar **filtros** sobre los registros, cómo realizar **cálculos sencillos**, y cómo **ordenarlos**. Por último aprendimos a **crear una tabla** con la información que genera una consulta.

Hasta ahora la sintaxis de nuestras consultas son de dos formas:

```
SELECT <columnas>
FROM nombre_de_tabla
[WHERE condiciones]
[ORDER BY columna_x [desc],...,columna_z]
```

O si vamos a crear una tabla:

```
CREATE TABLE nombre_de_tabla AS
   SELECT ...
```

MySQL DATA TYPES

DATE TYPE	SPEC	DATA TYPE	SPEC
CHAR	String (0 - 255)	INT	Integer (-2147483648 to 214748-3647)
VARCHAR	String (0 - 255)	BIGINT	Integer (-92233720368547758 08 to 9223372036854775807)
TINYTEXT	String (0 - 255)	FLOAT	Decimal (precise to 23 digits)
TEXT	String (0 - 65535)	DOUBLE	Decimal (24 to 53 digits)
BLOB	String (0 - 65535)	DECIMAL	"DOUBLE" stored as string
MEDIUMTEXT	String (0 - 16777215)	DATE	YYYY-MM-DD
MEDIUMBLOB	String (0 - 16777215)	DATETIME	YYYY-MM-DD HH:MM:SS
LONGTEXT	String (0 - 4294967295)	TIMESTAMP	YYYYMMDDHHMMSS
LONGBLOB	String (0 - 4294967295)	TIME	HH:MM:SS
TINYINT	Integer (-128 to 127)	ENUM	One of preset options
SMALLINT	Integer (-32768 to 32767)	SET	Selection of preset options
MEDIUMINT	Integer (-8388608 to 8388607)	BOOLEAN	TINYINT(1)

Valores nulos

En cualquier sistema de almacenamiento de datos se presentan situaciones en las cuales hay valores de un campo que están ausentes. Se denominan **valores nulos** (MISSING VALUES o NULL VALUES, en inglés) y debemos tratarlos de manera especial.

Una manera de detectar valores nulos en una columna es usando **IS NULL**:

```
select * from address where address2 IS NULL;
```

Joins de tablas:

Permiten combinar los registros y columnas de dos (o más) tablas mediante alguna/s columna/s que tengan en común.

- **INNER JOIN:** devuelve todos los registros donde hay correspondencia entre dos tablas.
- **LEFT JOIN:** devuelve todos los registros de la tabla que se encuentra a la **izquierda** de la sentencia y solo los campos donde haya correspondencia de la tabla derecha.
- **RIGHT JOIN:** devuelve todos los registros de la tabla que se encuentra a la **derecha** de la sentencia y solo los campos donde haya correspondencia de la tabla izquierda.
- **FULL JOIN:** devuelve todos los registros de ambas tablas independientemente de si hay o no correspondencia.

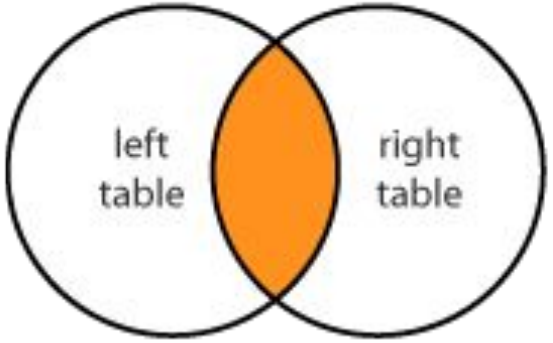
Observación: en el caso de LEFT, RIGHT o FULL JOIN cuando no hay correspondencia de un registro en la otra tabla se completa con **NULL** la columna donde falta ese registro (ver ejemplos).

Sintaxis:

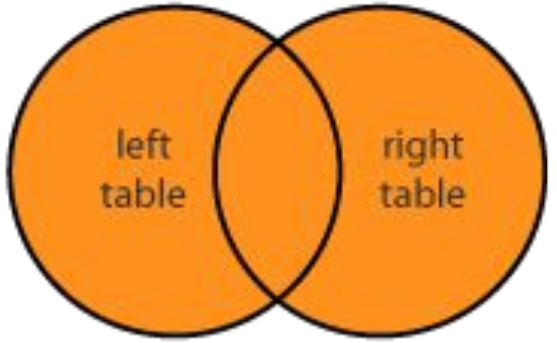
```
SELECT <columnas tabla_1, columnas tabla_2>
FROM tabla_1 a  [INNER/LEFT/RIGHT/FULL] JOIN tabla_2  b  ON (a.campo_tabla_1=b.campo_tabla_2);
```

Tipos de JOINS

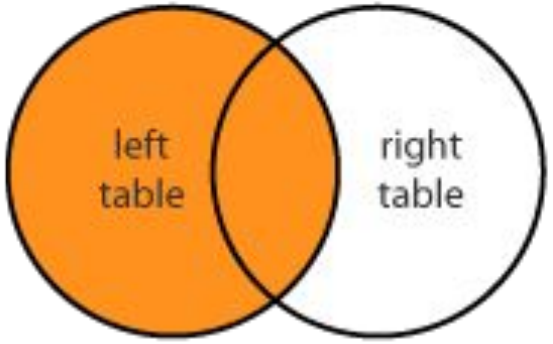
INNER JOIN



FULL JOIN



LEFT JOIN



RIGHT JOIN

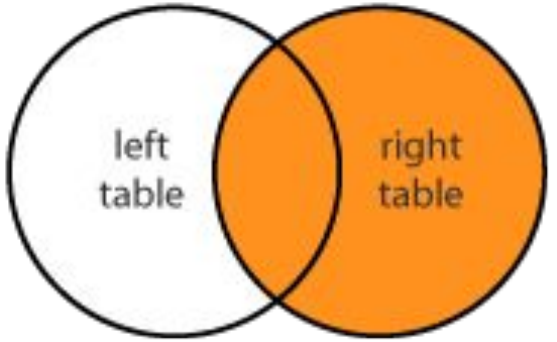


TABLA A	
ID_EMPLEADO	NOMBRE
1	JUAN
2	SEBASTIAN
3	AGUSTIN
4	PATRICIO
5	LUCIANO

TABLA B	
ID	DEPARTAMENTO
1	MARKETING
2	ADMINISTRACION
3	TESORERIA
5	TESORERIA
6	ADMINISTRACION

INNER JOIN

ID_EMPLEADO	NOMBRE	DEPARTAMENTO
1	JUAN	MARKETING
2	SEBASTIAN	ADMINISTRACION
3	AGUSTIN	TESORERIA
5	LUCIANO	TESORERIA

```
SELECT a.id_empleado,a.nombre,b.departamento
FROM tabla_a a INNER JOIN tabla_b b ON a.id_empleado=b.id
ORDER BY 1;
```

TABLA A	
ID_EMPLEADO	NOMBRE
1	JUAN
2	SEBASTIAN
3	AGUSTIN
4	PATRICIO
5	LUCIANO

TABLA B	
ID	DEPARTAMENTO
1	MARKETING
2	ADMINISTRACION
3	TESORERIA
5	TESORERIA
6	ADMINISTRACION

LEFT JOIN

ID_EMPLEADO	NOMBRE	DEPARTAMENTO
1	JUAN	MARKETING
2	SEBASTIAN	ADMINISTRACION
3	AGUSTIN	TESORERIA
4	PATRICIO	NULL
5	LUCIANO	TESORERIA

```
SELECT a.id_empleado,a.nombre,b.departamento
FROM tabla_a a LEFT JOIN tabla_b b ON a.id_empleado=b.id
ORDER BY 1;
```

TABLA A	
ID_EMPLEADO	NOMBRE
1	JUAN
2	SEBASTIAN
3	AGUSTIN
4	PATRICIO
5	LUCIANO

TABLA B	
ID	DEPARTAMENTO
1	MARKETING
2	ADMINISTRACION
3	TESORERIA
5	TESORERIA
6	ADMINISTRACION

RIGHT JOIN

ID_EMPLEADO	NOMBRE	DEPARTAMENTO
1	JUAN	MARKETING
2	SEBASTIAN	ADMINISTRACION
3	AGUSTIN	TESORERIA
5	LUCIANO	TESORERIA
6	NULL	ADMINISTRACION

```
SELECT b.id as id_empleado,a.nombre,b.departamento
FROM tabla_a a RIGHT JOIN tabla_b b ON a.id_empleado=b.id
ORDER BY 1;
```

TABLA A	
ID_EMPLEADO	NOMBRE
1	JUAN
2	SEBASTIAN
3	AGUSTIN
4	PATRICIO
5	LUCIANO

TABLA B	
ID	DEPARTAMENTO
1	MARKETING
2	ADMINISTRACION
3	TESORERIA
5	TESORERIA
6	ADMINISTRACION

FULL JOIN

ID_EMPLEADO	NOMBRE	DEPARTAMENTO
1	JUAN	MARKETING
2	SEBASTIAN	ADMINISTRACION
3	AGUSTIN	TESORERIA
4	PATRICIO	NULL
5	LUCIANO	TESORERIA
NULL	NULL	ADMINISTRACION

```
SELECT a.id_empleado,a.nombre,b.departamento
FROM tabla_a a FULL JOIN tabla_b b ON a.id_empleado=b.id
ORDER BY 1;
```

Uniones de tablas:

Sirven para unir los registros de dos tablas (una encima de la otra) que tengan columnas en común. Estas deben tener obligatoriamente el mismo tipo de dato.

- **UNION:** une dos tablas. Si un mismo registro está en ambas tablas mantiene uno solo.
- **UNION ALL:** une dos tablas. Si un mismo registro está en ambas tablas mantiene ambos casos.

Sintaxis:

```
SELECT <campos tabla_1>
FROM tabla_1
UNION [ALL]
SELECT <campos tabla_2>
FROM tabla_2;
```

TABLA_X	
CELULAR	PRECIO
SAMSUNG S8	22.000
IPHONE 7	23.500
SAMSUNG J7	7.000
SONY M2	7.300
SONY Z3	19.000

TABLA_Y	
CELULAR	PRECIO
SAMSUNG J7	7.000
MOTOROLA G2	6.500
HUAWEI P10	18.000
IPHONE 6S	17.500
SONY M2	7.300

UNION

CELULAR	PRECIO
SAMSUNG S8	22.000
IPHONE 7	23.500
SAMSUNG J7	7.000
SONY M2	7.300
SONY Z3	19.000
MOTOROLA G2	6.500
HUAWEI P10	18.000
IPHONE 6S	17.500

```
SELECT celular,precio
FROM tabla_x
UNION
SELECT celular, precio
FROM tabla_y;
```

Observaciónń: no necesariamente los nombres de las columnas tienen que ser iguales. Lo que sí tiene que coincidir es el **tipo de dato**.

TABLA_X	
CELULAR	PRECIO
SAMSUNG S8	22.000
IPHONE 7	23.500
SAMSUNG J7	7.000
SONY M2	7.300
SONY Z3	19.000

TABLA_Y	
CELULAR	PRECIO
SAMSUNG J7	7.000
MOTOROLA G2	6.500
HUAWEI P10	18.000
IPHONE 6S	17.500
SONY M2	7.300

UNION ALL

CELULAR	PRECIO
SAMSUNG S8	22.000
IPHONE 7	23.500
SAMSUNG J7	7.000
SONY M2	7.300
SONY Z3	19.000
SAMSUNG J7	7.000
MOTOROLA G2	6.500
HUAWEI P10	18.000
IPHONE 6S	17.500
SONY M2	7.300

```
SELECT celular,precio
FROM tabla_x
UNION ALL
SELECT celular,precio
FROM tabla_y;
```

Observaciónń: no necesariamente los nombres de las columnas tienen que ser iguales. Lo que sí tiene que coincidir es el **tipo de dato**.

Realizar la ejercitación que se encuentra en el archivo: