



# UT5. CONSULTAS DE RECUPERACIÓN MONOTABLA EN SQL.

Módulo: BASES DE DATOS

Curso 2022/2023. 1º DAM

Ruth Lospitao Ruiz

Invoice

Invoice_id
Customer_id
Order_id
Product_id
Date_time
Status
Total
Remark

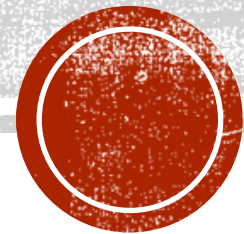


# CONTENIDOS

- Selección de registros (Sentencia SELECT Parte I):
- Filtrado
- Tratamiento de nulos
- Ordenamiento
- Limitar el resultado



# SELECCIÓN DE REGISTROS





# SINTAXIS DE SELECT

```
SELECT [ DISTINCT ] select_expr [, select_expr ...]  
FROM table_references  
[ WHERE filtro ]  
[ ORDER BY {col_name | expr | position} [ASC | DESC] , ... ]  
[ LIMIT [ desplazamiento , ] nfilas ]
```

Sintaxis completa de la sentencia SELECT la cual iremos viendo poco a poco



# RECUPERACIÓN DE DATOS: SENTENCIA SELECT

- Se utiliza para consultar la información de la base de datos.

**SELECT** [DISTINCT] *select\_expr* [, *select\_expr* ...]  
**FROM** *table\_references*

SELECT campo, campo2, campo3 FROM tabla1;



# RECUPERACIÓN DE DATOS: SENTENCIA SELECT

- En ***table\_references*** aparecerá el nombre de una tabla (de momento).
- ***select\_expr*** es: ***nombre\_columna*** [***AS alias***]
  - | \*
  - | ***expresión*** [***AS alias***]

Por tanto, entre las palabras SELECT y FROM van las columnas que saldrán como resultado.

Puede ser alguna columna de la tabla indicada en el FROM, todas las columnas (con \*), o expresiones (matemáticas, de cadena, booleanas) aplicadas sobre una o varias columnas.

Sobre cada fila de la tabla indicada en el FROM, se calculan las columnas a mostrar, indicadas en el SELECT.



# TABLAS DE EJEMPLOS

EDITORIALES			
Nombre_e	Direccion	Pais	Ciudad
Universal Books	Brown Sq. 23	EEUU	Los Ángeles
Rama	Canillas, 144	España	Madrid
Mc Graw-Hill	Basauri, 17	España	Madrid
Paraninfo	Virtudes, 7	España	Madrid

LIBROS				
Codigo	Titulo	Num_paginas	Fecha_publica	Editorial
34587	Int. Artificial	50	10/08/2017	Paraninfo
1022305	Concep. Y Dis.	48	07/05/2019	Rama
493942	Turbo C++	125	25/05/2003	Mc Graw-Hill
45307	Virus Informát.	50	12/12/2004	NULL
112313	Sist. Informac.	358	15/08/2013	Rama



# SELECT: EJEMPLOS

```
SELECT codigo, titulo  
FROM libros;
```

De la tabla libros, se toman todas las filas, y para cada fila, las columnas código y título

LIBROS			
Codigo	Titulo	Num_paginas	Editorial
34587	Int. Artificial	50	Paraninfo
1022305	Concep. Y Dis.	48	Rama
493942	Turbo C++	125	Mc Graw-Hill
45307	Virus Informát.	50	NULL
112313	Sist. Informac.	358	Rama



Codigo	Titulo
34587	Int. Artificial
1022305	Concep. Y Dis.
493942	Turbo C++
45307	Virus Informát.
112313	Sist. Informac.





# SELECT: EJEMPLOS

```
SELECT *
```

```
FROM libros;
```

De la tabla libros, se toman todas las columnas en el mismo orden en que aparecen.

Como en un SGBD el orden de las columnas no es determinante y no está garantizado, el uso de \* no debe utilizarse.

LIBROS			
Codigo	Titulo	Num_paginas	Editorial
34587	Int. Artificial	50	Paraninfo
1022305	Concep. Y Dis.	48	Rama
493942	Turbo C++	125	Mc Graw-Hill
45307	Virus Informát.	50	NULL
112313	Sist. Informac.	358	Rama



LIBROS			
Codigo	Titulo	Num_paginas	Editorial
34587	Int. Artificial	50	Paraninfo
1022305	Concep. Y Dis.	48	Rama
493942	Turbo C++	125	Mc Graw-Hill
45307	Virus Informát.	50	NULL
112313	Sist. Informac.	358	Rama



# SELECT: EJEMPLOS

alias

```
SELECT codigo, titulo,  
       num_paginas / 1000 AS miles_páginas  
FROM libros;
```

De la tabla libros, se toman las columnas código y título, y la columna num\_paginas que se divide por 1000.

A cada columna resultado, se puede asignar un nombre (alias). Cuando no se referencia a una columna concreta, se recomienda el uso de alias.

LIBROS			
Codigo	Titulo	Num_paginas	Editorial
34587	Int. Artificial	50	Paraninfo
1022305	Concep. Y Dis.	48	Rama
493942	Turbo C++	125	Mc Graw-Hill
45307	Virus Informát.	50	NULL
112313	Sist. Informac.	358	Rama

Codigo	Titulo	Miles_paginas
34587	Int. Artificial	0,05
1022305	Concep. Y Dis.	0,048
493942	Turbo C++	0,125
45307	Virus Informát.	0,05
112313	Sist. Informac.	0,358



# CONSIDERACIONES GENERALES

- Se puede poner un alias a cada tabla.
  - Se recomienda en el caso de utilizar varias tablas en la misma sentencia (se verá en la UT siguiente).
  - Esto se realiza igual que los alias de las columnas.
  - Tanto para la tabla, como para las columnas, en términos generales, no es necesario poner la palabra reservada 'AS', pero hay algunos casos concretos en los que es necesario ponerla.
- A las columnas se les puede llamar por su nombre de columna, o por el nombre de tabla, un punto y el nombre de columna.

Ejemplo: `SELECT libros.titulo FROM libros;`

- Esto último es obligatorio cuando hay dos columnas que se llaman igual (veremos los casos), ya que el SGBD no es capaz de distinguir a cuál de ellas nos referimos.
- Como una tabla puede tener un alias, en vez del nombre de la tabla, se puede utilizar el alias para calificar la columna.

Ejemplo: `SELECT lib.titulo FROM libros lib;`



# SELECT: EJEMPLOS

```
SELECT codigo, libros.titulo,  
        lib.num_paginas / 1000 AS miles_paginas  
FROM libros lib;
```

Es la misma consulta del ejemplo anterior, pero se usan las 3 formas de nombrar una columna:

- solo el nombre de la columna (codigo),
- el nombre de la tabla y el nombre de la columna (libros.titulo)
- o el alias de la tabla y el nombre de la columna (lib.num\_paginas).

LIBROS			
Codigo	Titulo	Num_paginas	Editorial
34587	Int. Artificial	50	Paraninfo
1022305	Concep. Y Dis.	48	Rama
493942	Turbo C++	125	Mc Graw-Hill
45307	Virus Informát.	50	NULL
112313	Sist. Informac.	358	Rama



Codigo	Titulo	Miles_paginas
34587	Int. Artificial	0,05
1022305	Concep. Y Dis.	0,048
493942	Turbo C++	0,125
45307	Virus Informát.	0,05
112313	Sist. Informac.	0,358




# SELECT: EJEMPLOS

```
SELECT codigo,  
       concat(titulo, ' (' , editorial,')' ) AS nombre  
FROM libros;
```

Una expresión puede no contener ninguna columna,  
contener una, o varias.

LIBROS			
Codigo	Titulo	Num_paginas	Editorial
34587	Int. Artificial	50	Paraninfo
1022305	Concep. Y Dis.	48	Rama
493942	Turbo C++	125	Mc Graw-Hill
45307	Virus Informát.	50	NULL
112313	Sist. Informac.	358	Rama



Codigo	nombre
34587	Int. Artificial (Paraninfo)
1022305	Concep. Y Dis. (Rama)
493942	Turbo C++ (Mc Graw-Hill)
45307	Virus Informát. (NULL)
112313	Sist. Informac. (Rama)



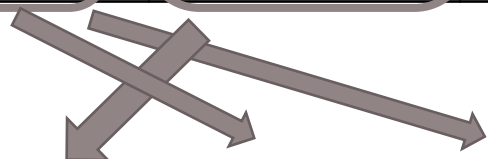
# SELECT: EJEMPLOS

```
SELECT titulo, codigo,  
       codigo AS codigo2,  
       codigo + 1 AS codigoSiguiente  
FROM libros;
```

Una columna puede aparecer varias veces (tanto en expresiones como directamente).

Se debería usar alias para dar nombre a cada columna de resultado.

LIBROS			
Codigo	Titulo	Num_paginas	Editorial
34587	Int. Artificial	50	Paraninfo
1022305	Concep. Y Dis.	48	Rama
493942	Turbo C++	125	Mc Graw-Hill
45307	Virus Informát.	50	NULL
112313	Sist. Informac.	358	Rama



Titulo	Codigo	Codigo2	CodigoSiguiente
Int. Artificial	34587	34587	34588
Concep. Y Dis.	1022305	1022305	1022306
Turbo C++	493942	493942	493943
Virus Informát.	45307	45307	45308
Sist. Informac.	112313	112313	112314





# SELECT: DISTINCT

La palabra reservada DISTINCT en un SELECT elimina filas repetidas.

Si hay dos filas resultado iguales, solo muestra una.

Muchas veces, la columna que va después de DISTINCT se pone entre paréntesis. Y también, muchas veces solo aparece esa columna.



EDITORIALES			
Nombre_e	Direccion	Pais	Ciudad
Universal Books	Brown Sq. 23	EEUU	Los Ángeles
Rama	Canillas, 144	España	Madrid
Mc Graw-Hill	Basauri, 17	España	Madrid
Paraninfo	Virtudes, 7	España	Madrid



```
SELECT DISTINCT ciudad  
FROM editoriales;
```

Ciudad
Los Ángeles
Madrid



**FILTRADO**



# FILTROS: WHERE

**SELECT** [DISTINCT] *select\_expr* [, *select\_expr* ...]

**FROM** *table\_references*

**WHERE** *filtro*

- El **filtro** es una expresión que indica la condición o condiciones que deben satisfacer las filas de la tabla indicada en el FROM para ser seleccionadas. Es decir, se comprueba fila a fila de la tabla indicada en el FROM, si se cumple la condición indicada en el WHERE. Si no se cumple, esa fila se elimina de los resultados.

Por tanto, al aplicar el filtro con los valores de una fila, debemos obtener un valor booleano (cierto o falso).



# WHERE: EJEMPLO

```
SELECT codigo, titulo
```

```
FROM libros
```

```
WHERE num_paginas < 60;
```

De la tabla libros, se toman las filas que tengan menos de 60 páginas, y para cada fila resultante, se toman las columnas código y título

También se puede entender como que se eliminan las filas que tengan 60 o más páginas,

LIBROS				
Codigo	Titulo	Num_paginas	Editorial	
34587	Int. Artificial	50	Paraninfo	✓
1022305	Concep. Y Dis.	48	Rama	✓
493942	Turbo C++	125	Mc Graw-Hill	✗
45307	Virus Informát.	50	NULL	✓
112313	Sist. Informac.	358	Rama	✗

Codigo	Titulo
34587	Int. Artificial
1022305	Concep. Y Dis.
45307	Virus Informát.



# EXPRESIONES

Las expresiones se pueden utilizar tanto en los filtros como en el select (más adelante veremos más sitios donde también se utilizan), y pueden devolver un valor de cualquier tipo. En el caso de usarse en un filtro de un WHERE, deben devolver un valor booleano (cierto o falso).

## ➤ Operadores aritméticos:

- $+$  → Suma
- $-$  → Resta
- $*$  → Producto o multiplicación
- $/$  → División
- $\%$  → Módulo o resto de la división entera

## ➤ Operadores relacionales:

- $x = y$  x es igual a y
- $x <> y$  ó  $x \neq y$  x es distinto de y
- $x < y$  x es menor que y
- $x > y$  x es mayor que y
- $x \leq y$  x es menor o igual que y
- $x \geq y$  x es mayor o igual que y



# EXPRESIONES

## ➤ Operadores Lógicos:

- AND: Se deben cumplir ambos.
- OR: Se debe cumplir al menos uno de ellos.
- NOT: Sobre un elemento, devuelve lo contrario.

Tabla de la verdad para el operador lógico AND

Operando 1	Operando 2	Resultado
TRUE (1)	TRUE (1)	TRUE (1)
TRUE (1)	FALSE (0)	FALSE (0)
FALSE (0)	TRUE (1)	FALSE (0)
FALSE (0)	FALSE (0)	FALSE (0)

Tabla de la verdad para el operador lógico OR

Operando 1	Operando 2	Resultado
TRUE (1)	TRUE (1)	TRUE (1)
TRUE (1)	FALSE (0)	TRUE (1)
FALSE (0)	TRUE (1)	TRUE (1)
FALSE (0)	FALSE (0)	FALSE (0)

Tabla de la verdad para el operador lógico NOT

Operando 1	Resultado
TRUE (1)	FALSE (0)
FALSE (0)	TRUE (1)





# EJEMPLOS CON OPERADORES LÓGICOS

```
SELECT titulo FROM libros WHERE num_paginas > 30;
```

```
SELECT codigo FROM libros WHERE editorial = 'Mc Graw-Hill' and num_paginas <=20;
```

```
SELECT titulo FROM libros WHERE num_paginas > 55 and editorial = 'Mc Graw-Hill' or editorial = 'Paraninfo';
```



# PRECEDENCIA DE OPERADORES

```
1  INTERVAL
2  BINARY, COLLATE
3  !
4  - (unary minus), ~ (unary bit inversion)
5  ^
6  *, /, DIV, %, MOD
7  -, +
8  <<, >>
9  &
10 |
11 = (comparison), <=>, >=, >, <=, <, <>, !=, IS, LIKE, REGEXP, IN
12 BETWEEN, CASE, WHEN, THEN, ELSE
13 NOT
14 AND, &&
15 XOR
16 OR, ||
17 = (assignment), :=
```

Con paréntesis se altera la precedencia



# EXPRESIONES

## ➤ Operador de rango BETWEEN

Se utiliza para seleccionar **registros que estén incluidos en un rango**. Se utiliza en fechas y números. Todos los elementos incluidos en el rango satisfacen la condición. Los valores indicados en el intervalo están también incluidos. Es decir, es inclusivo.

**columna BETWEEN *rangoInferiorIncluido* AND *rangoSuperiorIncluido***

Ejemplo:

```
SELECT titulo, num_paginas FROM libros WHERE num_paginas BETWEEN 50 and 358;
```

Titulo	Num_paginas
Int. Artificial	50
Turbo C++	125
Virus Informát.	50
Sist. Informac.	358



# EXPRESIONES

## ➤ Operador test de patrón LIKE:

Comprueba que la columna **cumpla una serie de características** (un patrón establecido).

En MySQL la búsqueda con LIKE no es case sensitive. Es decir, no distinguirá mayúsculas de minúsculas. En otros gestores si hay esta diferenciación.

***columna* LIKE '*patron*'**

- Caracteres comodines para utilizar con LIKE y NOT LIKE:
  - % → Comodín para un número indeterminado de caracteres (de 0 a N).
  - \_ → Comodín para un único carácter (1).



# EJEMPLO EXPRESIÓN LIKE

- Todos los libros que el título empiece por 'c'

```
SELECT titulo, num_paginas FROM libros WHERE titulo LIKE 'c%';
```

- Todos los libros que el título tenga en segunda posición una 'c'

```
SELECT titulo, num_paginas FROM libros WHERE titulo LIKE '_c%';
```

- Todos los libros que el título de dos letras tenga en segunda y última posición una 'c'

```
SELECT titulo, num_paginas FROM libros WHERE titulo LIKE '_c';
```

- Todos los libros que el título tenga alguna 'c'.

```
SELECT titulo, num_paginas FROM libros WHERE titulo LIKE '%c%';
```

- Todos los libros que el título tenga alguna 'c' a partir de la 3ª posición.

```
SELECT titulo, num_paginas FROM libros WHERE titulo LIKE '___%c%';
```



# EXPRESIONES

## ➤ Operador de pertenencia a conjuntos IN

Permite comprobar si una columna (o expresión) tiene un valor igual que cualquiera de los que están incluidos dentro del paréntesis. Dentro del paréntesis, aparecen de 1 a N elementos.

**columna IN (valor1, valor2, ...)**

Ejemplo:

```
SELECT titulo, num_paginas FROM libros WHERE num_paginas IN (40, 48, 125);
```

Titulo	Num_paginas
Concep. Y Dis.	48
Turbo C++	125





# FUNCIONES MATEMÁTICAS

Función	Ejemplo
ABS Devuelve el valor absoluto	SELECT abs(-15); → 15
MOD Resto de una división entera	SELECT mod(554, 10) → 4
RAND Valores aleatorios	SELECT rand(); → 0.80785042161585
ROUND Cálculo de redondeos	SELECT round(-1.78); → -2
SIGN Devuelve el signo	SELECT sign(-15); → -1
SQRT Cálculo de la raíz cuadrada	SELECT sqrt(4); → 2
TRUNCATE Elimina decimales	SELECT truncate (1.723,1); → 1.7



# FUNCIONES CON CADENAS

Función	Ejemplo
CHAR_LENGTH Cálculo de longitud de cadena en caracteres.	SELECT char_length('Hola') --> 4
CONCAT Concatena dos cadenas de caracteres.	SELECT concat('Ho', 'l', 'a') --> Hola
INSERT Inserta una cadena en otra.	SELECT insert('Edificio', 3, 4, 'What'); --> EdWhatio
INSTR Busca una cadena en otra.	SELECT instr('Edificio', 'ifi');-->3
REPLACE Busca una secuencia en una cadena y la sustituye por otra.	SELECT replace('Edificio', 'ifi', 'of'); --> Edofocio
REVERSE Invierte el orden de los caracteres de una cadena.	SELECT reverse('12345'); --> '54321'
STRCMP Compara cadenas.	SELECT STRCMP('text', 'text2'); → -1 SELECT STRCMP('text2', 'text'); → 1 SELECT STRCMP('text', 'text'); → 0



# FUNCIONES CON FECHAS

Función	Ejemplo
current_date() Fecha actual	SELECT current_date(); → 2017-12-17
current_time() Hora actual	SELECT current_time() → 10:40:07
now() Fecha y hora actuales	SELECT now(); → 2017-12-17 10:40:08
dateDiff(unafecha, otraFecha) Días entre dos fechas	SELECT dateDiff('2017-12-17', '2017-12-20'); → - 3
day(unafecha); El día de una fecha	SELECT day('2017-12-17'); → 27
dayName(unafecha); El nombre del día de una fecha	SELECT dayName('2017-12-17'); → Wednesday
month(unafecha); El mes de una fecha	SELECT month('2017-12-17'); → 12
monthName(unafecha); El nombre del mes de una fecha	SELECT monthName('2017-12-17'); → December
year(unafecha) ; El año de una fecha	SELECT year('2017-12-17'); → 2017

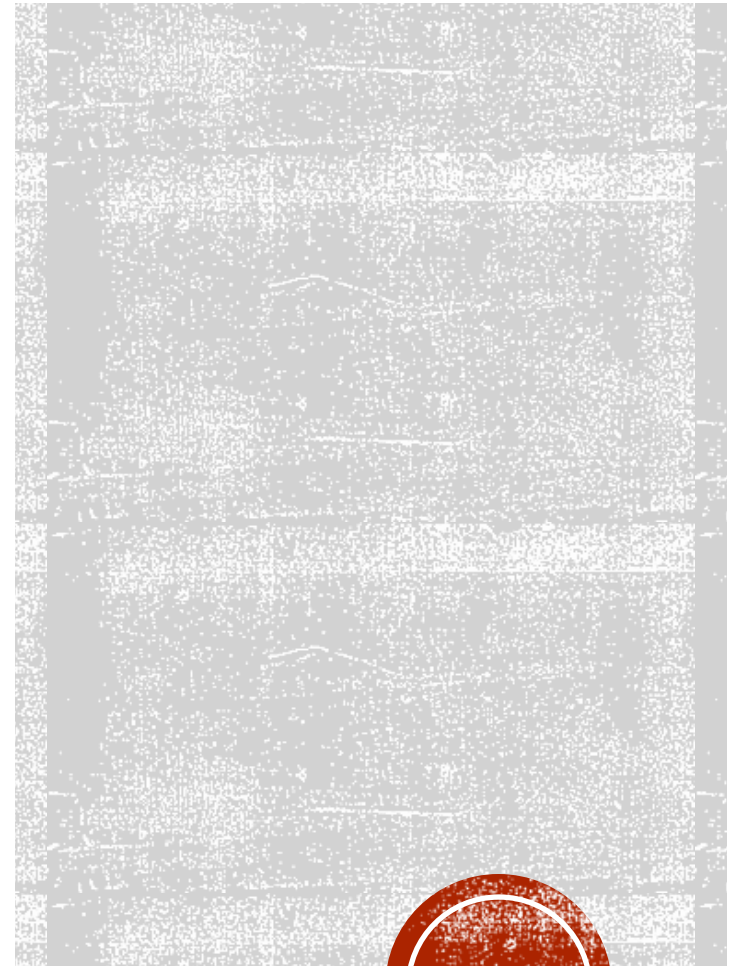
*Las fechas por defecto van en formato AAAA-MM-DD y en inglés*

*Cambiar de formato → date\_format(unafecha, "%d-%m-%Y")  
SELECT date\_format('2017-12-17', "%d-%m-%Y") → 17-12-2017*

*Cambiar de idioma para fechas → SET lc\_time\_names=es\_ES*



# TRATAMIENTO DE NULOS



# VALOR NULO (NULL) EN EL MODELO RELACIONAL

- Concepto.
  - Marca utilizada para representar **información desconocida, inaplicables, inexistente, no valida .....**
- Necesidad de los valores nulos:
  1. Por la existencia de tuplas con atributos desconocidos (año de edición de un libro).
  2. Necesidad de añadir nuevos atributos a una tabla ya existente (a esos atributos sin ningún valor se les da el valor nulo).
  3. Posibilidad de atributos inaplicables a ciertas tuplas.

Si tengo una tabla de ARTICULOS donde me puedo referir a COCHES, FRUTAS y LIBROS y entre otros atributos tengo EDITORIAL que sólo sirve para LIBROS, en el resto de filas, EDITORIAL tendrá el valor nulo.



# OPERACIONES CON VALORES NULOS

- =

NULL <> NULL

- AND

NULL AND TRUE → NULL

NULL AND FALSE → NULL

- OR

NULL OR TRUE → NULL

NULL OR FALSE → NULL

- NOT

NOT NULL → NULL

No se pueden utilizar un valor NULL para realizar ninguna comparación, AND, ...





# ORDENAMIENTO



# OPERACIONES CON VALORES NULOS

- IS NULL e IS NOT NULL

Permiten verificar si un campo es o no es nulo respectivamente. Es la única forma de trabajar con valores nulos. No se puede utilizar =.

IS NULL devuelve TRUE si la columna es nula, y falso en caso contrario. IS NOT NULL funciona justo al revés.

**columna IS NULL**

Ejemplo:

```
SELECT titulo, num_paginas FROM libros WHERE editorial IS NULL;
```

Titulo	Num_paginas
Virus Informát.	50



# ORDENAR RESULTADOS

- Los resultados de una sentencia **SELECT** se pueden ordenar por uno o varios criterios.
- Si se ordena por varios criterios, se ordena por el primero, y el caso de que varias filas tengan el mismo valor, se ordenan éstas por el segundo criterio, y así sucesivamente mientras haya criterios indicados.
- Hay que recordar que **MySQL es NO case sensitive**, y por tanto, para ordenar, **no distingue mayúsculas de minúsculas**.
- Para cada criterio, se puede indicar si se ordena de forma ascendente o descendente. Si no se indica nada, es de forma ascendente.
- Para ordenar, se utiliza las palabras **ORDER BY** después del filtro del **WHERE**.



# ORDENAR RESULTADOS

- Cada criterio de ordenación puede ser el nombre de la columna, la posición o una expresión.
  - La columna por la que se ordena es cualquier campo de la tabla indicada en el FROM, aunque no aparezca ni se utilice en el SELECT.
  - Si se indica una expresión, esta puede ser cualquiera de las ya vistas (si utiliza columnas, deben estar en la tabla indicada en el FROM).
  - Si se indica una posición, se refiere a la expresión (puede ser solo una columna) indicada en el SELECT que ocupa esa posición (se empieza a contar en 1).

**SELECT** [ **DISTINCT** ] *select\_expr* [, *select\_expr* ...]

**FROM** *table\_references*

[ **WHERE** *filtro* ]

[ **ORDER BY** {*col\_name* | *expr* | *position*} [**ASC** | **DESC**] , ... ]



# ORDER BY: EJEMPLOS

- Ordenar por título

```
SELECT titulo, num_paginas FROM libros ORDER BY titulo;
```

- Ordenar por número de páginas descendente

```
SELECT titulo, num_paginas FROM libros ORDER BY num_paginas DESC;
```

- Ordenar por número de páginas descendente, y en caso de iguales, por título ascendente

```
SELECT titulo, num_paginas FROM libros ORDER BY num_paginas DESC, titulo ASC;
```

- Ordenar por el cociente código dividido número de páginas

```
SELECT titulo, num_paginas FROM libros ORDER BY codigo / num_paginas;
```

- Ordenar por la segunda columna del select (num\_paginas)

```
SELECT titulo, num_paginas FROM libros ORDER BY 2;
```



# ORDER BY: EJEMPLOS

- Ordenar por la segunda columna y por la primera columna descendente  

```
SELECT titulo, num_paginas FROM libros ORDER BY 2, 1 DESC;
```
- Ordenar por la segunda columna del select (num\_paginas), y por el número de páginas descendente  

```
SELECT titulo, num_paginas FROM libros ORDER BY 2, num_paginas DESC;
```
- Filtrar los que tienen editorial, y ordenar por el título  

```
SELECT titulo, num_paginas FROM libros WHERE editorial IS NOT NULL ORDER BY titulo;
```
- Ordenar por la editorial. ¿Cómo se ordena un valor nulo?  

```
SELECT titulo, num_paginas FROM libros ORDER BY editorial
```

Los valores nulos, se consideran menores que cualquier otro valor. Si para un criterio, dos filas tienen valores nulos, se consideran que son iguales, y se utiliza el siguiente criterio. Es decir, solo para el order by, NULL = NULL.



# LIMITAR EL RESULTADO



# LIMITAR EL NÚMERO DE RESULTADOS A MOSTRAR

- Consiste en limitar el número de registros o filas devuelto por una consulta.
- Muchas veces aparece con un order by, pero no es necesario.
- Este tipo de filtro no es estándar, y su funcionamiento varía de un SGBD a otros.
- En MySQL se indica al final de la sentencia con

**LIMIT** [*desplazamiento* , ] *nfilas*

- **desplazamiento** es la fila desde la que se empieza a mostrar resultados.
- **nfilas** indica el número de filas a mostrar.
- Ejemplo (muestra 2 libros):

```
SELECT titulo, num_paginas FROM libros LIMIT 2;
```





# SINTAXIS DE SELECT

SELECT [ DISTINCT ] *select\_expr* [, *select\_expr* ...]

FROM *table\_references*

[ WHERE *filtro* ]

[ ORDER BY { *col\_name* | *expr* | *position* } [ASC | DESC] , ... ]

[ LIMIT [ *desplazamiento* , ] *nfilas* ]



# EJECUCIÓN DE UNA SENTENCIA SELECT

1. Se toma la tabla indicada en el FROM.
2. Para cada fila, se aplican las condiciones indicadas en el WHERE. Si en esa fila no se cumple (resultado falso) las condiciones, se elimina la fila del conjunto de resultados.
3. Para cada fila, se calculan las columnas que se devolverán, indicadas en la parte SELECT.
4. Se ordenan las filas según se indica en la parte ORDER BY
5. Solo se muestran las filas que cumplan la condición indicada en LIMIT





# UT5. CONSULTAS DE RECUPERACIÓN MONOTABLA EN SQL.

Módulo: BASES DE DATOS

Curso 2022/2023. 1º DAM

Ruth Lospitao Ruiz

Invoice

Invoice_id
Customer_id
Order_id
Product_id
Date_time
Status
Total
Remark

