

SQL

PRÁCTICA

Aquí encontrarás una recopilación de los contenidos y cómo aplicarlos dentro de los ejercicios de simulación.



SQL - PRÁCTICA

INSTRUCCIONES PREVIAS

Para esta práctica, el estudiante deberá descargar MySQL para realizar los ejercicios que se verán y explicarán en el vídeo tutorial. Para ello deberás seguir los siguientes pasos:

- En primer lugar, deberás dirigirte al apartado de **contenidos descargables** y buscar el **link a la “descarga de MySQL”**.
- En el apartado de **contenidos descargables** encontrarás también un archivo con las **“instrucciones para la instalación de MySQL”**. Una vez tengas instalada la herramienta, ¡podrás comenzar a practicar los ejercicios!

SQL - PRÁCTICA

1. IMPORTAR LA BASE DE DATOS EN MYSQL

Trabajas en una empresa de Renting de Coches, y debido al exceso de información por el aumento de la demanda, se te ha encargado la **creación de una base de datos digital mediante un gestor de bases de datos (MySQL)**.

Para poder realizar los distintos ejercicios que se proponen, lo primero que tenemos que hacer, es crear la BBDD sobre la que poder trabajar.

Para ello, debemos abrir la herramienta o aplicación MySQL, que previamente hemos tenido que instalar. Cuando intentamos acceder nos va a solicitar una "Password", en este caso, debemos introducir la password que pusimos en la instalación.

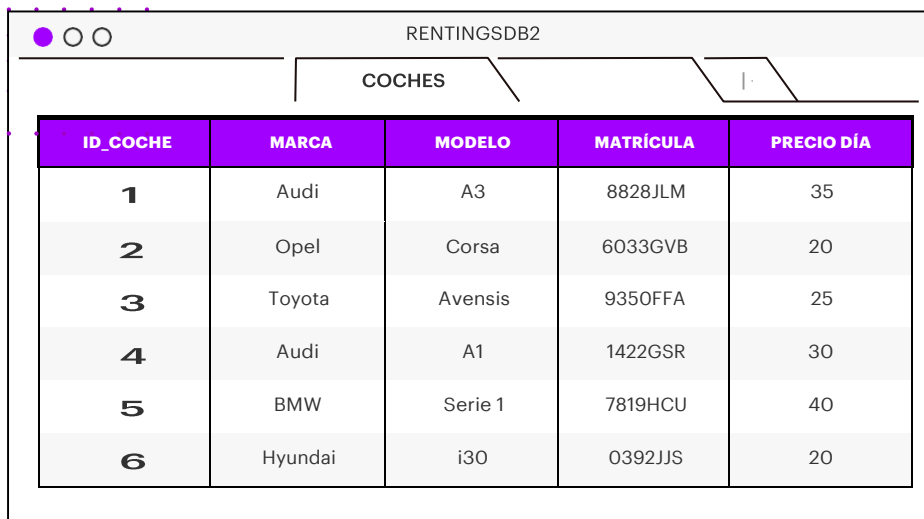
Una vez abierta la herramienta, importaremos el fichero [rentingsbd.sql](#) que se encuentra en el apartado de [contenidos descargables](#).

En el archivo tendrás la base de datos sobre la que haremos los ejercicios, que consta de las tres tablas que mostramos a continuación:

Tabla Clientes: Guarda la información personal de cada cliente

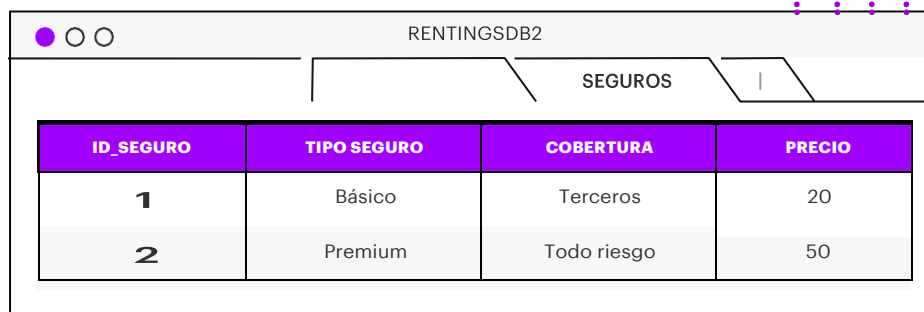
RENTINGSDB2					
CLIENTES					
ID_CLIENTE	NOMBRE	APELLIDO	DNI	FECHA CARNET	TELÉFONO
1	Beatriz	Alonso	22222222J	1995/10/15	666545757
2	Julia	Herrera	11111111H	1999/08/01	637451678
3	Alberto	López	70215788F	1992/06/05	643290777
4	Raquel	Ortiz	03411919Y	2000/11/20	673555666
5	Carlos	Gonzalez	50753246E	1993/03/03	649511558

Tabla Coches: Guarda la información de cada vehículo o coche de alquiler



ID_COCHE	MARCA	MODELO	MATRÍCULA	PRECIO DÍA
1	Audi	A3	8828JLM	35
2	Opel	Corsa	6033GVB	20
3	Toyota	Avensis	9350FFA	25
4	Audi	A1	1422GSR	30
5	BMW	Serie 1	7819HCU	40
6	Hyundai	i30	0392JJS	20

Tabla Seguros: Guarda la información de los distintos tipos de seguro disponibles



ID_SEGURO	TIPO SEGURO	COBERTURA	PRECIO
1	Básico	Terceros	20
2	Premium	Todo riesgo	50

2. CREATE, INSERT Y ALTER TABLE

A continuación, vamos a crear una tabla que registrará cada alquiler de vehículo realizado. Para ello, necesitaremos información de las tres tablas iniciales. En primer lugar, debes **crear una nueva tabla llamada Alquileres** usando el **comando CREATE** para guardar la relación de alquileres de la empresa. Las columnas que deberá tener la tabla son:

ID_ALQUILER

ID_CLIENTE

ID_COCHE

ID_SEGURO

FÓRMULA

CREATE TABLE nombre_tabla (
 nombre_columna TIPO DE VALOR,
 nombre_columna TIPO DE VALOR,
);

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
create table alquileres (  
    `alquilerId` int(5) not null auto_increment,  
    `clienteId` int(5) not null,  
    `cocheId` int(4) not null,  
    `seguroId` int(3) not null,  
    primary key (alquilerId) );
```

RENTINGSDB2

ALQUILERES

ID_ALQUILER	ID_CLIENTE	ID_COCHE	ID_SEGURO

Es importante acordarse de poner **''** alrededor de los **valores que sean texto**

Una vez creada la tabla con las columnas deseadas, es hora de **introducir los datos en la tabla**. Para ello utiliza el **comando INSERT**.

FÓRMULA

```
INSERT INTO nombre_tabla (columna1, columna2, columna3)
VALUES (valor1, valor2, valor3);
```

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
insert into alquileres (alquilerId,clienteId,cocheld,seguroid)
values (1,4,2,1);
```

```
insert into alquileres (alquilerId,clienteId,cocheld,seguroid)
values (2,2,3,1);
```

```
insert into alquileres (alquilerId,clienteId,cocheld,seguroid)
values (3,1,5,2);
```

```
insert into alquileres (alquilerId,clienteId,cocheld,seguroid)
values (4,3,1,1);
```

```
insert into alquileres (alquilerId,clienteId,cocheld,seguroid)
values (5,5,4,2);
```

RENTINGSDB2			
ALQUILERES			
ID_ALQUILER	ID_CLIENTE	ID_COCHE	ID_SEGURO
1	4	2	1
2	2	3	1
3	1	5	2
4	3	1	1
5	5	4	2

Añade dos columnas más a la tabla para que la información esté más completa. Con el **comando ALTER TABLE** añade la columna DIAS RENTING y PRECIO.

FÓRMULA

```
ALTER TABLE nombre_tabla  
ADD COLUMN nombre_columna TIPO DE VALOR;
```

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
alter table alquileres  
add column diasRenting int(3) not null,  
add column precio int(6) not null;
```

RENTINGSDB2					
ALQUILERES					
ID_ALQUILER	ID_CLIENTE	ID_COCHE	ID_SEGURO	DÍAS RENTING	PRECIO TOTAL
1	4	2	1		
2	2	3	1		
3	1	5	2		
4	3	1	1		
5	5	4	2		

3. UPDATE Y DELETE

Una vez que tenemos nuestras columnas, con el **comando UPDATE** introduciremos los valores en los diferentes campos. Para Beatriz Alonso, añade 8 días de renting y 180€ como precio.

FÓRMULA

```
UPDATE nombre_tabla  
SET columna2 = nuevo valor  
WHERE columna1 = valor1;
```

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
update alquileres  
set diasRenting = 8, precio=180  
where alquilerId=1;
```

RENTINGSDB2					
ALQUILERES					
ID_ALQUILER	ID_CLIENTE	ID_COCHE	ID_SEGURO	DÍAS RENTING	PRECIO TOTAL
1	4	2	1	8	180
2	2	3	1		
3	1	5	2		
4	3	1	1		
5	5	4	2		

Rellena el resto de la tabla de alquileres con la siguiente información:

- Julia alquila el coche 5 días por 145€.
- Alberto alquila el coche 3 días por 170€.
- Raquel alquila el coche 5 días por 195€.
- Carlos alquila el coche 7 días por 260€.

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
update alquileres  
set diasRenting = 5, precio=145  
where alquilerId=2;
```

```
update alquileres  
set diasRenting = 3, precio=170  
where alquilerId=3;
```

```
update alquileres  
set diasRenting = 5, precio=195  
where alquilerId=4;
```

```
update alquileres  
set diasRenting = 7, precio=260  
where alquilerId=5;
```

RENTINGSDB2					
ALQUILERES					
ID_ALQUILER	ID_CLIENTE	ID_COCHE	ID_SEGURO	DÍAS RENTING	PRECIO TOTAL
1	4	2	1	8	180
2	2	3	1	5	145
3	1	5	2	3	170
4	3	1	1	5	195
5	5	4	2	7	260

Si te equivocas introduciendo un valor, o ya **no quieres tener cierta información en tu tabla**, puedes usar el **comando DELETE para eliminarla**. Prueba a eliminar la fila de Julia Herrera que tiene el id=2

FÓRMULA

DELETE FROM nombre_tabla
WHERE columna1 = valor1;

Elimina un registro

DELETE FROM nombre_tabla;

Elimina una tabla completa

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
delete from alquileres  
where alquilerId=2;
```

RENTINGSDB2					
ALQUILERES					
ID_ALQUILER	ID_CLIENTE	ID_COCHE	ID_SEGURO	DÍAS RENTING	PRECIO TOTAL
1	4	2	1	8	180
3	1	5	2	3	170
4	3	1	1	5	195
5	5	4	2	7	260

SQL - PRÁCTICA

4. CONSULTAS CON SELECT FROM, WHERE, BETWEEN, AND, ORDER BY Y LIMIT

¡Has creado tu primera tabla! Ya puedes comenzar a hacer consultas, conocidas también como **queries**. Comencemos con una consulta básica. Con el **comando SELECT consulta** el listado de marcas de los coches que tenemos actualmente en la tabla Coches.

FÓRMULA

```
SELECT columna1 FROM nombre_tabla;
```

Este es comando a introducir y la consola mostrará lo siguiente:

```
select marca from coches;
```

Result Grid	
	marca
▶	Audi
	Opel
	Toyota
	Audi
	BMW
	Hyundai

Utiliza la **cláusula DISTINCT** para obtener los **valores únicos dentro de una columna**, eliminando así los duplicados.

FÓRMULA

```
SELECT DISTINCT columna1  
FROM nombre_tabla;
```

Este es el comando a introducir y la consola mostrará lo siguiente:

```
select distinct marca  
from coches;
```

Result Grid	
	marca
▶	Audi
	Opel
	Toyota
	BMW
	Hyundai

Para hacer consultas más concretas se utiliza la **cláusula WHERE**. Si le **añades un operador**, como podría ser "=", "<," ">.", **añades una condición** que los resultados deben cumplir. ¡Comencemos! Consulta los alquileres que hay en la BBDD de más de 185€.

FÓRMULA

```
SELECT columna1  
FROM nombre_tabla  
WHERE columna1 < 'valor'
```

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
select * from alquileres  
where precio > 185;
```

RENTINGSDB2					
ALQUILERES					
ID_ALQUILER	ID_CLIENTE	ID_COCHE	ID_SEGURO	DÍAS RENTING	PRECIO TOTAL
4	3	1	1	5	195
5	5	4	2	7	260

Otro operador que puedes utilizar es **BETWEEN**, el cual **limita la búsqueda dentro de un parámetro**. Por ejemplo, queremos consultar los alquileres que hay en la BBDD entre 4 y 7 días.

FÓRMULA

```
SELECT *  
FROM nombre_tabla  
WHERE columna1 BETWEEN 'valor' AND 'valor'
```

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
select * from alquileres  
where diasRenting between 4 and 7;
```

RENTINGSDB2					
ALQUILERES					
ID_ALQUILER	ID_CLIENTE	ID_COCHE	ID_SEGURO	DÍAS RENTING	PRECIO TOTAL
2	2	3	1	5	145
4	3	1	1	5	195
5	5	4	2	7	260

...

.....
.....
.....
.....

Si quieres hacer una **consulta muy específica** utiliza el **operador AND** que te permite **unir dos condiciones que deben cumplir los resultados**. Por ejemplo, sacar un listado de coches de alquiler de marca AUDI y que su precio de alquiler al día sea superior a 30€.

FÓRMULA

```
SELECT *  
FROM nombre_tabla  
WHERE columna1 > 'valor'  
AND columna2 = 'valor'
```

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
select * from coches  
where marca = 'AUDI' and  
precioDia > 30;
```

RENTINGSDB2				
COCHES				
ID_COCHE	MARCA	MODELO	MATRÍCULA	PRECIO DÍA
1	Audi	A3	8828JLM	35

Con el **operador ORDER BY** podemos **ordenar los resultados de nuestra búsqueda** (ascendente o descendente). Por ejemplo, sacar un listado de clientes ordenados por orden alfabético en función del apellido.

FÓRMULA

```
SELECT columna1, columna2  
FROM nombre_tabla  
WHERE columna1 > 'valor'  
ORDER BY columna2 ASC/DESC;
```

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
select * from clientes  
order by apellido ASC;
```

RENTINGSDB2					
CLIENTES					
ID_CLIENTE	NOMBRE	APELLIDO	DNI	FECHA CARNET	TELÉFONO
1	Beatriz	Alonso	22222222J	1995/10/15	666545757
5	Carlos	Gonzalez	50753246E	1993/03/03	649511558
2	Julia	Herrera	11111111H	1999/08/01	637451678
3	Alberto	López	70215788F	1992/06/05	643290777
4	Raquel	Ortiz	03411919Y	2000/11/20	673555666

Con la **cláusula LIMIT**, limitas el número de resultados **a un número específico de filas**. Siguiendo con el ejemplo anterior, limita los resultados a sólo 2 filas.

FÓRMULA

```
SELECT columna1, columna2  
FROM nombre_tabla  
LIMIT cifra;
```

El comando a introducir será el siguiente y la consola mostrará la correspondiente tabla:

```
select * from clientes  
order by apellido ASC  
limit 2;
```

RENTINGSDB2					
CLIENTES					
ID_CLIENTE	NOMBRE	APELLIDO	DNI	FECHA CARNET	TELÉFONO
1	Beatriz	Alonso	22222222J	1995/10/15	666545757
5	Carlos	Gonzalez	50753246E	1993/03/03	649511558

SQL - PRÁCTICA

5. FUNCIÓN COUNT

Ahora que dominas las consultas, veamos las funciones agregadas, las cuales se utilizan para realizar **cálculos sobre una tabla**. ¡Comencemos! Calcula cuántos clientes hay en la BBDD usando la **función COUNT**.

FÓRMULA

```
SELECT COUNT*  
FROM nombre_tabla;
```

El comando a introducir es el siguiente:

```
select count(*)  
from clientes;
```

Y la consola mostrará el resultado: **5**

Para un cálculo preciso, añade la **cláusula WHERE** a la función agregada. Calcula cuántos alquileres hay en la BBDD con un precio total de menos de 150€.

FÓRMULA

```
SELECT COUNT*  
FROM nombre_tabla  
WHERE columna1 = 'valor';
```

El comando a introducir es el siguiente:

```
select count(*)  
from alquileres  
where precio <150;
```

Y la consola mostrará el resultado: **1**

Calcula las ganancias **totales** de los alquileres con la **función agregada SUM()**.

FÓRMULA

```
SELECT SUM(columna1)
FROM nombre_tabla;
```

El comando a introducir es el siguiente:

```
select sum(precio)
from alquileres;
```

Y la consola mostrará el resultado: **950**

Calcula cuál es el **precio más alto** que está pagando un cliente por el alquiler de un coche con la función agregada **MAX()**. Después calcula el **mínimo** con la función agregada **MIN()**.

FÓRMULA

```
SELECT MAX(columna1)
FROM nombre_tabla

SELECT MIN(columna1)
FROM nombre_tabla
```

El comando a introducir es el siguiente:

```
select max(precio)
from alquileres;
```

Y la consola mostrará el resultado: **260**

```
select min(precio)
from alquileres;
```

Y la consola mostrará el resultado: **145**

Calcula **la media** de días que se alquilan los coches usando la **función agregada AVG()**.

FÓRMULA

```
SELECT AVG(columna1)
FROM nombre_tabla;
```

El comando a introducir es el siguiente:

```
select avg(diasRenting)
from alquileres;
```

Y la consola mostrará el resultado: **5,6000**

Añade la función agregada **ROUND()** al comando **AVG()** para **redondear el resultado anterior a cero decimales**.

FÓRMULA

```
SELECT ROUND(AVG(columna2,X)
FROM nombre_tabla;
```

El comando a introducir es el siguiente:

```
select round(avg(diasRenting),0)
from alquileres;
```

Y la consola mostrará el resultado: **6**

**FUNDACIÓN
ACCENTURE**

accenture