INICIAR SQL EN LA TERMINAL DE MAC

/Applications/MAMP/Library/bin/mysql --host=localhost -uroot -proot

INICIAR SQL EN LA CONSOLA DE WINDOWS

Para inciar mysql en Windows, primero tenemos que conectarlo al Path, para ello:

Ir a Configuración avanzada del sistema, Propiedades del sistema, Variables de entorno, dentro de la ficha Opciones avanzadas.

Se abrirá el cuadro de diálogo Variables de entorno.

En el panel inferior, Variables del sistema, y se encuentra una llamada Path. Pinchamos encima de ella y despues a Nuevo, y escribimos la ruta C:\Archivos de programa\MySQL\MySQL Server 5.6\bin\,12. El punto y coma que hay delante de la ruta, sirve para separar los diferentes elementos que componen la variable Path.

Ahora abrimos el símbolo del sistema (cmd) y escribimos mysql ya si la reconoce y lo abre

y seguidamente escribimos mysql -u root -p para acceder.

CREAR BASES DE DATOS, MOSTRARLAS Y USARLAS

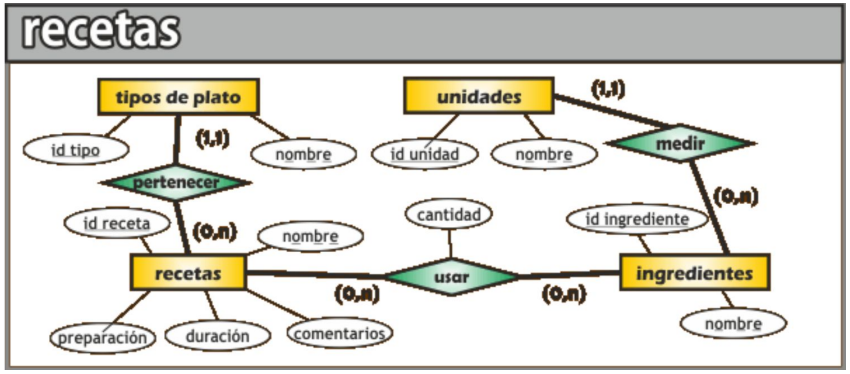
CREATE DATABASE nombre_base_de_datos;	**Creamos base de datos**
show databases;	**Para mostrar las bases de datos EXISTENTES**
use nombre_bd	**Para usar una base de datos ya creada. Es la única que no lleva PUNTO Y COMA al final**

CREAR TABLAS Y MOSTRARLAS	
CREATE TABLE as (id autores INT AUTO_INCREMENT NOT NULL, nombre VARCHAR(100) NOT NULL UNIQUE, PRIMARY KEY (id)) ENGINE=InnoDB;	Con esta sentencia de ejemplo estaría creando una tabla llamada autores, con un campo de enteros llamado id que se va a incrementar automáticamente con AUTO_INCREMENT, también un campo VARCHAR llamado nombre de hasta 100 caracteres y que no puede estar vacío con NOT NULL y además le puedo poner UNIQUE por si no querems que haya nombres repetidos, y para finalizar, le digo que el campo id va a ser la clave primaria. Por último, añadimos ENGINE=InnoDB;
SI TENEMOS UNA TABLA CON CLAVE FORÁNEA, ANTES DEBEMOS DE TENER CREADA LA TABLA A LA QUE HACE REFERENCIA. PARA DECLARAR UNA CLAVE FORÁNEA SE HACE ASI:	
CREATE TABLE volumenes (id INT AUTO_INCREMENT NOT NULL, id_libros INT NOT NULL, PRIMARY KEY (id), FOREIGN KEY (id_libros) REFERENCES libros (id)) ENGINE=InnoDB;	Donde ponemos todos los campos incluido el que va a ser la clave foránea, y luego al final indicamos cual es la clave foránea con: FOREIGN KEY (id_libros) REFERENCES libros (id), donde le estamos diciendo, que el registro id_libros es una clave foránea que relaciona con el registro id de la tabla libros
SI QUEREMOS DECIRLE A UN CAMPO YA CREADO QUE SEA CLAVE FORANEA LO HACEMOS ASI:	
ALTER TABLE nombre_tabla ADD FOREIGN KEY(nombre_campo) REFERENCES tabla_referencia (id)	Con esta sentencia, estariamos diciendole para la tabla nombre_tabla, que el campo nombre_campo, va a ser clave foránea y que esta referenciada con el campo id de la tabla, tabla_referencia
DECLARAR UN BOOLEAN	

deteriorado BOOLEAN NOT NULL DEFAULT 0	**y en INSERT INTO, no ponemos nada para el valor boolean. o bien podemos ponerle valor si queremos.**
show tables;	**Muestra las tablas existentes de la base de datos en uso**

MAS SOBRE CLAVES FORÁNEAS

CLAVES FORÁNEAS - FOREIGN KEY (id_libros) REFERENCES libros (id)



HABRA UNA CLAVE FORÁNEA SIEMPRE QUE HAYA ALGUNA RELACIÓN ENTRE LAS TABLAS.
(o lo que es lo mismo, cuando haya rombos verdes)

En el caso de las tabas tipos_de_plato y recetas, existe entremedias "pertenecer" y estan relacionadas de tal manera que tipos de plato tiene (1,1) y recetas (0,n) es decir, mramos los numeros finales y estan relacionadas 1:N, por tanto a la que tiene la n al final, hay que ponerle una clave foranea del otro, por tanto a recetas hay que ponerle una clave foranea que se llame id_tipos_de_plato. y que esta relacionada con el campo id de la tabla tipos_de_plato

CREATE TABLE recetas (id INT AUTO_INCREMENT NOT NULL,
nombre VARCHAR(100) NOT NULL UNIQUE,
preparacion VARCHAR (10000) NOT NULL,

duracion INT NOT NULL,
comentarios VARCHAR (500) NOT NULL, id_tipos_de_plato INT NOT NULL,
PRIMARY KEY (id),
FOREIGN KEY (id_tipos_de_plato) REFERENCES tipos_de_plato (id))
ENGINE=InnoDB;

En el caso de las tablas recetas e ingrediente, tenemos (0,n) y (0,n), osea N:N, por lo que SIEMPRE QUE TENGAMOS UNA RELACION N:M HAY QUE CREAR UNA TABLA APARTE, y en este caso creamos una tabla llamada "usar" y dentro hay que ponere una clave foranea de id_recetas y otra de id_ingredientes

CREATE TABLE usar (id_recetas INT,
id_ingredientes INT,
cantidad INT NOT NULL,
PRIMARY KEY (id_recetas, id_ingredientes),
FOREIGN KEY (id_recetas) REFERENCES recetas (id),
FOREIGN KEY (id_ingredientes) REFERENCES ingredientes (id))
ENGINE=InnoDB;

En el caso de las tablas unidades e ingredientes, tenemos (1,1) y (0,n), por tanto, a la que tiene la n, osea a ingredientes hay que ponerle una clave foranea llama id_unidades

VER UN EJEMPLO

https://drive.google.com/open?id=1wwcps_wxeM2BweCL20uG0b0ZKNdwZ6DjwlqZWj0SKbI

BORRAR BASES DE DATOS O TABLAS

DROP DATABASE nombre_bd	Borrar Base de datos
DROP TABLE nombre_tabla	Borrar Tabla
INSERTAR DATOS - REGISTROS	
INSERT INTO nombre_tabla (nombre, apellidos) VALUES ('Manuel', 'Gutierrez'); También se puede poner con el orden cambiado que el efecto es el mismo, podriamos ponerlo asi también; INSERT INTO nombre_tabla (apellidos, nombre) VALUES ('Gutierrez', 'Manuel');	Forma1: Nota: Al id como le hemos puesto auto_increment no hay que ponerle un valor. Por otro lado, SON comillas SIMPLES, y se ponen usando la misma tecla de interrogación ? pero sin pulsar mayúsculas
INSERT INTO nombre_tabla VALUES (DEFAULT,'Manuel', 'Gutierrez');	INSERTAR DATOS - Forma 2. Aquí no especificamos los campos, pero esto puede conllevar a equivocarnos mas facilmente Con este formato para el id hay que poner DEFAULT pero SIN comillas
INSERTAR MULTIPLES REGITROS	
INSERT INTO categories (category_name, min_limit) VALUES ('PAPELERÍA', 200000), ('ZAPATERÍA', 250000), ('TELAS', 700000);	
COMPROBAR DATOS INSERTADOS	
DESCRIBE nombre_tabla;	Para ver todas las características de las columnas

SELECT * FROM nombre_tabla;	Visualizando resultados en modo fila
AÑADIR UNA NUEVA COLUMNA A UNA TABLA DE UNA BASE DE DATOS YA CREADA	
ALTER TABLE autores ADD apellidos VARCHAR(200) NOT NULL AFTER nombre;	En este ejemplo, queremos añadir un nuevo campo llamado apellidos en la tabla autores, y la queremos poner debajo de un campo ya existente llamado nombre Si no hubieramos agregado el último comando, en este caso ALTER, colocaría el nuevo campo al final de los campos originales de la tabla. Si la situación deseada es distinta, por ejemplo deseamos crear el nuevo campo en la tabla y ubicarlo en la primera posición, entonces se usa la consulta SQL: ALTER TABLE nombre_tabla ADD ##nuevo_campo## FIRST;
ALTER TABLE nombre_tabla ADD COLUMN nombre_nueva_columna VARCHAR(255);	
AÑADIR MULTIPLES CAMPOS A UNA TABLA DE UNA BASE DE DATOS YA CREADA	
ALTER TABLE name_tabla ADD nombre_registro_nuevo VARCHAR(140) NOT NULL AFTER nombre_registro_ya_existente, ADD nombre_registro_nuevo_2 DATETIME NOT NULL AFTER nombre_registro_ya_existente;	
ELIMINANDO COLUMNA DE UNA TABLA	
ALTER TABLE nombre_tabla DROP COLUMN nombre_columna;.	
RENOMBRAR UNA COLUMNA, YA SEA EL NOMBRE, EL TIPO DE INT A VARCHAR, EL TAMAÑO DE (20) A (200) O AÑADIRLE O QUITAR NOT NULL	

ALTER TABLE nombre_tabla CHANGE COLUMN nombre_anterior_del_registro nombre_nuevo_del_registro CHAR(200);	Cambiar un campo o columna de nombre pero hay que decirle también que tipo va a ser (char, varchar). En caso de querer cambiar algo que no sea el nombre, entonces tanto en nombre_anterior_del_registro como en nombre_nuevo_del_registro ponemos el mismo nombre, y cambiamos solo lo que queramos cambiar
SI SOLO QUEREMOS CAMBIAR EL TIPO, TAMAÑO O AÑADIR O QUITAR NOT NULL, TAMBIEN PODEMOS USAR	
ALTER TABLE nombre_tabla MODIFY nombre_del_registro VARCHAR (200) NOT NULL;	
CAMBIAR UN CAMPO DE ORDEN	
ALTER table personas MODIFY COLUMN nombre VARCHAR (20) NOT NULL AFTER dni;	SI TENEMOS en una tabla persona un campo con estos datos: nombre VARCHAR (20) NOT NULL, y queremos ponerlo debajo de un campo que se llama dni, ponemos:
	tenemos que poner su tipo, su tamaño y si es null, sino tambien cambiara esas propiedades
MODIFICAR REGISTROS	
UPDATE nombre_tabla SET nombre_columna = 'valor_nuevo' WHERE id = 1; UPDATE personas SET nombre = 'Manuel' WHERE id = 1	Le decimo que a la tabla personas y en la columna llamada nombre, y al elemento que tiene un id=1 que nos ponga ahora el valor de alberto. El nombre viejo no es necesario ponerlo en el codigo para hacer este cambio), con que pongamos el id es suficiente

UPDATE nombre_de_la_tabla SET nombre_columna = 'valor_1', nombre_columna2 = 'valor_2' WHERE id=1, ide=2	Tambien podemos cambiar varios a la vez, en ese caso, UPDATE cambio1 SET nombre = 'Julian', apellidos = 'gomez' WHERE id=1, ide=2;
UPDATE nombre_de_la_tabla SET nombre_columna = valor_1, nombre_columna2 = valor_2 WHERE id=1,ide=2 ORDER BY columna (ASC|DESC) LIMIT n;	TAMBIEN PODEMOS LIMITAR LAS VECES QUE SE PUEDE ACTUALIZAR
BORRAR REGISTROS	
DELETE FROM nombre_de_tabla WHERE id=1; DELETE FROM personas WHERE id=1;	Borrar de la tabla personas el registro con id=1
GROUP BY - Para agrupar registros y evitar repetidos	
SELECT category_id, year_released FROM movies GROUP BY category_id, year_released;	
HAVING - Filtra los grupos resultantes	
SELECT * FROM movies GROUP BY category_id, year_released HAVING category_id = 8;	

Muestra todos los campos de la tabla movies, agrupando los campo category_id, year_released para no obtener repetidos, y filtrando con HAVING para que aparezcan solo los que tienen category_id = 8;	
OTRO EJEMPLO: **SELECT continente , SUM(superficie) FROM Paises GROUP BY continente HAVING SUM(superficie) >5000000;**	**En este caso, va a mostrar la columna continente, y la suma de la superficie de la tabla Paises, agrupandolos y a la vez con Having le decimos que del grupo, muestre solo los que la suma de la superficie sea mayor de 5000000.**
COUNT	
SELECT `gender`,COUNT(`membership_number`) FROM `members` GROUP BY `gender`;	**Con esta sentencia, va a agrupar el campo gender de la tabla members y va a contar cuantos seran hombre y mujeres, o de otra forma, los resultados se agrupan por cada valor de la columna gender y el número de las filas agrupadas se cuenta con la función COUNT.**
SUBCONSULTAS E INNER JOIN (para las subconsultas no olvidar poner paréntesis al final!!!!!!)	
EJEMPLO PARA LOS 3 TIPOS DE CONSULTAS: **Contar las cargas familiares del empleado Humberto Pons.**	

Forma 1 (Comparando tablas):
SELECT (dep_nom) AS 'Nombre carga familiar', Count(*) AS 'Numero de cargas familiares' FROM empleado,carga_f WHERE ci=eci and nombre='Humberto' and apellido='Pons' GROUP BY eci;

Forma 2 (Subconsultas):
SELECT (dep_nom) AS 'Nombre carga familiar', Count(*) AS 'Numero de cargas familiares' FROM carga_f WHERE eci = (SELECT ci FROM empleado WHERE nombre='Humberto' AND apellido='Pons') GROUP BY eci;

Primero se pone lo que se va a listar, y el segundo select ponemos la condicion de la otra tabla

Forma 3 (Con Join):
SELECT (dep_nom) AS 'Nombre carga familiar' FROM empleado INNER JOIN carga_f ON empleado.ci=carga_f.eci WHERE nombre='Humberto' AND apellido='Pons' GROUP BY carga_f.dep_nom;

OTRO EJEMPLO, en este caso para la consulta de tablas compartidas usamos GROUP BY
pero para obtener el restultado por subconsultas no es necesario usar GROUP BY,
sin embargo hay que usar IN para igualar las foraneas

Listar el nombre de los empleados que pertenezcan a más de un proyecto.

SELECT nombre, count(*) AS 'Numero proyectos en los que trabaja' FROM empleado, proyecto WHERE dnum=dno GROUP BY nombre HAVING COUNT(*)>1;

SELECT nombre FROM empleado WHERE dno IN (SELECT dnum FROM proyecto WHERE dnum>1);

CONSULTAS VARIAS

SELECT * FROM empleados;	

SELECT * FROM empleados WHERE cargoE = 'Secretaria';	
SELECT nomEmp,salEmp FROM empleados;	
SELECT * FROM empleados WHERE cargoE = 'Vendedor' ORDER BY nomEmp;	
SELECT (salEmp+500000) as 'Total a pagar' FROM empleados WHERE codDepto = '3000' ORDER BY nomEmp;	**Como necesito un alias para visualizar los resultados, pues uso as y le digo:** **SELECT (operacion_a_realizar) as 'Columna resultados' FROM empleados ....** **es decir, después del select pongo la operación que va a hacer entre paréntesis y seguidamente y antes del FROM pongo el as con su titulo,**
SELECT nomEmp FROM empleados WHERE comisionE>salEmp;	
SELECT nomEmp FROM empleados WHERE comisionE<=salEmp*0.3 ORDER BY nomEmp;	
SELECT nomEmp, cargoE FROM empleados WHERE nomEmp BETWEEN 'J' AND 'Z' ORDER BY nomEmp;	**Aquí le digo que me muestre las columnas nomEmp y cargoE de la tabla empleados, donde nomEmp empieza entre j y Z, y además lo ordeno alfabéticamente.**

SELECT nomEmp FROM empleados WHERE lower(nomEmp) NOT LIKE '%ma%' ORDER BY nomEmp;	**Hallar los empleados cuyo nombre no contiene la cadena "MA". Al poner la letra entre 2 signos de porcierto, significa que busque una cadena, pero si ponemos solo un astirisco al final indica que busque nombres que empiecen por esa letra. Con lower le decimos no importe si esta en mayusculas o minusculas, asi lo encuentra sin problema**
SELECT nombreDpto FROM departamentos WHERE nombreDpto NOT IN ('VENTAS','INVESTIGACION','MANTENIMIENTO');	**Obtener los nombres de los departamentos que no sean "Ventas" ni "Investigación" ni ‘MANTENIMIENTO'.**
SELECT nomEmp FROM empleados WHERE lower(nomEmp) LIKE '7' ORDER BY nomEmp;	**Obtener información de los empleados cuyo nombre tiene exactamente 11 caracteres**
SELECT nomEmp FROM empleados WHERE char_length(nomEmp) = 11 ORDER BY nomEmp;	**Obtener información de los empleados cuyo nombre tiene al menos 11 caracteres**
SELECT max(salEmp) as 'SALARIO MAS ALTO' FROM empleados;	**25. Mostrar el salario más alto de la empresa.**
SELECT max(nomEmp) as 'El ultimo empleado por orden alfabetico es:' FROM empleados;	**27. Mostrar el nombre del último empleado de la lista por orden alfabético.**
SELECT max(salEmp) as 'SALARIO MAS ALTO', min(salEmp) as 'SALARIO MAS BAJO', (max(salEmp)-min(salEmp)) as 'DIFERENCIA' FROM empleados;	**28. Hallar el salario más alto, el más bajo y la diferencia entre ellos.**
SELECT nomEmp, salEmp FROM empleados WHERE salEmp >= (SELECT avg(salEmp) FROM empleados) ORDER BY codDepto;	**31. Mostrar la lista de los empleados cuyo salario es mayor o igual que el promedio de la empresa. Ordenarlo por**

	departamento. Uso SELECT avg (), para calcular el valor medio (promedio) de una columna de tipo numérico.
SELECT * FROM recetas WHERE nombre LIKE 'a%' OR nombre LIKE 'e%' OR nombre LIKE 'i%' OR nombre LIKE 'o%' OR nombre LIKE 'u%';	
SELECT nomEmp, departamentos.nombreDpto FROM empleados, departamentos WHERE empleados.codDepto=departamentos.codDepto AND(departamentos.nombreDpto <> 'PRODUCCIÓN' AND cargoE='Secretaria' OR cargoE='Vendedor' AND salEmp>'1000000') ORDER BY fecIncorporacion;	**Obtener el nombre y el departamento de los empleados con cargo 'Secretaria' o 'Vendedor', que no trabajan en el departamento de "PRODUCCION", cuyo salario es superior a $1.000.000, ordenados por fecha de incorporación.** **En este caso, la tabla departamentos tiene una clave foranea llamada codDepto que relaciona con la tabla empleados**
SELECT nomEmp, empleados.codDepto, departamentos.nombreDpto FROM empleados, departamentos	**Listar los datos de los empleados cuyo nombre inicia por la letra 'M', su salario es mayor a $800.000 o reciben comisión y trabajan para el departamento de 'VENTAS'**

WHERE empleados.codDepto=departamentos.codDepto AND (departamentos.nombreDpto = 'VENTAS' AND nomEmp LIKE 'M%' AND (salEmp>'800000' or comisionE>'0')) ORDER BY nomEmp;	**Como dice u salario es mayor a $800.000 o reciben comisión, ambos los ponemos entre parentesis, ya que es una condicion de se da uno u otro mas otras condiciones.**
SELECT comisionE, COUNT(*) as 'Numero de empleados' FROM empleados GROUP BY comisionE HAVING comisionE>0 ;	**Mostrar cada una de las comisiones y el número de empleados que las reciben. Solo si tiene comisión.** **En este caso agrupamos las comisiones que son iguales, y contamos cuantas veces aparece cada una de ellas (empleados que reciben ese valor de comision), y de ellas con having le decimos que lo haga para las comisiones mayores de cero (Solo si tienen comision)**
SELECT codDepto, (sexEmp) as 'Sexo', COUNT(*) as 'Numero de empleados' FROM empleados GROUP BY codDepto, sexEmp;	**Mostrar el número de empleados de sexo femenino y de sexo masculino, por departamento.** **Muestro el codDepto y un alias con el sexo, y con GROUP, se agrupan los resultados de codDepto y el sexo (H o F), y con COUNT contamos el número de empleados que son hombres y mujeres**
SELECT empleados.jefeID, departamentos.codDirector, COUNT(*) as 'Num Empleados' FROM empleados, departamentos WHERE empleados.jefeID=departamentos.codDirector GROUP BY jefeID HAVING COUNT (*)>=2;	**33. Mostrar el código y nombre de cada jefe, junto al número de empleados que dirige. Solo los que tengan mas de dos empleados (2 incluido).**

SELECT departamentos.nombreDpto, sum(empleados.salEmp) FROM empleados, departamentos WHERE empleados.codDepto=departamentos.codDepto GROUP BY departamentos.nombreDpto ORDER BY sum(empleados.salEmp) desc LIMIT 1;	**. Mostrar el nombre del departamento cuya suma de salarios sea la más alta, indicando el valor de la suma.** **Agrupo el nombre de departamento de la tabla departamentos, Junto a la suma de salarios de la tabla empleados, y lo ordeno de modo descendente para que muestre el mayor a menor y con LIMIT le digo que solo muestre el primer valor descendente, es decir, el valor mas alto**
SELECT nombre, apellido, sum(horas) FROM empleado, trabaja_en WHERE ci=eci and eci='333445555';	**5.- Sumar el número de horas que ha trabajado el empleado Humberto Pons en los diferentes proyectos.** **Tenia puesto WHERE empleado.ci=trabaja_en.eci, pero he comprobado con WHERE ci=eci, y funciona igualmente. En caso de llamarse igual ambas claves, entonces si ponemos delante el nombre de la tabla con un punto** **Nos vamos a la tabla trabaja_en, le decimos que sume las horas, y como Humberto Pons tiene un eic=333445555, con where se lo digo**
SELECT nombre, apellido, (dep_nom) AS 'Nombre carga familiar', Count(*) AS 'Numero de cargas familiares' FROM empleado,carga_f WHERE ci=eci and nombre='Humberto' and apellido='Pons' GROUP BY eci;	**14.- Contar las cargas familiares del empleado Humberto Pons.**
SELECT nombre, apellido FROM empleado WHERE MONTH(fecha_n)=3;	**19.- Listar todos los empleados nacidos en el mes de marzo.**

	Con WHERE MONTH(fecha_n)=3; le decimos que para el valor de fecha_n, que su mes sea el 3 (Como nos dice MArzo, ponemos 3)
SELECT nombre, apellido FROM empleado WHERE YEAR(fecha_n) BETWEEN 1959 AND 1961;	**24.- Listar todos los empleados que hayan nacido entre el año 1959 y 1961.**
SELECT nombre, apellido, COUNT(*) FROM empleado, carga_f WHERE eci=ci AND carga_f.sexo='F' GROUP BY eci HAVING COUNT(*)>1;	**20.- Listar el nombre de los empleados que tienen más de 1 carga familiar de sexo femenino.**
SELECT dep_nom FROM empleado, carga_f WHERE ci=eci AND LENGTH(dep_nom) > (6)	**45.- Listar todas las cargas familiares del empleado Juan Polo que tengan 6 caracteres.**
SELECT nombre, sum(horas) as 'Horas' FROM empleado, trabaja_en, proyecto WHERE ci=eci AND pno=pnumero GROUP BY ci HAVING sum(horas)<20;	**40.- Listar el nombre de los empleados y del proyecto en el que han trabajado menos de 20 horas.**
SELECT pnombre, sum(horas) AS 'Horas trabajadas' FROM proyecto,trabaja_en WHERE pno=pnumero GROUP by pnombre HAVING sum(horas)>25;	**38.- Listar los proyectos cuyo total supere las 25 horas.**

SELECT pnombre, sum(horas) AS 'Horas trabajadas' FROM proyecto,trabaja_en WHERE pno=pnumero GROUP by pnumero;	**37.- Calcular el total de horas trabajadas en cada proyecto.**
SELECT salario, COUNT(*) AS 'Salarios diferentes' FROM empleado GROUP by salario;	**36.- Contar los salarios diferentes.**
ERRORES	
SI LA CONSOLA NOS DICE:	
Subquery returns more than 1 row	**Esto sucede cuando hacemos una subconsulta, y hay que poner un ON de esta forma:** **SELECT (dnombre) AS 'Nombre departamento' FROM departamento WHERE dnumero IN (SELECT dno FROM empleado WHERE salario between 2500 AND 4300);** **En este caso, después del WHERE no igualamos, sino que ponemos IN**
Operand should contain 1 column(s)	**Esto sucede cuando hacemos una subconsulta, y solo va a devolver un valor pero hemos puesto 2 argumentos en el segundo SELECT**

IMPORTAR BASES DE DATOS
EN WINDOWS:
source C:\Users\luisjaviergomezpriet\Desktop/database.sql
Donde reemplazamos database.sql por nombre de la base de datos a importar, por ejemplo;
SI DA ERROR ENTONCES HACER ESTO DE CREAR UNA BASE DE DATOS ANTES
Para importar una base de datos, antes tenemos que crear una base de datos vacía, por tanto, creamos una base de datos con:
CREATE DATABASE luis_bd;
Ahora comprobamos que se ha creado
show databases;
Ahora IMPORTANTE, salimos de la Terminal o consola (en windows) y la abrimos de nuevo. Si no hacemos esto, no podremos importar.
Ahora vamos a importar una base de datos dentro de la nueva creada, por tanto la llamamos primero
use nombre_bd
y ahora la importamos:
source RUTA/database.sql
Donde reemplazamos database.sql por la ruta al backup., por ejemplo;

source C:\Users\luisjaviergomezpriet\Desktop/database.sql

O en windows, partiendo desde C:/ , aunque la ruta relativa a tu instalación de MySQL también debería funcionar.

EN MAC:

En Mac OS X, ponemos en el escritorio, el archivo de la base de datos que queremos importar, despues escribios en la terminal, source y arrastramos el archivo sql en la terminal para obtener la ruta completa y le damos a enter, seria algu similar a esto

source /Users/luisjaviergomezprieto/Desktop/nombre_base_datod.sql

Es importante que pongamos antes el archivo en el escritorio, asi nos cogerá una ruta sin errores ya que cuando le ponemos una ruta de carpetas cuyos nombres tienen espacios, no lo reconoce bien y no la importa.

Ahora comprobamos con show databases;

EXPORTAR BASES DE DATOS

EN WINDOWS:

Abrimos cmd, el cual mostrará por defecto esta, C:\Users\luisjaviergomezpriet>

entonces pegamos a continuacion este codigo:

mysqldump -u root -p nombre_bd > C:\Users\luisjaviergomezpriet\Desktop\demo.sql

(no hay que poner el punto y coma)

Ahora nos vamos a la carpeta C:\Users\luisjaviergomezpriet\Desktop\ y ahi se abrá guardado
EN MAC (aun no me funciona por consola, solo con sequel pro)
Salimos de mysql con exit
Ahora nos aparecera la Terminal solo esto:
iMac-de-Luis:~ luisjaviergomezprieto$, pues a continuación le ponemos este otro codigo de abajo
$ mysqldump -u root -p nombre_db > /Users/luisjaviergomezprieto/Desktop/bd/nombre_base_datos.sql;
AHORA NOS DIRA QUE NO HA ENCONTRADO EL COMANDO SIN EMBARGO SI QUE LO HA GUARDADO