

# Tarea HITO2

Base de Datos II  
Unifranz Sede el Alto  
Hito2

Nombre: Daniel Escobar  
Saravia





# ¿A que se refiere cuando se habla de bases de datos relacionales?



Cuando hablamos de base de datos relacionales nos referimos a bases de datos que tienen tablas que están conectadas unas con otras.





# ¿A que se refiere cuando se habla de bases de datos no relacionales?



Cuando hablamos de bases de datos no relacionales nos referimos a una base de datos en la cual no hay tablas que están relacionadas entre sí

## Non Relational DB

Accommodation

Name 1	Review 1
	Review 2
	Review 3
Name 2	Review 1
	Review 2
	Review 3



# ¿Qué es MySQL y MariaDB?. Explique si existen diferencias o son iguales, etc.



Ambos son gestores de bases de datos relacionales.

Tienen el mismo objetivo que es permitir que varios usuarios tengan acceso a los datos permitiendo gestionar muy fácilmente la base de datos.

Aunque MariaDB es una bifurcación de MySQL MariaDB tiene licencia GPL mientras que MySQL tiene un enfoque de doble licencia. Cada mango se acumula de una manera diferente. MariaDB soporta muchos motores de almacenamiento diferentes.







## Funciones de Agregado

Función

Descripción

AVG

Utilizada para calcular el promedio de los valores de un campo determinado

COUNT

Utilizada para devolver el número de registros de la selección

SUM

Utilizada para devolver la suma de todos los valores de un campo determinado

MAX

Utilizada para devolver el valor más alto de un campo especificado

MIN

Utilizada para devolver el valor más bajo de un campo especificado

# ¿Qué son las funciones de agregación?

Una función de agregación es una función que resume las filas de un grupo en un solo valor.



# ¿Qué llegaría a ser XAMPP?

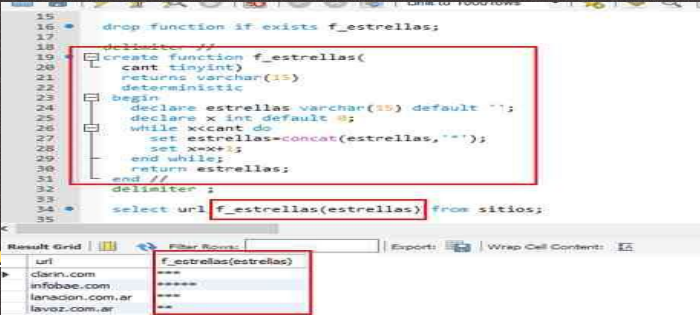
XAMPP es una herramienta de desarrollo que te permite probar el desarrollo web basado en PHP en nuestros ordenadores sin necesidad de tener acceso a internet.





# ¿Cual es la diferencia entre las funciones de agregación y funciones creados por el DBA? Es decir funciones creadas por el usuario.

Las funciones de agregación son funciones que ya están predefinidas en el programa y resume las filas de un grupo en un solo valor y las funciones creadas son funciones que el usuario creará para determinada tarea y puede generar más de un valor.



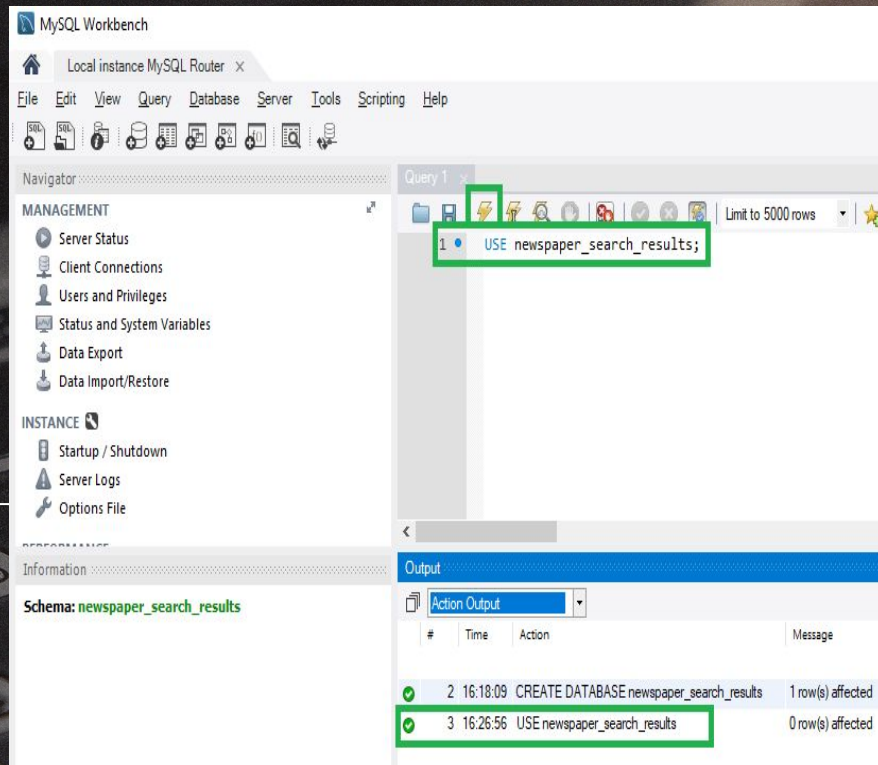
The screenshot shows a SQL script in a text editor. The script defines a function `f_estrellas` that takes a tinyint parameter and returns a varchar. The function uses a loop to concatenate asterisks. Below the script, the results of a query `select url, f_estrellas(estrellas) from sitios;` are shown in a table with two columns: `url` and `f_estrellas(estrellas)`. The results show four rows with different URLs and their corresponding star counts.

Funciones de Agregado	
Función	Descripción
AVG	Utilizada para calcular el promedio de los valores de un campo determinado
COUNT	Utilizada para devolver el número de registros de la selección
SUM	Utilizada para devolver la suma de todos los valores de un campo determinado
MAX	Utilizada para devolver el valor más alto de un campo especificado
MIN	Utilizada para devolver el valor más bajo de un campo especificado



# ¿Para qué sirve el comando USE?

El comando **USE** nos sirve para entrar y manipular una base de datos o una tabla la cual nosotros deseemos



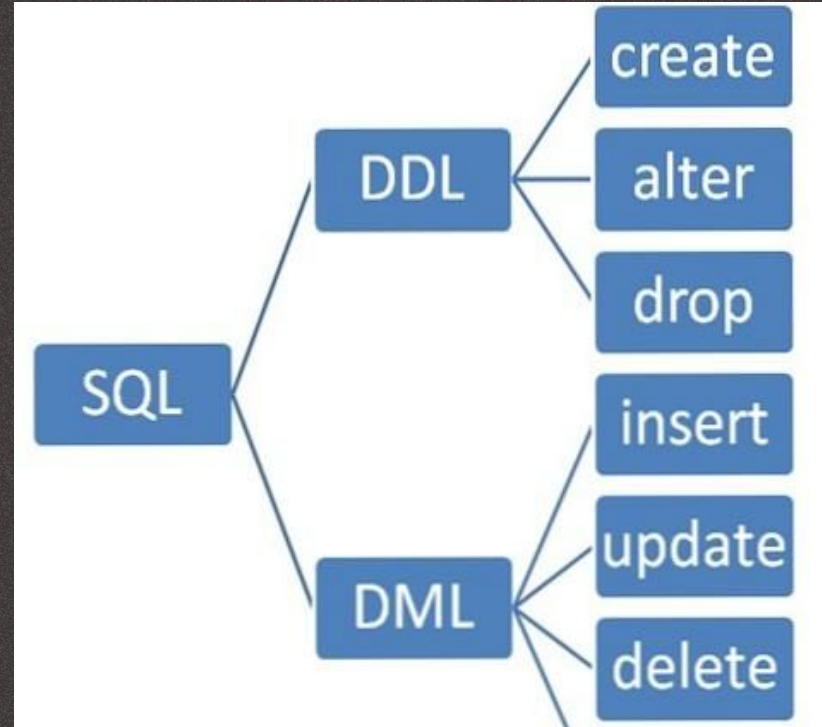


# ¿Que es DML y DDL?



DDL.- Es un lenguaje que permite a los programadores de un sistema gestor de base de datos, definir las estructuras que almacenarán los datos así como los procedimientos o funciones que permitan consultarlos.

DML.- permite a los usuarios introducir datos para posteriormente realizar tareas de consultas o modificación de los datos que contienen las Bases de Datos.





# ¿Qué cosas características debe de tener una función? Explique sobre el nombre, el return, parametros, etc



Para poder crear una función esta debe cumplir con ciertas características las cuales son:

- **Nombre.-** La función debe de tener un nombre para poder llamarlo cuando lo necesitemos.
- **Return.-** Una función siempre devolverá un valor de salida asociado al nombre de la función.
- **Parámetros.-** En una función todos los parámetros son de entrada por lo que no se necesita poner IN como palabra reservada delante del nombre.
- **Declare.-** En una función podemos declarar variables locales con la palabra reservada DECLARE.

```
drop function if exists f_tipositio;

delimiter //
create function f_tipositio(
cantidad int)
returns varchar(20)
deterministic
begin
case
when cantidad<20000000 then
return 'tráfico bajo';
when cantidad=20000000 and cantidad<40000000 then
return 'tráfico medio';
when cantidad=40000000 then
return 'tráfico alto';
end case;
end //
delimiter ;

select url,f_estrellas(estrellas), cantpaginas, f_tipositio(cantpaginas) from sitios;
```

url	f_estrellas(estrellas)	cantpaginas	f_tipositio(cantpaginas)
clarin.com	***	42000000	tráfico alto
infobae.com	****	33000000	tráfico medio
lanacion.com.ar	***	17000000	tráfico bajo
levoz.com.ar	**	25000000	tráfico medio



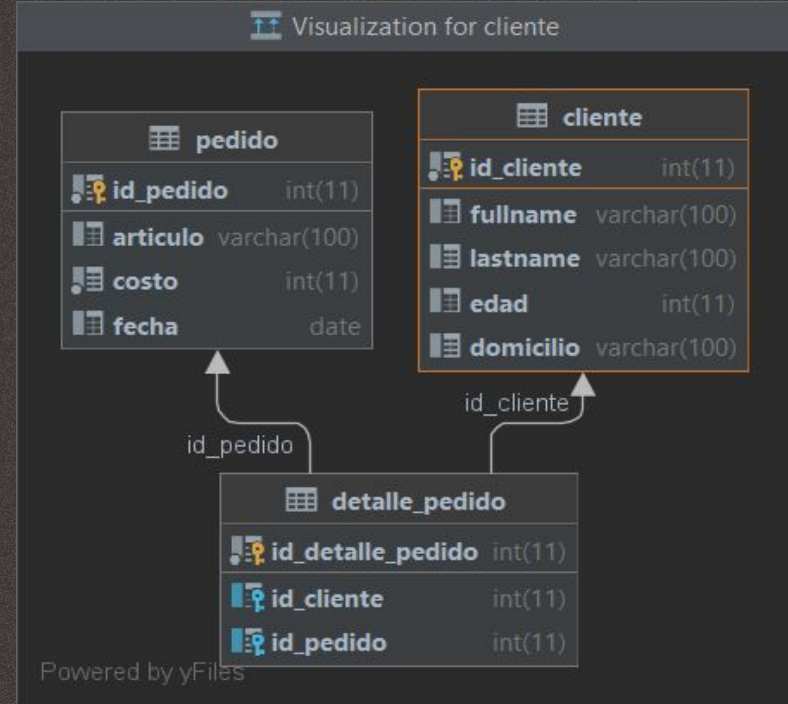
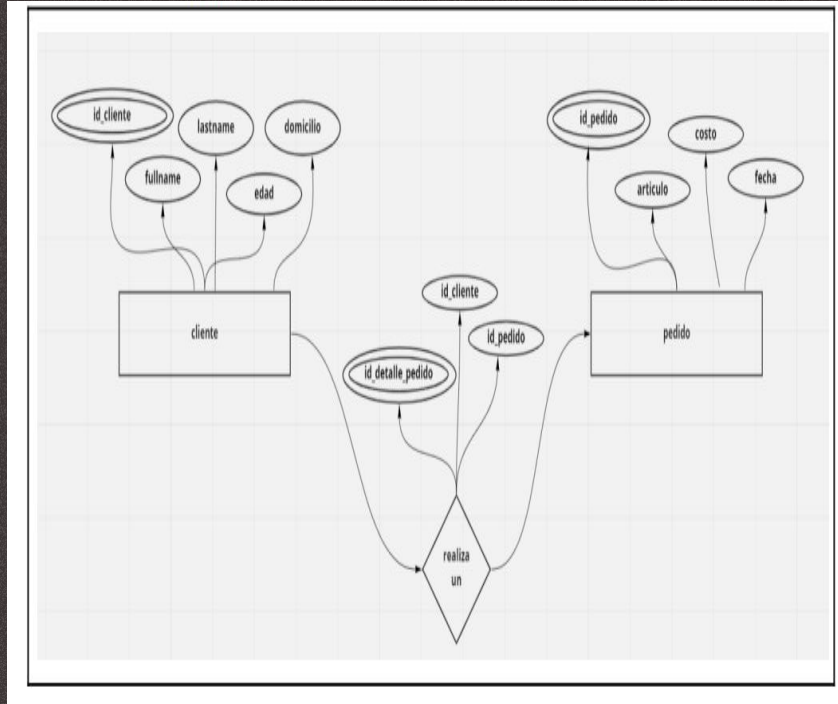
# ¿Cómo crear, modificar y cómo eliminar una función?

Se debe utilizar el lenguaje DDL con las siguientes palabras reservadas:

- CREATE.- Nos permitirá crear la función.
- ALTER.- Nos permitirá modificar la función.
- DELETE.- Nos permitirá eliminar la función.



# Crear las tablas y 2 registros para cada tabla para el siguiente modelo ER.





# Codigo generado



```
CREATE DATABASE
POLLOS COPA;
USE POLLOS COPA;
CREATE TABLE cliente
(
    id_cliente INTEGER
PRIMARY KEY NOT NULL,
    fullname
VARCHAR(100),
    lastname
VARCHAR(100),
    edad
INTEGER,
    domicilio
VARCHAR(100)
);
CREATE TABLE pedido
(
    id_pedido
INTEGER PRIMARY KEY
NOT NULL,
```

```
    articulo    VARCHAR(100),
    costo       INTEGER NOT
NULL,
    fecha       DATE
);
CREATE TABLE detalle_pedido
(
    id_detalle_pedido
INTEGER PRIMARY KEY NOT NULL,
    id_cliente  INTEGER,
    id_pedido   INTEGER,
    FOREIGN KEY
(id_cliente) REFERENCES
cliente(id_cliente),
    FOREIGN KEY (id_pedido)
REFERENCES pedido (id_pedido)
);
```

```
INSERT INTO cliente (id_cliente,
fullname, lastname, edad, domicilio)
VALUES
(111, 'Jose', 'Gonzales Veliz', 20,
'Plaza La Paz'),
(122, 'Maria', 'Mavir Uria', 22,
'Puente Rio Seco');
INSERT INTO pedido (articulo, costo,
fecha, id_pedido) VALUES
('Pollo a la
canasta', 35, '2022-03-30', 11),
('Amburguesa
doble', 25, '2022-04-01', 12);
INSERT INTO detalle_pedido
(id_detalle_pedido, id_cliente,
id_pedido) VALUES
(121, 111, 11),
(131, 122, 12);
```



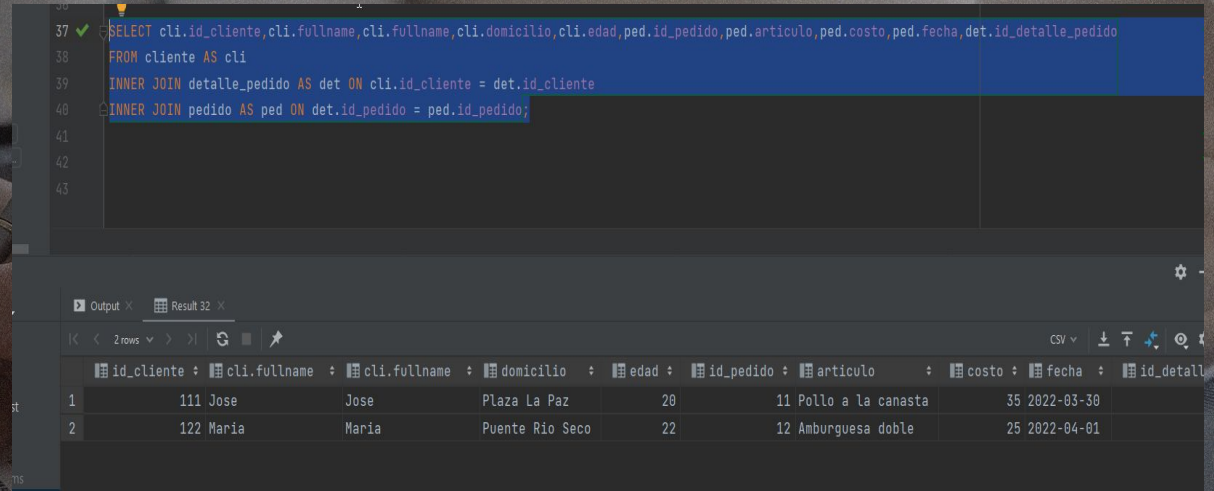
# Crear una consulta SQL en base al ejercicio anterior.

```
SELECT
cli.id_cliente,cli.fullname,cli.fullname,cli.domicilio,cli.edad,ped.id_pedido,
ped.articulo,ped.costo,ped.fecha,det.id_detalle_pedido

FROM cliente AS cli

INNER JOIN detalle_pedido
AS det ON cli.id_cliente =
det.id_cliente

INNER JOIN pedido AS ped
ON det.id_pedido =
ped.id_pedido;
```



The screenshot shows a SQL IDE with a query editor and an output window. The query in the editor is:

```
SELECT cli.id_cliente,cli.fullname,cli.fullname,cli.domicilio,cli.edad,ped.id_pedido,ped.articulo,ped.costo,ped.fecha,det.id_detalle_pedido
FROM cliente AS cli
INNER JOIN detalle_pedido AS det ON cli.id_cliente = det.id_cliente
INNER JOIN pedido AS ped ON det.id_pedido = ped.id_pedido;
```

The output window displays the results of the query, showing 2 rows. The columns are: id\_cliente, cli.fullname, cli.fullname, domicilio, edad, id\_pedido, articulo, costo, fecha, and id\_detalle\_pedido.

	id_cliente	cli.fullname	cli.fullname	domicilio	edad	id_pedido	articulo	costo	fecha	id_detalle_pedido
1	111	Jose	Jose	Plaza La Paz	20	11	Pollo a la canasta	35	2022-03-30	
2	122	Maria	Maria	Puente Rio Seco	22	12	Amburguesa doble	25	2022-04-01	





**GRACIAS POR SU  
ATENCION!!!**

