



# LAB - PROCESUAL PARA HITO 2

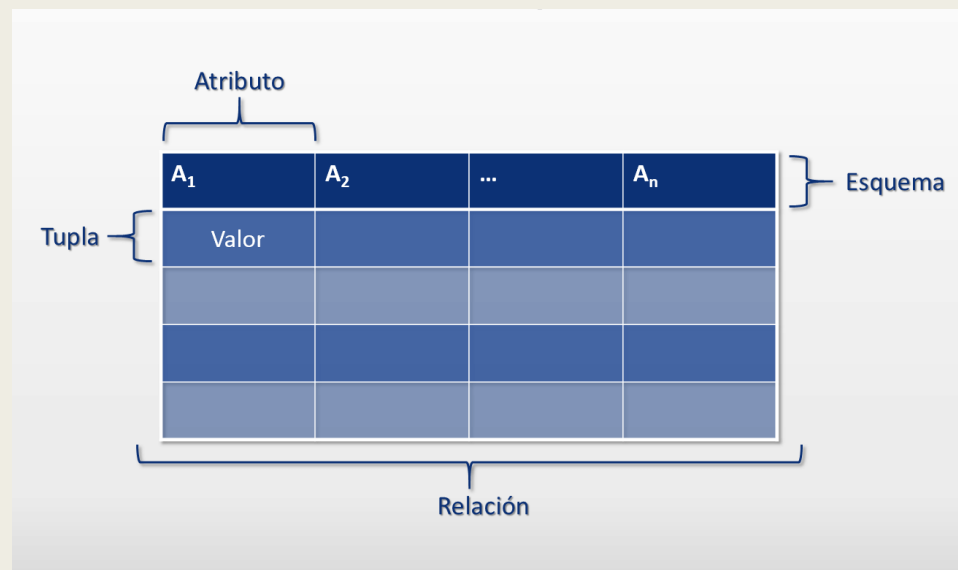
**Docente:** Ing. William Roddy Barra Paredes

**Nombre completo:** Ludwing Antoni Vargas Ibarra

**Semestre:** III/2022

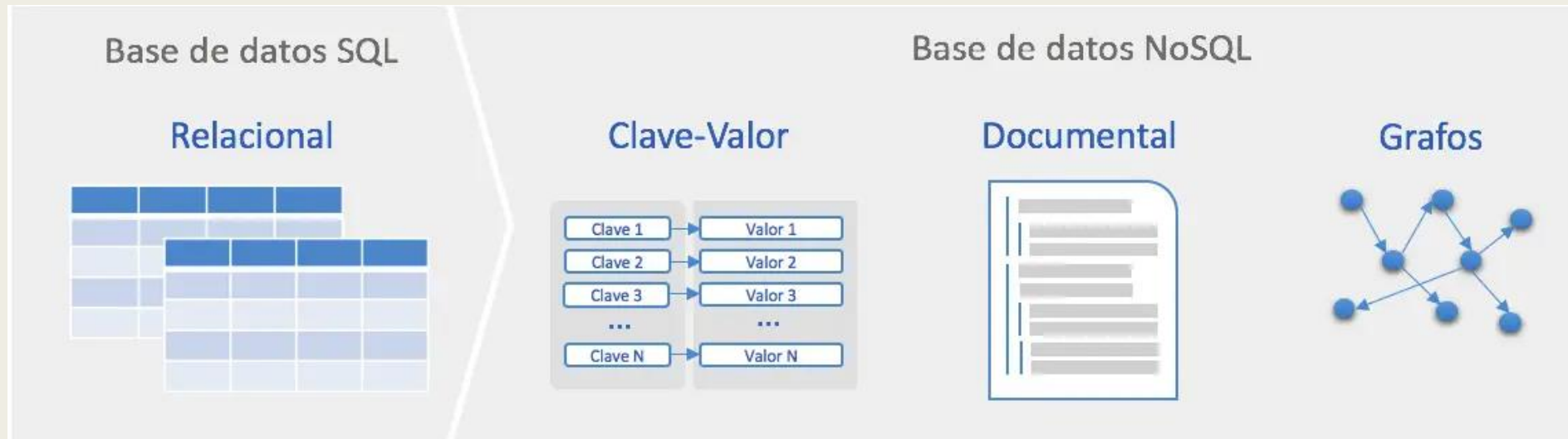
# ¿A que se refiere cuando se habla de bases de datos relacionales?

- Es un conjunto de tablas (relacionadas bidimensionalmente), similares a las tablas de una hoja de calculo, formadas por filas (registros) y columnas (campos)
- Ejemplo:



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

- Es una amplia clase de sistemas facilitando un crecimiento horizontal, enfocándose en rendimiento mas que gestión de datos.
- Ejemplo:



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

- MySQL es un sistema de gestión de base de datos relacional de código abierto, multihilo y multiusuario.

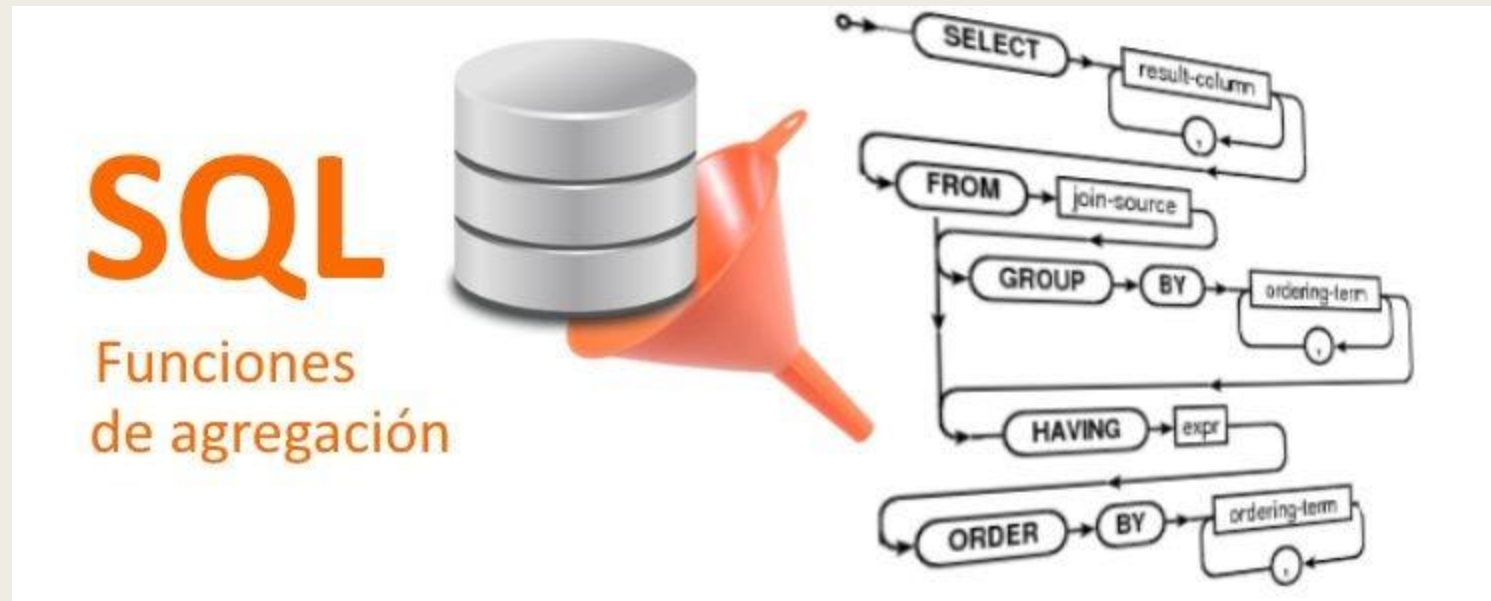


- MariaDB es un potente sistema de base de datos objeto-relacional de código abierto.



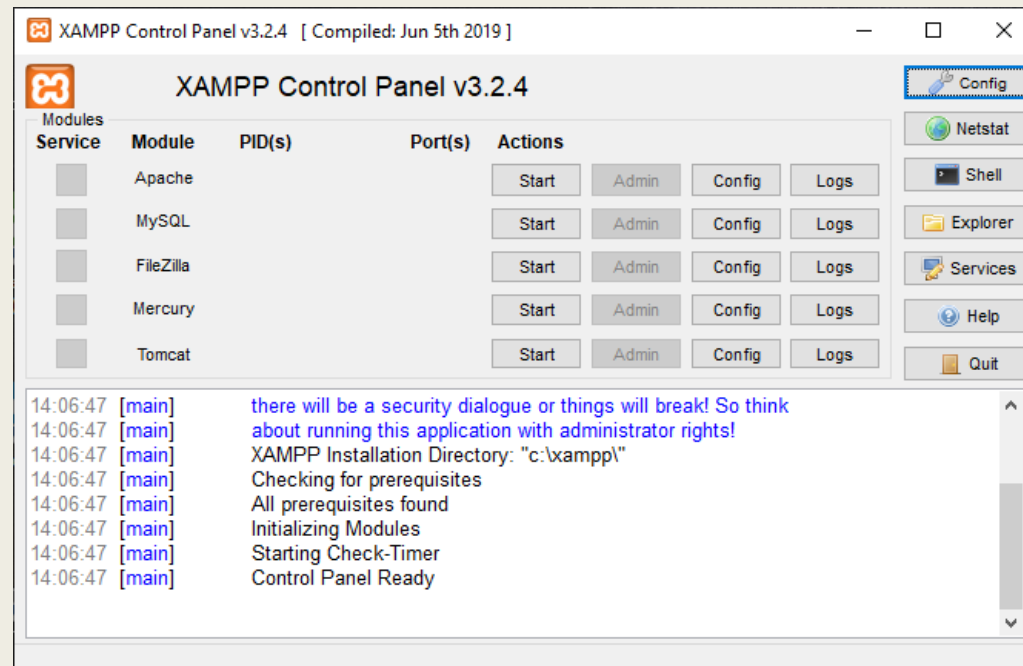
# ¿Qué son las funciones de agregación?

- Las funciones de agregación en SQL nos permiten efectuar operaciones sobre un conjunto de resultados, pero devolviendo un único valor agregado para todos ellos. Es decir, nos permiten obtener medias, máximos, etc... sobre un conjunto de valores.



# ¿Qué llegaría a ser XAMPP?

- XAMPP es un paquete formado por un servidor web Apache, una base de datos MySQL y los intérpretes para los lenguajes PHP y Perl. De hecho su nombre viene de hay, X (para cualquier sistema operativo), A (Apache), M (MySQL), P (PHP) y P (Perl). XAMPP es independiente de plataforma y tiene licencia GNU GPL.

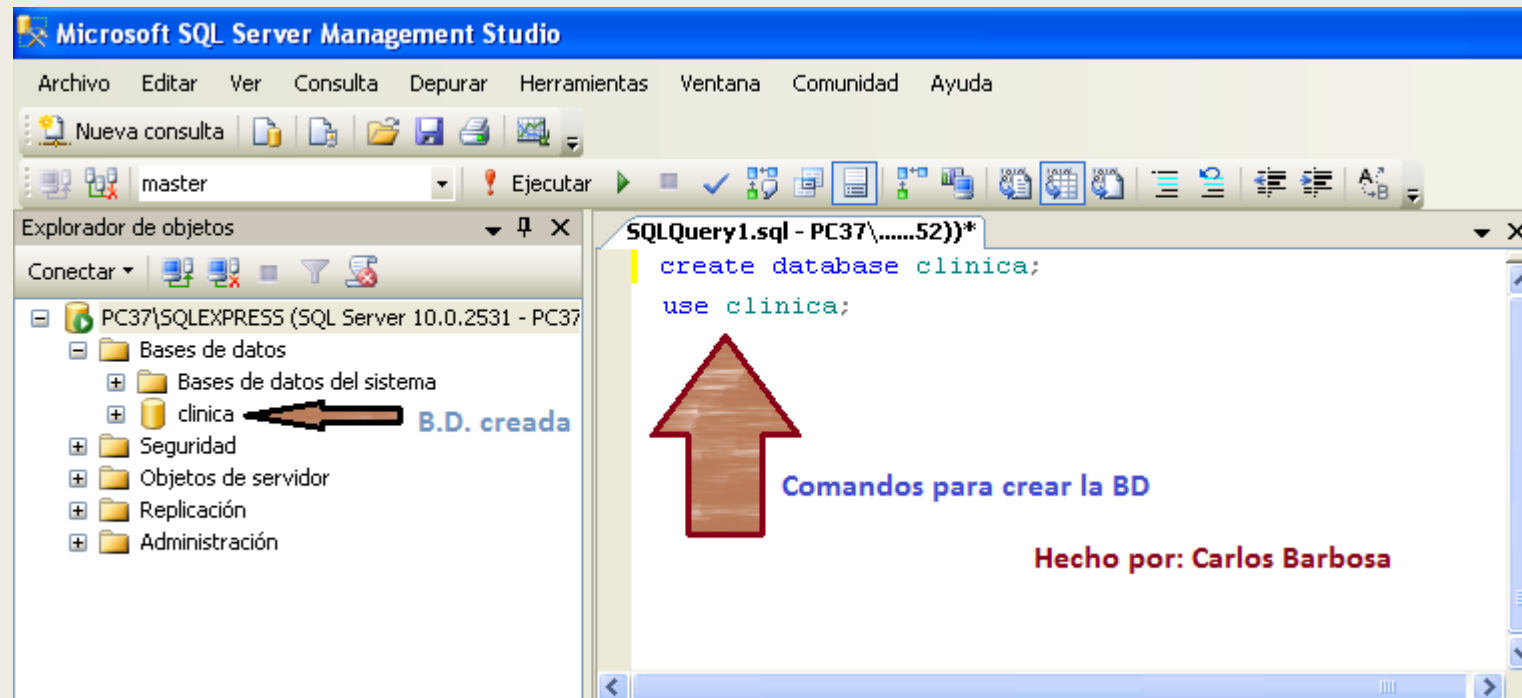


# ¿Cual es la diferencia entre las funciones de agregación y funciones creados por el DBA? Es decir funciones creadas por el usuario.

- Las funciones de agregación en SQL nos permiten efectuar operaciones sobre un conjunto de resultados, pero devolviendo un único valor agregado para todos ellos.
- Garantizar y optimizar la seguridad, integridad y estabilidad de las bases de datos, que administran la información de las operaciones del negocio, para que siempre estén disponibles, según las necesidades de las diferentes áreas de la compañía.

# ¿Para qué sirve el comando USE?

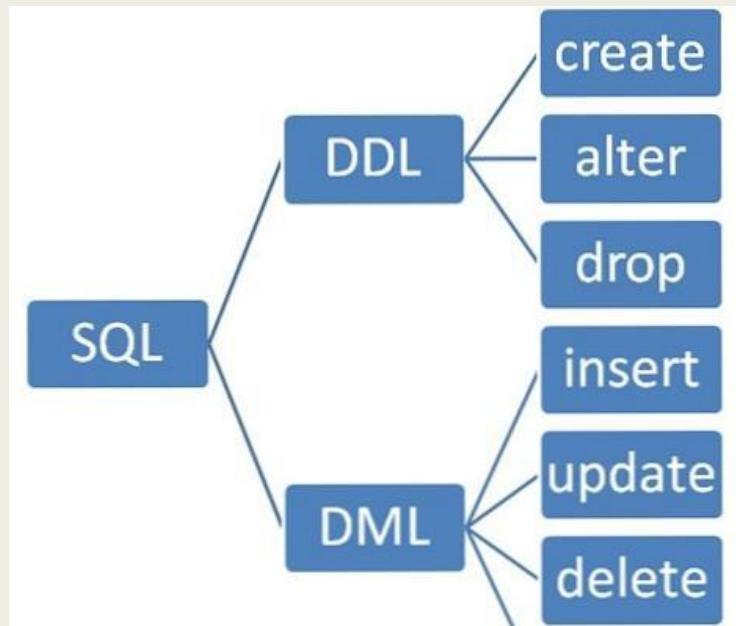
- Hacer que una base de datos determinada o creada sea actualmente la que estará en uso.





# Que es DML y DDL?

- Las sentencias DDL se utilizan para describir una base de datos, para definir su estructura, para crear sus objetos y para crear los sub-objetos de la tabla.
- Las sentencias DML se utilizan para controlar la información contenida en la base de datos.




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

- Los parámetros son variables locales a los que se les asigna un valor antes de comenzar la ejecución del cuerpo de una función. Su ámbito de validez, por tanto, es el propio cuerpo de la función. El mecanismo de paso de parámetros a las funciones es fundamental para comprender el comportamiento de los programas en C.

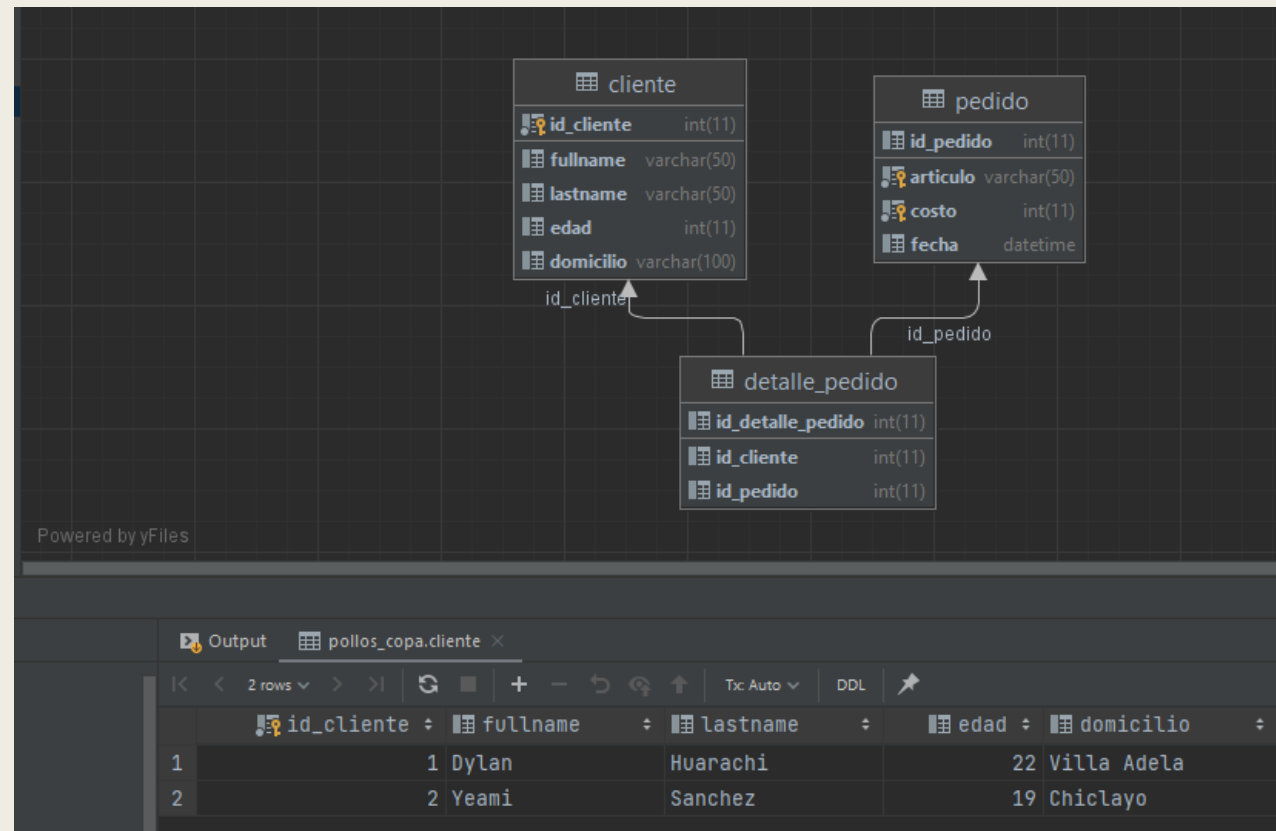
```
CREATE or replace FUNCTION min_edad_est(edad integer, genero varchar(20))
RETURNS INTEGER
BEGIN
    return
    (
        SELECT min(est.edad)
        FROM estudiantes AS est
        where est.sexo = genero and est.edad > edad
    );
END;
SELECT min_edad_est( edad: 18, genero: 'femenino');
```

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



```
CREATE or replace FUNCTION min_edad_est(edad integer, genero varchar(20))
RETURNS INTEGER
BEGIN
    return
    (
        SELECT min(est.edad)
        FROM estudiantes AS est
        where est.sexo = genero and est.edad > edad
    );
END;
SELECT min_edad_est( edad: 18, genero: 'femenino');
```

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



Crear una consulta SQL en base al ejercicio anterior.

```
SELECT cli.fullname, cli.lastname, ped.articulo, ped.costos  
FROM cliente AS cli  
INNER JOIN pedido AS ped ON cli.id_cliente = ped.id_cliente  
WHERE cli.id_cliente = 1;
```

# Crear un función que compare dos códigos de materia.

```
1 CREATE DATABASE tareaHito2;
2 USE tareaHito2;
3
4 CREATE TABLE estudiantes
5 (
6     id_est INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
7     nombres VARCHAR(50),
8     apellidos VARCHAR(50),
9     edad INTEGER,
10    gestion INTEGER,
11    fono INTEGER,
12    email VARCHAR(100),
13    direccion VARCHAR(100),
14    sexo VARCHAR(10)
15 );
```

```
CREATE TABLE materias
(
    id_mat INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    nombre_mat VARCHAR(100),
    cod_mat VARCHAR(100)
);

CREATE TABLE inscripcion
(
    id_ins INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    id_est INT NOT NULL,
    id_mat INT NOT NULL,
    semestre VARCHAR(20),
    gestion INTEGER,
    FOREIGN KEY (id_est) REFERENCES estudiantes (id_est),
    FOREIGN KEY (id_mat) REFERENCES materias (id_mat)
);
```

```
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)
VALUES ('Miguel', 'Gonzales Veliz', 20, 2_832_115, 'miguel@gmail.com', 'Av. 6 de Agosto', 'masculino');
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)
VALUES ('Sandra', 'Mavir Uribe', 25, 2_832_116, 'sandra@gmail.com', 'Av. 6 de Agosto', 'femenino');
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)
VALUES ('Joel', 'Adubini Mondar', 30, 2_832_117, 'joel@gmail.com', 'Av. 6 de Agosto', 'masculino');
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)
VALUES ('Andrea', 'Arias Ballesteros', 21, 2_832_118, 'andrea@gmail.com', 'Av. 6 de Agosto', 'femenino');
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)
VALUES ('Santos', 'Montes Valenzuela', 24, 2_832_119, 'santos@gmail.com', 'Av. 6 de Agosto', 'masculino');

SELECT est.*
FROM estudiantes AS est;

CREATE TABLE materias
```

```
INSERT INTO materias (nombre_mat, cod_mat) VALUES ('Introduccion a la Arquitectura', 'ARQ-101');
INSERT INTO materias (nombre_mat, cod_mat) VALUES ('Urbanismo y Diseno', 'ARQ-102');
INSERT INTO materias (nombre_mat, cod_mat) VALUES ('Dibujo y Pintura Arquitectonica', 'ARQ-103');
INSERT INTO materias (nombre_mat, cod_mat) VALUES ('Matematica discreta', 'ARQ-104');
INSERT INTO materias (nombre_mat, cod_mat) VALUES ('Ingenieria Sistemas', 'SIS-121');

INSERT INTO inscripcion (id_est, id_mat, semestre, gestion) VALUES (1, 1, '1er Semestre', 2015);
INSERT INTO inscripcion (id_est, id_mat, semestre, gestion) VALUES (1, 2, '2do Semestre', 2015);
INSERT INTO inscripcion (id_est, id_mat, semestre, gestion) VALUES (2, 4, '4to Semestre', 2017);
INSERT INTO inscripcion (id_est, id_mat, semestre, gestion) VALUES (3, 3, '2do Semestre', 2017);
INSERT INTO inscripcion (id_est, id_mat, semestre, gestion) VALUES (3, 1, '3er Semestre', 2017);
INSERT INTO inscripcion (id_est, id_mat, semestre, gestion) VALUES (4, 4, '4to Semestre', 2017);
INSERT INTO inscripcion (id_est, id_mat, semestre, gestion) VALUES (5, 5, '5to Semestre', 2017);

SELECT est.*
```

```
SELECT est.*
FROM estudiantes AS est;

SELECT mat.*
FROM materias AS mat;

SELECT ins.*
FROM inscripcion AS ins;

SELECT es.id_est, es.nombres, es.apellidos, mat.nombre_mat, mat.cod_mat
FROM estudiantes AS es
INNER JOIN inscripcion AS ins ON es.id_est = ins.id_est
INNER JOIN materias AS mat ON ins.id_mat = mat.id_mat
WHERE mat.cod_mat = 'ARQ-104';
```

id_est	nombres	apellidos	nombre_mat	cod_mat
2	Sandra	Mavir Uribe	Matematica discreta	ARQ-104
4	Andrea	Arias Ballesteros	Matematica discreta	ARQ-104

Crear una función que permita obtener el promedio de las edades del género masculino o femenino de los estudiantes inscritos en la asignatura ARQ-104.

```
CREATE FUNCTION promedio_edad_estudiantes() RETURNS int
BEGIN
  return
  (
    SELECT avg(est.edad)
    FROM estudiantes AS est
  );
END;

SELECT promedio_edad_estudiantes() as promedioEdad;
```

# Crear una función que permita concatenar 3 cadenas.

```
CREATE FUNCTION max_edad_est_4maculino(nombres VARCHAR(20), apellidos  
VARCHAR(20),eda integer)  
RETURNS INTEGER  
BEGIN  
  return  
  (  
    SELECT max(est.edad)  
    FROM estudiantes AS est  
    where est.nombres = nombres AND est.edad > eda  
  );  
END;  
select max_edad_est_4maculino('Miguel','Gonzales Veliz',18);
```



# Crear una función de acuerdo a lo siguiente:

```
CREATE or replace FUNCTION SUM_edad (genero varchar(20))
RETURNS INTEGER
BEGIN
    return
    (
        SELECT SUM(est.edad)
        FROM estudiantes AS est
        WHERE est.sexo = genero
    );
END;

SELECT est.nombres, est.apellidos
FROM estudiantes AS est
where SUM_edad('masculino') % 2 = 0;
```

# Crear una función de acuerdo a lo siguiente:

```
CREATE FUNCTION comparaNombre(nombres VARCHAR(50),apellidos VARCHAR(50))
RETURNS INTEGER
BEGIN
    return (
        select (est.nombres)
        from estudiantes as est
        where est.nombres='Miguel' and est.apellidos='Gonzales Veliz'
    );
end;

SELECT est.*
FROM estudiantes AS est
where comparaNombre( 'Santos', 'Montes Valenzuela');
```