BASE DE DATOS II
TAREA HITO 2

ESTUDIANTE: BRITTANY IBLING MARINO QUISPE
CODIGO: SIS13181271
DOCENTE: LIC. WILLIAM BARRA PAREDES
CARRERA: INGENIERIA DE SISTEMAS
SEMESTRE: 3
FECHA: 04-04-2022

MANEJO DE CONCEPTOS

1.- ¿A que se refiere cuando se habla de bases de datos relacionales?

Una base de datos relacional es un tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí. Las bases de datos relacionales se basan en el modelo relacional, una forma intuitiva y directa de representar datos en tablas.

2.- ¿A que se refiere cuando se habla de bases de datos no relacionales?

Las bases de datos no relacionales son un sistema de almacenamiento de información que se caracteriza por no usar el lenguaje SQL para las consultas. Esto no significa que no puedan usar el lenguaje SQL, pero no lo hacen como herramienta de consulta, sino como apoyo

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

MariaDB Server, conocida abreviadamente como MariaDB, es una base de datos de código abierto creada por los fundadores de MySQL. Tiene raíces similares a Percona Server para MySQL, pero se está apartando rápidamente de la compatibilidad con MySQL y creciendo por sí sola. Se ha convertido en la instalación de serie para varios Sistemas Operativos. Los cambios en las variables predeterminadas pueden marcar una gran diferencia en el rendimiento de la base de datos, por lo que conocer sus diferencias es importante.

MySQL ha adoptado una postura más conservadora cuando se trata del registro binario. En las versiones más recientes de MySQL 5.7, han actualizado dos variables para ayudar a garantizar que todos los datos aceptados permanezcan intactos e idénticos. Binlog_format ha sido actualizada a ROW en MySQL para evitar que las declaraciones no deterministas tengan resultados diferentes en el esclavo. Por el contrario, MariaDB tiene el formato mixto (MIXED) como predeterminado. Este utiliza un formato basado en declaraciones a menos que se cumplan ciertos criterios. En ese caso, usa el formato ROW.

MANEJO DE CONCEPTOS

4.- ¿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.

5.- ¿Qué llegaría a ser 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.

6.- ¿Cual es la diferencia entre las funciones de agresión yfunciones creados por el DBA? Es decir funciones creadas por el usuario

Las funciones de agregación se usan dentro de la cláusula SELECT en grupos de registros de devolver un único valor que se aplica a un grupo de registros.

El DBA es la persona con más conocimientos sobre base de datos en una organización. Como tal, debe entender las reglas básicas de la tecnología de base de datos relacional y ser capaz de comunicarlos con precisión a los demás. Un DBA debe ser un profesional experto en la recopilación y análisis de las necesidades del usuario para obtener modelos de datos conceptuales y lógicos.

7.- ¿Para qué sirve el comando USE?

a sentencia **USE nombre** indica a MySQL que use la base de datos **nombre** como la base de datos por defecto (actual) en sentencias subsiguientes.

MANEJO DE CONCEPTOS

8.- 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 subobjetos de la tabla.
La lista siguiente ofrece ejemplos de estos tipos de sentencias DDL:

•Creación de tablas (mandato CREATE)
•Modificación de la estructura de una tabla (mandato ALTER) sin suprimirla y volver a crearla, como añadir columnas, eliminar columnas o cambiar definiciones de columna (por ejemplo, longitud o los valores predeterminados)
•Eliminación de objetos (como tablas) de la base de datos (mandato DROP)

-Las sentencias DML se utilizan para controlar la información contenida en la base de datos. Las listas siguientes ofrecen ejemplos de estos tipos de sentencias DML:

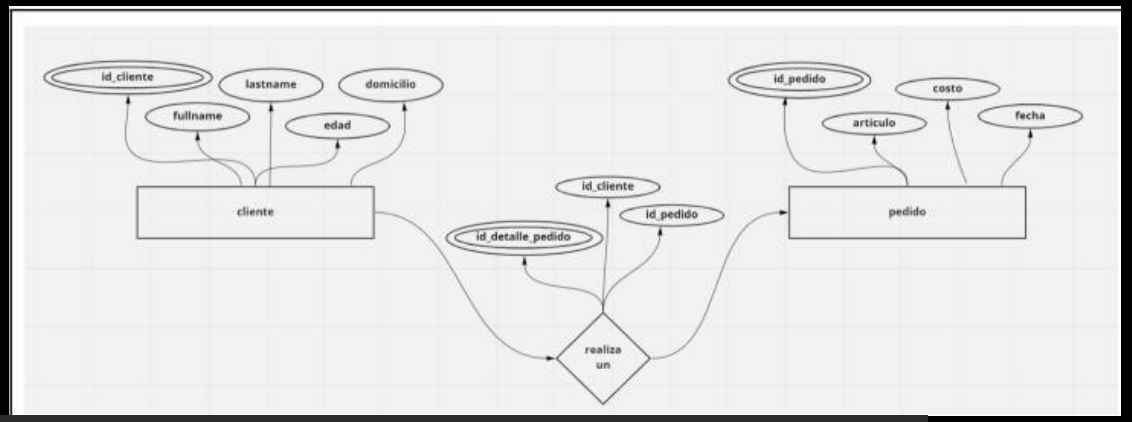
•Adición de registros a una tabla (mandato INSERT)
•Modificación de la información de una tabla (mandato UPDATE)
•Eliminación de registros de una tabla (mandato DELETE)

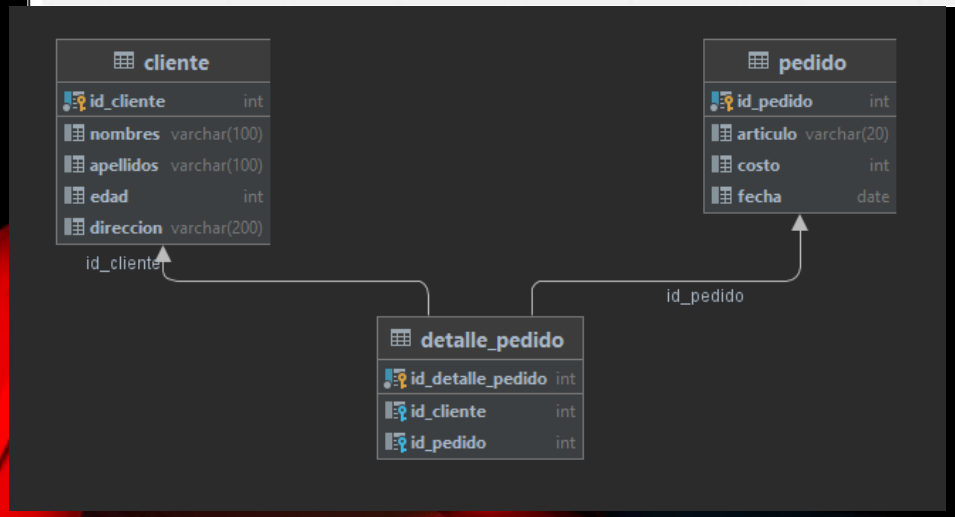
9.- ¿Qué cosas características debe de tener una función? Explique sobre el nombre, el return, parametros, etc.

Las funciones de agregación se usan dentro de la cláusula SELECT en grupos de registros de devolver un único valor que se aplica a un grupo de registros.
a función `return` tiene un parámetro y produce el siguiente efecto: termina la ejecución de la función en la que se ejecuta y el valor devuelto por la función es el parámetro con que es invocada `return`.
Un parámetro por defecto es un parámetro formal que se define terminándolo con un igual y una expresión.

PARTE PRACTICA

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






```sql
Create database POLLOS_COPA
USE POLLOS_COPA


CREATE TABLE cliente
(
 id_cliente INTEGER PRIMARY KEY NOT NULL,
 nombres VARCHAR(100),
 apellidos VARCHAR(100),
 edad INTEGER,
 direccion VARCHAR(200)
);
```

```sql
INSERT INTO cliente (id_cliente,nombres, apellidos, edad, direccion)
  VALUES (1,'Miguel' ,'Gonzales Veliz', 20,'Av. 6 de Agosto');
INSERT INTO cliente (id_cliente,nombres, apellidos, edad, direccion)
  VALUES (2,'Elizabeth' ,'Tnedo Flores', 19,'Av. Litoral');
INSERT INTO cliente (id_cliente,nombres, apellidos, edad, direccion)
  VALUES (3,'Abel' ,'Lopez Gonzales', 25,'Av. Marcelina');
```

```sql
CREATE TABLE pedido
(
  id_pedido INTEGER PRIMARY KEY  NOT NULL,
  articulo VARCHAR(20),
  costo INTEGER,
  fecha date
);
INSERT INTO pedido (id_pedido,articulo, costo, fecha)
  VALUES (22,'Broaster', 20,'04-04-22');
INSERT INTO pedido (id_pedido,articulo, costo, fecha)
  VALUES (21,'Pollo a la canasta', 25,'04-04-22');
INSERT INTO pedido (id_pedido,articulo, costo, fecha)
  VALUES (23,'Spiedo', 23,'04-04-22');

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)
);
```

```sql
);
INSERT INTO detalle_pedido (id_detalle_pedido, id_cliente, id_pedido)
  VALUES (52,1 ,22);
INSERT INTO detalle_pedido (id_detalle_pedido, id_cliente, id_pedido)
  VALUES (51,3 ,21);
INSERT INTO detalle_pedido (id_detalle_pedido, id_cliente, id_pedido)
  VALUES (53,2 ,23);


SELECT cli.id_cliente, cli.nombres, cli.apellidos, ped.id_pedido,ped.articulo
 FROM cliente AS cli
  INNER JOIN detalle_pedido AS det ON cli.id_cliente = det.id_cliente
  INNER JOIN pedido AS ped ON ped.id_pedido = det.id_pedido
WHERE ped.id_pedido <24;
```

12.Crear una consulta SQL en base al ejercicio anterior.

```sql
SELECT cli.id_cliente, cli.nombres, cli.apellidos, ped.id_pedido,ped.articulo
FROM cliente AS cli
  INNER JOIN detalle_pedido AS det ON cli.id_cliente = det.id_cliente
  INNER JOIN pedido AS ped ON ped.id_pedido = det.id_pedido
WHERE ped.id_pedido <24;
```

Output | Result 23

3 rows | CSV

	id_cliente	nombres	apellidos	id_pe...	articulo
1	2	Elizabeth	Tnedo Flores	23	Spiedo
2	3	Abel	Lopez Gonzales	21	Pollo a la canasta
3	1	Miguel	Gonzales Veliz	22	Broaster

13.Crear un función que compare dos códigos de materia.

```sql
CREATE DATABASE TareaHito2;
USE TareaHito2;


CREATE TABLE estudiantes
(
 id_est INTEGER PRIMARY KEY NOT NULL,
 nombres VARCHAR(50),
 apellidos VARCHAR(50),
 edad INTEGER,
 gestion INTEGER,
 fono INTEGER,
 email VARCHAR(100),
 direccion VARCHAR(100),
 sexo VARCHAR(10)
);
```

```sql
CREATE TABLE materias
(
  id_mat INTEGER PRIMARY KEY  NOT NULL,
  nombre_mat VARCHAR(100),
  cod_mat VARCHAR(100)
);
CREATE TABLE inscripcion
(
  id_ins INTEGER PRIMARY KEY  NOT NULL,
  semestre VARCHAR(20),
  gestion INTEGER,
  id_est INTEGER,
  id_mat INTEGER,
  FOREIGN KEY (id_est) REFERENCES estudiantes (id_est),
  FOREIGN KEY (id_mat) REFERENCES materias (id_mat)
);
```

```sql
INSERT INTO estudiantes (id_est,nombres, apellidos, edad, fono, email,direccion, sexo)
VALUES(1,'Miguel', 'Gonzales Veliz', 20, 2_832_115,'miguel@gmail.com', 'Av. 6 de Agosto', 'masculino'),
 (2,'Sandra', 'Mavir Uria', 25, 2_832_116, 'sandra@gmail.com','Av. 6 de Agosto', 'femenino'),
 (3,'Joel', 'Adubiri Mondar', 30, 2_832_117, 'joel@gmail.com','Av. 6 de Agosto', 'masculino'),
 (4,'Andrea', 'Arias Ballesteros', 21, 2_832_118,'andrea@gmail.com', 'Av. 6 de Agosto', 'femenino'),
(5,'Santos', 'Montes Valenzuela', 24, 2_832_119,'santos@gmail.com', 'Av. 6 de Agosto', 'masculino');

INSERT INTO materias (id_mat,nombre_mat, cod_mat)
VALUES (11,'Introduccion a la Arquitectura', 'ARQ-101'),
 (12,'Urbanismo y Diseno', 'ARQ-102'),
 (13,'Dibujo y Pintura Arquitectonico', 'ARQ-103'),
 (14,'Matematica discreta', 'ARQ-104'),
(15,'Fisica Basica', 'ARQ-105');
```

```sql
INSERT INTO inscripcion (id_ins,id_est, id_mat, semestre, gestion)
VALUES (21,1, 11, '1er Semestre', 2018);
      INSERT INTO inscripcion (id_ins,id_est, id_mat, semestre, gestion)
VALUES (22,1, 12, '2do Semestre', 2018);
      INSERT INTO inscripcion (id_ins,id_est, id_mat, semestre, gestion)
VALUES (23,2, 14, '1er Semestre', 2019);
      INSERT INTO inscripcion (id_ins,id_est, id_mat, semestre, gestion)
VALUES (24,2, 13, '2do Semestre', 2019);
      INSERT INTO inscripcion (id_ins,id_est, id_mat, semestre, gestion)
VALUES (25,3, 13, '2do Semestre', 2020);
      INSERT INTO inscripcion (id_ins,id_est, id_mat, semestre, gestion)
VALUES (26,3, 11, '3er Semestre', 2020);
      INSERT INTO inscripcion (id_ins,id_est, id_mat, semestre, gestion)
VALUES (27,4, 14, '4to Semestre', 2021);
      INSERT INTO inscripcion (id_ins,id_est, id_mat, semestre, gestion)
VALUES (28,5, 15, '5to Semestre', 2021);
```

Mostrar los nombres y apellidos de los estudiantes inscritos en la materia ARQ-105, adicionalmente mostrar el nombre de la materia.

```sql
SELECT est.nombres, est.apellidos, mat.nombre_mat,mat.cod_mat
FROM estudiantes AS est
 INNER JOIN inscripcion AS ins ON est.id_est = ins.id_est
  INNER JOIN materias AS mat ON ins.id_mat = mat.id_mat
WHERE mat.cod_mat='ARQ-105'
```

Output | Result 12

1 row

	nombres	apellidos	nombre_mat	cod_mat
1	Santos	Montes Valenzuela	Fisica Basica	ARQ-105

Deberá de crear una función que reciba dos parámetros y esta función deberá ser utilizada en la cláusula WHERE.

```sql
select est.nombres,est.apellidos,i.semestre
from estudiantes as est
INNER JOIN inscripcion  i ON est.id_est = i.id_est
where est.edad=dbo.sum_edad2022( @genedad: 'masculino', @edad1: 20);
```

Output | Result 2

2 rows

	nombres	apellidos	semestre
1	Miguel	Gonzales Veliz	1er Semestre
2	Miguel	Gonzales Veliz	2do Semestre

14.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. ○ La función recibe como parámetro solo el género. ○ La función retorna un valor numérico.

```
113
114  create function promedio(@genero varchar(40))
115  returns integer
116  begin
117      return(
118          select AVG(est.edad)
119          from estudiantes as est
120          where est.sexo = @genero
121          );
122  end;
123
124  SELECT dbo.promedio ( @genero: 'femenino') ;
```

promedio()

Output | dbo.promedio ('femenino'):int

1 row

	<anonymous>
1	23