BASE DE DATOS II

MANEJO DE FUNCIONES Y BASE DE DATOS RELACIONALES

NOMBRE: APAZA LARA KEVIN MICHAEL
CÓDIGO: SIS13814908
DOCENTE: WILLIAM RODDY BARRA PAREDES
FECHA: Martes 5 de abril, 2022

Manejo de conceptos

¿A que se refiere cuando se habla de base de datos relacionales?

Nombre	Apellido	Email	Dirección
Sara	Sanchez	sanchez98@gmail.com	C/Pantomima nº12
Ricardo	Estevez	RE765@hotmail.com	Av/ Blasco Ibáñez nº89
Guillermo	López	lopezgd@gmail.com	C/ Jazmin nº1
Ana	Marín	ana.a.ps@hotmail.es	C/ Condesa nº5
Gustavo	García	gargosar@yahoo.es	C/ Barrera nº10

Base de datos relacional es aquel que implementa el modelo relacional. Se basa en el almacenamiento de la información con cierta relación entre sí en filas y columnas (las filas los registros y las columnas los campos).

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

También llamado noSQL estas bases de datos a diferencia de las relaciones guardan la información en documentos donde, diseñados para ser escalables horizontalmente se ven principalmente implementados donde se requiera el manejo de grandes volúmenes de datos.

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

MySQL es un sistema de gestión de bases de datos relacional bajo licencia Licencia pública general, comercial por Oracle Corporation, sistema de gestión de bases de datos derivado de MySQL con licencia Licencia Pública General (software libre).

¿Qué son las funciones de agregación?

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

Funciones que por defecto ya vienen implementadas en el gestor de base de datos.

¿Qué llegaría a ser XAMPP?

XAMPP es un servidor independiente de plataforma de código libre. Que incorpora distintas bases de datos.

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

Las funciones de agregación pueden ser ejecutadas en SELECT, mientras que las creadas por el usuario pueden ser ejecutadas desde el WHERE.

¿Para qué sirve el comando USE?

Para posicionarse en una base de datos existente y acceder a esta.

Que es DML y DDL?

Lenguaje de definición de datos (DDL)
-Creación de tablas (mandato CREATE)
-Modificación de la estructura de una tabla (mandato ALTER) sin suprimirla y volver a crearla, cómo 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)
-Particionado de tablas (mandato PARTITION)

Lenguaje de manipulación de datos (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)

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

Así como una función en un lenguaje de programación en las base de datos se debe de cumplir con ciertas características:
-Debe de tener un nombre generalmente relacionado a lo que hace.
-Si es que tiene parámetros se les da un tipo ya sea VARCHAR() o INTEGER o otro tipo.
-Se debe de especificar el valor a devolver
-La ejecución debe ir entre BEGIN y END
-Siempre se debe de retornar el valor

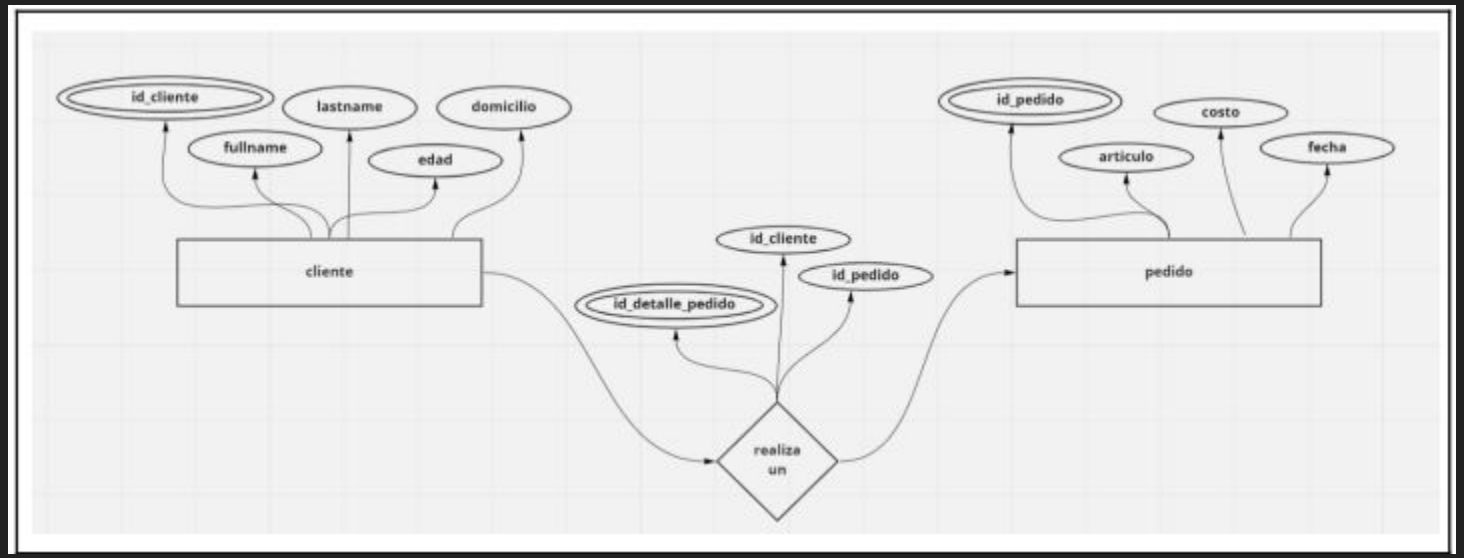
¿Cómo crear,modificar y cómo eliminar una función?

```
DROP FUNCTION IF EXISTS holaMundo;          //eliminar

CREATE OR REPLACE FUNCTION holaMundo(entrada VARCHAR(20))     //crear y modificar
RETURNS  //VARCHAR(20)
BEGIN
  DECLARE salida VARCHAR(20);
  SET salida = entrada;                              //proceso o funcionamiento
  RETURN salida;
END
```

PARTE PRÁCTICA

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



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

Se sugiere crear una base de datos de nombre POLLOS_COPA y en ella crear las tablas:

- cliente
- detalle_pedido
- pedido
  - Adjuntar el código SQL generado.

```sql
CREATE DATABASE POLLOS_COPA;
USE POLLOS_COPA;

CREATE TABLE cliente(
   id cliente VARCHAR(20)NOT NULL PRIMARY KEY,
   fullname VARCHAR(20) NOT NULL,
   lastname VARCHAR(30) NOT NULL,
   edad INTEGER,
   domicilio VARCHAR(30) NOT NULL
);


CREATE TABLE pedido(
   id pedido INT AUTO INCREMENT PRIMARY KEY,
   articulo VARCHAR(30)NOT NULL ,
   costo INTEGER NOT NULL ,
   fecha VARCHAR(15) NOT NULL
);


CREATE TABLE detalle pedido(
   id detalle pedido INT AUTO INCREMENT PRIMARY KEY,
   id cliente VARCHAR(20)NOT NULL,
   id pedido INT,
   FOREIGN KEY (id cliente) REFERENCES cliente (id cliente),
   FOREIGN KEY (id_pedido) REFERENCES pedido (id_pedido)
);
```

-Creación de la base de datos
-Posicionamiento en la BD
-Creación de tablas

Inserción de registros

```
INSERT INTO cliente(id_cliente, fullname, lastname, edad, domicilio)values
('ID-12345555','Mario Hugo','Fernandez Quispe',25,'av.15 Los Altos'),
('ID-12346666','Maria Maria','Delgado Jackson',30,'av.2 Bajo'),
('ID-12347777','Pedro Carlos','Ramires Peres',21,'av.2 Alto'),
('ID-12348888','Jose Jose','Peres Peres',19,'av.31 treinta y uno');
```

```
INSERT INTO pedido(articulo, costo, fecha) VALUES
('pollo frito',20,'15/03/22'),
('refresco',13,'15/03/22'),
('pollo horneado',62,'15/03/22'),
('pollo a la lena',50,'15/03/22');
```

```
INSERT INTO detalle_pedido(id_cliente, id_pedido) VALUES
('ID-12345555',1),
('ID-12346666',2),
('ID-12347777',4),
('ID-12348888',4);
```

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

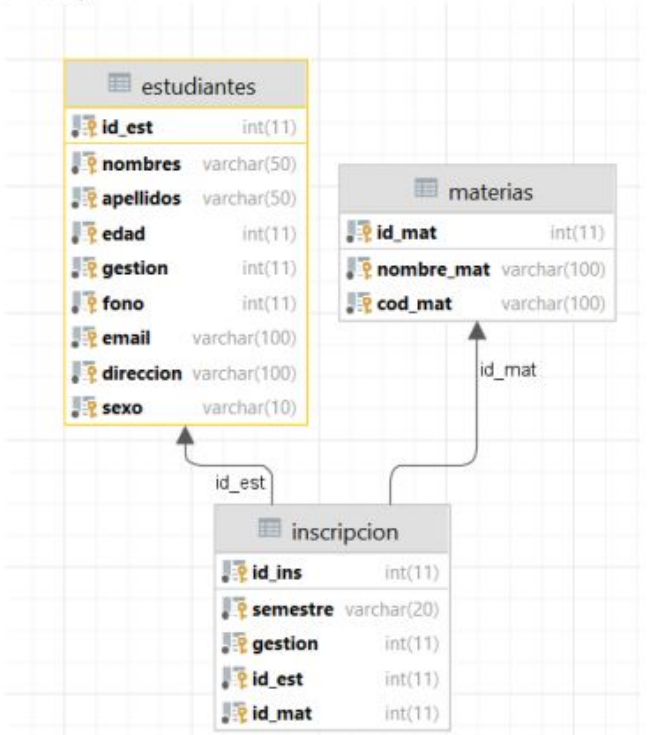
- Debe de utilizar las 3 tablas creadas anteriormente.
- Para relacionar las tablas utilizar JOINS.
- Adjuntar el código SQL generado.

```
SELECT dtp.id_cliente,
cl.fullname,
cl.lastname,ped.costo,ped.articu
lo
FROM detalle pedido AS dtp
JOIN cliente AS cl ON
dtp.id cliente = cl.id_cliente
JOIN pedido AS ped ON
dtp.id_pedido = ped.id_pedido
```

	id_cliente	fullname	lastname	costo	articulo
1	ID-12345555	Mario Hugo	Fernandez Quispe	20	pollo frito
2	ID-12346666	Maria Maria	Delgado Jackson	13	refresco
3	ID-12347777	Pedro Carlos	Ramires Peres	50	pollo a la lena
4	ID-12348888	Jose Jose	Peres Peres	50	pollo a la lena

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

```
CREATE DATABASE tareaHito2;
USE tareaHito2;
```



```
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email,
direccion, sexo)
VALUES ('Miguel', 'Gonzales Veliz', 20, 2832115,
'miguel@gmail.com', 'Av. 6 de Agosto', 'masculino'),
      ('Sandra', 'Mavir Uria', 25, 2832116, 'sandra@gmail.com',
'Av. 6 de Agosto', 'femenino'),
      ('Joel', 'Adubiri Mondar', 30, 2832117, 'joel@gmail.com',
'Av. 6 de Agosto', 'masculino'),
      ('Andrea', 'Arias Ballesteros', 21, 2832118,
```

```
'andrea@gmail.com', 'Av. 6 de Agosto', 'femenino'),
      ('Santos', 'Montes Valenzuela', 24, 2832119,
'santos@gmail.com', 'Av. 6 de Agosto', 'masculino');
```

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

```
INSERT INTO inscripcion (id_est, id_mat, semestre, gestion)
VALUES (1, 1, '1er Semestre', 2018),
      (1, 2, '2do Semestre', 2018),
      (2, 4, '1er Semestre', 2019),
      (2, 3, '2do Semestre', 2019),
      (3, 3, '2do Semestre', 2020),
      (3, 1, '3er Semestre', 2020),
      (4, 4, '4to Semestre', 2021),
      (5, 5, '5to Semestre', 2021);
```

```sql
CREATE DATABASE tareaHito2;
USE tareaHito2;

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

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

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

```sql
INSERT INTO estudiantes (nombres, apellidos, edad, fono,
email,direccion, sexo)VALUES
('Miguel', 'Gonzales Veliz', 20, 2832115,'miguel@gmail.com', 'Av. 6
de Agosto', 'masculino'),
('Sandra', 'Mavir Uria', 25, 2832116, 'sandra@gmail.com','Av. 6 de
Agosto', 'femenino'),
('Joel', 'Adubiri Mondar', 30, 2832117, 'joel@gmail.com','Av. 6 de
Agosto', 'masculino'),
('Andrea', 'Arias Ballesteros', 21, 2832118,'andrea@gmail.com', 'Av.
6 de Agosto', 'femenino'),
('Santos', 'Montes Valenzuela', 24, 2832119,'santos@gmail.com', 'Av.
6 de Agosto', 'masculino');
```

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

```sql
INSERT INTO inscripcion (id_est, id_mat, semestre, gestion)
VALUES (1, 1, '1er Semestre', 2018),
(1, 2, '2do Semestre', 2018),
(2, 4, '1er Semestre', 2019),
(2, 3, '2do Semestre', 2019),
(3, 3, '2do Semestre', 2020),
(3, 1, '3er Semestre', 2020),
(4, 4, '4to Semestre', 2021),
(5, 5, '5to Semestre', 2021);
```

Resolver lo siguiente:

- Mostrar los nombres y apellidos de los estudiantes inscritos en la materia ARQ-105, adicionalmente mostrar el nombre de la materia.
- Deberá de crear una función que reciba dos parámetros y está función deberá ser utilizada en la cláusula WHERE.

```
drop function if exists  comparaMaterias;

CREATE FUNCTION comparaMaterias(materia varchar(15), cod_mat
VARCHAR(50)) RETURNS bool
BEGIN
   return materia=cod_mat;
END;
```

```
select est.nombres,est.apellidos,mat.nombre_mat
from inscripcion as ins
join estudiantes as est on ins.id_est=est.id_est
join materias as mat on ins.id_mat = mat.id_mat
where comparaMaterias(mat.cod_mat,'ARQ-104');
```

	nombres	apellidos	nombre_mat
1	Sandra	Mavir Uria	Matematica discreta
2	Andrea	Arias Ballesteros	Matematica discreta

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.

```
CREATE FUNCTION min_edad_genero(genero VARCHAR(10)) RETURNS INTEGER
BEGIN
   DECLARE min_edad INTEGER;
   SET min_edad=(
       SELECT AVG(est.edad)
       FROM estudiantes as est, materias as mat
       WHERE est.sexo=genero and
             mat.cod_mat='ARQ-104'
       );
   return min_edad;
END;


SELECT min_edad_genero('femenino');
```

	`min_edad_genero('femenino')`
1	23

15.Crear una función que permita concatenar 3 cadenas.

- La función recibe 3 parámetros.
- Si la cadenas fuesen:
  - Pepito
  - Perez
  - 50
- La salida debería ser: Pepito - Perez - 50

```sql
create function concatenando(nombre varchar(50), apellido varchar(50), edad integer)
returns varchar(104)
begin
   declare info_per varchar(104);
   set info_per= concat(nombre,'-',apellido,'-',edad);
   return info_per;
end;

select concatenando('Pepito','Perez',21);
```

	`concatenando('Pepito','Perez',21)`
1	Pepito-Perez-21

16.Crear una función de acuerdo a lo siguiente:
○ Mostrar el nombre, apellidos y el semestre de todos los estudiantes que estén inscritos. Siempre y cuando la suma de las edades del sexo femenino o masculino sea par y mayores a cierta edad.
○ Debe de crear una función que sume las edades (recibir como parámetro el sexo, y la edad).
■ Ejemplo: sexo=’Masculino’ y edad=22
■ Note que la función recibe 2 parámetros.
○ La función creada anteriormente debe utilizarse en la consulta SQL. (Cláusula WHERE).

```
create function suma_edades(sexo1 varchar(10),edad1 integer)
returns integer
begin
   declare retornar integer;
   set retornar =(
       SELECT sum(est.edad)
       FROM estudiantes as est
       WHERE est.sexo=sexo1 and
             est.edad>edad1 and
             est.edad%2=0);

   return retornar;
end;

select est.nombres,est.apellidos,ins.semestre
from estudiantes as est
join inscripcion ins on est.id_est = ins.id_est
where suma_edades('masculino',22);
```

	nombres	apellidos	semestre
1	Miguel	Gonzales Veliz	1er Semestre
2	Miguel	Gonzales Veliz	2do Semestre
3	Sandra	Mavir Uria	1er Semestre
4	Sandra	Mavir Uria	2do Semestre
5	Joel	Adubiri Mondar	2do Semestre
6	Joel	Adubiri Mondar	3er Semestre
7	Andrea	Arias Ballesteros	4to Semestre
8	Santos	Montes Valenzuela	5to Semestre

17. Crear una función de acuerdo a lo siguiente:
○ Crear una función sobre la tabla estudiantes que compara un nombre y apellidos. (si existe este nombre y apellido mostrar todos los datos del estudiante).
■ La función devuelve un boolean.
■ La función debe recibir el nombre y sus apellidos.
○ Ejemplo:

```
create function comparaNombreLastonem(nombres varchar(50), apellidos varchar(50))
```

○ La función debería ser usada en la cláusula WHERE.

```sql
drop function if exists  comparaNombre;
create function  comparaNombre(nombres  varchar(50),apellidos  varchar(50),
nombresAcomparar  varchar(50),apellidosAcomparar  varchar(50))
returns boolean
begin

   declare  respuesta  bool default false;
   set  respuesta=(nombres=nombresAcomparar  and apellidos = apellidosAcomparar) ;
   return  respuesta;

end;


select est.*
from estudiantes  as est
where  comparaNombre('Miguel','Gonzales Veliz','Miguel','Gonzales Veliz');
```

	id_est	nombres	apellidos	edad	fono	email	direccion	sexo
1	1	Miguel	Gonzales Veliz	20	2832115	miguel@gmail.com	Av. 6 de Agosto	masculino
2	2	Sandra	Mavir Uria	25	2832116	sandra@gmail.com	Av. 6 de Agosto	femenino
3	3	Joel	Adubiri Mondar	30	2832117	joel@gmail.com	Av. 6 de Agosto	masculino
4	4	Andrea	Arias Ballesteros	21	2832118	andrea@gmail.com	Av. 6 de Agosto	femenino
5	5	Santos	Montes Valenzuela	24	2832119	santos@gmail.com	Av. 6 de Agosto	masculino