BASE DE DATOS 2
HITO 2

UNIFRANZ
Internacionalízate

PRESENTACION DE PREGUNTAS TEÓRICAS Y PRACTICAS

NOMBRE: JOEL REYNALDO
APELLIDO: CONDORI TUMIRI

UNIFRANZ Internacionalízate | INNOVACIÓN EN EDUCACIÓN | Blended Learning

OBJETIVOS A MOSTRAR

- MANEJO DE CONCEJOS
- PARTE PRACTICA

EMPEZAMOS

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

R.- Es un tipo de base de datos en la cual los datos están clasificados en tablas, estas tablas están relacionadas entre si

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

R.- Es aquella que no usa el esquema tabular de filas y columnas que se encuentra. A diferencia de la relaciones, no tienen un identificador que sirva de relación entre un conjunto de datos y otros

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

R.- Estos dos sistemas de gestión de bases de datos siguen siendo bastante diferentes

- MariaDB tiene licencia GPL mientras que MySQL tiene un enfoque de doble licencia.
- En muchos escenarios, MariaDB ofrece un mejor rendimiento.
- MariaDB soporta muchos motores de almacenamiento diferentes.

4. ¿Que son las funciones de agregación?

R.- nos permiten agrupar filas, según los campos específicos, es común que se utilicen en conjunto con las funciones de agregación para obtener resultados resumidos y agrupado.

5. ¿Que llegaría a ser XAMPP, WAMP SERVER o LAMP?

R.- XAMPP.- es un paquete de software libre, que consiste principalmente en el sistema de gestión de base de datos MySQL. También nos permite sin necesidad de tener conexión a internet

WAMPSERVER.- este permite subir paginas HTML a internet, además de poder gestionar datos en ellas. Al mismo tiempo un WAMP proporciona lenguajes de programación para desarrollar aplicaciones web.

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

R.- Las funciones agregación son aquellas que ya vienen con en el gestor de base de datos solo necesitan ser llamadas.

7.¿Para que sirve el comando USE?

R.- Se utiliza para designar una base externa como base de datos actual, en otras palabras, par hacer consultas en el proceso

8.¿Que es DML y DDL?

R.- DML .- Lenguaje de manipulación de datos es un idioma proporcionado por los sistemas gestores de bases de datos que permiten a los usuarios de la misma llevar a cabo las tareas de consulta o modificación de los datos contenidos en las bases de datos del sistema gestor de bases de datos

DDL .- Es un lenguaje para describir los datos y sus relaciones un una base de datos. Puede generar DDL en un

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

R.- Una función debe tener las siguientes partes

CREATE FUNCTION OR REPLACE NombreDeLaFuncion (parámetros a recibir)

RETURNS INTEGER Dato_A_devolver

BEGIN

DECLARE NUMERO INTEGER DEFAULT 0 declaración de datos para usar, si son necesarios

SET NUMERO = 2; Procedimiento de la función

RETURN NUMERO;

END;

SELECT

NombreDeLaFuncion (parámetros a enviar);

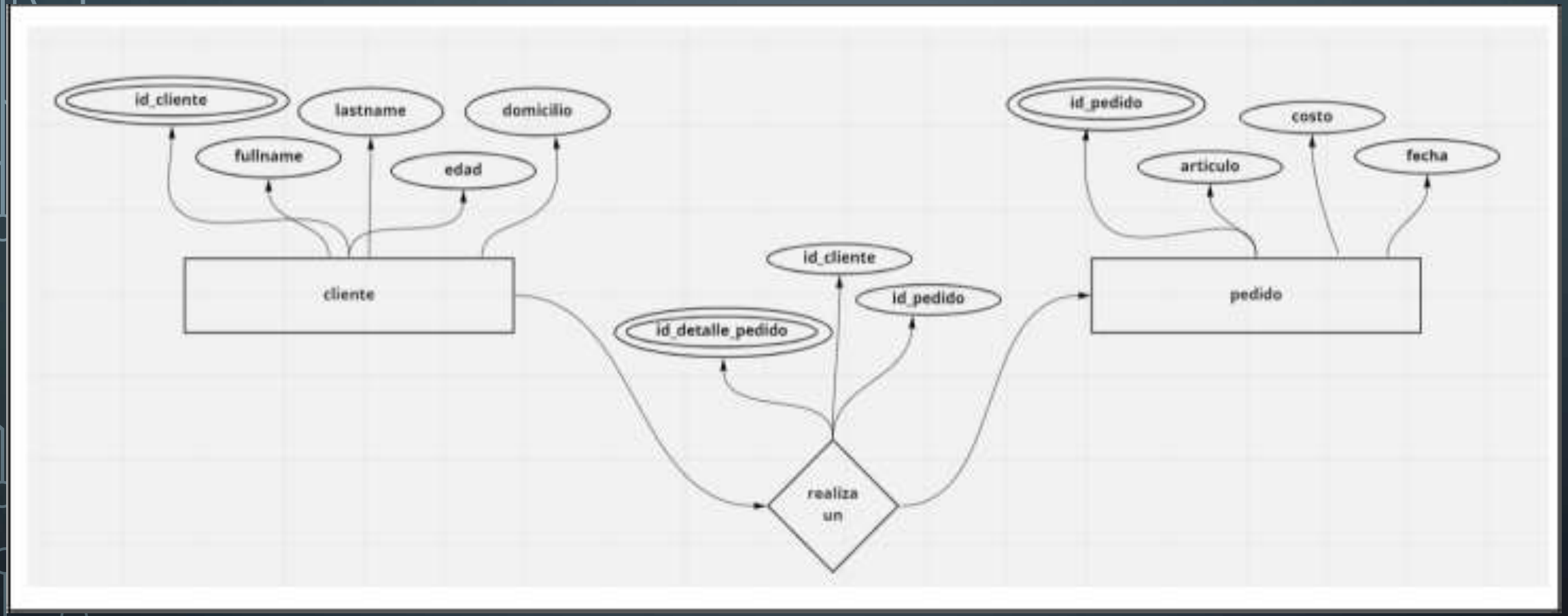
10. ¿Como crear, modificar y como eliminar una función?

R.- Para crear una función el código es **CREATE FUNCTION**

Para modificar una función **REPLACE**

Para eliminar la función es **DROP FUNCTION (nombre de la función)**

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



```sql
-- auto-generated definition
create table cliente
(
    id_cliente int auto_increment
        primary key,
    fullname   varchar(30) null,
    lastname   varchar(30) null,
    edad       int         null,
    domicilo   varchar(40) null
);
```

	id_cliente	fullname	lastname	edad	domicilo
1	1	piter	alejandro mamani	21	final castillo
2	2	deysi	achu	21	ballivian

```sql
-- auto-generated definition
create table pedido
(
    id_pedido int auto_increment
        primary key,
    articulo  varchar(30) null,
    costo     float       null,
    fecha     date        null
);
```

	id_pedido	articulo	costo	fecha
1	1	huevos	60	2022-08-12
2	2	carne	15	2022-08-12

```sql
-- auto-generated definition
create table datalle_pedido
(
    id_detalle_pedido int auto_increment
        primary key,
    id_pedido         int not null,
    id_cliente        int not null,
    constraint datalle_pedido_ibfk_1
        foreign key (id_pedido) references pedido (id_pedido),
    constraint datalle_pedido_ibfk_2
        foreign key (id_cliente) references cliente (id_cliente)
);

create index id_cliente
    on datalle_pedido (id_cliente);

create index id_pedido
    on datalle_pedido (id_pedido);
```

	id_detalle_pedido	id_pedido	id_cliente
1	1	1	2
2	2	2	1

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

R.-

```
select est.nombres, est.apellidos, mat.cod_mat, mat.nombre_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';
```

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

R.-

```
create function compara_materias(cod_mat varchar(20), nombre_mat varchar(20))
    returns boolean
begin
    declare respuesta boolean;
    if cod_mat = nombre_mat
    then
        set respuesta=1;
    end if;
    return respuesta;
end;
```

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

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.

R.-

```
SELECT avg(est.edad)
FROM estudiantes AS est
        inner join inscripcion ins on est.id_est = ins.id_est
        inner join materias mat on ins.id_mat = mat.id_mat
where est.sexo = 'femenino' and mat.cod_mat = 'ARQ-104';

CREATE OR REPLACE FUNCTION get_avg_est(genero varchar(10), codMateria varchar(10))
    RETURNS INTEGER
BEGIN
    declare avgEdad int default 0;
    SELECT avg(est.edad) into avgEdad
    FROM estudiantes AS est
            inner join inscripcion ins on est.id_est = ins.id_est
            inner join materias mat on ins.id_mat = mat.id_mat
    where est.sexo = genero and mat.cod_mat = codMateria;
    return avgEdad;
END;

select get_avg_est( genero: 'femenino',  codMateria: 'ARQ-104') as promedio;
```

Output | promedio:int(11)

1 row

	promedio
1	23

15. Crear una función que permita concatenar 3 cadenas

○ La función recibe 3 parámetros. ○ Si las cadenas fuesen: ■ Pepito ■ Pep ■ 50 ○ La salida debería ser: (Pepito), (Pep), (50) ○ La función creada utilizarlo en una consulta SQL.

R.-

```
create function getParametros(par1 varchar(20), par2 varchar(20), par3 varchar(20))
    returns varchar(60)
begin
    declare resultado varchar(60);
    set resultado = CONCAT(par1, ' ', par2, ' ', par3);
    return resultado;
end;

select getParametros( par1: 'pepito',  par2: 'pep',  par3: '50');
```

	`getParametros('pepito', 'pep', '50')`
1	pepito pep 50

16. Crear una función de acuerdo a lo siguiente:

- Mostrar el nombre, apellidos, edad y el semestre de todos los estudiantes que estén inscritos.
- Siempre y cuando la suma de las edades del sexo femenino(tambien puede ser 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).

R.-

```
create or replace function get_genero_edad(genero varchar(10), edad int)
    returns boolean
begin
    declare resultado int default 0;
    declare ifRes boolean;

    select sum(est.edad) into resultado
    from estudiantes as est
    where est.sexo=genero;

    if resultado%2=0 and resultado>edad
    then
        set ifRes=1;
    end if;
    return ifRes;
end;

select est.nombres, est.apellidos, i.semestre
 from estudiantes as est
 inner join inscripcion i on est.id_est = i.id_est
where get_genero_edad( genero: 'masculino',  edad: 22);
```

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). 7

- La función devuelve un boolean.
- La función debe recibir 4 parámetros, nombres y apellidos

R:-

```
create or replace function comparaNombre(nombre varchar(50),apellido varchar(50),nombreEst varchar(50), apellidoEst varchar(50))
    returns boolean
begin
    declare resultado boolean;
    if nombre=nombreEst and apellido=apellidoEst
    then
    set resultado=1;
end if;
    return resultado;
end;

select est.*
 from estudiantes as est
where comparaNombre( nombre: est.nombres, apellido: est.apellidos, nombreEst: 'Joel', apellidoEst: 'Adubiri Mondar')
```

	id_est	nombres	apellidos	edad	gestion	fono	email	direccion	sexo
1	3	Joel	Adubiri Mondar	30	<null>	2832117	joel@gmail.com	Av. 6 de Agosto	masculino