Base de datos II

UNIVERSIDAD PRIVADA FRANZ TAMAYO

DEFENSA HITO 4

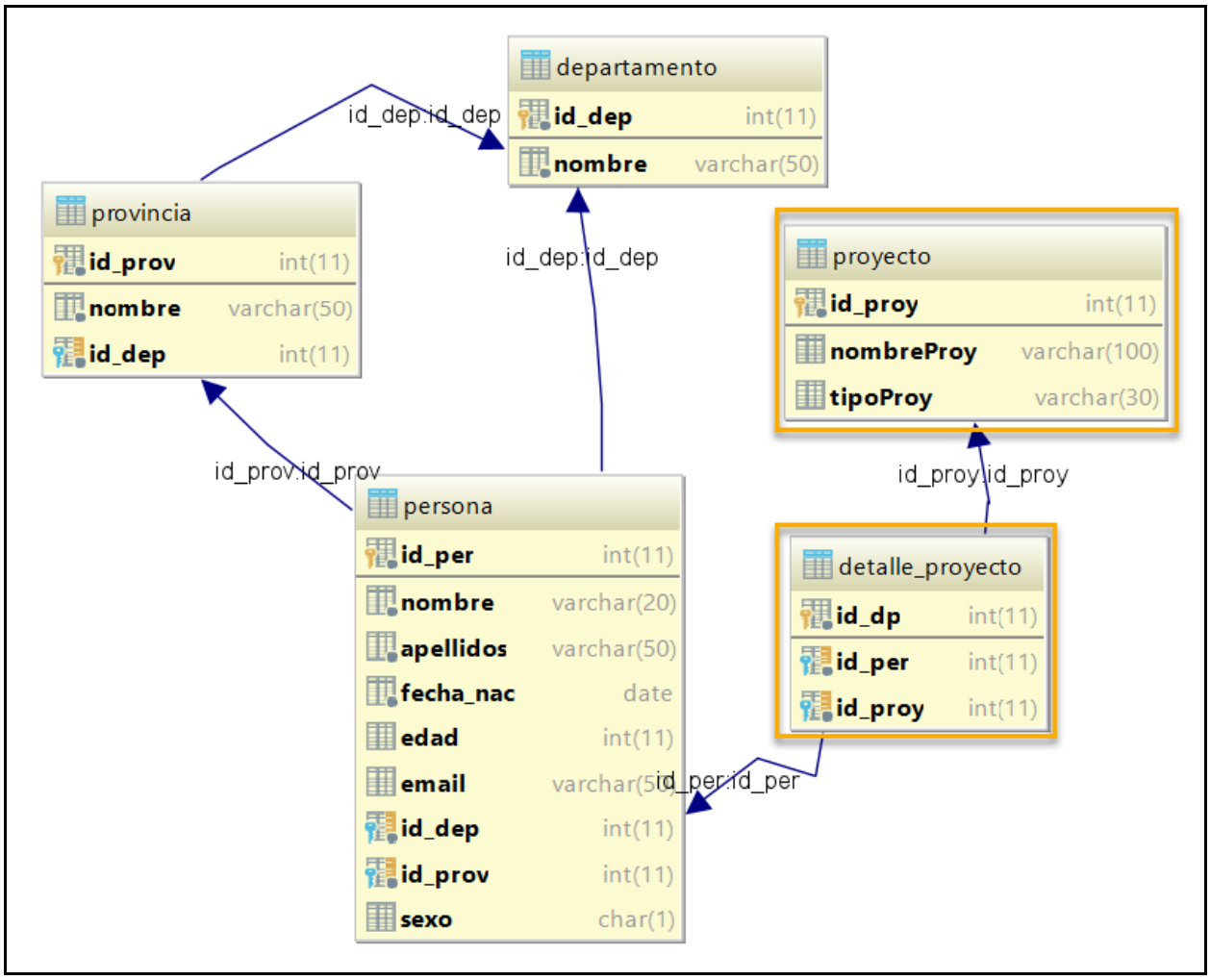
Nombre Completo: Diego Emiliano Rivera Tapia
Asignatura: BASE DE DATOS II
Carrera: INGENIERÍA DE SISTEMAS
Paralelo: BDA (1)
Docente: Lic. William R. Barra Paredes
Fecha: 03/12/2019

BASE DE DATOS II
LENGUAJE PROCEDURAL

TAREA ACUMULATIVA PARA EL HITO 4

Requisitos previos para poder dar solución:

- Crear la base de datos ONG.

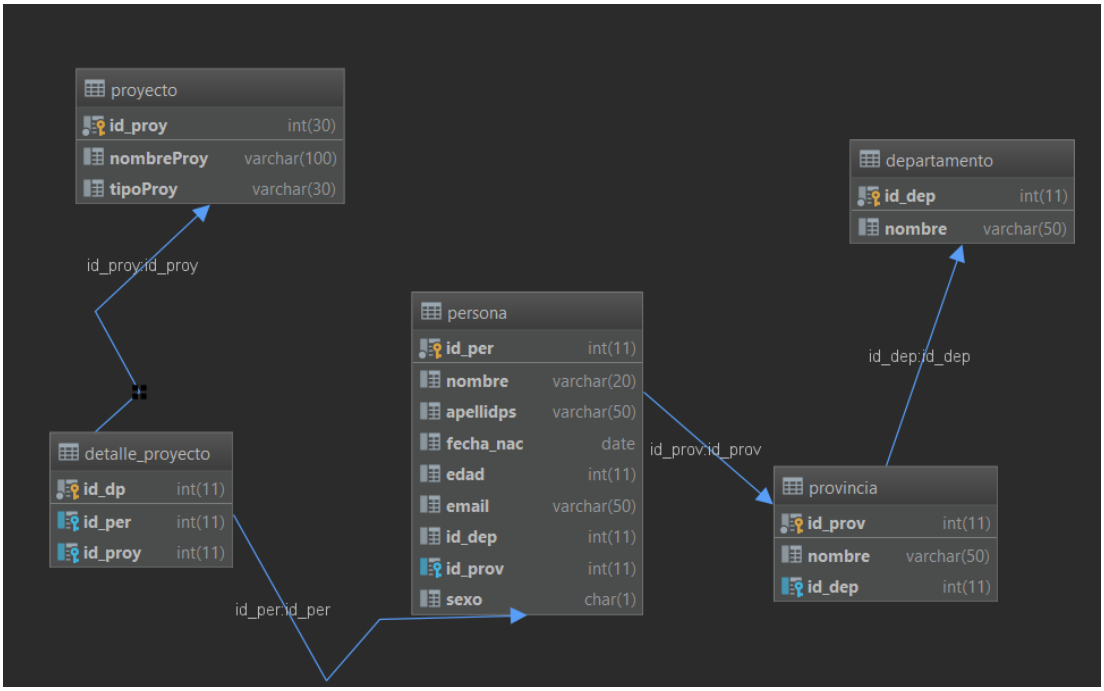


Resolver los siguientes ejercicios.

Empezamos creando la base de datos ONG y las tablas correspondientes:

```
CREATE DATABASE ONG;
USE ONG;
DROP TABLE ong.proyecto;

CREATE TABLE proyecto
(
  id_proy INT(30) AUTO_INCREMENT PRIMARY KEY NOT NULL,
  nombreProy VARCHAR(100),
  tipoProy VARCHAR(30)

);
DROP TABLE detalle_proyecto;
CREATE TABLE detalle_proyecto
(
  id_dp INT(11) AUTO_INCREMENT PRIMARY KEY NOT NULL,
  id_per INT(11),
  id_proy INT(11),
  FOREIGN KEY (id_proy) REFERENCES proyecto (id_proy)

);


DROP TABLE ong.provincia;
CREATE TABLE provincia
(
  id_prov int(11) AUTO_INCREMENT PRIMARY KEY NOT NULL,
  nombre varchar(50),
  id_dep int(11),
  FOREIGN KEY (id_dep) REFERENCES departamento (id_dep)
);

CREATE TABLE departamento
(
  id_dep INT(11) AUTO_INCREMENT PRIMARY KEY NOT NULL,
  nombre varchar(50)


);

CREATE TABLE persona
(
  id_per INT(11) AUTO_INCREMENT PRIMARY KEY NOT NULL,
  nombre varchar(20),
  apellidps varchar(50),
  fecha_nac date,
```

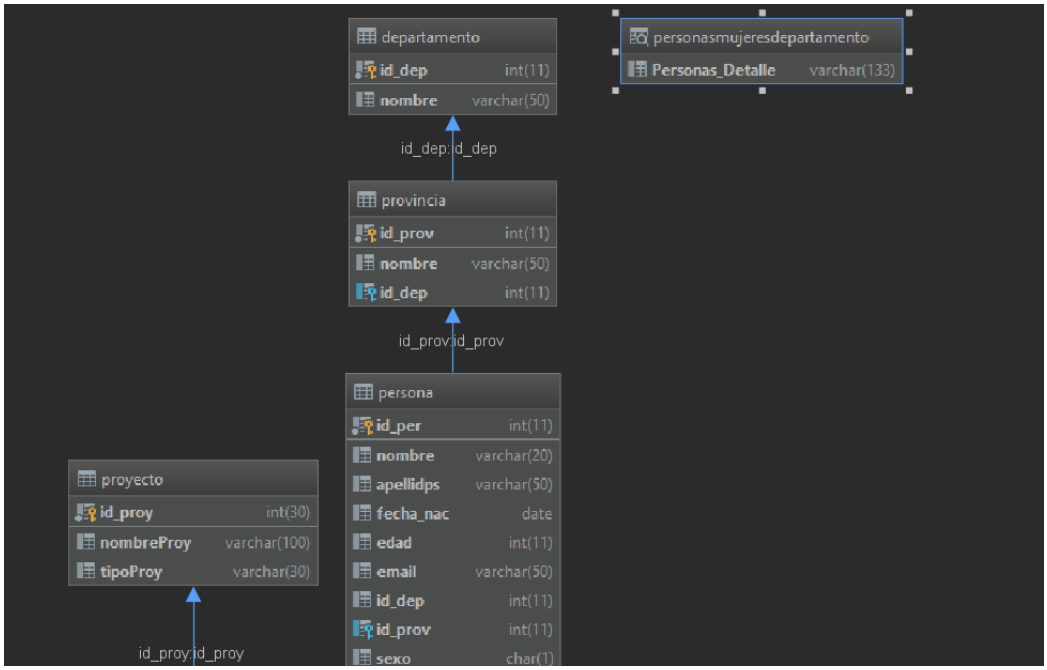
```
  edad int(11),
  email varchar(50),
  id_dep int(11),
  id_prov int(11),
  sexo char(1)

);
ALTER TABLE provincia ADD column id_dep integer;
ALTER TABLE ong.persona ADD FOREIGN KEY (id_prov) REFERENCES ong.provincia(id_prov);
```



- Crear una Vista.
  - La Vista debe de llamarse **personasMujeresDepartamento**.
  - La consulta de la vista debe reflejar como campos nombres y apellidos concatenados, la edad y la fecha de nacimiento.
  - Obtener todas las personas del sexo femenino que hayan nacido en el departamento de Cochabamba en donde la fecha sea:
    - **fecha_nac** = **'1993-10-10'**.

```
DROP FUNCTION Datos_Personas;
CREATE FUNCTION Datos_Personas(DNac date,dep VARCHAR(100),sexo CHAR) RETURNS BOOLEAN
 BEGIN
  DECLARE res BOOLEAN DEFAULT FALSE;
   SET res = IF(DNac = '1993-10-10' and dep ='Cochabamba' AND sexo = 'f',true,false);
  RETURN res;
 end;
```

```
SELECT Datos_Personas('1993-10-10','cochabamba','f');


CREATE VIEW personasMujeresDepartamento AS
  SELECT CONCAT(per.nombre,'-',per.apellidps,'-',per.fecha_nac,'-',dep.nombre) As Personas_Detalle
  FROM persona as per
    INNER JOIN departamento as dep on per.id_dep = dep.id_dep
  WHERE Datos_Personas(per.fecha_nac,dep.nombre,per.sexo);
```

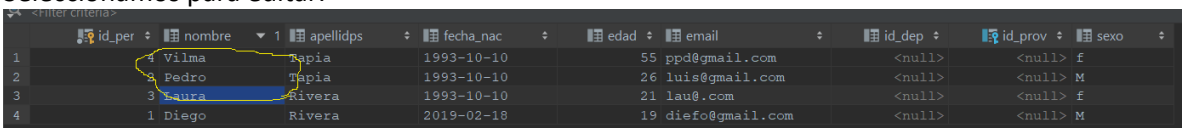


- Crear una TRIGGER.
  - El trigger debe de llamarse **backupPersonasUpdate**.
  - El evento debe de ejecutarse en un **BEFORE UPDATE**.
  - Crear un tabla llamada **backupPersonasUpdate**.
  - Esta nueva tabla tiene que tener 2 campos **oldNombre** y **newNombre**.
  - Cada vez que se modifique el registro de una persona, se debe de insertar un nuevo registro en la tabla **backupPersonasUpdate**, en donde el primer campo es el nombre que se está modificando y el segundo campo es el nuevo nombre.

```
DROP TABLE backupPersonasUpdate;
CREATE TABLE backupPersonasUpdate
(
  operation CHAR(1) NOT NULL,
  stamp TIMESTAMP NOT NULL,
  user TEXT NOT NULL,
  hostname TEXT NOT NULL,
  viejo_Nombre varchar(50),
  nuevo_Nombre VARCHAR(50),
```

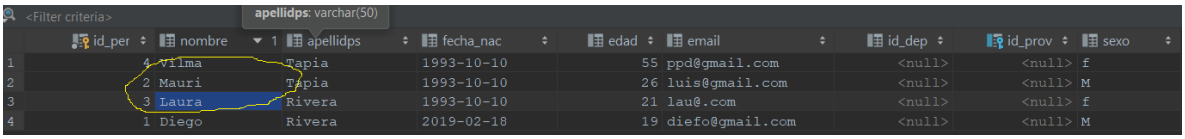
```
  id_Persona integer not null
);

DROP TRIGGER backupPersonasUpdate;
CREATE TRIGGER backupPersonasUpdate
  BEFORE UPDATE ON persona
  FOR EACH ROW
  BEGIN
    INSERT INTO backupPersonasUpdate(operation, stamp, user, hostname, viejo_Nombre,
nuevo_Nombre, id_Persona)
     SELECT 'U' ,now(),user(),@@hostname,OLD.nombre,NEW.nombre,OLD.id_per;
   END;
```

Seleccionamos para editar:



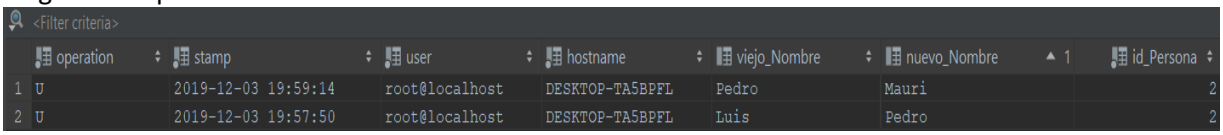
	id_per	nombre	apellidps	fecha_nac	edad	email	id_dep	id_prov	sexo
1	4	Vilma	Tapia	1993-10-10	55	ppd@gmail.com	<null>	<null>	f
2	2	Pedro	Tapia	1993-10-10	26	luis@gmail.com	<null>	<null>	M
3	3	Laura	Rivera	1993-10-10	21	lau@.com	<null>	<null>	f
4	1	Diego	Rivera	2019-02-18	19	diefo@gmail.com	<null>	<null>	M

Terminamos de editar



	id_per	nombre	apellidps	fecha_nac	edad	email	id_dep	id_prov	sexo
1	4	Vilma	Tapia	1993-10-10	55	ppd@gmail.com	<null>	<null>	f
2	2	Mauri	Tapia	1993-10-10	26	luis@gmail.com	<null>	<null>	M
3	3	Laura	Rivera	1993-10-10	21	lau@.com	<null>	<null>	f
4	1	Diego	Rivera	2019-02-18	19	diefo@gmail.com	<null>	<null>	M

Se guarda el proceso:

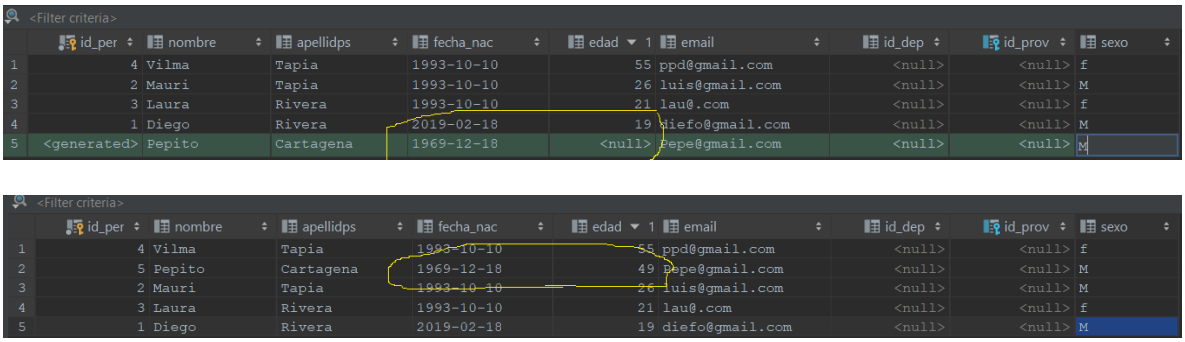


	operation	stamp	user	hostname	viejo_Nombre	nuevo_Nombre	id_Persona
1	U	2019-12-03 19:59:14	root@localhost	DESKTOP-TA5BPFL	Pedro	Mauri	2
2	U	2019-12-03 19:57:50	root@localhost	DESKTOP-TA5BPFL	Luis	Pedro	2

- Crear una TRIGGER.
  - El trigger debe de llamarse **calculaEdad**.
  - El evento debe de ejecutarse en un **BEFORE INSERT**.
  - Cada vez que se inserte un registro en la tabla PERSONA, el trigger debe de calcular la edad en función a la edad en función a la fecha de nacimiento.

```
use ong;
CREATE TRIGGER CEdad
BEFORE INSERT ON ong.persona
 FOR EACH ROW
 BEGIN

 DECLARE anios  INT(11) DEFAULT 0;

    SELECT TIMESTAMPDIFF(YEAR,NEW.fecha_nac,CURDATE())
     INTO anios;
```

```
   SET NEW.edad = anios;
END;
```

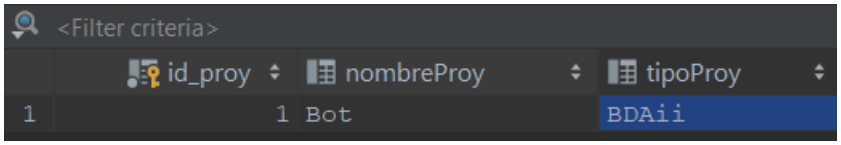


	id_per	nombre	apellidps	fecha_nac	edad 1	email	id_dep	id_prov	sexo
1	4	Vilma	Tapia	1993-10-10	55	ppd@gmail.com	<null>	<null>	f
2	2	Mauri	Tapia	1993-10-10	26	luis@gmail.com	<null>	<null>	M
3	3	Laura	Rivera	1993-10-10	21	lau@.com	<null>	<null>	f
4	1	Diego	Rivera	2019-02-18	19	diefo@gmail.com	<null>	<null>	M
5	<generated>	Pepito	Cartagena	1969-12-18	<null>	Pepe@gmail.com	<null>	<null>	M

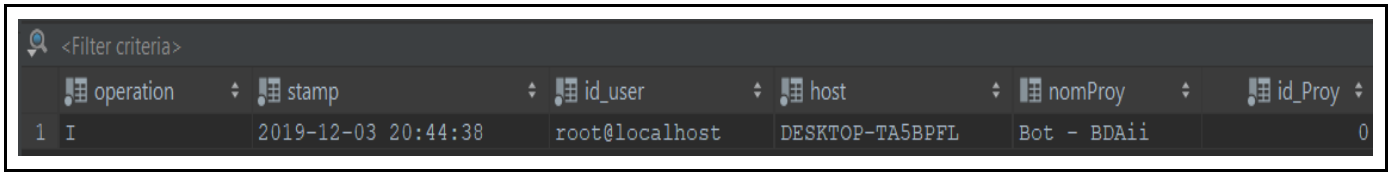
	id_per	nombre	apellidps	fecha_nac	edad 1	email	id_dep	id_prov	sexo
1	4	Vilma	Tapia	1993-10-10	55	ppd@gmail.com	<null>	<null>	f
2	5	Pepito	Cartagena	1969-12-18	49	Pepe@gmail.com	<null>	<null>	M
3	2	Mauri	Tapia	1993-10-10	26	luis@gmail.com	<null>	<null>	M
4	3	Laura	Rivera	1993-10-10	21	lau@.com	<null>	<null>	f
5	1	Diego	Rivera	2019-02-18	19	diefo@gmail.com	<null>	<null>	M

- Crear un TRIGGER BEFORE o AFTER INSERT para la tabla PROYECTO.
  - El nombre del TRIGGER deberá ser **triggerInsert_Proyecto**
  - Deberá de crear una tabla de AUDITORIA en donde esta tabla deberá de tener 2 columnas.
    - El 1er campo debe de guardar el nuevo idProy insertado.
    - El 2do campo debe de guardar el nombre de proyecto y tipo de proyecto concatenados separados por un espacio.
      - Ejemplo: nombreProy: ‘’Educacion para Ancianos”, tipoProy: “Educacion”.
      - Resultado: ‘’Educacion para Ancianos - Educacion”.

```
CREATE TABLE AUDITORIA
(
 operation CHAR(1) NOT NULL,
 stamp TIMESTAMP NOT NULL,
 id_user TEXT NOT NULL,
 host TEXT NOT NULL,
 nomProy varchar(100),
 id_Proy integer not null
);

CREATE TRIGGER triggerInsert_Proyecto
 BEFORE INSERT on proyecto
 FOR EACH ROW
 BEGIN
  INSERT INTO auditoria(operation, stamp, id_user, host, nomProy, id_Proy)
   SELECT 'I',NOW(),USER(),@@hostname,CONCAT(New.nombreProy,' - ',NEW.tipoproy),NEW.id_proy;
 END;
```



	id_proy	nombreProy	tipoProy
1	1	Bot	BDAii



	operation	stamp	id_user	host	nomProy	id_Proy
1	I	2019-12-03 20:44:38	root@localhost	DESKTOP-TA5BPFL	Bot - BDAii	0

- Crear un TRIGGER BEFORE o AFTER para INSERT, UPDATE y DELETE para la tabla DETALLE_PROYECTO.
  - El nombre del TRIGGER deberá ser **triggerForDetProy**
  - Deberá de crear una tabla de AUDITORIA similar al siguiente ejemplo.
    -

```
CREATE TABLE auditoria_detalle_proyecto (
 operation          CHAR(1)   NOT NULL, -- ('D', 'U', 'I')
 stamp              TIMESTAMP NOT NULL,
 userid             TEXT    NOT NULL,
 ……
 todos los campos de la tabla detalle_proyecto
);
```

    -
  - Debe de crear un Procedimiento Almacenado o Stored Procedure (SP).
    - Este SP debe recibir parámetros de entrada con los valores a insertar en la tabla de AUDITORIA.
  - Los TRIGGERS deben de utilizar este SP, cada trigger debe de enviar los parámetros de inserción de la tabla de AUDITORIA

```
CREATE TABLE auditoria_detalle_proyecto
(
 operation    CHAR(1)   NOT NULL,
 stamp        TIMESTAMP NOT NULL,
 userid       TEXT   NOT NULL,
 id_proy integer not null,
 nombreProy varchar(50),
 tipoproy TEXT,
 diaDelaSemana varchar(12)
);
```

- Crear un TRIGGER BEFORE INSERT para la tabla DETALLE_PROYECTO.
  - Si es LUNES o MARTES o MIÉRCOLES o JUEVES o VIERNES insertar adicionalmente los datos en su tabla de AUDITORÍA.
    - Adicionar un nuevo campo (**diaDelaSemana** varchar(12)) a la tabla **auditoria_detalle_proyecto.**
      - En este campo debe de almacenarse el dia en se insertó los nuevos registros.

- Si es SÁBADO o DOMINGO mostrar un mensaje indicando que no se permite inserciones los fines de semana. Por lo tanto no se debe de insertar en la tabla detalle_proyecto y tampoco en su tabla de auditoria.