UNIVERSIDAD PRIVADA "FRANZ TAMAYO"

DEFENSA HITO 4

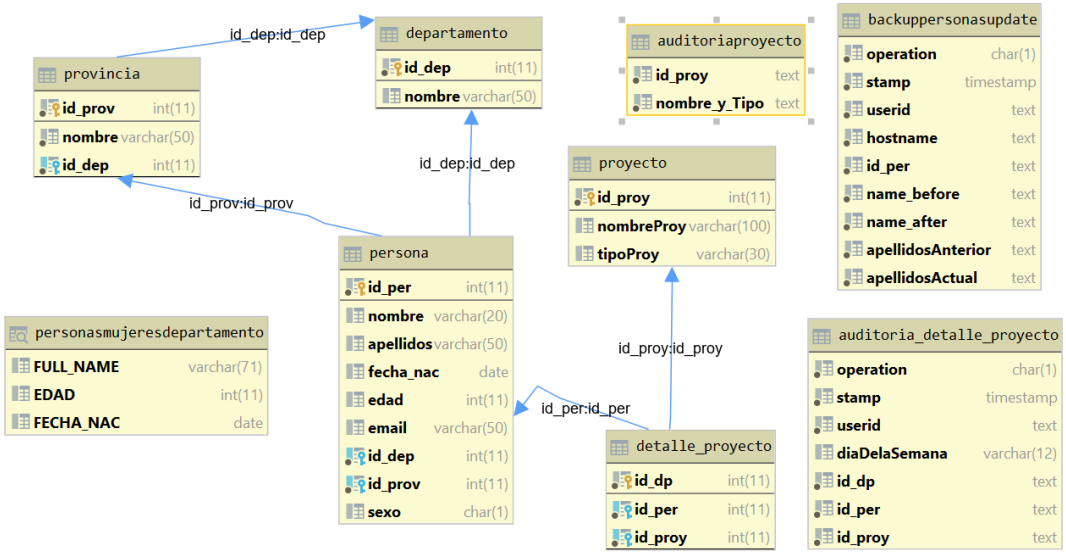
BASE DE DATOS II

ALUMNO: Adán Abinadi Tudela Cabero

Paralelo 1 - Ing. De Sistemas

DOCENTE: Lic. William R. Barra Paredes.

Cochabamba, 2 de diciembre de 2019



- 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'`.

```
# E1 CREAR UNA VISTA
CREATE VIEW personasMujeresDepartamento AS
    SELECT CONCAT(P.nombre,' ',P.apellidos) AS FULL_NAME,
           P.edad EDAD,P.fecha_nac FECHA_NAC
    FROM PERSONA AS P
    INNER JOIN DEPARTAMENTO D on P.id_dep = D.id_dep
    WHERE p.sexo='F' AND p.fecha_nac = '1993-10-10' AND D.nombre =
'Cochabamba';
```

- 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.

```
# E2 CREAR UN TRIGGER
CREATE TABLE backupPersonasUpdate
(
  #columnas de auditoria
   operation    CHAR(1)   NOT NULL, -- ('D', 'U', 'I')
   stamp        TIMESTAMP NOT NULL,
   userid       TEXT      NOT NULL,
   hostname     TEXT      NOT NULL,
   #columnas adicionales de la tabla persona
   id_per     TEXT      NOT NULL,
   name_before  TEXT      NOT NULL,
   name_after   TEXT      NOT NULL,
   apellidosAnterior TEXT      NOT NULL,
    apellidosActual TEXT      NOT NULL

   #fecha_nac date,
   #edad TEXT      NOT NULL
);
drop table backupPersonasUpdate;
CREATE TRIGGER backupPersonasUpdate
    BEFORE UPDATE ON PERSONA
   FOR EACH ROW
BEGIN
   INSERT INTO backupPersonasUpdate
       (operation, stamp, userid, hostname, id_per, name_before, name_after,
apellidosAnterior,apellidosActual)SELECT
        'U', now(), user(), @@hostname,OLD.id_per,
OLD.nombre,NEW.nombre,OLD.apellidos,NEW.apellidos;
END;
```

- 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.

```
# E3 CREAR UN TRIGGER edad

CREATE TRIGGER calculaEdadPersona
    before insert on PERSONA
    for each row
    begin
        declare edad integer default  0;
        select timestampdiff(year ,NEW.fecha_nac,curdate())as edad into
edad;
        set new.edad = edad;
    end;
```

- 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 auditoriaProyecto
(
  id_proy    TEXT      NOT NULL,
  nombre_y_Tipo     TEXT      NOT NULL
);

create trigger triggerInsert_Proyecto
    before insert on PROYECTO
    for each row
    begin
        insert into auditoriaProyecto (id_proy, nombre_y_Tipo) SELECT
        new.id_proy,concat(new.nombreProy,' - ',new.tipoProy);
    end;
```

- 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.
  - 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, -- ('D', 'U', 'I')
    stamp     TIMESTAMP NOT NULL,#(now(), user())
    userid    TEXT      NOT NULL,
    diaDelaSemana VARCHAR(12),
    #DETALLE_PROYECTO
    id_dp text not null ,
    id_per text not null,
    id_proy text not null
);
DROP TABLE auditoria_detalle_proyecto;
CREATE TRIGGER triggerForDetProy
    BEFORE INSERT ON DETALLE_PROYECTO
   FOR EACH ROW
BEGIN
   INSERT INTO auditoria_detalle_proyecto
       (operation, stamp, userid, id_dp, id_per, id_proy) SELECT
        'I', now(), user(),NEW.id_dp,NEW.id_per,NEW.id_proy;
END;
DROP TRIGGER triggerForDetProy;

CREATE TRIGGER triggerForDetProyUPDATE
    BEFORE UPDATE ON DETALLE_PROYECTO
   FOR EACH ROW
BEGIN
   INSERT INTO auditoria_detalle_proyecto
       (operation, stamp, userid,diaDelaSemana, id_dp, id_per, id_proy)
SELECT
        'U', now(),
user(),DAYNAME(CURRENT_DATE()),NEW.id_dp,NEW.id_per,NEW.id_proy;
END;
CREATE TRIGGER triggerForDetProyDELETE
    BEFORE DELETE ON DETALLE_PROYECTO
   FOR EACH ROW
BEGIN
   INSERT INTO auditoria_detalle_proyecto
       (operation, stamp, userid,diaDelaSemana, id_dp, id_per, id_proy)
SELECT
        'D', now(),
user(),DAYNAME(CURRENT_DATE()),OLD.id_dp,OLD.id_per,OLD.id_proy;
END;
```

- 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.

```
#E5 TRIGGER
SELECT DAYNAME(CURRENT_DATE());
SELECT CURRENT_DATE();

CREATE TRIGGER triggerForDetProyINSERT2
     BEFORE INSERT ON DETALLE_PROYECTO
   FOR EACH ROW
BEGIN
    DECLARE DIA VARCHAR(12) DEFAULT DAYNAME(CURRENT_DATE());
    if DIA = 'Saturday'or DIA = 'Sunday'
        then
            signal sqlstate '45000' set message_text = 'NO SE ADMITE
INSERCICIONES FINES DE SEMANA';
    ELSE
            INSERT INTO auditoria_detalle_proyecto
               (operation, stamp, userid,diaDelaSemana, id_dp, id_per,
id_proy) SELECT
                'I', now(), user(),DIA,NEW.id_dp,NEW.id_per,NEW.id_proy;
    end if ;
END;
```