BASES DE DATOS 2

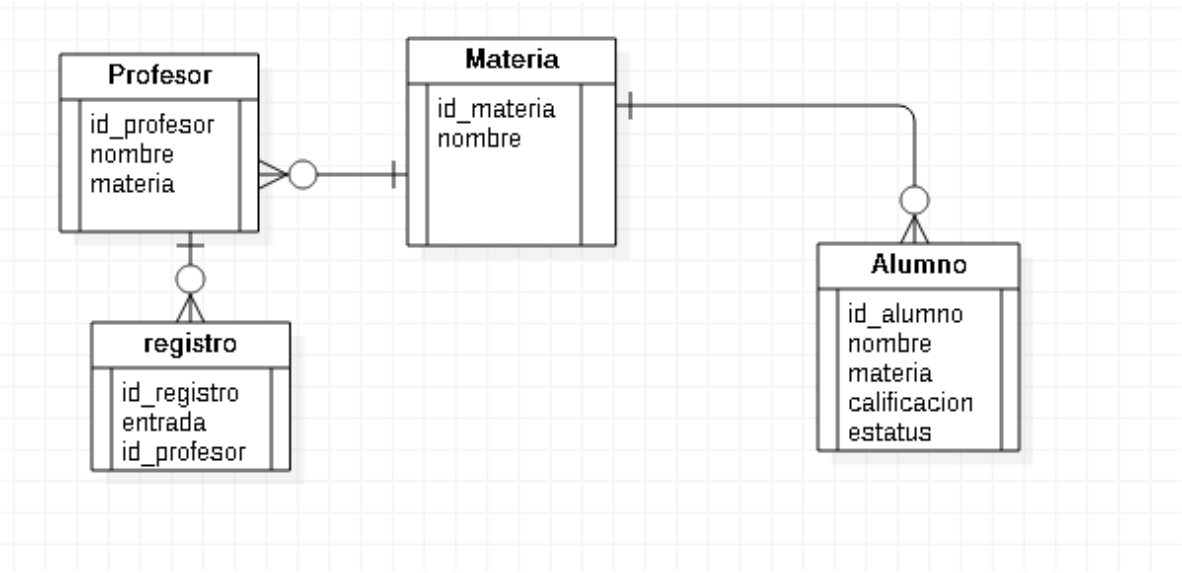
Triggers de automatización 2

Descripción breve

Una universidad desea elaborar un sistema data-céntrico para validar calificaciones de alumnos y la hora de registro de los profesores. El profesor se identifica por su clave de profesor, y se le asigna nombre de materia que impartirá y alumnos registrados a dicha materia, además de un campo de tipo Date para que diariamente actualize su hora de entrada a su materia (campo transitivo). La materia se identifica por el nombre de la materia y un número de secuencia consecutivo iniciando desde 1. Los alumnos se identifican con su número de cuenta y por el identificador de la materia, que debe de coincidir con el del profesor. También cuenta con campo nombre, calificación final y estatus. Este último campo puede ser solamente “aprobado” o “reprobado”, dependiendo de la calificación final

Luis Ivan Herrera Equihua
12386391

El diagrama Entidad-relación de este problema



Guardar un maestro cuyo nombre sea Juan López y que esté asignado a la materia Programación. Tendrá un solo alumno cuyo nombre es Maria Sanchez y cuya calificación final es 8.5.

```
create or replace procedure guardar_profesor(
  my_id in varchar,
  my_nombre in varchar,
  my_materia in integer)
  as
  begin
    insert into profesor values (my_id,my_nombre,my_materia);
  end;
/
```

alida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 2 en 0.015 segundos

	ID_PROFESOR	NOMBRE	MATERIA
1	1234567	Luis	1
2	1234565	Juan López	1

```
create or replace procedure guardar_alumnol(
  my_id in varchar,
  my_nombre in varchar,
  my_materia in integer,
  my_calificacion in float)
  as
  begin
    insert into alumno (id_alumno,nombre,materia,calificacion)
      values(my_id,my_nombre,my_materia,my_calificacion);
  end;
/
```

alida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 1 en 0.089 segundos

	ID_ALUMNO	NOMBRE	MATERIA	CALIFICACION	ESTATUS
1	123456	MAría Sanchez	1	8.5	(null)

Genere un disparador de tipo automatización para impedir que el profesor se pueda registrar después del minuto 15 de cada hora (para probarlo, cambia el minuto 15, por los 15 segundos de cada minuto).

```
74 create or replace procedure guardar_registro(
75   my_id out integer,
76   my_profesor in varchar,
77   my_entrada out char)
78   as
79   valor char(60);
80   begin
81     valor:=to_char(sysdate,'DD-MON-YYYY HH:MI:SS');
82     my_entrada:=valor;
83     select sec_registro.nextval into my_id from dual;
84     insert into registro values(my_id,my_profesor,my_entrada);
85   end;
86 /
```

```
135 --creación de disparadores
136 create or replace trigger disp_entrada before insert on registro
137   for each row
138   declare
139     minuto char(40);
140   begin
141     minuto:=to_char(sysdate,'mi');
142     if to_number(minuto) > 15 then
143       raise_application_error(-20001,'Ha llegado tarde!, no se puede regitrar');
144     else
145       DBMS_OUTPUT.PUT_LINE('Registrado!');
146     end if;
147   end;
148   /
```

Prueba pasados los minutos limite

```
182 --Prueba de registros
183 declare
184 vl integer;
185 valor char(60);
186   begin
187   guardar_registro(vl,'1234567',valor);
188 end;
189 /
190
```

Salida de Script | Resultado de la Consulta

Tarea terminada en 0.193 segundos

```
Informe de error -
ORA-20001: Ha llegado tarde!, no se puede regitrar
```

```
207 select * from registro;
208
```

Salida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 0 en 0.002 segundos

ID_REGIS...	ID_PROF...	ENTRADA

Antes de los minutos limite

```
183 declare
184 vl integer;
185 valor char(60);
186   begin
187   guardar_registro(vl,'1234567',valor);
188 end;
189 /
190
```

Salida de Script | Resultado de la Consulta

Tarea terminada en 0.006 segundos

```
Procedimiento PL/SQL terminado correctamente.
```

```
207 select * from registro;
208
```

Salida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 1 en 0.006 segundos

	ID_REGISTRO	ID_PROFESOR	ENTRADA
1	34	1234567	01-ABR-2018 02:38:29

Genere un disparador que dependiendo de la calificación inserte automáticamente aprobado o reprobado en el campo “estatus”, sin que este se tenga que hacer manualmente, todo ello dependiendo de la calificación ingresada.

intenté de todas las maneras para hacer un trigger y tenía el error de que la tabla se encontraba mutando inclusive usando cursores, tablas :new-:old, after y before así que la validación la puse dentro del SP :^)

```
create or replace procedure guardar_alumno2(
   my_id in varchar,
   my_nombre in varchar,
   my_materia in integer,
   my_calificacion in float)
   as
   begin
   if my_calificacion < 6 then
     insert into alumno (id_alumno,nombre,materia,calificacion,estatus)
       values(my_id,my_nombre,my_materia,my_calificacion,'Reprobado');
   else
     insert into alumno (id_alumno,nombre,materia,calificacion,estatus)
       values(my_id,my_nombre,my_materia,my_calificacion,'Aprobado');
   end if;
   end;
 /
```

```
create or replace procedure update_alumno(
   my_id in varchar,
   my_calif in varchar)
   as
   begin
   if my_calif < 6 then
     update alumno set calificacion=my_calif,
                       estatus='Reprobado'
       where  id_alumno=my_id;
   else
         update alumno set calificacion=my_calif,
                       estatus='Aprobado'
       where  id_alumno=my_id;
   end if;
   end;
 /
```

Prueba

```
declare
begin
  update_alumno('123456',9.3);
end;
/

select * from alumno;
select * from materia;
select * from profesor;
select * from registro;
```

Salida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 1 en 0.006 segundos

	ID_ALUMNO	NOMBRE	MATERIA	CALIFICACION	ESTATUS
1	123456	MAría Sanchez	1	9.3	Aprobado

```
declare
begin
  update_alumno('123456',4.3);
end;
/

select * from alumno;
select * from materia;
select * from profesor;
select * from registro;
```

ida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 1 en 0.002 segundos

	ID_ALUMNO	NOMBRE	MATERIA	CALIFICACION	ESTATUS
1	123456	MAría Sanchez	1	4.3	Reprobado

Código

```
--PRactica automatización
--drop table profesor;
create table profesor(
  id_profesor varchar(9) primary key,
  nombre varchar(60),
  materia integer);

create table registro(
  id_registro integer primary key,
  id_profesor varchar(9),
  entrada varchar(60));

create table alumno(
  id_alumno varchar(9) primary key,
  nombre varchar (60),
  materia integer,
  calificacion float,
  estatus varchar(10));

create table materia(
  id_materia integer primary key,
  nombre varchar (80)
  );

--Creación de foreign keys
alter table profesor
  add constraint fk1_profesor_materia
  foreign key (materia)
  references materia(id_materia);

alter table registro
  add constraint fk1_registro_profesor
  foreign key (id_profesor)
  references profesor (id_profesor);

alter table alumno
  add constraint fk1_alumno_materia
  foreign key (materia)
  references materia (id_materia);

--creación de secuencia para la materia y registros
create sequence sec_materia
  start with 1
  increment by 1
  nomaxvalue;
```

```
create sequence sec_registro
  start with 1
  increment by 1
  nomaxvalue;

--procedimiento almacenado para guardar
create or replace procedure guardar_materia(
  my_id out integer,
  my_nombre in varchar)
  as
  begin
    select sec_materia.nextval into my_id from dual;
    insert into materia values (my_id,my_nombre);
  end;
  /
  select  * from profesor;

create or replace procedure guardar_profesor(
  my_id in varchar,
  my_nombre in varchar,
  my_materia in integer)
  as
  begin
    insert into profesor values (my_id,my_nombre,my_materia);
  end;
/

create or replace procedure guardar_registro(
  my_id out integer,
  my_profesor in varchar,
  my_entrada out char)
  as
  valor char(60);
  begin
    valor:=to_char(sysdate,'DD-MON-YYYY HH:MI:SS');
    my_entrada:=valor;
    select sec_registro.nextval into my_id from dual;
    insert into registro values(my_id,my_profesor,my_entrada);
  end;
/

create or replace procedure guardar_alumno1(
  my_id in varchar,
  my_nombre in varchar,
  my_materia in integer,
```

```
  my_calificacion in float)
  as
  begin
    insert into alumno (id_alumno,nombre,materia,calificacion)
      values(my_id,my_nombre,my_materia,my_calificacion);
  end;
/


create or replace procedure guardar_alumno2(
  my_id in varchar,
  my_nombre in varchar,
  my_materia in integer,
  my_calificacion in float)
  as
  begin
  if my_calificacion < 6 then
    insert into alumno (id_alumno,nombre,materia,calificacion,estatus)
      values(my_id,my_nombre,my_materia,my_calificacion,'Reprobado');
  else
    insert into alumno (id_alumno,nombre,materia,calificacion,estatus)
      values(my_id,my_nombre,my_materia,my_calificacion,'Aprobado');
  end if;
  end;
/

create or replace procedure update_alumno(
  my_id in varchar,
  my_calif in varchar)
  as
  begin
  if my_calif < 6 then
    update alumno set calificacion=my_calif,
                      estatus='Reprobado'
      where  id_alumno=my_id;
  else
        update alumno set calificacion=my_calif,
                      estatus='Aprobado'
      where  id_alumno=my_id;
  end if;
  end;
/

--creación de disparadores
create or replace trigger disp_entrada before insert on registro
  for each row
```

```
  declare
    minuto char(40);
  begin
    minuto:=to_char(sysdate,'mi');
    if to_number(minuto) > 15 then
      raise_application_error(-20001,'Ha llegado tarde!, no se puede
regitrar');
    else
      DBMS_OUTPUT.PUT_LINE('Registrado!');
    end if;
  end;
  /

create or replace trigger disp_calificacion before update on alumno
  for each row
  declare
  begin
      if :new.calificacion < 6 then
        update alumno set estatus='Reprobado' where
:old.id_alumno=:new.id_alumno;
      else
        update alumno set estatus='Aprobado' where
:old.id_alumno=:new.id_alumno;
      end if;
    end;
  /

drop trigger disp_calificacion;


declare
  valor integer;
begin
  guardar_materia(valor,'Programación');
end;
/
select * from materia;

begin
  guardar_profesor('1234565','Juan López',1);
end;
/
select * from profesor;

--Prueba de registros
declare
```

```
v1 integer;
valor char(60);
  begin
  guardar_registro(v1,'1234567',valor);
end;
/

--Prueba de reprobar
declare
begin
  guardar_alumno2('123456','MAría Sanchez',1,8.5);
end;
/

declare
begin
  update_alumno('123456',4.3);
end;
/

select * from alumno;
select * from materia;
select * from profesor;
select * from registro;

drop trigger disp_calificacion;
```