Control de Evaluaciones

DESCRIPCIÓN BREVE

El presente documento presento todo lo visto y aprendido en la asignatura de Base de Datos como parte de la entrega del proyecto integrador.

MONTERO MUÑIZ MONSERRAT

Base de Datos

Contenido

Control de evaluaciones

Los profesores de la asignatura de Bases de Datos de una Facultad deciden crear una base de datos que contenga la información de los resultados de las evaluaciones realizadas a los alumnos.

Para realizar el diseño se sabe que:

- Los alumnos están definidos por su número de matrícula, nombre y el grupo al que asisten a clase.
- Dichos alumnos realizan dos tipos de evaluaciones a lo largo del curso académico:
  - Exámenes escritos: cada alumno realiza varios a lo largo del curso, y se definen por el número de examen, el número de preguntas de que consta y la fecha de realización (la misma para todos los alumnos que realizan el mismo examen). Evidentemente, es importante almacenar la calificación de cada alumno por examen.
  - Prácticas: se realiza un número indeterminado de ellas durante el curso académico, algunas serán en grupo y otras individuales. Se definen por un código de práctica, título y el grado de dificultad. En este caso los alumnos pueden examinarse de cualquier práctica cuando lo deseen, debiéndose almacenar la fecha y nota obtenida.
- En cuanto a los profesores, únicamente interesa conocer (además de sus datos personales: número de personal y nombre), quien es el que ha diseñado cada práctica, sabiendo que en el diseño de una práctica puede colaborar más de uno, y que un profesor puede diseñar más de una práctica. Interesa, además, la fecha en que ha sido diseñada cada práctica por el profesor correspondiente.

Modelo E-R

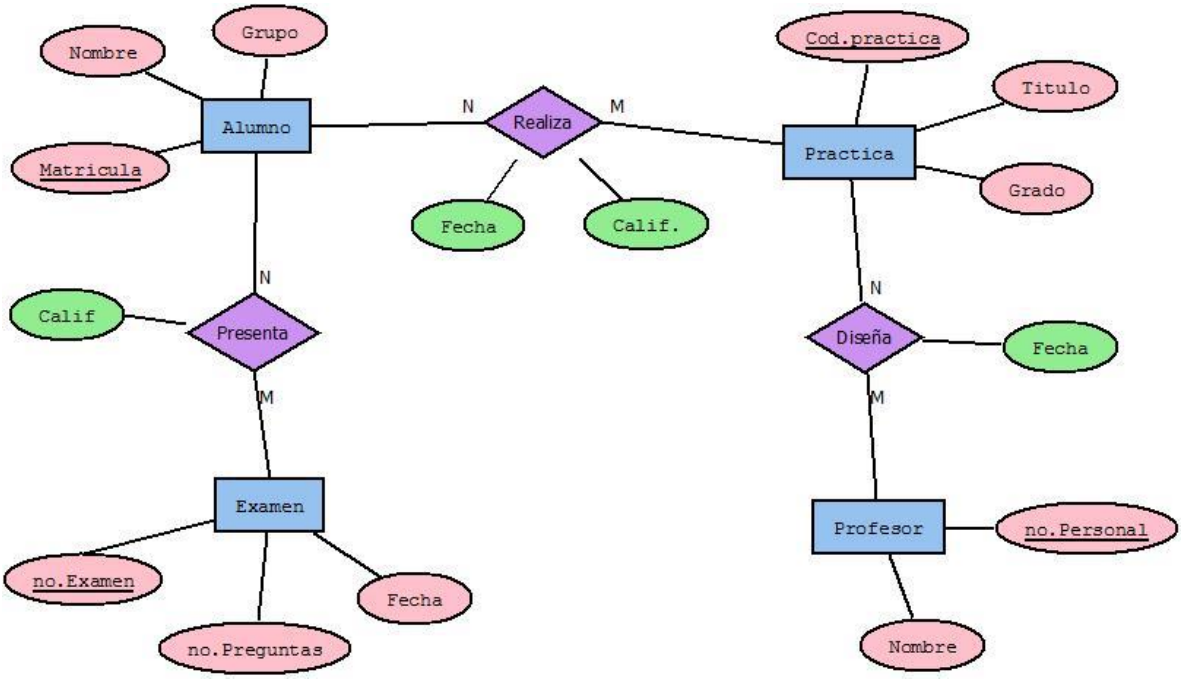

Ilustración 1 MER Control de Evaluaciones

Modelo relacional

Alumno (Matricula, Nombre, Grupo)

Examen (no.examen, no.preguntas, Fecha)

Practica (cod.practica, Titulo, Grado)

Profesor (no.Personal, Nombre)

Presenta (no.examen, matricula,Calif)

Realiza (matricula, cod.practica, Fecha, Calif)

Diseña (cod.practica, no.personal, Fecha)

Diccionario de datos

Tabla Alumno				
Nombre	Tipo	Longitud	Descripción	Llave
matricula	varchar	(10)	Es un numero con el cual se identifica el alumno ante la escuela	PK
nombre	varchar	(30)	Nombre completo del alumno	-
grupo	Int	(2)	Es el numero con el que se identifica el grupo en el que esta inscrito el alumno	-

Tabla Profesor				
Nombre	Tipo	Longitud	Descripción	Llave
no_personal	varchar	(10)	Es un numero con el cual es identificado cada una de los profesores	PK
nombre	varchar	(30)	Nombre completo del profesor	-

Tabla Realiza				
Nombre	Tipo	Longitud	Descripción	Llave
matricula	varchar	(10)	Es un numero con el cual se identifica el alumno ante la escuela	FK
cod_practica	int	(5)	Código con el que se identifica la practica que realizo el alumno	FK
fecha	varchar	(10)	Es el dato en el que el alumno realizo dicha practica	-

calif	float	(5)	Es la calificación que obtiene el alumno en la practica entregada	-

Normalización a las tablas

Una relación está en primera forma normal si todo atributo contiene un valor unitario. Por lo tanto, todas las tablas cumplen con dicha forma.

Una relación se encuentra en segunda forma normal si y solo si la relación está en 1FN, y cada atributo de la tabla que no forma parte de la clave depende funcionalmente de forma completa de la llave primaria. Es por esto que las tablas también cumplen con la 2FN.

Una relación está en 3FN si está en 2FN y todo atributo que no sea llave no depende de un atributo que no sea llave. La 3FN permite eliminar las redundancias para que ningún atributo dependa de otro de forma transitiva. Por lo tanto, todas las tablas cumplen con dicha forma.

En base a ello concluyo que mi esquema se encuentra Normalizado.

Alumno (Matricula, Nombre, Grupo)

Examen (no.examen, no.preguntas, Fecha)

Practica (cod.practica, Titulo, Grado)

Profesor (no.personal, Nombre)

Presenta (no.examen, matricula,Calif)

Realiza (matricula, cod.practica, Fecha, Calif)

Diseña (cod.practica, no.personal, Fecha)

Crear en MySQL la base de datos y las tablas

```
create database Control;
use Control;
create table Alumno (matricula varchar(10) primary key, nombre varchar(30), grupo int);

create table Examen (no_examen int primary key, no_preguntas int, fecha varchar(10));

create table Practica (cod_practica int primary key, titulo varchar(20), grado varchar (10));

create table Profesor (no_personal varchar(10) primary key, nombre varchar(30));

create table Presenta (no_examen int, matricula varchar(10), calif float (5), foreign key (no_examen) references Examen (no_examen), foreign key (matricula) references Alumno(matricula));
```

create table Realiza (matricula varchar(10), cod_practica int , fecha varchar (10), calif float (5), foreign key (matricula) references Alumno(matricula), foreign key (cod_practica) references Practica(cod_practica));

create table Diseña (cod_practica int, no_personal varchar(10), fecha varchar(10), foreign key (cod_practica) references Practica (cod_practica), foreign key (no_personal) references Profesor (no_personal));

Agregar entre 5 y 10 registros a cada tabla

insert into Alumno values ('s16013961', 'Monserrat Montero Muñiz',2);

insert into Alumno values ('s16014096', 'Carlos Huerta Araujo',1);

insert into Alumno values ('s16012594', 'Miguel Perez',1);

insert into Alumno values ('s16011020', 'Isabel Lopez',2);

insert into Alumno values ('s12018602','Arturo Jimenez',2);

insert into Alumno values ('s12011069', 'Hector Figueroa',2);

insert into Alumno values ('s15015824', 'Luis Antonio Del Moral',1);

insert into Examen values (10,50, '21/10/2017');

insert into Examen values (20,82, '21/10/2017');

insert into Examen values (30,40, '21/10/2017');

insert into Examen values (40,25, '21/10/2017');

insert into Examen values (50,100, '21/10/2017');

insert into Examen values (60,65, '21/10/2017');

insert into Examen values (70,10,'21/10/2017');

insert into Practica values (5,'Algebra Relacional', 'Medio');

insert into Practica values (10,'Base de Datos', 'Bajo');

insert into Practica values (15,'Modelo ER', 'Medio');

insert into Practica values (20,'Modelo Relacional', 'Alto');

insert into Practica values (25,'Normalizacion', 'Alto');

insert into Practica values (30,'Diccionario de Datos', 'Bajo');

insert into Practica values (35,'SQL', 'Medio');

```
insert into Profesor values (100,'Lorena Alonso Ramirez');
insert into Profesor values (200,'Jose de Jesus Martinez');
insert into Profesor values (300,'Elizabeth Guevara Roa');
insert into Profesor values (400,'Luis Gerardo Montane Jimenez');
insert into Profesor values (500,'Candy Obdulia Sosa Jimenez');
insert into Profesor values (600,'Angel Juan Sanchez ');
insert into Profesor values (700,'Edna Lopez Olan');

insert into Presenta values (10, 's16013961',7.5);
insert into Presenta values (30, 's15015824',8.5);
insert into Presenta values (60, 's16014096',10);
insert into Presenta values (10, 's16011020',5.5);
insert into Presenta values (20, 's16012594',6.2);
insert into Presenta values (20, 's12011069',7.1);
insert into Presenta values (60, 's12018602',4.5);

insert into Realiza values ('s16013961',20, '01/12/2017',7.2);
insert into Realiza values ('s16014096', 10, '01/12/2017',9);
insert into Realiza values ('s16012594',5, '01/12/2017',8);
insert into Realiza values ('s16011020',20, '01/12/2017',6.9);
insert into Realiza values ('s12018602', 5, '01/12/2017',7.5);
insert into Realiza values ('s12011069', 10, '01/12/2017',8.2);
insert into Realiza values ('s15015824', 30, '01/12/2017',10);

insert into Diseña values (20,100, '01/12/2015');
insert into Diseña values (5,700, '.21/02/2016');
insert into Diseña values (30,500, '05/12/2015');
insert into Diseña values (35,200, '09/03/2017');
insert into Diseña values (10,300, '16/05/2017');
insert into Diseña values (25,600, '30/11/2016');
```

insert into Diseña values (15,400, '27/02/2017');

Realizar las siguientes consultas:

4 consultas que involucren una sola tabla

Enunciado	Conocer las matrículas de los Alumnos
Algebra Relacional	π matricula (Alumno)
SQL	select matricula from Alumno;

Enunciado	Conocer el nombre de los alumnos que están en el grupo 2
Algebra Relacional	π nombre[σ grupo=2 (Alumno)]
SQL	select matricula from Alumno where grupo=2;

Enunciado	Conocer todos los datos de los alumnos que están en el grupo 1
Algebra Relacional	σ grupo=1 (Alumno)
SQL	select * from Alumno where grupo =1;

Enunciado	Conocer las matriculas y los nombre de los Alumnos donde la matricula contenga 160
Algebra Relacional	π matricula,nombre [σ matricula like “160” (Alumno)]
SQL	select matricula, nombre from Alumno where matricula like ‘s160%’;

4 consultas que incluyan una subconsulta

Enunciado	Nombre de los alumnos que hayan realizado la practica de Modelo Relacional
Algebra Relacional	ρ R1 [σ Alumno.matricula=Realiza.matricula (Alumno×Realiza)] ρ R2 [σ R2.cod_practica= Practica.cod_practica(R2×Practica)] π nombre { σ titulo like ‘Modelo Relacional’
SQL	select nombre from Alumno a, (select * from Practica where titulo = 'Modelo Relacional')p, Realiza r where a.matricula = r.matricula and r.cod_practica = p.cod_practica;

Enunciado	Conocer el nombre del profesor que realizo la practica Normalizacion
Algebra Relacional	ρ R1 (Diseña ⋈ Practica) ρ R2 (R1 ⋈ Profesor) π nombre { σ titulo='Modelo ER' (R2)}
SQL	Select nombre from Profesor natural join ((select * from Practica where titulo = 'Normalizacion')p natural join Diseña);

Enunciado	Conocer todos los datos de los alumnos que presentaron un examen el dia 21 de octubre de 2017
Algebra Relacional	ρ R1 (Examen ⋈ Presenta) ρ R2 (R1 ⋈ Alumno) π matricula,nombre, grupo, no_examen,no_preguntas fecha, calif { σ fecha='21/10/2017' (R2)}
SQL	Select nombre from Alumno natural join ((select * from Examen where fecha='21/10/2017')e natural join Presenta);

Enunciado	Obtener el nombre del maestro que diseño una practica en el 2016
Algebra Relacional	ρ R1 [σ Alumno.matricula=Realiza.matricula (Alumno×Realiza)] ρ R2 [σ R2.cod_practica= Practica.cod_practica(R2×Practica)] π nombre { σ titulo like 'Modelo Relacional'
SQL	select nombre from Profesor p1, (select * from Diseña where fecha like '%2016')d, Practica p2 where p1.no_personal = d.no_personal and d.cod_practica = p2.cod_practica ;

4 consultas que involucren más de una tabla

Enunciado	Nombre de los alumnos que haya sacado mas de 8 de calificación en la realización de la practica
Algebra Relacional	π nombre { σ calif>8[σ Alumno.matricula=Realiza.matricula (Alumno×Realiza)]}
SQL	Select nombre from Alumno a, Realiza r where a.matricula=r.matricula and calif >8;

Enunciado	Conocer el nombre del profesor que realizo la practica modelo ER
Algebra Relacional	ρ R1 (Diseña ⋈ Practica) ρ R2 (R1 ⋈ Profesor) π nombre { σ titulo='Modelo ER' (R2)}
SQL	Select nombre from Profesor natural join (Diseña natural join Practica) R1 where titulo='Modelo ER' ;

Enunciado	Conocer todos los datos de los alumnos que presentaron una practica de grado 'Alto'
Algebra Relacional	π matricula,nombre, grupo, cod_practica, titulo { σ grado = ' Alto' (Alumno ⋈ Practica)}
SQL	select * from Alumno natural join Practica where grado = 'Alto';

Enunciado	Obtener el nombre del maestro que diseño una practica en el 2015
Algebra Relacional	Π nombre { σ fecha like '%2015' (Alumno ⋈ Diseña)}
SQL	select nombre from Alumno natural join Diseña where fecha like '%2015';

4 consultas que incluyan funciones de agregación

Enunciado	Mostar el promedio obtenido, obtenida por los alumnos al realizar sus practicas
Algebra Relacional	G avg (calif) Realiza
SQL	Select avg(calif) from Realiza;

Enunciado	Mostrar la calificación máxima que se obtuvo al presentar los exámenes
Algebra Relacional	G max (calif) Presenta;
SQL	select max(calif) from Presenta;

Enunciado	Contar cuantas practicas son de grado 'Bajo'
Algebra Relacional	G count(grado)[σ grado= 'Bajo'(Practica)]
SQL	select count (grado) fron Practica where grado= 'Bajo';

Enunciado	Mostrar cuantos son los exámenes de mas de 35 preguntas
Algebra Relacional	G count (no_preguntas)[σ no_preguntas>35(Examen)]
SQL	select count (no_preguntas) from Examen where no_preguntas>35;

2 consultas update, 2 consultas delete

Enunciado	Escribir el nombre completo de Miguel Perez
SQL	update Alumno set nombre = 'Miguel Antonio Santiago Perez' where nombre = 'Miguel Perez';

Enunciado	Actualizar el grado de la practica base de datos a Media
SQL	update Practica set grado = 'Medio' where titulo = 'Base de Datos';

Enunciado	Eliminar al alumno que haya presentado una calificación menor a 5
SQL	delete from Presenta where calif <5;

Enunciado	Eliminar los exámenes que se diseñaron en el año 2015
SQL	delete from Diseña where fecha like '%2015';

2 vistas

Crear una vista que muestre a los alumnos de 3er semestre (matricula s160)

create view Alumnos3er as select matricula, nombre, grupo from Alumno where matricula like '%s160';

crear una vista de los alumnos que realizaron un examen de mas de 40 preguntas

create view TipoExamen as select matricula,calif,fecha from Examen e, Presenta p where no_preguntas >40 and e.no_examen = p.no_examen;

2 cuentas de usuario con diferentes privilegios

Usuario Monse Contraseña proyecto, con privilegios globales.

Create user ‘Monse’@'localhost' identified by ‘proyecto’;

grant all privileges on *.* to Monse@'localhost' ;

flush privileges;

Usuario prueba contraseña 12345, con privilegios de tabla.

Create user ‘prueba’@'localhost' identified by ‘12345’;

Grant select on Control.Alumnos to prueba@'localhost';

flush privileges;

2 triggers

Crear un triggers que calcule el promedio de calificación de un alumnos

create table Calificaciones (matricula varchar(10), calif1 float (5), calif2 float (5), prom float (5), foreign key (matricula) references Alumno(matricula));

insert into Calificaciones values ('s16013961', 9.2,7,0);

insert into Calificaciones values ('s16014096',9.2,6.5,0);

insert into Calificaciones values ('s16012594',6.8,8,0);

insert into Calificaciones values ('s16011020' ,8.9,9.1,0);

insert into Calificaciones values ('s12018602',5.5,6.2,0);

insert into Calificaciones values ('s12011069',6.2,8,0);

insert into Calificaciones values ('s15015824',7,9,0);

```
Delimiter //
Create trigger  calpromedio before insert on Calificaciones
for each row
begin
set new.prom= (new.calif1 +new.calif2 )/2;
end;
//
Delimiter ;
```

Crear un trigger que agregue a la tabla profesor la tabla comisión por el diseño de Practicas.

create table Comisiones (no_personal varchar(10), comision float, foreign key (no_personal) references Profesor (no_personal));

```
insert into Comisiones values (100, 250);
insert into Comisiones values (200, 400);
insert into Comisiones values (300, 650);
insert into Comisiones values (400, 300);
insert into Comisiones values (500, 250);
insert into Comisiones values (600, 400);
insert into Comisiones values (700, 500);

Delimiter //
Create trigger agregarComision after insert on Profesor
For each row
Begin
Insert into Comisiones values(new.no_personal, comision =comision*.10);
End;
//
Delimiter ;
```

2 procedimientos almacenados

Crear procedimiento que aumente el número de preguntas en 10 de cada examen.

```
Delimiter //
create procedure aumentarpreguntas2()
begin
update Examen set no_preguntas=no_preguntas + 10;
end
//
Delimiter;

call aumentarpreguntas2;
```

Crear un procedimiento que calcule el valor del examen en base al número de preguntas.

create table Examen2 (no_preguntas int, porcentaje int, foreign key (no_preguntas) references Examen (no_preguntas);

```
insert into Examen2 values (100, 60);

insert into Examen2 values (80, 50);
```

```
insert into Examen2 values (60, 40);

insert into Examen2 values (125, 70);


Delimiter //
create procedure calcularCalificacion(in x int)
begin
update Examen2 set porcentaje=(x*porcentaje/no_preguntas);
end
//
Delimiter ;
call CalcularCalificacion(36);
```