

Reporte: Práctica 2

Carlos Tonatihu Barrera Pérez
Profesor: Hernández Contreras Euler
Bases de Datos
Grupo: 2CM1

3 de marzo de 2017

Índice

1. Marco Teórico	1
2. Desarrollo	2
3. Conclusiones	13
Referencias	14

1. Marco Teórico

En bases de datos existe el termino **modelo de datos** el cual es un conjunto de herramientas conceptuales para describir los datos, las relaciones entre los datos, la semántica de los datos y las restricciones de los datos.[1]

Además, existe el **modelo de datos relacional** el cual es el mas implantado para el almacenamiento de datos. Existen otros como el modelo de datos orientado a objetos, el modelo relacional orientado a objetos y los modelos de datos semiestructurados.

Un componente importante en el diseño de bases de datos es el diseño del esquema de la base de datos, para realizar dicho esquema se suele usar el **modelo de datos entidad-relación (E-R)** el cual proporciona una representación gráfica conveniente para los datos, las relaciones y las restricciones.[1]

Para esta practica se continuaron usando los conceptos antes vistos en la práctica uno como son tablas, claves primarias y foraneas y atributos.

2. Desarrollo

El primer paso en esta practica fue crear la base de datos que se iba a utilizar junto a las relaciones que la integrarían, para esto se utilizaron los siguientes comandos.

```
create database cine; --Crea la base de datos
use cine; -- Necesario para poder trabajar con la base recién creada
```

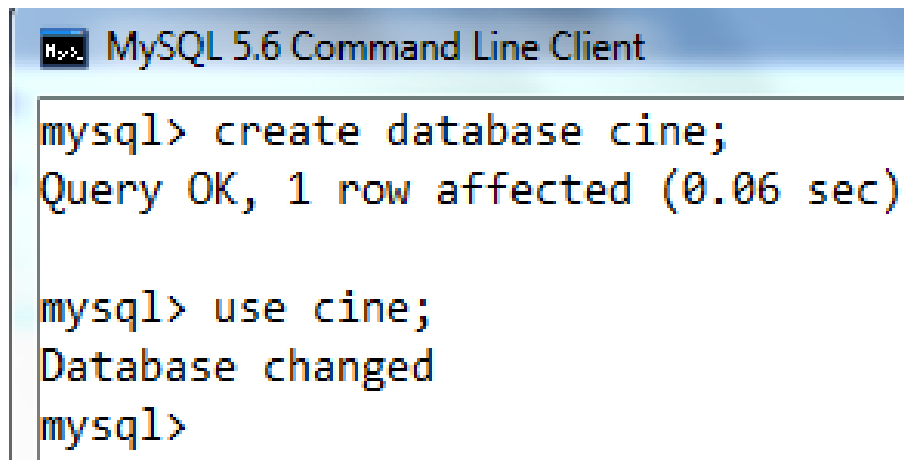


Figura 1: Creación y uso de la base.

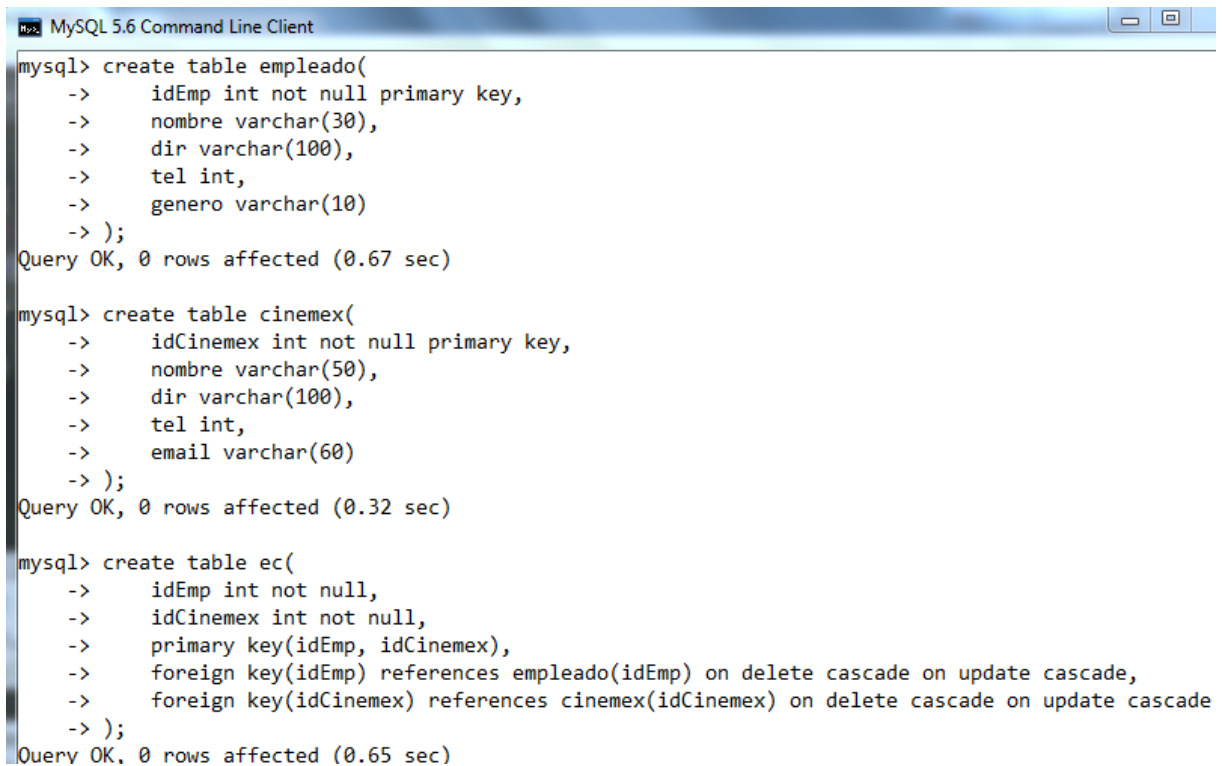
Se crean las tablas empleado, cinemex y ec.

```
create table empleado(
    idEmp int not null primary key,
    nombre varchar(30),
    dir varchar(100),
    tel int,
    genero varchar(10)
);

create table cinemex(
    idCinemex int not null primary key,
    nombre varchar(50),
    dir varchar(100),
    tel int,
    email varchar(60)
);

create table ec(
    idEmp int not null,
    idCinemex int not null,
    primary key(idEmp, idCinemex),
```

```
foreign key(idEmp) references empleado(idEmp) on delete cascade on
update cascade,
foreign key(idCinemex) references cinemex(idCinemex) on delete cascade
on update cascade
);
```



```
mysql> create table empleado(
-> idEmp int not null primary key,
-> nombre varchar(30),
-> dir varchar(100),
-> tel int,
-> genero varchar(10)
-> );
Query OK, 0 rows affected (0.67 sec)

mysql> create table cinemex(
-> idCinemex int not null primary key,
-> nombre varchar(50),
-> dir varchar(100),
-> tel int,
-> email varchar(60)
-> );
Query OK, 0 rows affected (0.32 sec)

mysql> create table ec(
-> idEmp int not null,
-> idCinemex int not null,
-> primary key(idEmp, idCinemex),
-> foreign key(idEmp) references empleado(idEmp) on delete cascade on update cascade,
-> foreign key(idCinemex) references cinemex(idCinemex) on delete cascade on update cascade
-> );
Query OK, 0 rows affected (0.65 sec)
```

Figura 2: Creación de tablas.

```
MySQL 5.6 Command Line Client
mysql> desc empleado;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idEmp | int(11)        | NO   | PRI | NULL    |       |
| nombre | varchar(30)    | YES  |     | NULL    |       |
| dir   | varchar(100)   | YES  |     | NULL    |       |
| tel   | int(11)        | YES  |     | NULL    |       |
| genero | varchar(10)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)

mysql> desc cinemex;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idCinemex | int(11)        | NO   | PRI | NULL    |       |
| nombre    | varchar(50)    | YES  |     | NULL    |       |
| dir       | varchar(100)   | YES  |     | NULL    |       |
| tel       | int(11)        | YES  |     | NULL    |       |
| email     | varchar(60)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)

mysql> desc ec;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idEmp      | int(11)        | NO   | PRI | NULL    |       |
| idCinemex | int(11)        | NO   | PRI | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.01 sec)
```

Figura 3: Descripción.

Ahora comenzamos con la modificación de las tablas, primero agregamos 2 columnas que permitan almacenar el salario y el correo electrónico en los empleados.

```
alter table empleado add column email varchar(60);
alter table empleado add column salario double;
```

```
MySQL 5.6 Command Line Client
mysql> alter table empleado add column email varchar(60);
Query OK, 0 rows affected (0.78 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> alter table empleado add column salario double;
Query OK, 0 rows affected (0.58 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc empleado;
```

Field	Type	Null	Key	Default	Extra
idEmp	int(11)	NO	PRI	NULL	
nombre	varchar(30)	YES		NULL	
dir	varchar(100)	YES		NULL	
tel	int(11)	YES		NULL	
genero	varchar(10)	YES		NULL	
email	varchar(60)	YES		NULL	
salario	double	YES		NULL	

```
7 rows in set (0.01 sec)
```

Figura 4: Resultado del alter table.

Después procedemos a crear una nueva relación gerente

```
create table gerente(
    idGerente int not null primary key,
    nombre varchar(30),
    turno varchar(15),
    noCel int,
    salario double,
    idCinemex int,
    foreign key(idCinemex) references cinemex(idCinemex) on delete cascade on
        update cascade;
);
```

```
MySQL 5.6 Command Line Client
mysql> create table gerente(
-> idGerente int not null primary key,
-> nombre varchar(30),
-> turno varchar(15),
-> noCel int,
-> salario double,
-> idCinemex int,
-> foreign key(idCinemex) references cinemex(idCinemex) on delete cascade on update cascade
-> );
Query OK, 0 rows affected (0.59 sec)

mysql> desc gerente
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idGerente  | int(11)       | NO   | PRI | NULL    |       |
| nombre     | varchar(30)   | YES  |     | NULL    |       |
| turno      | varchar(15)   | YES  |     | NULL    |       |
| noCel      | int(11)       | YES  |     | NULL    |       |
| salario    | double        | YES  |     | NULL    |       |
| idCinemex  | int(11)       | YES  | MUL | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.01 sec)
```

Figura 5: Nuestra nueva tabla.

Ahora modificamos las gerente, empleado y asociado y observamos los cambios.

```
-- Cambiar el tipo de dato del nocel del gerente a varchar
alter table gerente modify column noCel varchar(15); --modifica la
descripcion del tipo de dato

-- Renombrar la relacion empleado a asociado
alter table empleado rename as asociado; -- SOLO modifica el nombre

-- Aumentar el tamaño del tipo de dato de la dir en asociado
alter table asociado modify column dir varchar(200);
```

```

mysql> alter table gerente modify column noCel varchar(15);
Query OK, 0 rows affected (0.85 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> alter table empleado rename as asociado;
Query OK, 0 rows affected (0.16 sec)

mysql> alter table asociado modify column dir varchar(200);
Query OK, 0 rows affected (0.67 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc gerente;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idGerente | int(11) | NO | PRI | NULL | |
| nombre | varchar(30) | YES | | NULL | |
| turno | varchar(15) | YES | | NULL | |
| noCel | varchar(15) | YES | | NULL | |
| salario | double | YES | | NULL | |
| idCinemex | int(11) | YES | MUL | NULL | |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.03 sec)

mysql> desc asociado;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idEmp | int(11) | NO | PRI | NULL | |
| nombre | varchar(30) | YES | | NULL | |
| dir | varchar(200) | YES | | NULL | |
| tel | int(11) | YES | | NULL | |
| genero | varchar(10) | YES | | NULL | |
| email | varchar(60) | YES | | NULL | |
| salario | double | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
7 rows in set (0.01 sec)

```

Figura 6: Los cambios que sucedieron.

Procedemos a modificar la llave primaria de cinemex que ahora sera compuesta por el id y el nombre por lo que también se tienen que modificar las tablas que esten relacionadas con esta por lo que seguiremos los siguientes pasos:

1. Eliminar PK de cinemex, pero antes eliminar la llave foránea con gerente y ec.
2. Definir PK compuesta.

```

show create table gerente; -- Muestra la descripción de la relación de una
    forma mas completa a comparación del comando desc para poder obtener el
    constraint de lo contrario no se podría eliminar la primary key de cinemex
    ya que es utilizada en esta relación
alter table gerente drop foreign key gerente_ibfk_1;

```

```
MySQL 5.6 Command Line Client
mysql> show create table gerente;
+-----+-----+
| Table | Create Table |
+-----+-----+
| gerente | CREATE TABLE `gerente` (
  `idGerente` int(11) NOT NULL,
  `nombre` varchar(30) DEFAULT NULL,
  `turno` varchar(15) DEFAULT NULL,
  `noCel` varchar(15) DEFAULT NULL,
  `salario` double DEFAULT NULL,
  `idCinemex` int(11) DEFAULT NULL,
  PRIMARY KEY (`idGerente`),
  KEY `idCinemex` (`idCinemex`),
  CONSTRAINT `gerente_ibfk_1` FOREIGN KEY (`idCinemex`) REFERENCES `cinemex` (`idCinemex`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+-----+-----+
1 row in set (0.02 sec)

mysql> alter table gerente drop foreign key gerente_ibfk_1;
Query OK, 0 rows affected (0.08 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

Figura 7: Los cambios que sucedieron.

Lo mismo para ec pero hay que tener cuidado de no borrar la de la relación asociado.

```
show create table ec;
alter table ec drop foreign key ec_ibfk_2;
```

```

MySQL 5.6 Command Line Client
mysql> show create table ec;
+-----+
| Table | Create Table
+-----+
| ec    | CREATE TABLE `ec` (
  `idEmp` int(11) NOT NULL,
  `idCinemex` int(11) NOT NULL,
  PRIMARY KEY (`idEmp`,`idCinemex`),
  KEY `idCinemex` (`idCinemex`),
  CONSTRAINT `ec_ibfk_1` FOREIGN KEY (`idEmp`) REFERENCES `asociado` (`idEmp`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `ec_ibfk_2` FOREIGN KEY (`idCinemex`) REFERENCES `cinemex` (`idCinemex`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+-----+

1 row in set (0.00 sec)

mysql> alter table ec drop foreign key ec_ibfk_2;
Query OK, 0 rows affected (0.07 sec)
Records: 0 Duplicates: 0 Warnings: 0

```

Figura 8: Los cambios que sucedieron.

```

alter table cinemex drop primary key; -- Ahora si podemos eliminar la clave
primaria
alter table cinemex add primary key(idCinemex, nombre); -- Creamos nuestra
nueva PK compuesta

```

```

MySQL 5.6 Command Line Client
mysql> alter table cinemex drop primary key;
Query OK, 0 rows affected (0.62 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> alter table cinemex add primary key(idCinemex, nombre);
Query OK, 0 rows affected (0.96 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc cinemex
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idCinemex  | int(11)       | NO   | PRI | NULL    |       |
| nombre     | varchar(50)   | NO   | PRI |         |       |
| dir        | varchar(100)  | YES  |     | NULL    |       |
| tel        | int(11)       | YES  |     | NULL    |       |
| email      | varchar(60)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)

```

Figura 9: Cambios en la tabla.

Es necesario agregar el campo que falta para la relación y poder crear la clave foránea compuesta en las tablas de gerente y ec. Es importante que sean iguales al campo de la tabla cinemex aunque pueden llevar nombres distintos.

```

alter table gerente add column nomCine varchar(50);
alter table ec add column nomCine varchar(50);
-- Se crea la FK compuesta
alter table gerente add foreign key(idCinemex, nomCine) references
    cinemex(idCinemex, nombre) on delete cascade on update cascade;

alter table ec add foreign key(idCinemex, nomCine) references
    cinemex(idCinemex, nombre) on delete cascade on update cascade;

```

```
MySQL 5.6 Command Line Client

mysql> alter table gerente add foreign key(idCinemex, nomCine) references cinemex(idCinemex, nombre) on delete cascade on update cascade;
Query OK, 0 rows affected (0.80 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> alter table ec add foreign key(idCinemex, nomCine) references cinemex(idCinemex, nombre) on delete cascade on update cascade;
Query OK, 0 rows affected (0.85 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc gerente;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idGerente | int(11) | NO | PRI | NULL | |
| nombre | varchar(30) | YES | | NULL | |
| turno | varchar(15) | YES | | NULL | |
| noCel | varchar(15) | YES | | NULL | |
| salario | double | YES | | NULL | |
| idCinemex | int(11) | YES | MUL | NULL | |
| nomCine | varchar(50) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
7 rows in set (0.01 sec)

mysql> desc ec;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idEmp | int(11) | NO | PRI | NULL | |
| idCinemex | int(11) | NO | PRI | NULL | |
| nomCine | varchar(50) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.01 sec)
```

Figura 10: Tablas modificadas.

Por ultimo se crea la relación cartelera y se asocia con cinemex.

```
create table cartelera(
    idCartelera int not null primary key,
    nombre varchar(50),
    fechaInicio date,
    fechaFin date,
    clasificacion varchar(5)
);
```

```
MySQL 5.6 Command Line Client
mysql> create table cartelera(
->     idCartelera int not null primary key,
->     nombre varchar(50),
->     fechaInicio date,
->     fechaFin date,
->     clasificacion varchar(5)
-> );
Query OK, 0 rows affected (0.30 sec)

mysql> desc cartelera
-> ;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idCartelera    | int(11)       | NO   | PRI | NULL    |       |
| nombre         | varchar(50)   | YES  |     | NULL    |       |
| fechaInicio    | date          | YES  |     | NULL    |       |
| fechaFin       | date          | YES  |     | NULL    |       |
| clasificacion  | varchar(5)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)
```

Figura 11: Tablas modificadas.

Finalmente creamos una llave foránea en cinemex que se asocia con la cartelera.

```
alter table cinemex add column idCartelera int;
alter table cinemex add foreign key(idCartelera) references
    cartelera(idCartelera) on delete cascade on update cascade;
```

```
MySQL 5.6 Command Line Client

mysql> alter table cinemex add column idCartelera int;
Query OK, 0 rows affected (0.53 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> alter table cinemex add foreign key(idCartelera) references cartelera(idCartelera) on delete cascade on update cascade
Query OK, 0 rows affected (0.79 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc cinemex;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| idCinemex  | int(11)   | NO   | PRI | NULL    |       |
| nombre     | varchar(50) | NO   | PRI |         |       |
| dir        | varchar(100) | YES  |     | NULL    |       |
| tel        | int(11)   | YES  |     | NULL    |       |
| email      | varchar(60) | YES  |     | NULL    |       |
| idCartelera | int(11)   | YES  | MUL | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.03 sec)
```

Figura 12: Tablas modificadas.

Para terminar se exporto la base de datos a un archivo .sql

```
C:\Program Files\MySQL\MySQL Server 5.6\bin>mysqldump -u root -p cine>J:\bases-datos\practica2\respaldo2.sql
Enter password: *****

C:\Program Files\MySQL\MySQL Server 5.6\bin>
```

Figura 13: Exportar la base.

3. Conclusiones

Esta segunda práctica me permitió profundizar y reforzar lo aprendido en la primera práctica para poder aprender más rápida y clara los comandos que se usan en la creación y trabajo de una base de datos, a pesar de ser comandos básicos considero que pueden llegar a complicarse a medida de que la complejidad de una base de datos aumenta. Por lo que es importante continuar practicando para no olvidar el como se declara cada sentencia sql y no depender de guías salvo para casos mas complejos.

Referencias

- [1] H. F. K. Abraham Silberschatz, *Fundamentos de Diseño de Bases de Datos*. McGraw-Hill, 2007.