DDL

Lenguaje de Definición de Datos

- CREATE
- DROP

Creación de tablas

- Una base de datos almacena la información en tablas
- **¿Qué es una tabla?**
  - Es una estructura que organiza los datos en filas y columnas.
  - Cada columna es un campo o atributo.
  - Cada fila es un registro
  - Cada campo debe tener un nombre único que hace referencia a la información que almacenará
  - Cada columna tiene asociado un **tipo de dato** **(integridad de dominio)**
  - Esta información se usa al momento de crear la tabla

Tipos de datos

- **Numéricos:**
  - **tinyint** [unsigned] → 1 byte (con signo: -128 a 127, sin signo: 0 a 255)
  - **smallint** [unsigned] → 2 bytes (con signo: -32,768 a 32,767, sin signo: 0 a 65,535)
  - **mediumint** [unsigned] → 3 bytes (con signo: -8,388,608 a 8,388,607, sin signo: 0 a 16,777,215)
  - **int** [unsigned] → 4 bytes (con signo: -2,147,483,648 a 2,147,483,648, sin signo: 0 a 4,294,967,295)
  - **bigint** [unsigned] → 8 bytes

(con signo: -9,223,372,036,854,775,808 a 9,223,372,036,854,775,807,
sin signo: 0 a 18,446,744,073,709,551,615

Tipos de datos

- **Numéricos de tipo flotante:**
  - **float (e,d)** total de dígitos de **e** enteros y **d** decimales (4 bytes)
  - **double (e,d)** igual a float pero de doble precisión (8 bytes)
  - **decimal (e,d)** igual a float, se utiliza para manejar cantidades de dinero (4 bytes)

Tipos de datos

- **Texto:**
  - **char(n)**: texto fijo (n byte)
  - **varchar(n)** : texto variable (n bytes)
  - **tinytext** (16 kb)
  - **tinyblob** (16kb)
  - **text** (64kb)
  - **blob** (64kb)
  - **enum(valores)**

Tipos de datos

- **Tiempo:**
  - **Date:** Tipo fecha 'YYYY-MM-DD' ó 'YY-MM-DD' ó 'YYMMDD'
  - **Time**: Tipo hora 'HH:MM:SS' ó 'HHMMSS'
  - **DateTime:** 'YYYY-MM-DD HH-MM-SS'
  - **Year:** Tipo año 'YYYY'
  - **Timestamp**: Tipo instante 'YYYYMMDDhhmmss'

Sintaxis básica para crear tablas

```
CREATE TABLE nombre_tabla(
nombre_campo1 TIPO_DATO,
nombre_campo2 TIPO_DATO,
....
nombre_campoN TIPO_DATO);
```

Convención: nombres de tablas en mínusculas, sin espacios y sin caracteres especiales como acentos.

Ejemplos:

- Habilitar la base de datos a usar: **use ejemplo1**
- Crear la tabla usuario

```
CREATE TABLE usuario(
id int, nombre varchar(30), rfc char(13));
```

Comandos para manejo de tablas

- Mostrar las tablas de la BD:
  - **SHOW TABLES;**
- Mostrar la **estructura** de las tablas:
  - **SHOW CREATE TABLE** nombre_tabla;
- Ejemplo: show create table usuario;
- Mostrar la **descripción** de las tablas:
  - **DESCRIBE** nombre_tabla;
  - **DESC** nombre_tabla;
- Ejemplo: desc usuario;
- **Borrar** una tabla de la BD:
  - **DROP TABLE** nombre_tabla;
- Ejemplo: drop table usuario;

```
(BD:ejemplo1) mysql> CREATE TABLE usuario(
    -> id int, nombre varchar(30), rfc char(13));
Query OK, 0 rows affected (1.69 sec)

(BD:ejemplo1) mysql> show tables;
+--------------------+
| Tables_in_ejemplo1 |
+--------------------+
| usuario            |
+--------------------+
1 row in set (0.02 sec)

(BD:ejemplo1) mysql> show create table usuario;
+---------+-------------------------------------------------------------------
| Table   | Create Table
+---------+-------------------------------------------------------------------
| usuario | CREATE TABLE `usuario` (
  `id` int DEFAULT NULL,
  `nombre` varchar(30) DEFAULT NULL,
  `rfc` char(13) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+---------+-------------------------------------------------------------------
1 row in set (0.03 sec)

(BD:ejemplo1) mysql> desc usuario;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| id     | int         | YES  |     | NULL    |       |
| nombre | varchar(30) | YES  |     | NULL    |       |
| rfc    | char(13)    | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.04 sec)

(BD:ejemplo1) mysql> describe usuario;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| id     | int         | YES  |     | NULL    |       |
| nombre | varchar(30) | YES  |     | NULL    |       |
| rfc    | char(13)    | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

(BD:ejemplo1) mysql> notee
```

Implementación de Relaciones

Tipos de relaciones

a) 1 : 1 – uno a uno (PK)
b) 1 : M – uno a muchos (FK)
c) M : 1 – muchos a uno (FK)
d) M : M – muchos a muchos (FK)

Tipos de índices

- **PK (Primary Key):** Llave primaria es el atributo para identificar de manera única a los elementos de la entidad, la llave puede ser compuesta si hay más de un atributo. No puede haber nulos.

- **FK (Foreign Key):** Llave foránea, se encarga de relacionar las entidades y está representada por la llave primaria de otra entidad. **La FK garantiza la integridad referencial**.

- **UK (UNIQUE KEY):** Este índice se utiliza para definir llaves de negocio sobre un atributo o conjunto de atributos elegidos con la finalidad de no permitir valores duplicados **(integridad de entidad)**

- **INDEX (KEY):** Índice utilizado para optimización de búsquedas sobre un atributo, conjunto de atributos o parte de un atributo elegido.

Tipos de restricciones

- **NULL:** Admite valores nulos.
- **NOT NULL:** Rechaza dejar el campo en blanco
- **DEFAULT:** Permite establecer un valor por defecto
- **AUTO_INCREMENT:** Genera identificadores únicos consecutivos o valores en serie

Tipos de motores de datos

- **Engines o motores de datos:** Administran la información física dentro de MySQL.
  - **MyISAM:** Sirve para las necesidades del usuario medio, compatible con todos los tipos de campos y parámetros.
  - **MEMORY:** Se utiliza por su velocidad, no admite auto_increment ni campos de tipo blob o text, los datos se almacenan sólo en memoria.
  - **InnoDB:** Utilizada para aplicaciones de gran tamaño a las que se accede con frecuencia, cuenta con un mecanismo de bloqueo de filas para evitar que los usuarios modifiquen o añadan la misma fila a una tabla. Soportan transacciones e integridad referencial (FK).
- **show engines;** -- para ver los motores disponibles en mysql

Sintaxis completa para creación de tablas

CREATE TABLE nombre_tabla**(**

nombre_campo TIPO_DATO [NOT NULL | NULL] [DEFAULT valor][AUTO_INCREMENT]**,**

[**,**PRIMARY KEY (campo)]

[**,**INDEX (campo)]

[**,**UNIQUE (campo)]

[**,** [CONSTRAINT FK_PAIS] FOREIGN KEY (campo) REFERENCES tabla_padre (campo)

ON DELETE CASCADE | SET NULL | NO ACTION | RESTRICT

ON UPDATE CASCADE | SET NULL | NO ACTION |

RESTRICT]**)**ENGINE=TIPO_DE_TABLA**;**

Ejemplos

1. Crear la base de datos: *Tienda*
2. Habilitar la base *Tienda*
3. Crear la tabla *producto*:

```
create table producto (
id int unsigned not null auto_increment primary key,
nombre varchar(30) not null unique,
precio decimal(6,2) not null default 9.99
);
```

4. desc producto;
5. show create table producto;

Ejemplos

6. Crear la tabla empleado:

```
create table empleado(
id smallint unsigned not null auto_increment,
nombre varchar(30) not null,
apellido1 varchar(30) not null,
apellido2 varchar(30),
primary key(id),
unique key(nombre,apellido1,apellido2)
);
```

7. desc empleado;

8. show create table empleado;

Relación uno a uno

1:1

Sintaxis completa para creación de tablas

CREATE TABLE nombre_tabla**(**

nombre_campo TIPO_DATO [NOT NULL | NULL] [DEFAULT valor][AUTO_INCREMENT]**,**

[**,**PRIMARY KEY (campo)]

[**,**INDEX (campo)]

[**,**UNIQUE (campo)]

[, [CONSTRAINT FK_name] FOREIGN KEY (campo) REFERENCES tabla_padre (campo)

ON DELETE CASCADE | SET NULL | NO ACTION | RESTRICT

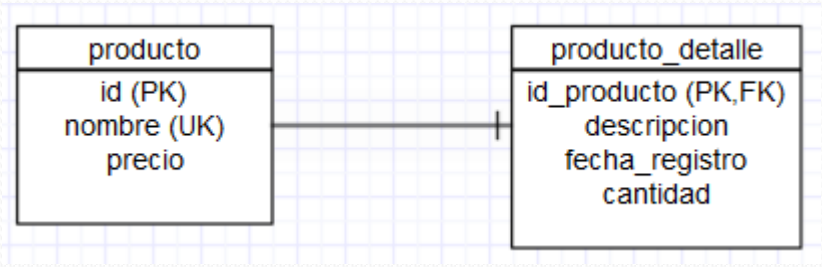
ON UPDATE CASCADE | SET NULL | NO ACTION |

RESTRICT]**)**ENGINE=TIPO_DE_TABLA**;**

Relación uno a uno (PK,FK)

- Una relación uno a uno ocurre cuando cada registro de una tabla está asociado con sólo un registro de su tabla asociada

- Ejemplo1:
  - Tabla padre (base): producto
  - Tabla hija (asociada): producto_detalles



Ejemplos

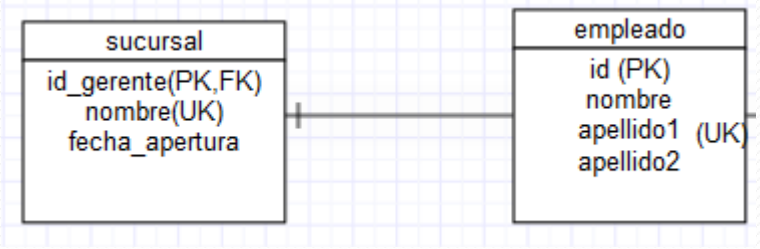
```
create table producto (
id int unsigned not null auto_increment primary key,
nombre varchar(30) not null unique,
precio decimal(6,2) not null default 9.99
);
```

```
CREATE TABLE producto_detalles (
id_producto INT UNSIGNED NOT NULL,
descripcion VARCHAR(100) NOT NULL,
fecha_registro DATE NOT NULL,
cantidad smallint unsigned not null,
PRIMARY KEY (id_producto),
FOREIGN KEY (id_producto) REFERENCES producto (id)
);
```

Relación uno a uno (PK,FK)

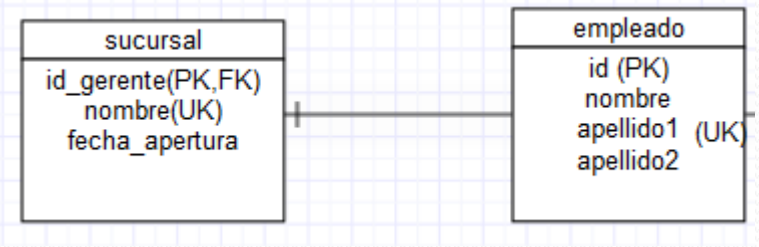
- Una relación uno a uno ocurre cuando cada registro de una tabla está asociado con sólo un registro de su tabla asociada

- Ejemplo:
  - Tabla padre (base): empleado
  - Tabla hija (asociada): sucursal



Relación uno a uno

```
create table empleado(
id smallint unsigned not null auto_increment,
nombre varchar(30) not null,
apellido1 varchar(30) not null,
apellido2 varchar(30),
primary key(id),
unique key(nombre,apellido1,apellido2)
);
```



Crear la tabla sucursal:

```
create table sucursal(
id_gerente smallint unsigned not null,
nombre_suc varchar(30) not null unique,
fecha_apertura date not null,
primary key(id_gerente),
FOREIGN KEY (id_gerente) REFERENCES empleado(id)
);
```

Tabla padre

Relación uno a uno

desc sucursal;
show create table sucursal;

```
mysql> show create table sucursal;
+----------+----------------------------------------------------------------------------------
| Table    | Create Table
+----------+----------------------------------------------------------------------------------
| sucursal | CREATE TABLE `sucursal` (
  `id_gerente` smallint unsigned NOT NULL,
  `nombre_suc` varchar(30) NOT NULL,
  `fecha_apertura` date NOT NULL,
  PRIMARY KEY (`id_gerente`),
  UNIQUE KEY `nombre_suc` (`nombre_suc`),
  CONSTRAINT `sucursal_ibfk_1` FOREIGN KEY (`id_gerente`) REFERENCES `empleado` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+----------+----------------------------------------------------------------------------------
1 row in set (0.07 sec)
```

Al crear la llave foránea, se asigna un nombre por default y se muestra en el **constraint**

En el siguiente ejemplo se verá el uso de constraint para asignar un nombre (al gusto) a la llave foránea.

Drop table sucursal; -- se borra la tabla para que no marque error

```
create table sucursal(
id_gerente smallint unsigned not null,
nombre_suc varchar(30) not null unique,
fecha_apertura date not null,
primary key(id_gerente),
CONSTRAINT FK_sucursal_empleado
FOREIGN KEY (id_gerente) REFERENCES empleado(id)
);
```

```
mysql> show create table sucursal;
+----------+----------------------------------------------------------------------------------
| Table    | Create Table
+----------+----------------------------------------------------------------------------------
| sucursal | CREATE TABLE `sucursal` (
  `id_gerente` smallint unsigned NOT NULL,
  `nombre_suc` varchar(30) NOT NULL,
  `fecha_apertura` date NOT NULL,
  PRIMARY KEY (`id_gerente`),
  UNIQUE KEY `nombre_suc` (`nombre_suc`),
  CONSTRAINT `FK_sucursal_empleado` FOREIGN KEY (`id_gerente`) REFERENCES `empleado` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+----------+----------------------------------------------------------------------------------
1 row in set (0.00 sec)
```

¿Preguntas?