Curso 2019-20

BASES DE DATOS

CAPITULO 3:
LENGUAJE SQL

BASES DE DATOS RELACIONALES

3. LENGUAJE SQL

3.1 Lenguaje SQL

3.2 Tipos de Datos MySQL

NOT NULL, UNIQUE, …

3.3 Lenguaje DDL

3.3.1 Creación de Schemas

3.3.2 Creación de tablas

3.3.3 Modificación de tablas

LENGUAJE SQL

3.1LENGUAJE SQL

VERSIONES. (ISO/IEC 9075 por IBM)

SQL-87

SQL-2: revisión mayor en 1992

SQL2000: consultas recursivas, triggers.

SQL2005: Uso conjunto XML y SQL.

SQL2008: añade sentencia TRUNCATE y clausula ORDER BY.

SQL 2016: Compatibilidad con ficheros JSON.

LENGUAJE SQL

DDL: Data Definition Language

Descripción

LMD: Data Manipulatuion Language

Manipulación

DCL: Data Control Language

Utilización

TCL: Transaction Control Languaje

Control de transacciones

LENGUAJE DDL

Permite crear toda la estructura de una base de datos: tablas, usuarios y diferentes objetos.

Sus sentencias son:

DROP →Elimina objetos

CREATE →Crea objetos

ALTER →Modifica objetos existentes

LENGUAJE DML

Permite jugar con los datos de una tabla mediante cuatro sentencias:

SELECT→ Borrar datos

INSERT → Insertar datos

UPDATE → Modificar Datos

DELETE → Borrar datos

LENGUAJE DCL

- Permite al administrador de la base de datos (DBA) gestionar el acceso a los contenidos en la base de datos.

Sus sentencias son:

GRANT → Permite el acceso

REVOKE → Niega el acceso

LENGUAJE TCL

- Permite ejecutar varios comandos de forma simultánea como si fuera en un único comando indivisible.

Si se pueden ejecutar todas las ordenes se aplica la transacción **COMMIT.**

Si sucede algo inesperado se pueden deshacer todos los pasos **ROLLBACK.**

SQL: Instrucciones

DML
SELECT
INSERT
UPDATE
DELETE

DDL
DROP
CREATE
ALTER

DCL
GRANT
REVOKE

TCL
COMMIT
ROLLBACK

ENLACES SQL

SQL: http://es.wikipedia.org/wiki/SQL

DDL:
http://en.wikipedia.org/wiki/Data_definition_language

DML:
http://en.wikipedia.org/wiki/Data_manipulation_language

DCL:
http://en.wikipedia.org/wiki/Data_Control_Language

DTL:
http://es.wikipedia.org/wiki/Transacci%C3%B3n_(base_de_datos)

TIPOS DE DATOS DE MYSQL

https://dev.mysql.com/doc/refman/5.7/en/data-types.html

3.2 TIPOS DE DATOS versión 5.8

- NUMÉRICOS

  signed/unsigned integers 1,2,3,4 y 8 bytes

  long, float, double, bool, boolean

- CADENAS DE TEXTO

  char, varchar, binary, varbinarym text, blob

- FECHAS

  date, time, datetime, timestamp, year,

DATOS MULTIVALUADOS

  set, enum

- DEFINICIÓN DE CADA COLUMNAS

3.2 TIPOS DE DATOS

DATOS NUMERICOS

	CARACTERISTICAS	DESDE	HASTA
TinyInt	Número entero con o sin signo.	CON SIGNO -128	127
		SIN SIGNO 0	255
Bit ó **Bool**	Número entero que puede ser 0 ó 1.	0	1
SmallInt	Número entero con o sin signo a diferencia del Tinyint maneja un mayor rango.	CON SIGNO -32768	32767
		SIN SIGNO 0	65535
MediumInt	Número entero con o sin signo, su diferencia con los demás tipos de datos numéricos es el tipo de datos que maneja	CON SIGNO -8.388.608	
		SIN SIGNO 0	16.777.215
Integer ó **Int**	Número entero con o sin signo, maneja un mayor rango que los demás tipos de datos.	CON SIGNO -2.147.483.648	2.147.483.647
		SIN SIGNO 0	4.294.967.295
BigInt	Número entero con o sin signo con un mayor rango de los demás.	CON SIGNO -9.223.372.036.854.775.808	9.223.372.036.854.775.807
		SIN SIGNO 0	18.446.744.073.709.551.615
Float	Número pequeño en coma flotante de precisión simple.	-3.402823466 e+38	-1.175494351 e-38,
		1.175494351 e+38	3.402823466 e+38.
Double	Número en coma flotante de precisión doble.	-1.7976931348623157 e+308	-2.2250738585072014 e+308
		2.2250738585072014 e+308	1.7976931348623157 e+308
Decimal, **Dec** ó **Numeric**	Número en coma flotante desempaquetado. El número se almacena como una cadena.	----------	----------

FECHAS

CADENAS

Date	guarda una fecha, el formato es de	1 de enero del 1001	31 de diciembre de 9999
	AAAA-MM--DD		
DateTime	Guarda combinación de fecha y hora, el formato es de AAAA-MM-DD HH:MM:SS	1 de enero del 1001 a las 0 horas, 0 minutos y 0 segundos	31 de diciembre del 9999 a las 23 horas, 59 minutos y 59 segundos
TimeStamp	también combina fecha y hora con la diferencia que almacena menos años atrás y menos años adelante, el formato es de AAAAMMDDHHMMSS	1 de enero de 1970	1 de enero de 2037.
Time	almacena una hora(únicamente la hora), el formato es HH:MM:SS	-838 horas, 59 minutos y 59 segundos	838, 59 minutos y 59 segundos
Year	almacena un año.(únicamente el año), El campo puede tener tamaño dos o tamaño 4 dependiendo de si queremos almacenar el año con dos o cuatro dígitos. (AAAA/AA)	año 1901	año 2155.
Char	Guarda una cadena de longitud fija.	0	255
Varchar	guarda una cadena de longitud variable	0	255
Blob O TEXT	Es un objeto binario que puede tratar una cantidad de datos variables. Los cuatro tipos BLOB son TINYBLOB, BLOB, MEDIUMBLOB, y LONGBLOB los cuales difieren sólo en la longitud máxima de los valores que pueden tratar.	---------- VARIAN DEACUERDO A LOS DATOS INTRODUCIDOS	----------
Text	Son cadenas de caracteres no binarias. Los cuatro tipos TEXT son TINYTEXT, TEXT, MEDIUMTEXT, y LONGTEXT. Se corresponden a los cuatro tipos BLOB y tienen las mismas longitudes y requerimientos de almacenamiento.	---------- VARIAN DEACUERDO A LOS DATOS INTRODUCIDOS	----------
MEDIUMTEXT, MEDIUMBLOB	textos	16777215 caracteres	-16777215 caracteres

3.3 CREACION DE COLUMNAS

- SINTAXIS DE UN TIPO DE DATO

```
tipo_dato [NOT NULL | NULL] [DEFAULT valor_defecto]
[AUTO_INCREMENT] [UNIQUE [KEY] | [PRIMARY] KEY]
[COMMENT 'string']
[COLUMN_FORMAT {FIXED|DYNAMIC|DEFAULT}]
[STORAGE {DISK|MEMORY|DEFAULT}]
[reference_definition]
```

3.2.1 Valores sin signo

UNSIGNED, define números positivos. Duplica el valor del límite máximo, se elimina la opción de almacenar valores negativos.

edad TinyInt UNSIGNED;

precio float UNSIGNED;

cantidad INT(10) UNSIGNED;

Atributo de columna **UN** → [UNSIGNED]

3.2.2 Valores con ceros

ZEROFILL, rellena con ceros la parte entera. Añade automáticamente UNSIGNED.

edad TinyInt ZEROFILL;

precio float ZEROFILL;

cantidad INT(10) ZEROFILL;

3.2.3 Números decimales

Sirven para almacenar Salarios, importes, cuentas bancarias, etc…

Se llaman datos en ***coma flotante***, *pero MySQL* los almacena usando un punto como separador.

Tipos: **FLOAT, DOUBLE** y **DECIMAL**

3.2.3 Números decimales

ESTRUCTURA.

- **Longitud total** (decimales y la coma)
- **Parte decimal** (nº dígitos)

precio FLOAT(6,2);

➔ Valor mínimo: -999.99

➔ Valor máximo: 999.99

3.2.3 Números decimales

ESTRUCTURA. Longitud/valores

Precisión SIMPLE rango entre 0 y 24:

precio FLOAT(28,24);

Precisión DOBLE rango entre 25 y 53:

precio DOBLE(29,25);

*Ambos pueden traer problemas de redondeo y pérdida de los decimales restantes.

3.2.3 Números decimales

DECIMAL. Longitud/valores

Máximo de dígitos totales es de 64 de los cuales 30 es el máximo permitido.

gasoleob DECIMAL(8,5); 121,01547 €

*Ideal para almacenar valores monetarios donde se requiera menor longitud, pero la máxima exactitud (sin redondeos).

*Similar a phpMyAdmin.

3.2.4 BOOL y BOOLEAN

Son sinónimos de TINYINT (1)

Valor cero es considerado falso y valores distintos de cero se consideran verdaderos.

```
mysql> SELECT IF(0, 'true', 'false');
+------------------------+
| IF(0, 'true', 'false') |
+------------------------+
| false                  |
+------------------------+

mysql> SELECT IF(1, 'true', 'false');
+------------------------+
| IF(1, 'true', 'false') |
+------------------------+
| true                   |
+------------------------+

mysql> SELECT IF(2, 'true', 'false');
+------------------------+
| IF(2, 'true', 'false') |
+------------------------+
| true                   |
+------------------------+
```

3.2.5 CHAR

RESERVAMOS UNA CADENA DE 14 CARACTERES

1	2	3	4	5	6	7	8	9	10	11	12	13	14
J	u	a	n		P	e	r	e	z				
C	a	r	l	o	s		G	a	r	c	i	a	
J	o	s	e		R	a	m	i	r	e	z		
l	u	i	s		F	e	r	n	a	n	d	e	z
P	e	p	e		L	o	p	e	z				

Ejemplo

char (4) ‘ab’ → 4 bytes

varchar (4) ‘ab’ → 3 bytes

A partir de la versión 5.0.3 se empezó a dejar de utilizarse **TEXT**.

3.2.6 VARCHAR

HASTA LA VERSIÓN 5.0.3 EL MÁXIMO ERA DE **255 CARACTERES**.

VERSIÓN SUPERIORES PERMITEN HASTA **65.535 CARACTERES**.

Solo consume los caracteres que contiene.

Reservamos 14 CARACTERES y escribimos Juan solo consumirá 5 CARACTERES.

nombre VARCHAR (14)= 'Juan' → 4 +1 bytes

3.2.7 BLOB

Guarda información en **formato binario**.

Se utiliza desde PHP para almacenar en la BD el contenido de un archivo binario (imagen, *.zip,…), leyéndolo bit a bit.

Tamaño máximo **65.535 bytes**.

- ➔ Convierte la base de datos a muy pesada.
- ➔ Lo ideal es guardar la URL de la imagen, para que el navegador muestre la imagen, mediante un varchar

Tamaño medio MEDIUMBLOB (16MB)

Mayor tamaño tipo de datos LONGBLOB (4GB)

3.2.8 TIPOS DE DATOS

- CAMPOS MULTIVALUADOS

ENUM	Lista de valores permitidos separados por comas y envueltos entre comillas simples.	Máximo de valores 65.535
ENUM ‘maestro’, ‘profesor’		
SET	Lista de valores permitidos separados por comas y envueltos entre comillas simples. Se pueden dejar celdas vacías.	Máximo de valores 64

3.2.8 ENUM

Permite establecer los valores que puede escoger ó **valores permitidos**.

No se autoriza el ingreso de un valor diferente, fuera de la lista.

Valores separados por comas y envueltos entre comillas simples.

Máximo de valores **65.535.**

3.2.8 SET

Similar a ENUM pero con una lista máxima de 64 opciones.

A diferencia de ENUM:

- Permite valores vacíos.
- Se puede escoger más de un valor de la lista

3.2.9 DATE

Almacenas fechas en formato: **AAAA-MM-DD**.

Se pueden insertar los datos tanto con guiones como sin guiones:

- 20151231
- 2015-12-31, 2015.12.31, 2015/12/31, 15-12-31, 15@12@31
- NOW()

El rango comienza desde el **1000-01-01** hasta **9999-12-31**. Valores anteriores no podremos almacenarlos con este tipo.

3.2.10 DATETIME

Almacenas fechas y horas con el formato:

AAAA-MM-DD HH:MM:SS

El rango comienza desde el **1000-01-01** hasta **9999-12-31**, desde las **00:00:000** hasta las **23:59:59**.

Valores anteriores no podremos almacenarlos con este tipo.

3.2.11 TIME

Almacenas horas, minutos y segundos con el formato:

HH:MM:SS

El rango comienza desde el **-839:59:59** hasta **839:59:59** (35 días hacia atrás y delante de la fecha actual).

Ideal para calcular tiempos transcurridos entre dos momentos cercanos.

3.2.12 TIMESTAMP

Almacena una fecha y horario similar al DATETIME pero con formatos diferentes:

- **AAAA-MM-DD HH:MM:SS**
- **AAAA-MM-DD**
- **AA-MM-DD**

La longitud puede ser de 14, 8 ó 6 dígitos.

El rango de fechas va desde 1970-01-01 hasta el 2038-01-19.

Se puede mantener actualizado cada vez que se inserte o actualice un registro, sin necesidad de programar nada.

3.2.13 YEAR

YEAR. Permite 4 dígitos como dos para almacenar años, desde **1901** hasta **2155** ó de 70 hasta 99.

- Equivaldría a 1970 hasta 1999

3.2 RESUMEN SINTAXIS

id	name	name_en	a	b	descr	descr_en	pos	created_at
1	Новый	New	01	0001	NULL	NULL	0	2016-12-02 21:35:59

- TINYINT [(m)] [UNSIGNED][ZEROFILL]
- INT[(M)] [UNSIGNED] [ZEROFILL]
- FLOAT[(M,D)] [UNSIGNED] [ZEROFILL]
- DOUBLE[(M,D)] [UNSIGNED] [ZEROFILL]
- DECIMAL[(M[,D])] [UNSIGNED] [ZEROFILL]
- SET('value1','value2',...) [CHARACTER SET charset_name] [COLLATE collation_name]

3.2 EQUIVALENCIAS MYSQL 5.7

- INT1 → TINYINT (4)
- INT2 → SMALLINT (6)
- INT3 → MEDIUMINT (9)
- INT4, INTEGER → INT (11)
- INT8 → BIGINT(20)
- DEC → DECIMAL (10,0)
- CHARACTER → CHAR(1)

3.2 NOT NULL ó NULL

- Se utiliza para agregar registros sin que sus valores sean completados.
  - Se pondrán valores **NULL** (nulos), para ir añadiendo registros.
  - Ejemplo Cod_producto y su nombre. Despues el disñador añadirá una *imagen* y el gerente fijara su *precio*.
- Con un valor **NOT NULL**, es necesario ingresar un valor en cada registro de la tabla

3.2 DEFAULT

- Indicas un valor por defecto o predeterminado para agilizar la carga de datos.
- Se puede definir el valor por omisión de **NULL**.
- Se puede definir en **TIMESTAMP** un valor actual por defecto (Current_Timestam)

Ver ejemplo de tabla 1:

3.2 UNIQUE

- Garantiza que no se escriben valores duplicados en columnas que no forman parte de la PRIMARY KEY.
- A diferencia de las PK, la restricción UNIQUE permite valores NULL.
- Sin embargo, solo se admite un valor null por columna.

3.2 EJEMPLOS

```
id int(11) NOT NULL auto_increment,
edad TINYINT UNSIGNED,
apellidos varchar(50) NOT NULL,
codigoPostal int(5) DEFAULT NULL,
fecha date DEFAULT NULL,
city varchar(50) DEFAULT 'MADRID'
```

3.2 DETALLES DE UNA COLUMNA

- COLUMNAS DE UNA TABLA:
  - PK → PRIMARY KEY
  - NN → NOT NULL
  - UQ → UNIQUE
  - BIN → BINARIO
  - UN → UNSIGNED
  - ZF → ZEROFILL
  - AI → AUTO_INCREMENT
  - DEFAULT

LENGUAJE DDL
1. Creación de esquemas

https://dev.mysql.com/doc/refman/5.7/en/sql-data-definition-statements.html

3.3.1 CREACION DE ESQUEMAS

A partir de la versión 5.0.2, en MySQL la creación de esquemas y bases de datos es exactamente lo mismo.

DATABASE = SCHEMA

MySQL soporta múltiples esquemas.

COMANDOS:

create database COLEGIO;

create schema CLUB;

3.3.1 VERSIÓN INSTALADA

Mostrar variables de entorno:

mysql>SHOW VARIABLES LIKE "%version%";

```
+-------------------------+-------------------------+
| Variable_name           | Value                   |
+-------------------------+-------------------------+
| innodb_version          | 5.5.62                  |
| protocol_version        | 10                      |
| slave_type_conversions  |                         |
| version                 | 5.5.62-0ubuntu0.14.04.1 |
| version_comment         | (Ubuntu)                |
| version_compile_machine | x86_64                  |
| version_compile_os      | debian-linux-gnu        |
+-------------------------+-------------------------+
7 rows in set (0.00 sec)
```

jaime@Profesor:~$ mysql --version

3.3.1 CREACION DE ESQUEMAS

CREATE DATABASE STATEMENT

CREATE {DATABASE | SCHEMA} [IF NOT EXISTS]
db_name [create_specification] ...

create_specification:
[DEFAULT] CHARACTER SET [=] charset_name
| [DEFAULT] COLLATE [=] collation_name
| DEFAULT ENCRYPTION [=] {'Y' | 'N'}

```
1  shell> mysqladmin [options] command [command-arg] [command [command-arg]] ...
```

3.3.1 CREACION DE ESQUEMAS

ALTER DATABASE STATEMENT

```
ALTER {DATABASE | SCHEMA} [db_name]
[create_specification] ...

create_specification:
    [DEFAULT] CHARACTER SET [=] charset_name
  | [DEFAULT] COLLATE [=] collation_name
  | DEFAULT ENCRYPTION [=] {'Y' | 'N'}
```

3.3.1 CREACION DE ESQUEMAS

DROP DATABASE STATEMENT

DROP {DATABASE | SCHEMA} [IF EXISTS] db_name

- Borra todas las tablas y datos existentes.
- Devuelve el número de tablas que fueron eliminadas.
- No elimina las tablas temporales que fueron creadas en dicha base de datos. Se requiere terminar tu sesión para ser eliminadas.

3.3.1 TABLESPACE

Los espacios de tabla son unidades de almacenamiento lógicas de motores de base de datos relacionales como **InnoDB**, que contienen todos los datos del sistema de base de datos.

Cada uno de los espacios de tabla contiene como mínimo un fichero de datos físico del sistema operativo subyacente en el que se almacenan tanto tablas de bases de datos como índices.

3.3.1 TABLESPACE

Por tanto, el espacio de tabla del sistema contiene (versión MySQL 5.6):

- Diccionario de datos InnoDB
- DubleWrite Buffer
- Cambiar Búfer
- Deshacer registros
- Tablas
- Índice de datos

Los ibd son donde se guardan los tablespace.

3.3.1 TABLESPACE

CREATE TABLESPACE Statement:

CREATE [UNDO] TABLESPACE tablespace_name

InnoDB and NDB:

[ADD DATAFILE 'file_name']

InnoDB only:

[FILE_BLOCK_SIZE = value]
[ENCRYPTION [=] {'Y' | 'N'}]

NDB only:

USE LOGFILE GROUP logfile_group
[EXTENT_SIZE [=] extent_size]
[INITIAL_SIZE [=] initial_size]
[AUTOEXTEND_SIZE [=] autoextend_size]
[MAX_SIZE [=] max_size]
[NODEGROUP [=] nodegroup_id]
[WAIT]
[COMMENT [=] 'string']

InnoDB and NDB:

[ENGINE [=] engine_name]

InnoDB debe de incluir un fichero con su extensión *.idb

3.3.1 TABLESPACE

CREATE TABLESPACE Statement: ejemplos

```
mysql> CREATE TABLESPACE `ts2` ADD DATAFILE 'ts2.ibd' FILE_BLOCK_SIZE = 8192 Engine=InnoDB;
///Se crea un multiplo de 8 bytes

mysql> CREATE TABLE t4 (c1 INT PRIMARY KEY) TABLESPACE ts2 ROW_FORMAT=COMPRESSED KEY_BLOCK_SIZE=8;

/// Añade una table comprimida al tablespace
```

+INFO:

https://dev.mysql.com/doc/refman/5.5/en/create-tablespace.html

3.3.1 TABLESPACE

Ejemplo 01: Crear ESQUEMA y una tabla.

```
mysql> CREATE DATABASE prueba;
mysql> USE prueba;

mysql> CREATE TABLE IF EXISTS `primeratabla`;
CREATE TABLE `primeratabla` (
`id` int(11) DEFAULT NULL,
`primervalor` varchar(100) DEFAULT NULL,
`segundovalor` varchar(50) DEFAULT NULL
PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

3.3.1 TABLESPACE

Ejemplo 01:

mysql> ALTER TABLE primeratabla DISCARD TABLESPACE;

<<<Descartamos el Tablespace de la tabla primeratabla

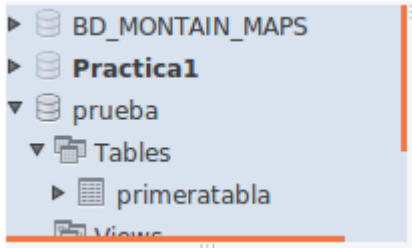
mysql> ALTER TABLE primeratabla IMPORT TABLESPACE;

3.3.1 TABLESPACE

Ejemplo 01: Resultado.

```
mysql> CREATE TABLE primeratabla (id int(11) DEFAULT NULL, primervalor varchar(100) DEFAULT
NULL, segundovalor varchar(50) DEFAULT NULL)ENGINE=InnoDB DEFAULT CHARSET=utf8;
Query OK, 0 rows affected (0.02 sec)
```

SHOW tables;




```
mysql> show tables;
+------------------+
| Tables_in_prueba |
+------------------+
| primeratabla     |
+------------------+
1 row in set (0.00 sec)
```

Al dar un error por el servidor se tiene que ir a ver el server LOG.

Server → Server logs

InnoDB:	Error: table `prueba`.`primeratabla` InnoDB: is in the system tablespace 0 which cannot be discarded

Configuration File: /etc/mysql/my.cnf

3.3.1 TABLESPACE

Ejemplo 01: Resultado.

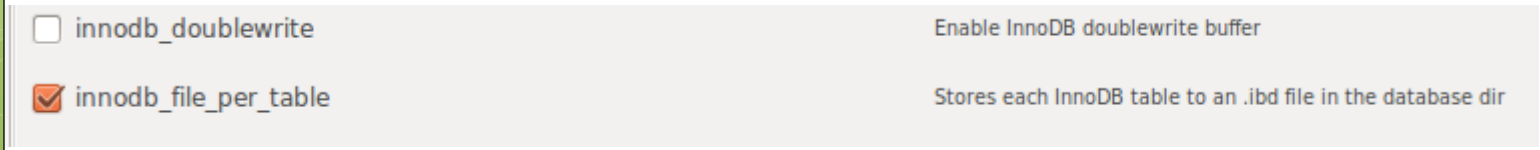
Edito el fichero con superusuario mysql.cnf o mysql.ini (Windows):

gedit /etc/mysql/my.cnf

Localizo la sección [mysqld] y añado:

[mysqld]innodb_file_per_table=1

Reinicio y miro el resultado



3.3.1 TABLESPACE

Ejemplo 01: Resultado.

¿Dónde se me ha creado la base de datos?

/var/lib/mysql/directorio

3.3.1 DOCUMENTACION

MySQLAdmin:
https://dev.mysql.com/doc/refman/8.0/en/mysqladmin.html
Comando créate database
https://dev.mysql.com/doc/refman/8.0/en/drop-database.html
Comando alter Database
https://dev.mysql.com/doc/refman/8.0/en/alter-database.html
Comando drop Database
https://dev.mysql.com/doc/refman/8.0/en/drop-database.html
EJEMPLO DATABASE, TABLE and DEFAULT CHARACTER
https://dev.mysql.com/doc/refman/5.7/en/charset-examples.html
TABLESPACE: https://dev.mysql.com/doc/refman/8.0/en/create-tablespace.html

3.3.1 ACTIVIDAD

Actividad CLASE:

- Observa la versión de tu SGBD MySQL/Oracle instalado.
- Create un esquema para tu instituto con el nombre del instituto.
- Añade una tabla con el nombre de la clase que tenga el campo nombre.
- Comprueba cuantos esquemas tienes en tu gestor de Bases de Datos.
- Visualízalo en WorkBench.
- Selecciona tu base de datos.
- Elimina el esquema que has creado.
- Verifica que se ha eliminado.

LENGUAJE DDL
2. Creación de tablas

3.3.2 CREACION DE TABLAS

- Es uno de las sintaxis más larga de SQL.
- Ver enlace: https://dev.mysql.com/doc/refman/5.7/en/create-table.html
- RESUMEN:

CREATE [TEMPORARY] TABLE [IF NOT EXISTS] *tbl_name*
(*create_definition*,...)
[*table_options*]
[*partition_options*]

3.3.2 CREACION DE TABLAS

- *create_definition*:

col_name column_definition

| {INDEX | KEY} [*index_name*] [*index_type*] (*key_part*,...) [*index_option*] ...

| {FULLTEXT | SPATIAL} [INDEX | KEY] [*index_name*] (*key_part*,...) [*index_option*] ...

| [CONSTRAINT [*symbol*]] PRIMARY KEY [*index_type*] (*key_part*,...) [*index_option*] ...

| [CONSTRAINT [*symbol*]] UNIQUE [INDEX | KEY] [*index_name*] [*index_type*] (*key_part*,...) [*index_option*] ...

| [CONSTRAINT [*symbol*]] FOREIGN KEY [*index_name*] (*col_name*,...) *reference_definition*

| CHECK (*expr*)

column_definition:
data_type

3.3.2 CREACION DE TABLAS

***column_definition*:**

data_type [NOT NULL | NULL] [DEFAULT *default_value*]
[AUTO_INCREMENT] [UNIQUE [KEY]] [[PRIMARY] KEY]
[COMMENT '*string*']
[COLLATE *collation_name*]
[COLUMN_FORMAT {FIXED | DYNAMIC | DEFAULT}]
[STORAGE {DISK | MEMORY}]
[*reference_definition*]
| *data_type*
[COLLATE *collation_name*]
[GENERATED ALWAYS] AS (*expr*)
[VIRTUAL | STORED] [NOT NULL | NULL]
[UNIQUE [KEY]] [[PRIMARY] KEY]
[COMMENT '*string*']
[*reference_definition*]

3.3.2 CREACION DE TABLAS. Ejemplo1

```
CREATE TABLE IF NOT EXISTS users (
user_id     INT(8) NOT NULL AUTO_INCREMENT,
user_name   VARCHAR(30) NOT NULL,
user_date   DATETIME NOT NULL,
user_level  INT(8) NOT NULL DEFAULT 0,
UNIQUE INDEX user_name_unique (user_name),
PRIMARY KEY (user_id)
) ENGINE=INNODB DEFAULT CHARSET=utf8;
```

El nombre puede ser especificado como: db_name.table_name

Si los dejo en blanco por defecto son NULL

Solo aplicable a tipos de datos INTEGER y FLOAT

3.3.2 MOSTRAR COLUMNAS DE TABLA.

SINTAXIS (versión 5.0.1 ó superior)

SHOW COLUMNS FROM <BBDD>.<table>

Ejemplo

SHOW COLUMNS FROM prueba.primeratabla;

SHOW COLUMNS FROM primeratabla FROM prueba;

```
mysql> show columns from primeratabla FROM prueba;
+--------------+--------------+------+-----+---------+-------+
| Field        | Type         | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| id           | int(11)      | NO   | PRI | 0       |       |
| primervalor  | varchar(100) | YES  |     | NULL    |       |
| segundovalor | varchar(50)  | YES  |     | NULL    |       |
+--------------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

OTRA FORMA

USE prueba;

DESCRIBE primeratabla;

3.3.2 OTRAS TIPOS DE TABLA.

DUPLICAR UNA TABLA

CREATE TABLE *new_table* LIKE *origin_table*; // tabla vacia

CREATE TABLE *new_table* AS SELECT * FROM *origin_table*;
//tabla llena

TABLAS TEMPORALES → DEPRECATED AS OF SQL 5.7

ALMACENAMIENTO EN MEMORIA O ALMACENAMIENTO EN DISCO.

STORAGE DISK

STORAGE MEMORY

→ SE REQUIERE CREAR UN TABLESPACE

3.3.2 ENCODING WITH UTF8

UTF8 es una codificación de amplitud variable (variable-width encoding) que puede representar todos los caracteres en el conjunto de caracteres Unicode.

- Todos los archivos HTML, PHP y Scripts deben ser codificados en **UTF8**.
  - NOTEPAD++ te permite elegir el formato de codificación.

NOTAS:

- **UTF-8** no existe en MySQL sino UTF8.

3.3.2 ENCODING WITH UTF8

PROBLEMAS:

- SELECT con palabras acentuadas.
- Codificar nombres de todo el mundo.
- Migrar bases de datos codificadas en Latin1.

3.3.2 ENCODING WITH UTF8

- **Comprobar la codificación:**

mysql> show variables like 'char%';

```
mysql> show variables like 'char%';
+--------------------------+----------------------------+
| Variable_name            | Value                      |
+--------------------------+----------------------------+
| character_set_client     | utf8                       |
| character_set_connection | utf8                       |
| character_set_database   | latin1                     |
| character_set_filesystem | binary                     |
| character_set_results    | utf8                       |
| character_set_server     | latin1                     |
| character_set_system     | utf8                       |
| character_sets_dir       | /usr/share/mysql/charsets/ |
+--------------------------+----------------------------+
8 rows in set (0.00 sec)
```

```
'char%';
+----------------------------+
| Value                      |
+----------------------------+
| utf8                       |
| utf8                       |
| utf8                       |
| binary                     |
| utf8                       |
| utf8                       |
| utf8                       |
| /usr/share/mysql/charsets/ |
+----------------------------+
```

PROBLEMA: MySQL UTF-8 utiliza 3 bytes mientras que UTF-8 requiere 4 bytes.

Solución: cambiar codificación

3.3.2 ENCODING WITH UTF8

- **Cambio de codificación** fichero /etc/mysql/my.cnf

[mysql]

default-character-set=utf8

[mysqldump]

default-character-set=utf8

[mysqld]

character-set-server=utf8
character_set_client=utf8

```
mysql> show variables like 'char%';
+--------------------------+-----------------
| Variable_name            | Value
+--------------------------+-----------------
| character_set_client     | utf8
| character_set_connection | utf8
| character_set_database   | utf8
| character_set_filesystem | binary
| character_set_results    | utf8
| character_set_server     | utf8
| character_set_system     | utf8
| character_sets_dir       | /usr/share/mysql/
+--------------------------+-----------------
8 rows in set (0.00 sec)
```

- **Crear esquemas y tablas con UTF8**

3.3.2 ENCODING WITH UTF8

Para exportar tablas, la primera sentencia antes de exportar datos:

```
set names utf8;
```

Para cambiar tablas con el comando:

```
ALTER TABLE <tabla> CONVERT TO CHARSET UTF8;
```

Para definir COLLATION usar:

utf8_general_ci

utf8_spanish_ci

Ver configuración: mysql> `show character set;`

3.3.2 CREACIÓN DE TABLAS con UTF8

```
CREATE TABLE ‘ejemplo1’{

….

} ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE utf8_general_ci
COMMENT=‘comentario1’;
```

3.3.2 OTRAS TIPOS DE TABLA.

ALMACENAMIENTO EN MEMORIA O ALMACENAMIENTO EN DISCO.

```
mysql> CREATE TABLE t1 (
    -> c1 INT STORAGE DISK,
    -> c2 INT STORAGE MEMORY
    -> ) TABLESPACE ts_1 ENGINE NDB;
Query OK, 0 rows affected (1.06 sec)
```

EJERCICIO DE CREACION/BORRADO de TABLAS.

Añadir a la base de datos "***prueba***" una nueva tabla que se llame personas formado por los campos de máximo 20 caracteres.

personas(dni, nombre, apellido1, apellido2)

- Visualiza las tablas que component la BD prueba.
- Elimina la segundatabla de la BD prueba.

LENGUAJE DDL

3.3.3 Modificar TABLAS

ALTER TABLE

3.3.3 MODIFICAR TABLAS.

Permite:

- Añadir una columna,
- Modificar una columna,
- Eliminar una columna,
- Renombrar una Columna o una tabla.

URL:
https://dev.mysql.com/doc/refman/5.7/en/alter-table.html

3.3.3 MODIFICAR TABLAS.

SINTAXIS RESUMIDA:

ALTER TABLE tbl_name [alter_specification [, alter_specification] ...] [partition_options]

alter_specification:

ADD

CHANGE

DROP

MODIFY

RENAME

3.3.3 MODIFICAR TABLAS.

SINTAXIS Añadir columna:

ALTER TABLE table_name ADD new_column_name column_definition [FIRST | AFTER column_name];

Ejemplo BD SAKILA:

```
ALTER TABLE sakila.actor
  ADD fecha_nacimiento TIMESTAMP NOT NULL,
  ADD email varchar(30) NOT NULL,
AFTER last_update;
```

3.3.3 MODIFICAR TABLAS.

SINTAXIS Modificar columna:

ALTER TABLE table_name MODIFIY column_name column_definition [FIRST | AFTER column_name];

Ejemplo BD SAKILA:

```
ALTER TABLE sakila.actor
  MODIFY email varchar(40) NULL PRIMARY KEY
AFTER last_update;
```

3.3.3 MODIFICAR TABLAS.

SINTAXIS eliminar columna:

ALTER TABLE table_name DROP COLUMN column_name;

Ejemplo BD SAKILA:

```
ALTER TABLE sakila.actor
    DROP COLUMN email;
```

3.3.3 MODIFICAR TABLAS.

SINTAXIS Renombrar columna:

ALTER TABLE table_name CHANGE COLUMN old_column_name new_name *datatype*

[FIRST | AFTER column_name];

Ejemplo BD SAKILA:

ALTER TABLE sakila.actor

CHANGE COLUMN fecha_nacimiento birthday **tipodeDATO**;

3.3.3 MODIFICAR TABLAS.

SINTAXIS Renombrar tabla:

ALTER TABLE table_name RENAME TO new_table_name;

*nota: se puede omitir el TO

Ejemplo BD SAKILA:

ALTER TABLE sakila.actor **RENAME TO** actor2;

ALTER TABLE sakila.actor **RENAME** actor2;

3.3.3 MODIFICAR TABLAS.

SINTAXIS Reasignar PK:

ALTER TABLE table_name DROP PRIMARY KEY (columns);

ALTER TABLE table_name ADD PRIMARY KEY (columns);

Ejemplo BD SAKILA:

ALTER TABLE sakila.actor **ADD PRIMARY KEY** (actor_id,first_name);

```
jaime@Profesor:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 40
Server version: 5.5.62-0ubuntu0.14.04.1 (Ubuntu)
```

```
mysql> ALTER TABLE clientes ADD PRIMARY KEY(dni);
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> SHOW COLUMNS FROM clientes;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| dni       | varchar(9)  | NO   | PRI |         |       |
| apellido1 | varchar(20) | YES  |     | NULL    |       |
| apellido2 | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

3.3.3 MODIFICAR TABLAS.

SINTAXIS Reasignar UNIQUE:

ALTER TABLE table_name ADD UNIQUE (column);

SINTAXIS Eliminar una restricción:

ALTER TABLE table_name DROP <restriccion> (column);

Ej.: ALTER TABLE personas3 DROP INDEX email;

3.3.3 MODIFICAR TABLAS.

SINTAXIS DEFAULT:

ALTER TABLE table_name

ALTER column_name SET DEFAULT "column";

Ej1. MySQL:

ALTER TABLE personas3 ALTER COLUMN ciudad SET DEFAULT 'Madrid';

Ej2. MSSQL: ALTER TABLE personas3 ADD CONSTRAINT df_1 DEFAULT 'Madrid' FOR ciudad;

Ej2. ORACLE: ALTER TABLE personas3

MODIFY (ciudad varchar2(25) DEFAULT 'Madrid' ;

EJERCICIO 3.1DE MODIFICACION de TABLAS. (parte1)

- De la base de datos "***prueba***" visualiza las tablas.
- Modifica el nombre de la table personas para que se llamen ***clientes***.
- Añade dos nuevas columnas que se llamen ***provincial y localidad,*** *con valor MADRID*.
- Añade una nueva columna que almacene la ***fechadenacimiento*** de tipo date.

clientes(dni, nombre, apellido1, apellido2, provincial, localidad, fechadeNacimiento)

EJERCICIO 3.1DE MODIFICACION de TABLAS(parte 2)

- Cambia el nombre de un campo o columna fechadenacimiento por fecha_nacimiento.

clientes(dni, nombre, apellido1, apellido2, telefono, fecha_nacimiento)

- Cambia el formato del campo nombre y añádele 5 carácteres más.
- Cambia el campo fecha_nacimiento de tipo date a tipo datetime.
- Pon el campo telefono como PK y que sea único

EJERCICIO 3.1DE MODIFICACION de TABLAS(parte 2)

- Cambia el nombre de un campo o columna fechadenacimiento por fecha_nacimiento.

clientes(dni, nombre, apellido1, apellido2, telefono, fecha_nacimiento)

- Cambia el formato del campo nombre y añádele 5 carácteres más.
- Cambia el campo fecha_nacimiento de tipo date a tipo datetime.
- Pon el campo telefono como PK y que sea único

ACTIVIDAD PARA CASA. MODIFICACION de TABLAS 2.

- Realizar el ejemplo guiado con una tabla T1 de la documentación official de MySQL 5.5 ó 5.6 ó 5.7

https://dev.mysql.com/doc/refman/5.7/en/alter-table-examples.html

ACTIVIDAD PARA CASA. MODIFICACION de TABLAS 3.

Crea en la base de datos **ejemplo01**, la siguiente tabla denominada **alumno**.

```
+------------+---------------+------+-----+---------+----------------+
| Field      | Type          | Null | Key | Default | Extra          |
+------------+---------------+------+-----+---------+----------------+
| id         | int(11)       | NO   | PRI | NULL    | auto_increment |
| Nombre     | text          | NO   |     | NULL    |                |
| Ap_Paterno | text          | NO   |     | NULL    |                |
| Ap_Materno | text          | NO   |     | NULL    |                |
| Sexo       | text          | NO   |     | NULL    |                |
| Edad       | varchar(3)    | NO   |     | NULL    |                |
| Municipio  | text          | NO   |     | NULL    |                |
| Telefono   | text          | NO   |     | NULL    |                |
| Correo     | text          | NO   |     | NULL    |                |
| Redes      | text          | NO   |     | NULL    |                |
| Tarjeta    | text          | NO   |     | NULL    |                |
| Monedero   | decimal(65,0) | NO   |     | NULL    |                |
+------------+---------------+------+-----+---------+----------------+
12 rows in set (0.003 sec)
```

ACTIVIDAD PARA CASA. MODIFICACION de TABLAS 3.

1º)Cambia los campos y asigna los siguientes valores:

telefono, **correo**, **redes** y **tarjeta** por defecto ‘ ’.

monedero asígnale el valor 0.

municipio asígnale tu municipio.

2º) Cambia edad por un dato de tipo entero positivo y máximo tres cifras.

3º) Inserta un campo llamado **estudios** de tipo lista que admita solo “DAM”,”DAW” y “ASIR”.

ACTIVIDAD PARA CASA. MODIFICACION de TABLAS 3.

4º) Cambia el campo **sexo** por una lista con dos opciones.:

5º) Cambia el campo **nombre** por **nombre_completo** y que sea una cadena de 50 caracteres.

6º) El campo **telefono** ha de ser de tipo numérico y que ocupe al menos 9 digitos.

BIBLIOGRAFIA

- SQL TUTORIAL.

https://www.w3schools.com/sql/default.asp

- Primary KEY.

https://www.w3schools.com/sql/sql_primarykey.asp

- FOREIGN KEY CONSTRAINTS

https://dev.mysql.com/doc/refman/5.7/en/create-table-foreign-keys.html

LENGUAJE DDL
3.4. CONSTRAINT

3.4 DEFINIR PRIMARY KEY.

Existen 3 formas de definir un atributo como PK:

- Al definir el dato (1) → ORACLE, Access, SQL Server, ***MySQL***
- Al definir la tabla (2) → MySQL
- Al modificar la tabla (3) → ORACLE, MySQL, Access, SQL Server

3.4 DEFINIR PRIMARY KEY.

Existen 3 formas de definir un atributo como PK:

- Al definir los datos (1):

```
CREATE TABLE personas (
    dni varchar(9) NOT NULL PRIMARY KEY,
    nombre varchar(25) NOT NULL,
    apellidos varchar(50) NOT NULL,
    salario int
);
```

3.4 DEFINIR PRIMARY KEY.

Existen 3 formas de definir un atributo como PK:

- Al definir la tabla (2):

```
CREATE TABLE Persons (
    dni varchar(9) NOT NULL,
    nombre varchar(25) NOT NULL,
    apellidos varchar(50),
    salario int,
    PRIMARY KEY (dni)
);
```

3.4 DEFINIR PRIMARY KEY.

Existen 3 formas de definir un atributo como PK:

- Al modificar la tabla (3)

```
ALTER TABLE Persons ADD PRIMARY KEY (dni);
```

(ORACLE, MySQL, Access, SQL Server)

3.4 DEFINIR VARIAS PRIMARY KEY.

```
CREATE TABLE persons (
    dni varchar(9) NOT NULL,
    nombre varchar(25) NOT NULL,
    apellidos varchar(50),
    salario int,
    CONSTRAINT PK_PERSON PRIMARY KEY (dni, name) );
```

Para modificar una tabla:

```
ALTER TABLE persons
ADD CONSTRAINT PK_PERSON PRIMARY KEY (dni,nombre);
```

2.6 RESTRICCIONES: PRIMARY KEY.

Para eliminar una PK de una table se utiliza la siguiente sintaxis:

```
ALTER TABLE Persons
DROP PRIMARY KEY;
```

⬅ MySQL

Para el resto de SGBD:

```
ALTER TABLE Persons
DROP CONSTRAINT PK_PERSON;
```

⬅ ORACLE, SQL SERVER, ACCESS

3.4 CONSTRAINTS-RESTRICCIONES

Dan la **validez** e **integridad** de los datos:

- Asegura evitar datos duplicados.
- Los valores introducidos tienen sentido lógico.

Ejemplo: Un campo de edad de alumno este entre 18 y 35.

NO FUNCIONA EN MySQL

Aplicable a: PRIMARY KEY, FOREIGN KEY, NOT NULL, DEFAULT, CHECK y UNIQUE.

3.4 RESTRICCIONES: CONSTRAINTS.

NOT NULL,

No se permitirá el ingreso o la actualización de un valor que sea igual a NULL.

```
mysql> show COLUMNS FROM prueba.personas;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| dni       | varchar(9)  | NO   | PRI | NULL    |       |
| name      | varchar(25) | NO   |     | NULL    |       |
| apellidos | varchar(50) | NO   |     | NULL    |       |
| salario   | int(11)     | YES  |     | NULL    |       |
| edad      | int(11)     | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

DEFAULT,

Valor por defecto a un campo que no se asigne explícitamente.

3.4 RESTRICCIONES: CONSTRAINTS.

UNIQUE,

Asegura que todos los valores presentes en una columna sean diferentes.

Al crear una PK se establece automáticamente el valor de UNIQUE.

CHECK

Campo de valores permitidos en una campo dado, utilizando los valores presentes de otras columnas mediante una expresión lógica.

(Valido para ORACLE, SQL SEERVER)

3.4 RESTRICCIONES: CONSTRAINTS.

PRIMARY KEY

Los campos que lo definen tienen que ser valores únicos y no acepta valores NULL.

FOREIGN KEY,

Para establecer la relación entre dos tablas, cuya clave externa contiene la clave primaria.

3.4 RESTRICCIONES: EJEMPLO 1.

```
CREATE TABLE IF NOT EXISTS libros(
id_book varchar(15) NOT NULL PRIMARY KEY ,
isbn varchar(20) NOT NULL UNIQUE,
n_page decimal(5,0) CHECK (no_page>0) ,
price decimal(8,2),
date_public DATE CHECK (date_public LIKE '--/--/----')
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE utf8_general_ci;
```

3.4 RESTRICCIONES: EJEMPLO 2.

```
CREATE TABLE IF NOT EXISTS ejemplo2(
dni varchar(9) NOT NULL,
country varchar(25) NOT NULL CHECK (country IN ('ESPAÑA','FRANCIA','PORTUGAL')),
city varchar(25) NOT NULL DEFAULT 'MADRID',
PRIMARY KEY (dni)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE utf8_general_ci;
```

3.4 RESTRICCIONES: EJEMPLO 2b.

```
CREATE TABLE IF NOT EXISTS ejemplo2b(
dni varchar(9) NOT NULL,
country varchar(25),
city varchar(25),
CHECK ((country='Spain' AND city='Madrid')
OR (country='Spain' AND city='Barcelona')) ,
PRIMARY KEY (dni)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE utf8_general_ci;
```

3.4 RESTRICCIONES: RESUMEN.

En el ejemplo 1 y 2 se han visto dos formas diferentes de definir las PRIMARY KEY sobre el campo DNI:

```
CREATE TABLE persona(
dni varchar(9) PRIMARY KEY,
...);
```

Oracle, SQL SERVER, MySQL, MS Access.

```
CREATE TABLE persona(
dni varchar(9),
....
PRIMARY KEY (dni));
```

MySQL.

```
ALTER TABLE persona ADD PRIMARY KEY(dni);
```

3.4 RESTRICCIONES: SET Y ENUM

Proporcionan integridad a los valores que puede contener un campo.

Equivalen a LISTAS definidas.

- **ENUM** contiene un solo valor de la lista.
- **SET** contiene múltiples valores de la lista.

Si es nulo el índice devuelve cero y el primer valor retorna un 1.

3.4 RESTRICCIONES: ENUM

```
CREATE TABLE sizes (
size ENUM('small', 'medium', 'large') NOT
NULL DEFAULT 'medium' );
```

- Se pueden definir hasta 65.535 valores en la lista.
- El valor por defecto es **NULL** sino el primer valor**.**
- Cada valor de la lista es enumerado con un índice:
  - 1 → small,
  - 2 → médium
  - 3 → large
- Si se introduce un valor no perteneciente a la lista el valor es una cadena vacía o índice cero.
- Para retornar el índice se puede sumar zero al campo:

  SELECT size+0 FROM …

3.4 RESTRICCIONES: SET

```
CREATE TABLE letters (
      letter SET('a', 'b', 'c') );
```

- Se pueden definir hasta 64 valores en la lista.
- Contiene Zero, uno o varios valores.
- Los valores no pueden contener comas.
- Cada valor de la lista representa un bit, si lo contiene representa un “1” sino un “0”.
- Ejemplo 7 en binario sería 0111 y contiene a,b,c:
  - ‘a’ 1 → 0001
  - ‘b’ 2 → 0010
  - ‘c’ 4 → 0100
  - ‘d’ 8 → 1000

LENGUAJE DDL
3.5. Relacionar TABLAS FOREIGN KEY

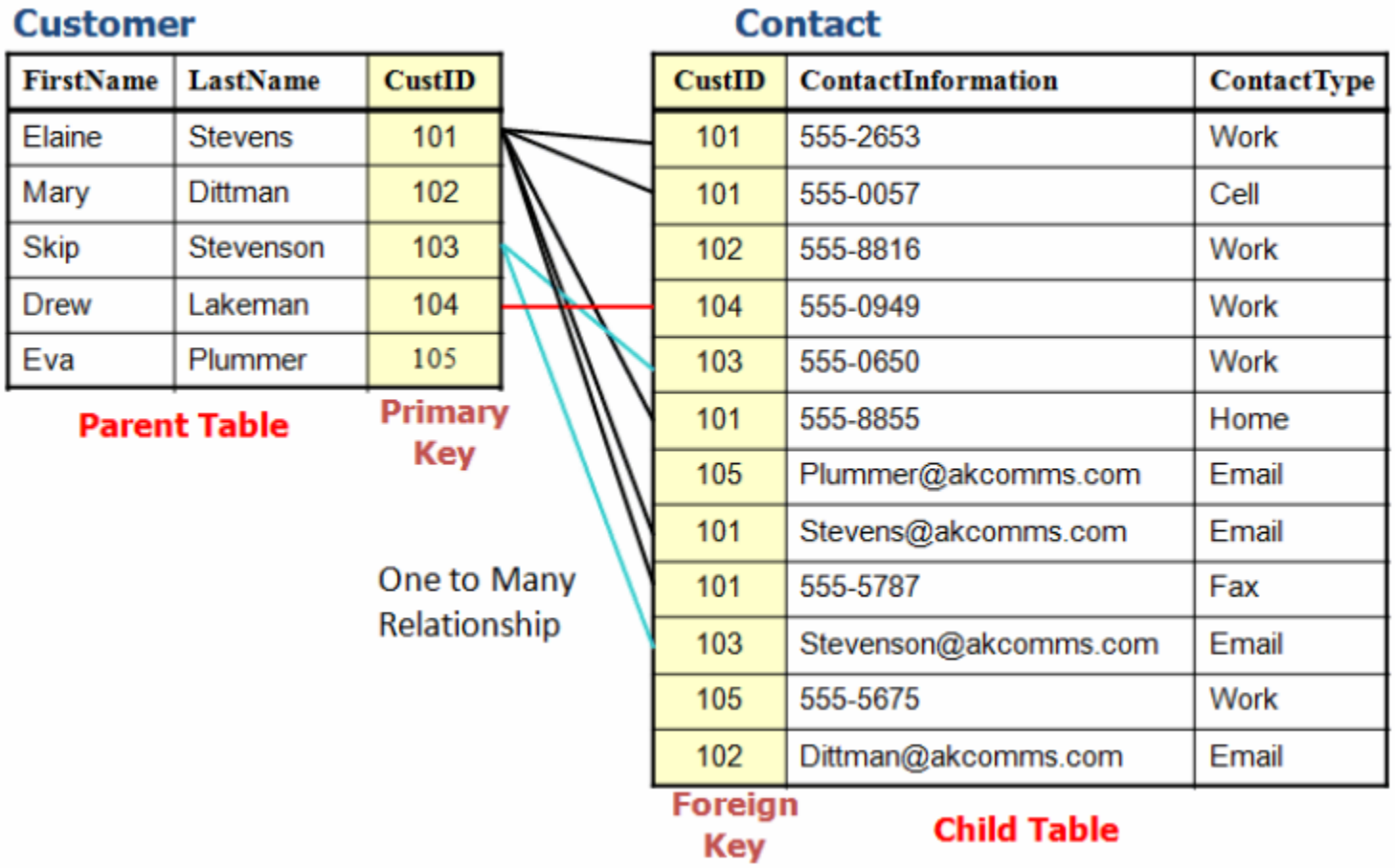
3.5 FOREIGN KEY

- Sirven para unir dos tablas:
  - Tabla referencia con PK.
  - Tabla secundaria con FK.

- Para definer claves foráneas ambas tablas deben de ser **InnoDB**, ninguna **temporal**.

- Si se proporciona un símbolo **CONSTRAINT**, este debe de ser único en la base de datos.

3.5 FOREIGN KEY



FirstName	LastName	CustID
Elaine	Stevens	101
Mary	Dittman	102
Skip	Stevenson	103
Drew	Lakeman	104
Eva	Plummer	105

CustID	ContactInformation	ContactType
101	555-2653	Work
101	555-0057	Cell
102	555-8816	Work
104	555-0949	Work
103	555-0650	Work
101	555-8855	Home
105	Plummer@akcomms.com	Email
101	Stevens@akcomms.com	Email
101	555-5787	Fax
103	Stevenson@akcomms.com	Email
105	555-5675	Work
102	Dittman@akcomms.com	Email

3.5 RESTRICCIONES: FOREIGN KEY.

- Las columnas de tipo **BLOB** y **TEXT** no pueden incluirse en una clave foránea, pues la longitud debe estar prefijada.

- En la tabla referenciada las columnas se deben de listar en primer lugar, con un **indice** creado automáticamente por innoDB.

(en versiones antiguas de MySQL debían crearse los índices manualmente 4.1.8)

3.5 EJEMPLO 03: FOREIGN KEY.

TABLA PERSONAS

dni	9 caracteres, not null, **PK**
nombre	30 caracteres, not null
apellidos	50 caracteres, not null
ciudad	Por defecto MADRID

TABLA TELEFONOS

dni_persona	**FK**
Telefono	Entero de 9

FOREIGN KEY (campo_FK) REFERENCES tabla_PK(campo_PK)

3.5 EJEMPLO 03: FOREIGN KEY.

```
CREATE TABLE IF NOT EXISITS personas (
dni varchar(9) NOT NULL PRIMARY KEY,
nombre varchar(30) NOT NULL,
apellidos varchar(50) NOT NULL,
ciudad varchar(15) DEFAULT 'MADRID');
```

```
CREATE TABLE IF NOT EXISITS telefonos(
dni_person varchar(9) NOT NULL,
telefono int(9) NOT NULL,
PRIMARY KEY (dni_person, telefono),
FOREIGN KEY (dni_person) REFERENCES
personas(dni));
```

3.5 EJEMPLO 03: FOREIGN KEY.

```
CREATE TABLE IF NOT EXISITS personas (
dni varchar(9) NOT NULL PRIMARY KEY,
nombre varchar(30) NOT NULL,
apellidos varchar(50) NOT NULL,
ciudad varchar(15) DEFAULT 'MADRID'
);
```

```
CREATE TABLE IF NOT EXISITS telefonos(
dni_person varchar(9) NOT NULL PRIMARY KEY
FOREIGN KEY REFERENCES personas(dni),
telefono int(9) NOT NULL PRIMARY KEY
);
```

3.5 EJEMPLO 03: FOREIGN KEY.

Con **ALTER TABLE**, una vez creada la tabla:

```
ALTER TABLE telefonos
ADD FOREIGN KEY (dni_person) REFERENCES personas(dni));
```

```
ALTER TABLE telefonos
ADD CONSTRAINT FK_dniPerson FOREIGN KEY (dni_person) REFERENCES personas(dni));
```

```
ALTER TABLE telefonos
DROP FOREIGN KEY 'FK_dniPerson';
```

3.5 EJEMPLO 03: FOREIGN KEY.

ERROR ( ):

No deja eliminar CONTRAINT!!!

```
ALTER TABLE telefonos
DROP CONSTRAINT FK_dniPerson;
```

SQL SERVER / ORACLE / MS Access

Alternativa Temporal para deshabilitar FK (diapo 132):

```
SET FOREIGN_KEY_CHECKS=0;
```

Volver a habilitar:

```
SET FOREIGN_KEY_CHECKS=1;
```

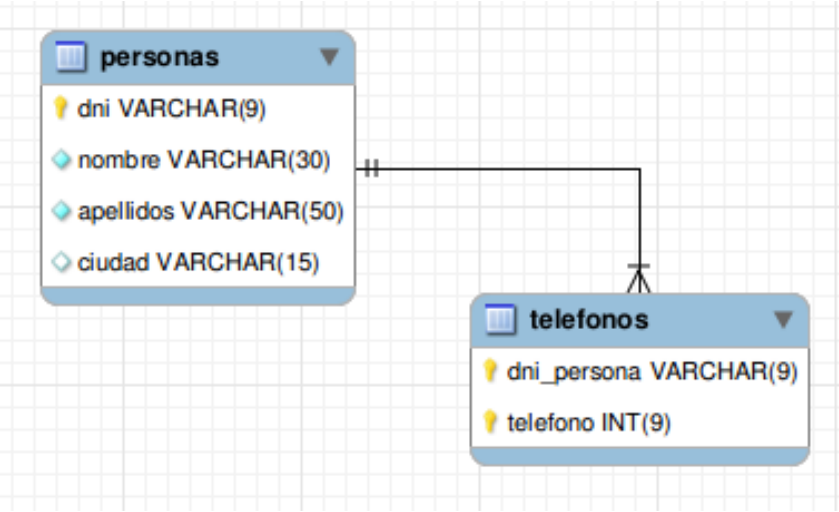
3.5 EJEMPLO3: FOREIGN KEY.

TABLA PERSONAS

dni	nombre	apellidos	ciudad
00000000A	Juan	Torres	Madrid
11111111B	Sara	Gómez	Barcelona
22222222C	Cristina	Almansa	Madrid

TABLA TELEFONOS

dni_persona	TELEFONO
00000000A	911234567
00000000A	912222222
00000000A	913333333
00000000A	913333333



3.5 EJEMPLO3: FOREIGN KEY.

```
INSERT INTO personas(dni,nombre, apellidos) VALUES
('00000000A','Juan','Torres');

INSERT INTO personas VALUES ('11111111B', 'Sara', 'Gomez',
'Barcelona');

INSERT INTO personas(dni,nombre, apellidos) VALUES
('22222222C',' Cristina',' Almansa');

SELECT * FROM personas;

INSERT INTO telefonos VALUES ('00000000A',911111111);
INSERT INTO telefonos VALUES ('00000000A',912222222);
INSERT INTO telefonos VALUES ('00000000A',913333333);
INSERT INTO telefonos VALUES ('00000000A',912222222);
SELECT * FROM telefonos;
```

EJERCICIO 3.2 DE CREACION de TABLAS.

- Creación de una base de datos de Aspirantes al centro IES Virgen de la Paz.

3.5 RESTRICCIONES: FOREIGN KEY.

SINTAXIS:

[**CONSTRAINT** símbolo] **FOREIGN KEY** [id] (nombre_índice, ...)

REFERENCES nombre_de_tabla (nombre_índice, ...)

[**ON DELETE** *reference_option*]

[**ON UPDATE** *reference_option*]

reference_option

{RESTRICT | CASCADE | SET NULL | NO ACTION | SET DEFAULT}

3.5 RESTRICCIONES: FOREIGN KEY.

SINTAXIS de ALTER TABLA:

ALTER TABLE nombre_de_tabla

ADD [**CONSTRAINT** símbolo] **FOREIGN KEY** [id] (nombre_índice, ...)

REFERENCES nombre_de_tabla (nombre_índice, ...)

[**ON DELETE** *reference_option*]

[**ON UPDATE** *reference_option*]

reference_option

{RESTRICT | CASCADE | SET NULL | NO ACTION | SET DEFAULT}

3.5 INTEGRIDAD REFERENCIAL

- Permite controlar que datos pueden almacenarse en la tabla secundaria de la clave externa (FK).

- Con una restricción de clave externa evita la perdida de vínculos entre dos tablas al eliminar o actualizar los datos.

3.5 INTEGRIDAD REFERENCIAL

- **NO ACTION**

El motor de la base de datos genera un error y se revierte la acción de eliminación o actualización de la fila de la tabla primaria.

- **RESTRICT**

Rechaza la opción de eliminación o actualización en la tabla padre.

Omiten la clausula ON DELETE y ON UPDATE.

Es similar a No ACTION.

3.5 INTEGRIDAD REFERENCIAL

- **CASCADE.**

Si se actualiza o elimina en la tabla primaria un dato, se actualiza o elimina en la tabla secundaria el dato relacionado.

ON DELETE CASCADE

ON UPDATE CASCADE

3.5 INTEGRIDAD REFERENCIAL

- **SET NULL.**

Si se actualiza o elimina en la tabla primaria un dato, se establecen en NULLO todos los valores que componen la clave externa.

- **SET DEFAULT.**

Si se actualiza o elimina en la tabla primaria un dato, se establecen los valores predeterminados que componen la clave externa.

3.5 EJEMPLO 03: CONSTRAINT FOREIGN KEY.

```
CREATE TABLE IF NOT EXISITS telefonos(
dni_person varchar(9) NOT NULL,
telefono int(9) NOT NULL,
PRIMARY KEY (dni_person, telefono),
CONSTRAINT fk_dniPerson FOREIGN KEY
(dni_person) REFERENCES personas(dni)
ON DELETE CASCADE ON UPDATE CASCADE);
```

```
UPDATE personas SET dni=00000001A WHERE
nombre = 'Juan';
```

3.5 ELIMINAR UNA TABLA REFERENCIADA.

Para ver las restricciones de una clave foránea:

SHOW TABLE STATUS FROM nombre_BD LIKE 'nombre_tabla';

Para borrar una tabla y limitar las restricciones del FOREIGN KEY se utiliza el comando:

```
mysql> SET FOREIGN_KEY_CHECKS = 0;
```

Para volver a las restricciones originales:

```
mysql> SET FOREIGN_KEY_CHECKS = 1;
```

3.5 BIBLIOGRAFIA

MANUAL OFICIAL MySQL 5.7.

FOREGIN KEY

- https://dev.mysql.com/doc/refman/5.7/en/create-table-foreign-keys.html

NOVEDADES

- https://dev.mysql.com/doc/refman/5.7/en/mysql-nutshell.html

3.4 BIBLIOGRAFIA.

FOREGIN KEY Constraints.

https://dev.mysql.com/doc/refman/5.7/en/create-table-foreign-keys.html

LENGUAJE DDL
3.6. Indices en las TABLAS

3.6 Índices

- Ayudan a obtener datos de las tablas de forma más rápida al realizar una consulta SELECT.

- Un índice puede contener una o más columnas.

CREATE INDEX "nombre_indice" ON "nombre_Tabla" (columna1, columna2);

3.6 Índices con PK.

- Cuando especifica una restricción de clave principal en una tabla, Motor de base de datos exige la unicidad de los datos mediante la creación automática de un **índice único** para las columnas de clave principal (PK).
- Permite el acceso rápido a los datos cuando se usa la clave principal en la consulta.

3.6 Índices con FK.

- La creación de una clave externa o foránea (FK) no se crea automáticamente un índice.

```
CREATE TABLE IF NOT EXISITS telefonos(
        dni_person varchar(9) NOT NULL,
        telefono int(9) NOT NULL,
PRIMARY KEY (dni_person, telefono),
INDEX idx_telefono(telefono),
CONSTRAINT fk_dniPerson FOREIGN KEY
(dni_person) REFERENCES personas(dni)
ON DELETE CASCADE ON UPDATE CASCADE);
```


3.6 MOSTRAR INDICES

```
USE prueba;
SHOW INDEX FROM telefonos;
```


3.6 BIBLIOGRAFIA

- MySQL 5.7 Reference Manual.

https://dev.mysql.com/doc/refman/5.7/en/optimization-indexes.html

2.6 CREACION DE TABLAS

- EJEMPLO 1: date + autoincrement

HTTPS://FORUMS.MYSQL.COM/READ.PHP?71,154336,154380