DDL

Línea de comandos:

Iniciar:
```
mysql -u user -p
```
Ejecutar un fichero bach.

```
mysql -u user -p < batch_file
```

Crear una copia de seguridad de una BD.

```
mysqldump --opt -u username -p database
> database_backup.sql

mysqldump   --opt   --all-databases   >
all_backup.sql
```

Mostrar Información de Tablas y BD

Seleccionar una BD:
```
USE database;
```
Listar las BD existentes:
```
SHOW DATABASES;
```
Mostrar las tablas de una BD:
```
SHOW TABLES;
```
Mostrar/Describir el formato/diseño de una tabla:
```
DESCRIBE table;
```

Crear BD y Tablas

Crear una BD:
```
CREATE DATABASE db_name;
```
Crear una tabla:
```
cREATE TABLE pet (
name VARCHAR(20) NOT NULL AUTO_INCREMENT
PRIMARY KEY,
sex CHAR(1),
birth DATE,
age INT(2),
raza ENUM('salchicha','pastor alemán')
);
```

Cambiar Sistema de almacenamiento

```
ALTER TABLE clientes
ENGINE MyISAM;
```

Insertar/Modificar/eliminar campos:

Añadir una columna:
```
ALTER TABLE clientes
ADD COLUMN direccion VARCHAR(40)
AFTER apellido2;
```
Modificar una columna:
```
ALTER TABLE clientes
CHANGE dni  nif VARCHAR(10);
```
Eliminar una columna:
```
ALTER TABLE clientes
DROP COLUMN dni;
```

Juegos de Caracteres Y colaciones:

Consultar juego de caracteres:
```
SHOW GLOBAL VARIABLES LIKE
'character_set_server';
```
Consultar colación:
```
SHOW GLOBAL VARIABLES LIKE
'collation_server';
```
Cambiar juego de caracteres en MySQL (Global):
```
SET GLOBAL
character_set_server='latin1';
```
Cambiar la colación en el MySQL (global):
```
SET GLOBAL
collation_server='latin1_spanish_ci';
```
Crear una BD con un juego de caracteres y colación determinados:
```
CREATE DATABASE hipermercado
CHARACTER SET latin1
COLLATE latin1_spanish_ci;
```
Modificar una BD con un juego de caracteres y colación determinados:
```
ALTER DATABASE hipermercado
CHARACTER SET latin1
COLLATE latin1_spanish_ci;
```

Borrar BD, Tablas y campos

Eliminar DB:
```
DROP DATABASE nombre_bd;
```
Eliminar TABLA:
```
DROP TABLA nombre_tabla;
```
Eliminar un campo:
```
ALTER TABLE tbl DROP COLUMN col;
```

Cambiar Nombre a Tabla y a Campo.

```
RENAME TABLE clientes  TO clientes2009;
```
```
ALTER TABLE clientes
CHANGE dni  nif VARCHAR(10);
```

Crear/Eliminar llave primaria y ajena.

//Crear Llave primaria

```
ALTER TABLE jugadores
ADD PRIMARY KEY (id_equipo);
```

```
//Crear Llave ajena
ALTER TABLE jugadores
ADD FOREIGN KEY (id_equipo)
REFERENCES equipo(id_equipo);
```

```
//Eliminar Llave primaria.
ALTER TABLE nombretabla
DROP PRIMARY KEY;
```

```
//Eliminar llave ajena.
ALTER TABLE nombretabla
DROP FOREIGN KEY nombre_fk;
```

DML
TRABAJAR CON DATOS

Cargar/Insertar datos

```
Cargar datos de un fichero TABULADO:

LOAD DATA LOCAL INFILE "fichero.txt"
INTO TABLE table_name;
(Use \n for NULL)
```

```
Insertar un registro:
INSERT INTO clientes
  VALUES ('Pío Pérez', 'Gran Plaza 12',
          '2002-08-31',NULL);
```

```
Reloading a new data set into existing table:

mysql> SET AUTOCOMMIT=1; # used for
quick recreation of table
mysql> DELETE FROM pet;
```

```
mysql> LOAD DATA LOCAL INFILE
"infile.txt" INTO TABLE table;
```

Actualizar datos.

```
UPDATE clientes
SET dni = '2603232'
WHERE nombre ="Pedro" AND ape1='Pérez';
```

Eliminar Datos

```
DELETE FROM clientes
where sexo='v';

//Borra todos los datos.
TRUNCATE TABLE clientes;
```

DML
CONSULTAS

Funciones

```
Valor máximo y mínimo:
SELECT MAX(edad) AS edad_maxima
FROM alumnos;

SELECT MIN(edad) AS edad_mInima
FROM alumnos;
```

```
Contar columnas:

SELECT COUNT(*)
FROM clientes;
```

```
//Media: media de los porcentajes de aquellas lenguas cuyo porcentaje supere el 50%.
SELECT AVG(Porcentaje) AS
mediaporcentaje
FROM lenguas
WHERE (Porcentaje>50.0);
```

```
Suma:
SELECT SUM(Superficie) AS
superficietotal
FROM paises;
```

Union

```
//Crea una unión con las filas de las dos tablas (han de coincidir las columnas).

TABLE jugadores_nuevos
UNION ALL
TABLE jugadores_antiguos
ORDER BY nombre_jugador;
```

```
//Unión de select:
SELECT nombre_alumn
FROM curso0708
WHERE ciclo='ESI'
UNION
SELECT nombre_alumn
FROM curso0809
WHERE ciclo='ESI';
```

Múltiples tablas

```
Producto cartesiano
SELECT nombre_equipo, COUNT(id_jugador)
FROM jugadores, equipos
WHERE
jugadores.id_equipo=equipos.id_equipo;
```

```
INNER JOIN
//Similar al producto cartesiano, pero más rápido si las columnas de emparejamiento están indexadas.

SELECT nombre_equipo, COUNT(id_jugador)
FROM jugadores INNER JOIN equipos
ON jugadores.id_equipo=equipos.id_equipo;
```

LEFT JOIN

```
//Aparecen todos los registros de la tabla izquierda (todos los jugadores), aunque no se correspondan con ningún registro de la derecha (aunque no juegen en ningún equipo).

SELECT  *
FROM jugadores LEFT JOIN equipos
ON jugadores.id_equipo = equipos.id_equipo;
```

RIGHT JOIN

```
//Aparecen todos los registros de la tabla derecha (todos los equipos), aunque no se correspondan con ningún registro de la izquierda (aunque no tengan ningún jugador).

SELECT  *
FROM empleados RIGHT JOIN oficinas
ON empleados.oficina = oficinas.oficina;
```

Seleccionar Registros distintos (DISTINCT)

SELECT (general):

```
SELECT dni,nombre
FROM clientes
WHERE dni='22234432' OR 'dni=25343234';
```

```
SELECT * FROM pedidos;
```

Listado de todos los nombres de clientes distintos:

```
SELECT DISTINCT nombre FROM clientes;
```

Subconsultas

```
SELECT nombre_equipo
FROM equipos
WHERE (id_equipo=
          (SELECT DISTINCT id_equipo
           FROM jugadores
           WHERE numero_goles>0)
);
```

Subconsultas con ANY, IN y SOME

```
//ANY o IN (ALIAS): es true si la condición se cumple con cualquiera de los valores de la subconsulta.
SELECT s1
FROM t1
WHERE s1 > ANY (SELECT s1 FROM t2);


//ALL: es true si la condición se cumple con todos los valores devueltos por la subconsulta.
SELECT s1
FROM t1
WHERE s1 > ALL (SELECT s1 FROM t2);
```

Subconsultas con EXISTS y NOT EXISTS

```
//Qué país tiene una o más ciudades.
SELECT DISTINCT nombre
FROM paises
WHERE EXISTS
  ( SELECT *
    FROM ciudades
    WHERE
ciudades.Cod_pais=paises.Cod_pais);

//Qué país NO tiene una o más ciudades.
SELECT DISTINCT nombre
FROM paises
WHERE NOT EXISTS
  ( SELECT *
    FROM ciudades
    WHERE
ciudades.Cod_pais=paises.Cod_pais);
```

Subconsultas en FROM

```
//Es obligatorio el AS dentro de la subconsulta.
SELECT AVG(porcentmayor)
FROM (SELECT Porcentaje AS porcentmayor
      FROM lenguas
      WHERE Porcentaje>50.0);
```

Ordenar Registros

Ordenar:

```
SELECT nombre, edad FROM amigos ORDER BY edad DESC;
```

Cálculos con fechas:

```
SELECT CURRENT_DATE,
(YEAR(CURRENT_DATE)-YEAR(fechanac)) AS time_diff
FROM amigos;
```

Selección con caracteres comodín:

```
SELECT *
FROM clientes
WHERE nombre LIKE "Jua%";
```

Agrupamientos (group by)

```
SELECT id_equipo, COUNT(*)
FROM jugadores
GROUP BY id_equipo;
```

Condiciones con HAVING

```
SELECT id_proveedor, MAX(precio_compra)
FROM compras
GROUP BY id_proveedor
HAVING MAX(precio_compra)>100;
```

TIPOS DE DATOS

TINYINT	1 byte
SMALLINT	2 bytes
MEDIUMINT	3 bytes
INT	4 bytes
INTEGER	4 bytes
BIGINT	8 bytes
FLOAT(X)	4 ú 8 bytes
FLOAT	4 bytes
DOUBLE	8 bytes
DOUBLE PRECISION	8 bytes
REAL	8 bytes
DECIMAL(M,D)	M+2 bytes sí D > 0, M+1 bytes sí D = 0
NUMERIC(M,D)	M+2 bytes if D > 0, M+1 bytes if D = 0

Date	FECHA (3 bytes)
DateTime	FECHA Y HORA (8bytes)
TimeStamp	FECHA Y HORA (4bytes)
Time	HORA (3bytes)
Year	AÑO (1901-2155) (1byte)

Char(n)	Longitud fija. De 0 a 255 caract. (n bytes)
VarChar(n)	Longitud variable. De 0 a 255 caracteres. (n+1 bytes)
TinyText	Máx 255 char.
TinyBlob	Máx. 255 bytes (binarios).
Text	Máx. 65535 char.
Blob	Máx. 65535 bytes (bin).
MediumText	Máx. 16 Mill. char.
MediumBlob	Máx 16 Mill. bytes (bin)
LongText	Máx 4294 mill. Char.
LongBlob	Máx 4294 mill. Bytes (bin).

Enum	Hasta 65535 valores. Fruta ENUM ('limón','naranja');
Set	puede contener ninguno, uno ó varios valores de una lista. (Máx 64 valores).