# TEORÍA GENERAL

Una sentencia SQL se compone de Comandos + Cláusulas + Operadores + Función de Agregado

Para que una sentencia sql esté completa se necesitan al menos los dos primeros elementos

## Tipos de objeto

TABLE

CONSTRAINT

DATABASE

TRIGGER

TRANSACTION

COLUMN

## Comandos

USE NombreBaseDatos: se utiliza para cambiar la base de datos que se está utilizando

BEGIN

DDL: Data definition language:

CREATE TipoObjeto NombreObjeto -- Utilizado para crear nuevas tablas, campos e índices.

ALTER TipoObjeto NombreObjeto. -- Utilizado para modificar las tablas agregando campos o cambiando la definición de los campos.

DROP TipoObjeto NombreObjeto --Empleado para eliminar tablas e índices

TRUNCATE

DML: Data manipulation language: consultas de selección y de acción

SELECT Utilizado para consultar registros de la base de datos que satisfagan un criterio determinado

Utilizado para cargar lotes de datos en la base de datos en una única operación.

UPDATE Utilizado para modificar los valores de los campos y registros especificados

DELETE Utilizado para eliminar registros de una tabla

TRUNCATE

DCL: Data Control Language: proporcionan seguridad a los datos,

GRANT

REVOKE

TLC: Transaction Control Language: gestión en los cambios

COMMIT

ROLLBACK

SAVEPOINT

## Cláusulas

FROM

WHERE

GROUP BY

HAVING

ORDER BY

JOIN

LEFT JOIN

RIGHT JOIN

UNION –Operación de conjuntos, en las operaciones de conjuntos no pueden existir datos repetidos dentro de un conjunto, por ende, cuando se usa unión sólo se toman en cuenta los registros únicos y no los repetidos del conjunto

UNION ALL – Considera todos los registros, no importa que tengan valores repetidos.

ADD

## Orden de escritura

SELECT

FROM

WHERE

GROUP BY

HAVING

ORDER BY

## Orden lógico

FROM

ON

OUTER

WHERE

GROUP BY

CUBE | ROLLUP

HAVING: se usa para señalar criterios en reemplazo del WHERE en sentencias de agrupación y totales, el criterio implica funciones de agregado

SELECT

DISTINCT

ORDER BY… ASC/DESC: orden  según el nombre del campo resultante en la consulta,no el nombre del campo de la tabla que puede variar

TOP

## Operadores lógicos

AND Es el “y” lógico. Evalúa dos condiciones y devuelve un valor de verdad sólo si ambas son ciertas.

OR Es el “o” lógico. Evalúa dos condiciones y devuelve un valor de verdad si alguna de las dos es cierta.

NOT Negación lógica. Devuelve el valor contrario de la expresión.

## Operadores de comparación

< Menor que

> Mayor que

<> Distinto de

<= Menor o igual que

>= Mayor o igual que

BETWEEN Intervalo

LIKE Comparación, se usa con caracteres comodín.

In Especificar registros en un campo concreto

## Funciones de agregado

Las funciones de agregado se usan dentro de una cláusula SELECT en grupos de registros para devolver un único valor que se aplica a un grupo de registros.

AVG() Utilizada para calcular el promedio de los valores de un campo determinado

COUNT() Utilizada para devolver el número de registros de la selección, no cuenta celdas vacias

SUM() Utilizada para devolver la suma de todos los valores de un campo determinado

MAX() Utilizada para devolver el valor más alto de un campo especificado

MIN() Utilizada para devolver el valor más bajo de un campo especificado

## Consultas de cálculo

Funciones frecuentes

Now()

Datediff()

Date\_format()

Concat()

Round

## Campos calculados

NombreCampo AS ExpresiónDelCampo

## Restricciones

PRIMARY KEY

FOREIGN KEY REFERENCES NombreTabla (NombreCampo)

ON UPDATE {CASCADE, SET NULL, NO ACTION}

ON DELETE {CASCADE, SET NULL, NO ACTION]

CHECK ExpresiónDeRestricción

WITH CHECK ExpresiónDeRestricción

NOT NULL

UNIQUE

IDENTITY (NúmeroInicial,Step) – Los campos identificados como IDENTITY son afectados por el siguiente comportamiento: antes de realizar una inserción, lo primero que realiza el manejador de BBDD es incrementar el campo identity, en caso que la transacción falle el campo ya habrá sido incrementado y no podrá ser reutilizado, esto se hace para evitar conflictos con la concurrencia de usuarios y procesos, que pueden estar insertando valores al mismo tiempo y por ello una vez incrementado un campo IDENTITY, éste no puede ser reversado.

DEFAULT (ValorPorDefecto)

## Alias

AS : se usa en consulta select para cambiar nombre al campo

## Character set

Se refiere al juego de carácteres, el abecedario, números y caracteres acentuados, etc y su código identificativo. Si se escoge mal el character set cuando un usuario ingrese información esta puede ser TRUNCADA. Esto aplica a campos tipo string.

utf8\_spanish\_ci: español moderno, incluye Ñ como letra independiente entre la N y O

uf8\_spanish2\_ci: español tradicional, además de la Ñ incluye la CH y la LL como carácteres.

Se puede cambiar para toda las tablas de la DB en pestaña OPERACIONES (phpmyadmin) para que al ingresar un nuevo campo escoja el character set seleccionado por defecto.

Se debe marcar "Cambiar la intercalación de todas las columnas de las tablas" para que haga efectivo el cambio a todas las columnas de todas las tablas.

## Collation

Tiene que ver cómo el RDBMS reconoce los caracteres y esto es relevante para efectos de ordenar, si el RDBMS tiene como equivalentes los caracteres Ñ y N al momento de ordenar, los tomará como equivalentes. También al momento de hacer consultas es relevante cómo son reconocidos los carácteres.

## Tipos de datos

TinyInt = -128 a 123 Unsigned = 0 a 255

SmallInt = -32.768 a 32.767 Unsigned = 0 a 65535

Int = -2.147.483.648 a 2.147.483.647 Unsigned = 0 a 4.294.967.296

BigInt = -9.223.372.036.854.775.808 a 9.223.372.036.854.775.807

Varchar(longitud) – longitud obligatoria

Decimal / Numeric(p,s): To store numbers that have fixed precision and scale, you use the DECIMAL data type.

p is the precision which is the maximum total number of decimal digits that will be stored, both to the left and to the right of the decimal point. The precision has a range from 1 to 38. The default precision is 38.

s is the scale which is the number of decimal digits that will be stored to the right of the decimal point. The scale has a range from 0 to p (precision). The scale can be specified only if the precision is specified. By default, the scale is zero.

The maximum storage sizes vary, depending on the precision as illustrated in the following table:

Precision Storage bytes

1 – 9 5

10-19 9

20-28 13

29-38 17

The NUMERIC and DECIMAL are synonyms, therefore, you can use them interchangeably.

## Restricciones

UNSIGNED

NULL (Por defecto, no es necesario colocarla)

NOT NULL

PRIMARY KEY

AUTO\_INCREMENT / INDENTITY

UNIQUE INDEX

# QUERIES

## Configuraciones de campos por defecto

LONGITUD: Varchar(255)

COLLATION: utf8\_spanish\_ci

CHARACTER SET: utf8\_spanish\_ci

UNSIGNED: Por defecto todos los campos numéricos deben ser UNSIGNED, la excepción es que se permitan números negativos. Aplica sólo para campos de tipo número.

NOT NULL: Por defecto todos los campos deben ser NOT NULL, la excepción es que sean nullable

## Biblioteca de campos

**`**NameTableId**`** BIGINT NOT NULL AUTO\_INCREMENT

**`**first\_name` VARCHAR(50)NOT NULL

**`**last\_name` VARCHAR(50)NULL

`address\_name` VARCHAR(100)NOT NULL

**`**address\_num**`** INT (7) UNSIGNED NOT NULL

`address\_comuna` VARCHAR(20)NOT NULL

`email` VARCHAR(255) NULL

`active` BOOLEAN NOT NULL

`phone` INT (15) UNSIGNED NOT NULL

`prefix\_phone` VARCHAR (5) NOT NULL

`price` DECIMAL(10,2) UNSIGNED NOT NULL

`status` VARCHAR(255) NOT NULL

`comment` VARCHAR(255) NULL

`latitude` DOUBLE NULL

`longitude` DOUBLE NULL

`date` DATE NOT NULL

`nac\_id` INT

`digito\_verificador\_id` CHAR(1)

Comillas especiales backticks

`` este tipo de comillas, en el teclado de RE se digitan con la siguiente combinación de teclas: CTRL + ALT + 2 veces [tecla “}” o “ç”]

El nombre de la BBDD es case-sensitive, puede dar error al referenciar el nombre de una bbdd

## Modificar columnas

Con GUI (recomendado)

* Con la GUI puedes modificar el atributo que desees sin necesidad de incluir al resto como sí hay que hacerlo de manera manual.
* Debido a que este tipo de operaciones realmente son delicadas, modificar una BBDD siempre va a ser un proceso crítico, lo mejor es hacer las operaciones con GUI que elabora el SQL y permite omitir errores.
* Hay una opción en el GUI phpmyadmin para editar y cambiar nombres, atributos de todos los campos de una tabla al mismo tiempo.

Con SQL

* Hacer una query de este tipo para obtener los campos y los atributos de toda la BBDD:

Select COLUMN\_NAME, IS\_NULLABLE, COLUMN\_TYPE, EXTRA, TABLE\_NAME, COLLATION\_NAME, CHARACTER\_SET\_NAME

From INFORMATION\_SCHEMA.COLUMNS

Where TABLE\_SCHEMA Like 'savinglc\_querier'

ORDER BY `COLUMNS`.`COLUMN\_TYPE` ASC

* Dividir pantalla, con copy-paste copiar el nombre de los campos y los atributos necesarios, hay que incluir todos los atributos que deseamos mantener en la tabla para que ninguno sea eliminado.

ALTER TABLE savinglc\_Querier.CLIENTS

MODIFY `Clt\_AddressComuna` varchar(255) NOT NULL,

MODIFY `Clt\_AddressName` varchar(255) NOT NULL,

MODIFY `Clt\_AddressNum` INT(10) UNSIGNED NOT NULL,

MODIFY `Clt\_IdNac` varchar(15) NOT NULL,

MODIFY `Clt\_LegalName` varchar(255) NOT NULL;

## Insertar Valores

INSERT INTO table\_name (column1, column2, column3, ...)

VALUES (value1, value2, value3, ...);

INSERT INTO table\_name

VALUES (value1, value2, value3, ...);

INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)

VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway');

## Crear tabla

CREATE TABLE `Savinglc\_Integrer`.`CONDUCTORES` (

 `ID\_CONDUCTOR` INT NOT NULL AUTO\_INCREMENT,

 `RUT` VARCHAR(45) NOT NULL,

 `NOMBRE` VARCHAR(45) NULL,

 `¿ACTIVO?` TINYINT NULL,

 `USER\_ML` VARCHAR(45) NULL,

 `PASS\_ML` VARCHAR(45) NULL,

 `EMAIL` VARCHAR(45) NULL,

 `PASS\_EMAIL` VARCHAR(45) NULL,

 `CELULAR` VARCHAR(15) NULL,

 `DIRECCION\_RESIDENCIA` VARCHAR(45) NULL,

PRIMARY KEY (`ID\_CONDUCTOR`, `RUT`),

UNIQUE INDEX `ID\_CONDUCTOR\_UNIQUE` (`ID\_CONDUCTOR` ASC) VISIBLE,

UNIQUE INDEX `RUT\_UNIQUE` (`RUT` ASC) VISIBLE)

COMMENT = 'Conductores internos y externos';

## Crear columnas

ALTER TABLE table\_name

  ADD new\_column\_name column\_type

    [ FIRST | AFTER column\_name ];

ALTER TABLE CLIENTS

  ADD Clt\_AddressNum INT(10) UNSIGNED

    AFTER Clt\_AddressName;

ALTER TABLE CLIENTS

ADD Clt\_AddressNum INT(10) UNSIGNED AFTER Clt\_AddressName,

     ADD Clt\_AddressComuna VARCHAR(50) AFTER Clt\_AddressNum

## Eliminar columna

ALTER TABLE CLIENTS

   DROP Clt\_AddressNum

## Enlistar todos los campos de la BBDD

INFORMATION\_SCHEMA es una base de datos que contiene información relativa a todas las base de datos almacenadas en el servidor.

Select \*

From INFORMATION\_SCHEMA.COLUMNS

Where TABLE\_SCHEMA Like 'Database\_Name'

ACCESS

Tomar BBDD y modificar para que sean aleatorios:  
-- Truncar una tabla en access  
Delete \* from (Your Table Name);

-- Resetea el autoincrement

Alter Table (Your Table Name) Alter Column (Your Column Id) Counter(1,1);

-- Generar datos aleatorios (procura que si los vas a insertar en access generes .csv o algo que no sea SQL  
[Generatedata.com](http://generatedata.com/)

# MICROSOFT SQL SERVER

## Bases de datos del sistema

Master: se guarda la configuración de la instancia de SQL server, siempre que se hagan cambios en la configuración lo mejor es tomar respaldo de esta BBDD.

Model: plantilla para la creación de nuevas bases de datos

Msdb: la utiliza el agente de SQL Server. Almacena información de tareas programadas, alertas, correos electrónicos, programar respaldos y otros servicios.

Tempdb: BBDD global para guardar información temporal, puedes crear objetos acá, pero la información no es persistente. Su configuración puede impactar mucho el rendimiento del sistema, debe residir en discos rápidos y debe si lo necesita.

# MARIADB **-** 10.3.30-MariaDB - MariaDB Server

## Conectarse servidor

1. Crear una base de datos, se crea desde CPANEL, no desde phpMyAdmin
2. Debe adicionarse permiso a la máquina local en el CPANEL para que se pueda conectar a la BBDD . IP local 186.11.85.232:
3. Crear un usuario de MySQL con privilegios para una determinada BBDD, una vez creado se le debe añadir privilegios a determinada BBDD
4. En MySQL WorkBench crear la conexión

HostName: savingl.cl / 192.154.227.209

Port: 3306

Database Name: savinglc\_NombreDeBbdd

Username: MySQL user creado previamente

**Error MySQL failed connection**

Puede suceder que falla la conexión al intentar conectarse con Workbench, hay que revisar:

* Usuario bien tipeado
* Clave bien tipeada
* Nombre del servidor bien escrito
* Usuario tiene privilegios sobre la BBDD (ver en cpanel si tiene privilegios)
* IP de la máquina local con permiso de acceso remoto en servidor: En  MySQL remoto® está la opción de listar las IP con permiso para acceder a las bases de datos. Para saber que ips deben tener permiso revisamos en cmd: ipconfig/all y buscamos IPv4. Además en el mensaje de error del workbench indica: Nombredeusuario@DirecciónIpQueIntentaConectar, ésta dirección es la que debe tener permiso de acceso.

**Reverse Engineering**

Se usa para traer un esquema físico y lógico al workbench, se le pueden aplicar cambios y enviarlos al servidor (forward engineering).