

# 6

## SQL - DDL

Structured Query Language

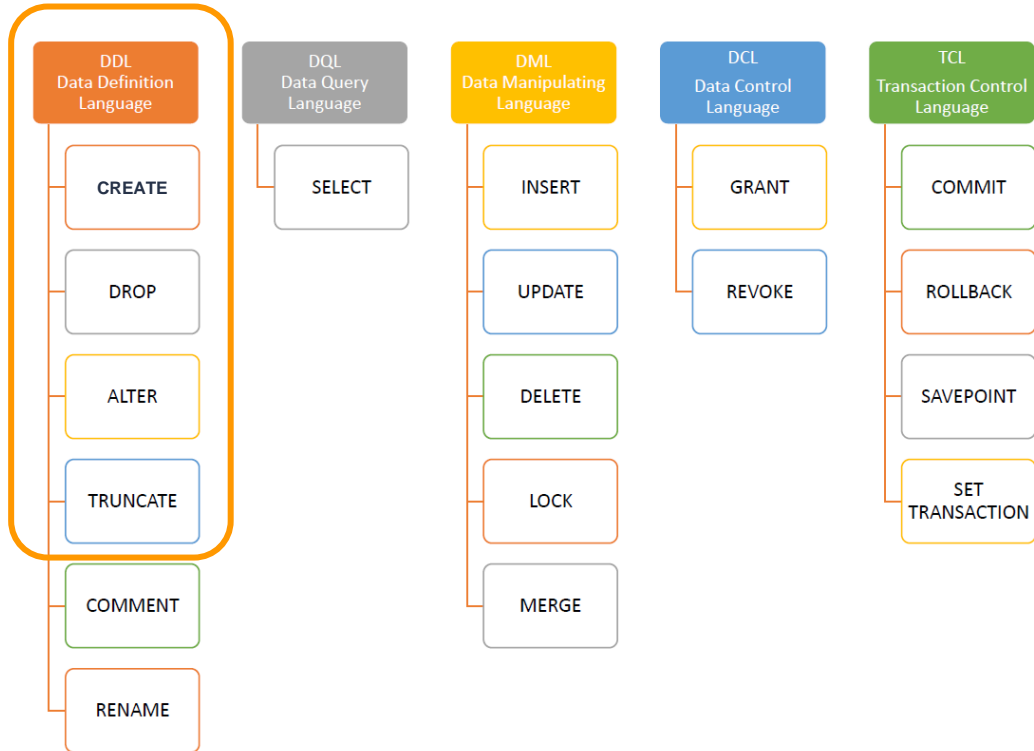
# DDL

Data Definition Language





# SQL





# SQL

- SQL – *Structured Query Language* (Lenguaje Estructurado de Consulta)
- Fue desarrollado inicialmente por IBM (Donald Chamberlin y Raymond F. Boyce) luego de estudiar acerca del modelo relacional propuesto por Edgar F. Codd a inicios de los '70s
- Primero se llamó SEQUEL (StructuredEnglish QueryLanguage) y su objetivo era extraer y manejar datos almacenados en un sistema “cuasi” relacional propietario de IBM.
- Luego se lo renombró a SQL debido a que el nombre SEQUEL ya estaba patentado por una empresa de UK



## SQL

- Sin embargo, no fue hasta varios años después, que el lenguaje SQL estuvo disponible públicamente. En 1979, una empresa llamada Relational Software, que luego se convirtió en Oracle, lanzó comercialmente su propia versión de SQL, llamada Oracle V2
- Desde entonces, el Instituto Nacional Estadounidense de Estándares (ANSI) y la Organización Internacional para la Estandarización han considerado que SQL es el lenguaje estándar en la comunicación de bases de datos relacionales. Si bien los principales proveedores de SQL modifican el lenguaje según sus deseos, la mayoría basa sus programas SQL en la versión aprobada por ANSI.



# CREATE...

Creación de estructuras...





# SQL

## ■ CREACIÓN DE BASES DE DATOS

```
CREATE DATABASE NombreBaseDatos
```

*\*NombreBaseDatos: No debe existir una BD con ese nombre creada anteriormente.*



# SQL

## ■ CREACIÓN DE TABLAS

De acuerdo a lo visto, la estructura de almacenamiento de los datos del modelo relacional son las tablas, se crean dentro de una BD.

```
CREATE TABLE NombreTabla ( definiciones de columnas...  
                             restricciones de tabla... );
```

*\*NombreTabla: No debe existir una tabla con ese nombre creada anteriormente en esa BD.*





# SQL

## DEFINICIÓN DE COLUMNAS

Definamos las columnas que formarán parte de las tablas.

```
NombreColumna { tipo dato | dominio } [def defecto] [restricción]
```

*\*NombreColumna: No debe existir una columna con ese nombre creada anteriormente en esa Tabla.*



# SQL

## EJEMPLO TIPOS DE DATOS MÁS UTILIZADOS EN SQL

	DESCRIPCIÓN	EJEMPLO
<b>NUMERIC(8)</b>	Número entero largo 8	35478221
<b>NUMERIC(4,2)</b>	Número largo 4 con 2 decimales incluidos	12,94
<b>CHAR(4)</b>	Alfanumérico largo 4 fijo – (máx 255)	A3rK
<b>VARCHAR(50)</b>	Alfanumérico largo 50 variable	'Hola... ayer.'
<b>DATE</b>	Fecha	'31/03/2001'
<b>DATETIME</b>	Fecha y hora	'31/03/2001 11:45'
<b>BIT</b>	Símil booleano (0 ó 1)	1



# SQL

## ■ CREACIÓN DE TABLAS – sin restricciones.

```
CREATE TABLE Empleados ( cedula NUMERIC (8),  
                           nombre VARCHAR (30),  
                           fecha_nacimiento DATE,  
                           sueldo NUMERIC(12,2),  
                           tipo CHAR(1) );
```



## RESTRICCIONES – *CONSTRAINTS*

Las restricciones más utilizadas son:

	DESCRIPCIÓN
<b>NULL - NOT NULL</b>	Especifica que la columna puede o no (según corresponda) tener un valor nulo.
<b>PRIMARY KEY</b>	Identifica de manera única cada fila de una tabla
<b>FOREIGN KEY</b>	Establece una relación entre una columna de la tabla y una columna de otra tabla. Clausula de integridad referencial
<b>UNIQUE</b>	La columna no puede tener valores repetidos. Es una clave alternativa
<b>CHECK</b>	La columna debe cumplir una condicion establecida
<b>DEFAULT</b>	Especifica el valor por defecto que va a tener la celda



# SQL

## CREACIÓN DE TABLAS – con restricciones.


```
CREATE TABLE Empleados ( cedula NUMERIC (8),  
                           nombre VARCHAR (30) NOT NULL,  
                           fecha_nacimiento DATE,  
                           sueldo NUMERIC(12,2),  
                           tipo CHAR(1) ,  
                           CONSTRAINT PK_Empleado PRIMARY KEY (cedula),  
                           CONSTRAINT CK_Sueldo CHECK (sueldo > 5000) );
```



# SQL

```
CREATE TABLE Ciudades ( nombre VARCHAR (30),  
                           fecha_fundacion DATE,  
                           poblacion NUMERIC(10),  
                           PRIMARY KEY (nombre) );
```

```
CREATE TABLE Empleados ( cedula NUMERIC (8),  
                           nombre VARCHAR (30) NOT NULL,  
                           fecha_nacimiento DATE,  
                           sueldo NUMERIC (12,2),  
                           tipo CHAR (1) ,  
                           ciudad VARCHAR (30),  
                           CONSTRAINT PK_Empleado PRIMARY KEY (cedula),  
                           CONSTRAINT CK_Sueldo CHECK (sueldo > 5000),  
                           CONSTRAINT FK_Ciudad FOREIGN KEY (ciudad) REFERENCES  
                               Ciudades(nombre) );
```





# SQL

## AUTOINCREMENTAL

Un campo numérico puede tener un atributo extra "**identity**".

Identity(50,3) - Genera valores secuenciales que se inician en 50 y se incrementan en 3.

Se utiliza generalmente en campos correspondientes a códigos de identificación para generar valores únicos para cada nuevo registro que se inserta.

Sólo puede haber un campo "identity" por tabla.

Cuando un campo tiene el atributo "identity" no se puede ingresar valor para él, se inserta automáticamente tomando el último valor como referencia..

```
codigo NUMERIC(10) PRIMARY KEY IDENTITY(1,1);
```



# SQL

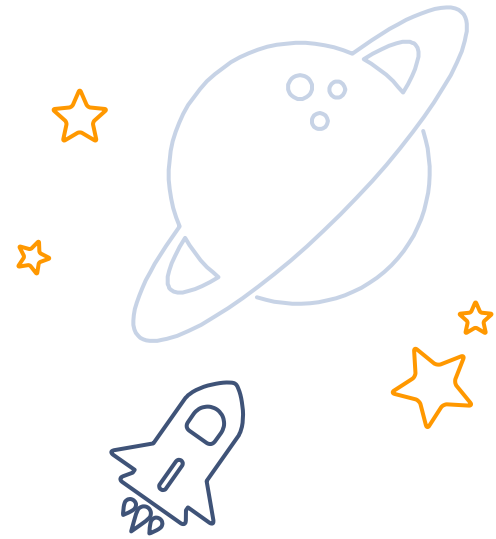
## CREACIÓN DE TABLAS – con IDENTITY.

```
CREATE TABLE Empleados ( codigo NUMERIC (8) IDENTITY(1,1),  
                           nombre VARCHAR (30) NOT NULL,  
                           fecha_nacimiento DATE,  
                           sueldo NUMERIC(12,2),  
                           tipo CHAR(1) ,  
                           CONSTRAINT PK_Empleado PRIMARY KEY (codigo),  
                           CONSTRAINT CK_Sueldo CHECK (sueldo > 5000) );
```



# ALTER...

Modificación de estructuras...





# SQL

## ■ AÑADIR COLUMNA A UNA TABLA

```
ALTER TABLE NombreTabla ADD NombreColumna TIPODATO {restricción}
```

*\*Modifica la tabla agregando una nueva columna a la estructura.*



# SQL

## ELIMINAR COLUMNA DE UNA TABLA

```
ALTER TABLE NombreTabla DROP COLUMN NombreColumna
```

*\*Modifica la tabla eliminando una de sus columnas y perdiendo la información contenida en dicha columna.*



# SQL

## MODIFICAR TAMAÑO DE COLUMNA

```
ALTER TABLE NombreTabla ALTER COLUMN NombreColumna TIPODATO
```

```
ALTER TABLE Empleado ALTER COLUMN nombre VARCHAR(50)
```



# SQL

## HACER *NOT NULL* O *NULL* UNA COLUMNA

```
ALTER TABLE NombreTabla ALTER COLUMN NombreColumna TIPODATO NOT NULL
```

```
ALTER TABLE Empleados ALTER COLUMN nombre VARCHAR(50) NOT NULL
```

*\*Convierte la columna Nombre de la tabla Empleados a Not Null*



# SQL

## ■ AÑADIR RESTICCIONES A UNA TABLA EXISTENTE

```
ALTER TABLE NombreTabla ADD CONSTRAINT Nombre { LA RESTRICCIÓN AQUÍ }
```

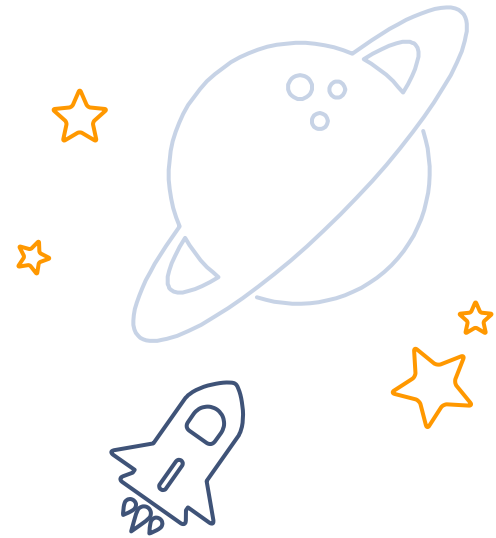
```
ALTER TABLE Empleados ADD CONSTRAINT CK_Sueldo CHECK (sueldo < 1000000)
```

\*Agrega la restricción Check tal que sueldo adminta valores menores a 1000000.



# DROP...

Eliminar de estructuras...





# SQL

## ELIMINAR DE TABLAS

```
DROP TABLE NombreTabla
```

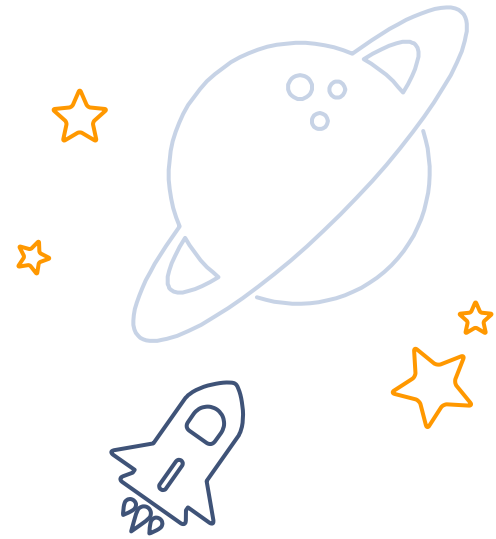
*\*Elimina la tabla de nuestra base de datos.*

*\*\* No podrá eliminarse la tabla cuya clave sea clave foránea en otra tabla de la base de datos.*



# TRUNCATE...

Eliminar de estructuras...





# SQL

## ■ BORRAR DATOS DE UNA TABLA

```
TRUNCATE TABLE NombreTabla
```

# CONSTRAINT...

Ejemplo de constraints...





# SQL

```
CREATE TABLE Empleados( cedula NUMERIC (8),
                          nombre VARCHAR (30) NOT NULL,
                          email VARCHAR (100),
                          fecha_ingreso DATE,
                          sueldo NUMERIC (12,2),
                          tipo VARCHAR (10),
                          ciudad VARCHAR (30) DEFAULT 'Montevideo',
                          CONSTRAINT PK_Empleados PRIMARY KEY (cedula),
                          CONSTRAINT FK_Empleados_Ciudad FOREIGN KEY (ciudad) REFERENCES Ciudades(nombre),
                          CONSTRAINT UC_Empleados_Email UNIQUE (email),
                          CONSTRAINT CK_Empleados_Sueldo CHECK (sueldo > 5000),
                          CONSTRAINT CK_Empleados_Tipo CHECK (tipo IN('Jornalero','Zafra1')),
                          CONSTRAINT CK_Empleados_Fecha CHECK (fecha_ingreso <= GETDATE()))
```



**Mauro Arrieta**

mauro.arrieta@fi365.ort.edu.uy

