



Introducción a Mysql

Comenzamos con la práctica



Clase 2

Temas

Puesta en común:

- Ejercicio práctico clase anterior

Teórico práctico:

- Diferencias entre DDL, DML y DCL
- Sintaxis
- Crear una base de datos
- Crear las tablas definidas en el problema planteado la semana anterior
- Generar un diagrama entidad relación
- Insertar datos
- Consultar datos
- Actualizar datos
- Eliminar datos
- Alter table add column
- Drop column



Empezamos

DDL, DML y DCL



Lenguaje de Definición de Datos (DDL)

Es un lenguaje de programación para definir estructuras de datos, proporcionado por los sistemas gestores de bases de datos, en este caso MySQL.

En inglés **Data Definition Language**, de ahí sus siglas **DDL**.

Este lenguaje permite definir las estructuras que almacenarán los datos así como los procedimientos o funciones que permitan consultarlos.



Lenguaje de Definición de Datos (DDL)

Para definir la estructura disponemos de tres sentencias:

- **CREATE**, se usa para crear una base de datos, tabla, vistas, etc.
- **ALTER**, se utiliza para modificar la estructura, por ejemplo añadir o borrar columnas de una tabla.
- **DROP**, con esta sentencia, podemos eliminar los objetos de la estructura, por ejemplo un índice o una secuencia.



Lenguaje de Manipulación de Datos (DML)

También es un lenguaje proporcionado por los sistemas gestores de bases de datos. En inglés, **Data Manipulation Language (DML)**.

Utilizando instrucciones de SQL, se le permite a los usuarios introducir datos para posteriormente realizar tareas de consultas o realizar modificaciones en la base.



Lenguaje de Manipulación de Datos (DML)

Los elementos que se utilizan para manipular los datos, son los siguientes:

- **SELECT**, esta sentencia se utiliza para realizar consultas sobre los datos.
- **INSERT**, con esta instrucción podemos insertar los valores en una base de datos.
- **UPDATE**, sirve para modificar los valores de uno o varios registros.
- **DELETE**, se utiliza para eliminar las filas de una tabla



Lenguaje de Control de Datos (DCL)

Estos comandos permiten al Administrador del sistema gestor de base de datos, controlar el acceso a los objetos, es decir, podemos otorgar o denegar permisos a uno o más roles para realizar determinadas tareas.

Sus siglas son **DCL** por su nombre en inglés, **Data Control Language**.




Lenguaje de Control de Datos (DCL)

Los comandos para controlar los permisos son los siguientes:

- **GRANT**, permite otorgar permisos.
- **REVOKE**, elimina los permisos que previamente se han concedido.



Empezamos con el código



id_producto	detalle_producto	id_categoria
1	Notebook	1
2	Pelota	2
3	Lavandina	3

id_categoria	detalle_categoria
1	Tecnologia
2	Deporte
3	Limpieza
4	Pesca



Crear una base de datos

CREATE DATABASE **mi_negocio**;

	#	Time	Action	Message	Duration / Fetch
✓	1	01:12:18	CREATE DATABASE mi_negocio	1 row(s) affected	0,00062 sec



Usar una base de datos

USE **mi_negocio**;

#	Time	Action	Message	Duration / Fetch
✓ 1	01:13:08	USE mi_negocio	0 row(s) affected	0.00030 sec

Creo una tabla empezando por las que no tienen clave foránea

```
CREATE TABLE categoria(  
id_categoria int not null auto_increment,  
detalle_categoria varchar(50) not null,  
PRIMARY KEY (id_categoria)  
)
```

- Creo la tabla con sus columnas.
- Creo un identificador único y lo hago auto_increment (es un número que se va a cargar solo aumentando a medida que se insertan más registros)
- Y por último con el comando primary key hago que id_categoria sea la clave primaria de la tabla.

#	Time	Action	Message	Duration / Fetch
✓ 1	01:24:50	CREATE TABLE categoria(id_categoria int not null auto_incre...	0 row(s) affected	0,269 sec

Creo las demás tablas y hago la referencias con sus claves foráneas

```
CREATE TABLE producto(  
  id_producto int not null auto_increment,  
  detalle_producto varchar(50) not null,  
  id_categoria int,  
  PRIMARY KEY(id_producto),  
  FOREIGN KEY (id_categoria) REFERENCES categoria(id_categoria)  
)
```

- Creo la tabla con sus columnas.
- Creo un identificador único y lo hago auto_increment (es un número que se va a cargar solo aumentando a medida que se insertan más registros)
- Con el comando primary key hago que id_producto sea la clave primaria de la tabla.
- Y por último con el comando FOREIGN KEY conecto las dos tablas y hago que id_categoria en la tabla producto sea la clave foránea y que haga referencia a la tabla categoría al campo id_categoria

#	Time	Action	Message	Duration / Fetch
1	01:25:40	CREATE TABLE producto(id_producto int not null auto_increm...	0 row(s) affected	0,269 sec



Ya creamos nuestra primer base de datos

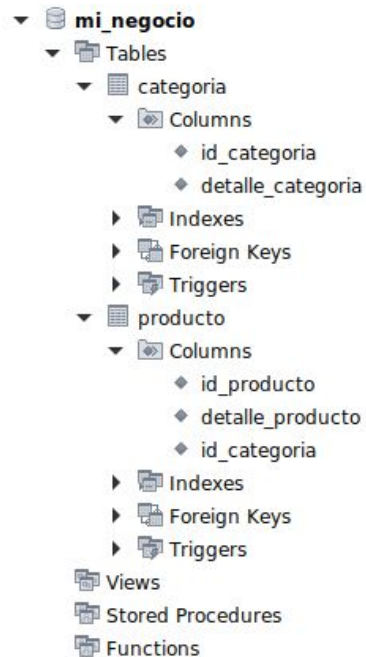




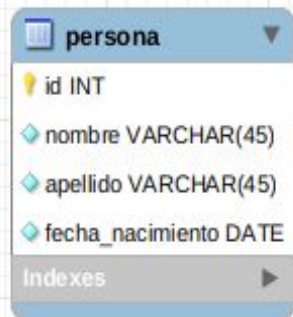
Diagrama entidad-relación



Ejemplos

persona			
id (int)	nombre (nvarchar)	apellido (nvarchar)	fecha_nacimiento (date)

producto	
id (int)	detalle (nvarchar)



persona - Table 

[illegible]

```
ejemplo_uno.sql
1  -- MySQL Script generated by MySQL Workbench
2  -- lun 06 jul 2020 01:04:57 -03
3  -- Model: New Model      Version: 1.0
4  -- MySQL Workbench Forward Engineering
5
6  SET @OLD_UNIQUE_CHECKS=@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
7  SET @OLD_FOREIGN_KEY_CHECKS=@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
8  SET @OLD_SQL_MODE=@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
9
10 -----
11 -- Schema mydb
12 -----
13
14 -----
15 -- Schema mydb
16 -----
17 CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
18 USE `mydb` ;
19
20 -----
21 -- Table `mydb`.`persona`
22 -----
23 CREATE TABLE IF NOT EXISTS `mydb`.`persona` (
24   `id` INT NOT NULL AUTO_INCREMENT,
25   `nombre` VARCHAR(45) NOT NULL,
26   `apellido` VARCHAR(45) NOT NULL,
27   `fecha_nacimiento` DATE NOT NULL,
28   PRIMARY KEY (`id`))
29 ENGINE = InnoDB;
30
31 -----
32 -- Table `mydb`.`producto`
33 -----
34
35 CREATE TABLE IF NOT EXISTS `mydb`.`producto` (
36   `id` INT NOT NULL AUTO_INCREMENT,
37   `detalle` VARCHAR(45) NOT NULL,
38   PRIMARY KEY (`id`))
39 ENGINE = InnoDB;
40
41
42 SET SQL_MODE=@OLD_SQL_MODE;
43 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
44 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Query 1

Limit to 1000 rows

```
1 • SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
2 • SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
3 • SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
4
5 • CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
6 • USE `mydb` ;
7
8 • CREATE TABLE IF NOT EXISTS `mydb`.`persona` (
9     `id` INT NOT NULL AUTO_INCREMENT,
10     `nombre` VARCHAR(45) NOT NULL,
11     `apellido` VARCHAR(45) NOT NULL,
12     `fecha_nacimiento` DATE NOT NULL,
13     PRIMARY KEY (`id`))
14     ENGINE = InnoDB;
15
16 • CREATE TABLE IF NOT EXISTS `mydb`.`producto` (
17     `id` INT NOT NULL AUTO_INCREMENT,
18     `detalle` VARCHAR(45) NOT NULL,
19     PRIMARY KEY (`id`))
20     ENGINE = InnoDB;
21
22
23 • SET SQL_MODE=@OLD_SQL_MODE;
24 • SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
25 • SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

ManagementSchemas

SCHEMAS

Filter objects

mydb

Tables

persona

Columns

id

nombre

apellido

fecha_nacimiento

Indexes

Foreign Keys

Triggers

producto

Columns

id

detalle

Indexes

Foreign Keys

Triggers

Views

Stored Procedures

Functions


sys

	#	Time	Action	Message	Duration / Fetch
✓	1	01:11:24	SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE...	0 row(s) affected	0,00033 sec
✓	2	01:11:24	SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECK...	0 row(s) affected	0,00030 sec
✓	3	01:11:24	SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITI...	0 row(s) affected	0,00043 sec
✓	4	01:11:24	CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTE...	1 row(s) affected	0,00061 sec
✓	5	01:11:24	USE `mydb`	0 row(s) affected	0,00021 sec
✓	6	01:11:24	CREATE TABLE IF NOT EXISTS `mydb`.`persona` (`id` INT N...	0 row(s) affected	1,310 sec
✓	7	01:11:26	CREATE TABLE IF NOT EXISTS `mydb`.`producto` (`id` INT ...	0 row(s) affected	0,287 sec
✓	8	01:11:26	SET SQL_MODE=@OLD_SQL_MODE	0 row(s) affected	0,00020 sec
✓	9	01:11:26	SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS	0 row(s) affected	0,00012 sec
✓	10	01:11:26	SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS	0 row(s) affected	0,00011 sec

Action Output

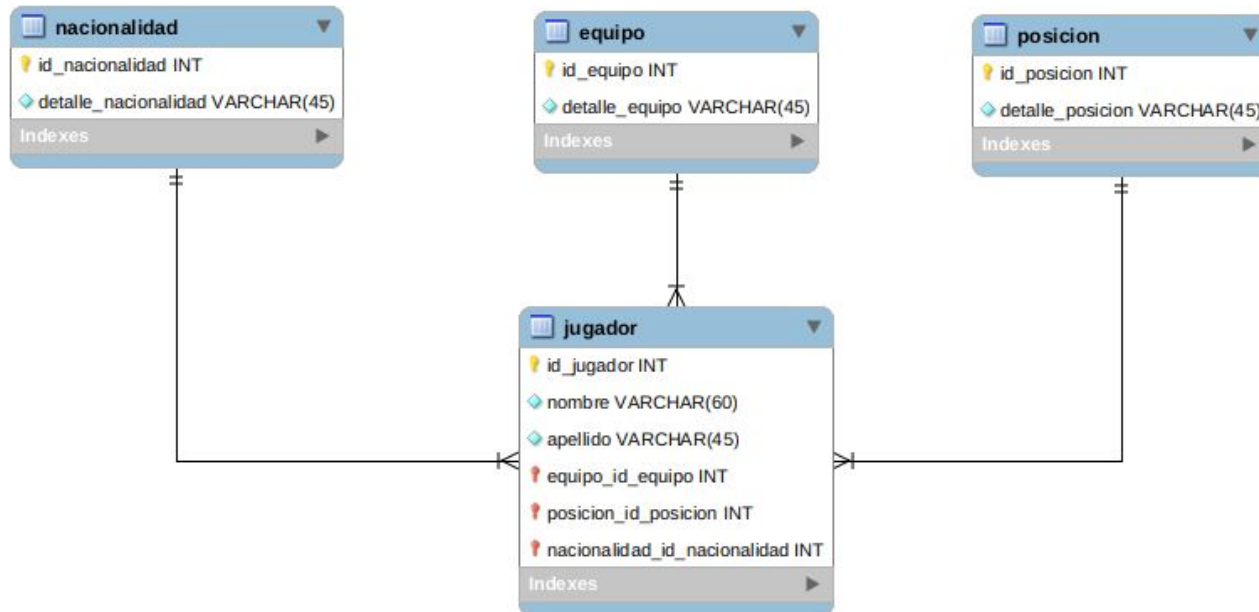


**Crear las tablas definidas en el problema
planteado la semana anterior**

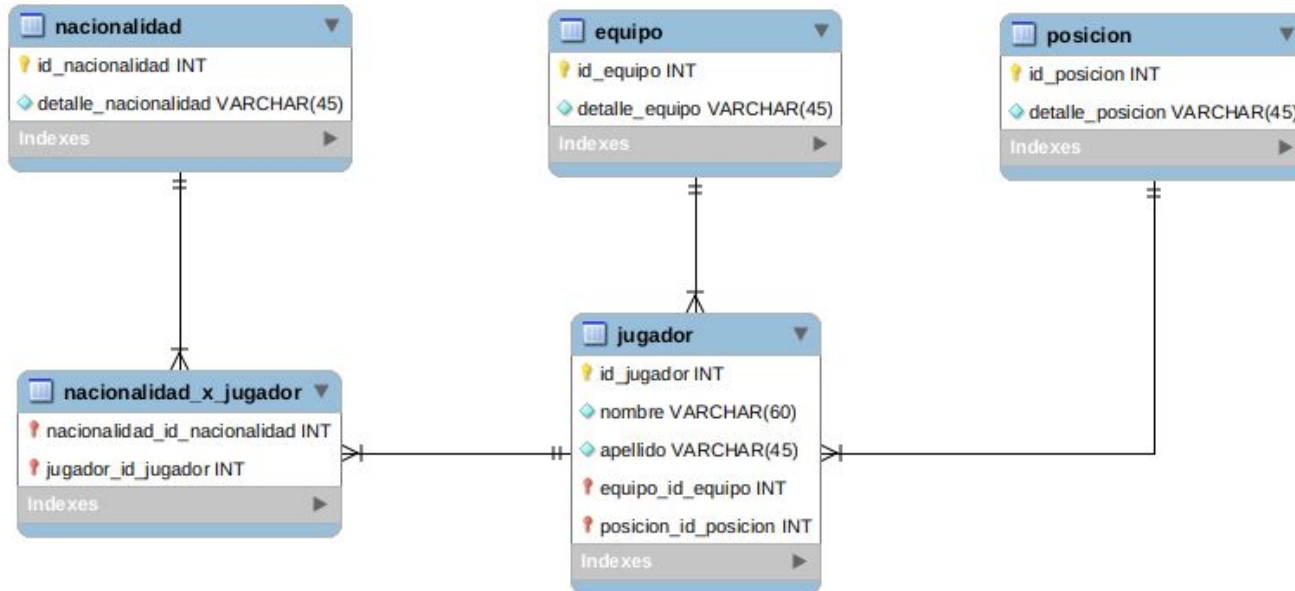


nombre	apellido	nacionalidad	equipo	posicion
Lionel Andres	Messi	Argentina	FC Barcelona	Extremo derecho
Cristiano	Ronaldo	Portugal	Juventus	Extremo izquierdo
Neymar da Silva	Santos Junior	Brasil	Paris Saint-Germain	Extremo izquierdo
Jan	Oblak	Eslovenia	Atletico Madrid	Arquero
Eden	Hazard	Belgica	Real Madrid	Extremo izquierdo
Marc-Andre	ter Stegen	Alemania	FC Barcelona	Arquero
Virgil	Van Dijk	Holanda	Liverpool	Defensa central
Sergio Leonel	Aguero	Argentina	Manchester City	Centro delantero

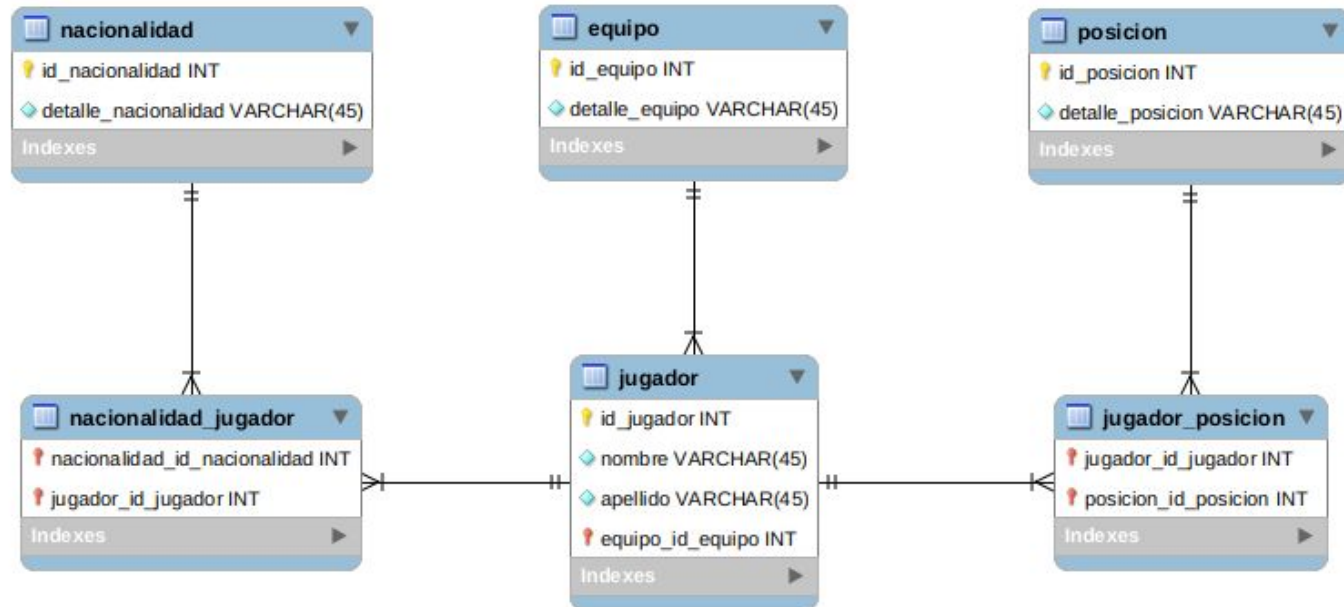
Crear las tablas definidas en el problema planteado la semana anterior




¿Y qué pasa si los jugadores pudieran tener varias nacionalidades? Ej: David Trezeguet (Francia | Argentina)

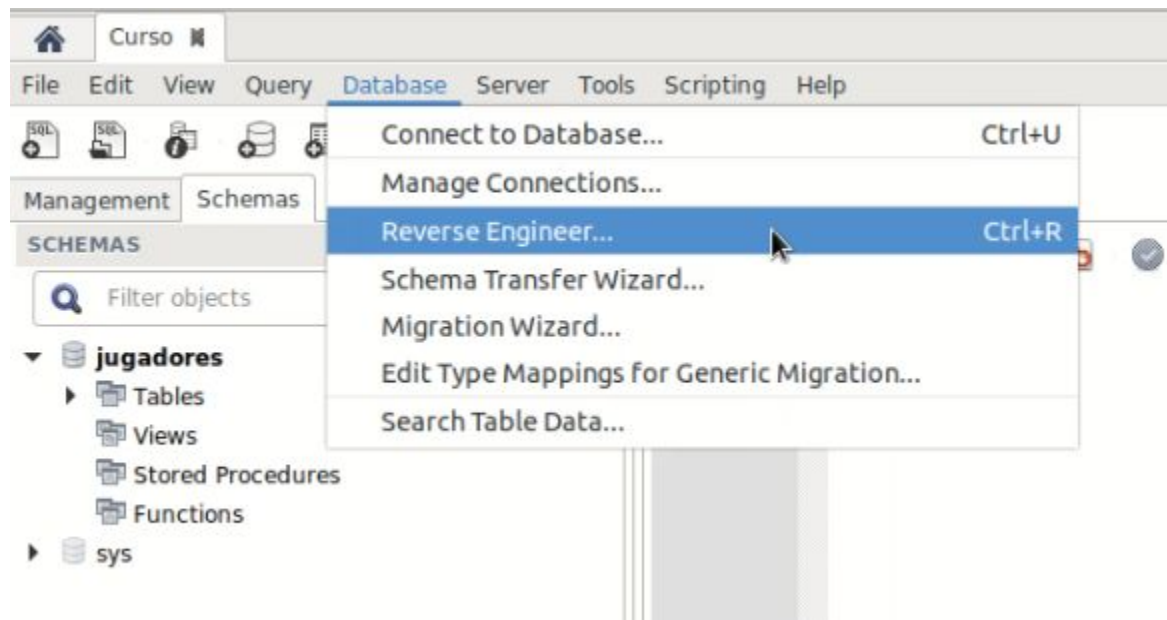


¿Y si tenemos algún jugador que puede jugar en 1 o 2 posiciones?





**La forma en que vamos a armar la
base va a depender del análisis que
hagamos**





Insertar datos



Primero tenemos que usar la base que corresponde

USE **jugadores**;

#	Time	Action	Message	Duration / Fetch
✓ 1	01:36:12	use jugadores	0 row(s) affected	0,00030 sec



Insertar datos tabla posición

Primero tenemos que insertar datos en las tablas que no tienen claves foráneas (Porque para que esa clave foránea exista primero tiene que estar cargada en otra tabla:

Sintaxis:

INSERT INTO <nombre_de_tabla> (campos de la tabla) VALUES (los valores que queremos insertar)

id_posicion	detalle_posicion
1	Extremo derecho
2	Extremo izquierdo
3	Arquero
4	Defensa central
5	Centro delantero



Insertar datos

Podemos insertar de a un valor:

```
INSERT INTO posicion (detalle_posicion) VALUES ('Extremo derecho')
```

#	Time	Action	Message	Duration / Fetch
✓ 1	01:38:30	INSERT INTO posicion (detalle_posicion) VALUES ('Extremo der...	1 row(s) affected	0,032 sec



Insertar datos

O podemos hacer insert multiples:

```
INSERT INTO posicion (detalle_posicion) VALUES ('Extremo izquierdo'), ('Arquero'), ('Defensa central'), ('Centro delantero')
```

#	Time	Action	Message	Duration / Fetch
✓ 1	01:39:55	INSERT INTO posicion (detalle_posicion) VALUES ('Extremo izq...	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0	0,032 sec

Insertar datos tabla nacionalidad

id_nacionalidad	detalle_nacionalidad
1	Argentina
2	Portugal
3	Brasil
4	Eslovenia
5	Belgica
6	Alemania
7	Holanda

INSERT INTO nacionalidad (detalle_nacionalidad) VALUES ('Argentina'), ('Portugal'), ('Brasil'), ('Eslovenia'), ('Belgica'), ('Alemania'), ('Holanda')

#	Time	Action	Message	Duration / Fetch
✓ 1	01:47:42	INSERT INTO nacionalidad (detalle_nacionalidad) VALUES ('Arg...	7 row(s) affected Records: 7 Duplicates: 0 Warnings: 0	0.033 sec

Insertar datos tabla nacionalidad

id_equipo	detalle_equipo
1	FC Barcelona
2	Juventus
3	Paris Saint-Germain
4	Atletico Madrid
5	Real Madrid
6	Liverpool
7	Manchester City

INSERT INTO equipo (detalle_equipo) VALUES ('FC Barcelona'), ('Juventus'), ('Paris Saint-Germain'), ('Atletico Madrid'), ('Real Madrid'), ('Liverpool'), ('Manchester City')

#	Time	Action	Message	Duration / Fetch
✓ 1	01:48:59	INSERT INTO equipo (detalle_equipo) VALUES ('FC Barcelona'),...	7 row(s) affected Records: 7 Duplicates: 0 Warnings: 0	0,032 sec

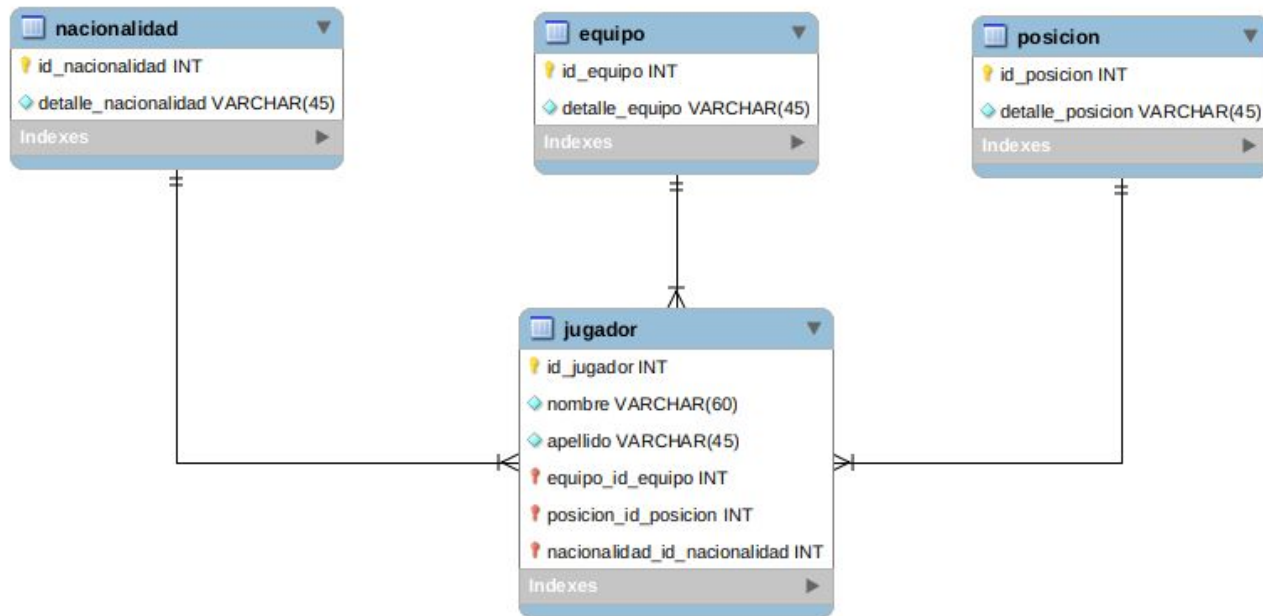
Ahora hay que insertar datos en la tabla principal: jugador

id_jugador	nombre	apellido	id_nacionalidad	id_equipo	id_posicion
1	Lionel Andres	Messi	Argentina	FC Barcelona	Extremo derecho
2	Cristiano	Ronaldo	Portugal	Juventus	Extremo izquierdo
3	Neymar da Silva	Santos Junior	Brasil	Paris Saint-Germain	Extremo izquierdo
4	Jan	Oblak	Eslovenia	Atletico Madrid	Arquero
5	Eden	Hazard	Belgica	Real Madrid	Extremo izquierdo
6	Marc-Andre	ter Stegen	Alemania	FC Barcelona	Arquero
7	Virgil	Van Dijk	Holanda	Liverpool	Defensa central
8	Sergio Leonel	Aguero	Argentina	Manchester City	Centro delantero

id_posicion	detalle_posicion
1	Extremo derecho
2	Extremo izquierdo
3	Arquero
4	Defensa central
5	Centro delantero

id_equipo	detalle_equipo
1	FC Barcelona
2	Juventus
3	Paris Saint-Germain
4	Atletico Madrid
5	Real Madrid
6	Liverpool
7	Manchester City

id_nacionalidad	detalle_nacionalidad
1	Argentina
2	Portugal
3	Brasil
4	Eslovenia
5	Belgica
6	Alemania
7	Holanda



Cambiamos los valores por las claves, renombramos y ordenamos las columnas

id_jugador	nombre	apellido	id_nacionalidad	id_equipo	id_posicion
1	Lionel Andres	Messi	Argentina	FC Barcelona	Extremo derecho
2	Cristiano	Ronaldo	Portugal	Juventus	Extremo izquierdo
3	Neymar da Silva	Santos Junior	Brasil	Paris Saint-Germain	Extremo izquierdo
4	Jan	Oblak	Eslovenia	Atletico Madrid	Arquero
5	Eden	Hazard	Belgica	Real Madrid	Extremo izquierdo
6	Marc-Andre	ter Stegen	Alemania	FC Barcelona	Arquero
7	Virgil	Van Dijk	Holanda	Liverpool	Defensa central
8	Sergio Leonel	Aguero	Argentina	Manchester City	Centro delantero

id_jugador	nombre	apellido	equipo_id_equipo	posicion_id_posicion	nacionalidad_id_nacionalidad
1	Lionel Andres	Messi	1	1	1
2	Cristiano	Ronaldo	2	2	2
3	Neymar da Silva	Santos Junior	3	2	3
4	Jan	Oblak	4	3	4
5	Eden	Hazard	5	2	5
6	Marc-Andre	ter Stegen	1	3	6
7	Virgil	Van Dijk	6	4	7
8	Sergio Leonel	Aguero	7	5	1



Insertamos los jugadores individualmente:

INSERT INTO jugador (nombre, apellido, equipo_id_equipo, posicion_id_posicion, nacionalidad_id_nacionalidad) VALUES ('Lionel Andres', 'Messi', 1, 1, 1)

	#	Time	Action	Message	Duration / Fetch
✓	1	02:02:16	INSERT INTO jugador (nombre, apellido, equipo_id_equipo, po...	1 row(s) affected	0,00060 sec



O múltiple:

```
INSERT INTO jugador (nombre, apellido, equipo_id_equipo, posicion_id_posicion, nacionalidad_id_nacionalidad)
VALUES
('Cristiano', 'Ronaldo', 2, 2, 2),
('Neymar da Silva', 'Santos Junior', 3, 2, 3),
('Jan', 'Oblak', 4, 3, 4),
('Eden', 'Hazard', 5, 2, 5),
('Marc-Andre', 'ter Stegen', 1, 3, 6),
('Virgil', 'Van Dijk', 6, 4, 7),
('Sergio Leonel', 'Aguero', 7, 5, 1)
```

#	Time	Action	Message	Duration / Fetch
✓ 2	02:07:26	INSERT INTO jugador (nombre, apellido, equipo_id_equipo, po...	7 row(s) affected Records: 7 Duplicates: 0 Warnings: 0	0.00080 sec



Consultar datos



Consultar datos









Vamos a empezar con consultas simple, en la próxima clase vamos a agregar mas cosas a las querys (consultas)

Sintaxis:


```
SELECT * FROM <nombre_de_tabla>
```


Consultar datos

SELECT * FROM equipo

Result Grid   Filter Rows: Edit:    Export/Import:   Wrap Cell Content: 

#	id_equipo	detalle_equipo
1	1	FC Barcelona
2	2	Juventus
3	3	Paris Saint-Germain
4	4	Atletico Madrid
5	5	Real Madrid
6	6	Liverpool
7	7	Manchester City

equipo 9 

Action Output 

#	Time	Action	Message	Duration / Fetch
1	02:30:28	SELECT * FROM equipo LIMIT 0, 1000	7 row(s) returned	0,00053 sec / 0,000...

Consultar datos

```
SELECT * FROM posicion
```

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Content:

#	id_posicion	detalle_posicion
1	1	Extremo derecho
2	2	Extremo izquierdo
3	3	Arquero
4	4	Defensa central
5	5	Centro delantero









posicion 4

Action Output


#	Time	Action	Message	Duration / Fetch
1	02:27:46	SELECT * FROM posicion LIMIT 0, 1000	5 row(s) returned	0,00036 sec / 0,000...

Consultar datos

SELECT * FROM nacionalidad

Result Grid   Filter Rows: Edit:    Export/Import:   Wrap Cell Content: 

#	id_nacionalidad	detalle_nacionalidad
1	1	Argentina
2	2	Portugal
3	3	Brasil
4	4	Eslovenia
5	5	Belgica
6	6	Alemania
7	7	Holanda









nacionalidad 7 

Action Output ▼


#	Time	Action	Message	Duration / Fetch
1	02:29:31	select * from nacionalidad LIMIT 0, 1000	7 row(s) returned	0,00054 sec / 0,000...


Consultar datos

SELECT * FROM jugador

Result Grid   Filter Rows:  Edit:    Export/Import:   Wrap Cell Content: ☐

#	id_jugador	nombre	apellido	equipo_id_equipo	posicion_id_posicion	nacionalidad_id_nacionalidad
1	1	Lionel Andres	Messi	1	1	1
2	2	Cristiano	Ronaldo	2	2	2
3	3	Neymar da Silva	Santos Junior	3	2	3
4	4	Jan	Oblak	4	3	4
5	5	Eden	Hazard	5	2	5
6	6	Marc-Andre	ter Stegen	1	3	6
7	7	Virgil	Van Dijk	6	4	7
8	8	Sergio Leonel	Aguero	7	5	1

jugador 10 

Action Output 

#	Time	Action	Message	Duration / Fetch
1	02:31:17	SELECT * FROM jugador LIMIT 0, 1000	8 row(s) returned	0,00037 sec / 0,000...



Actualizar datos



Actualizar datos

Vamos a empezar con consultas simple, en la próxima clase vamos a agregar mas cosas a las queries (consultas)

Sintaxis:

UPDATE <nombre_tabla>

SET <columna> = <valor>, <columna> = <valor>, ...

WHERE <condición para que encuentre la fila que necesitamos, en este caso vamos a usar el id_jugador>;



Consultar datos

Ej: Queremos que Neymar en nuestra base aparezca como Neymar Jr y no como Neymar da Silva Santos Junior

id_jugador	nombre	apellido	equipo_id_equipo	posicion_id_posicion	nacionalidad_id_nacionalidad
3	Neymar da Silva	Santos Junior	3	2	3
NULL	NULL	NULL	NULL	NULL	NULL

```
UPDATE jugador SET nombre = 'Neymar', apellido= 'Jr' WHERE id_jugador = 3;
```



Consultar datos

id_jugador	nombre	apellid	equipo_id_equipo	posicion_id_posicion	nacionalidad_id_nacionalidad
3	Neymar	Junior	3	2	3
NULL	NULL	NULL	NULL	NULL	NULL



Eliminar datos



Eliminar datos

Vamos a poder eliminar registros si no están relacionados con otras tablas, por ejemplo podemos eliminar jugadores pero no podemos eliminar nacionalidades que están siendo usadas en algún jugador. Para eliminar una nacionalidad tendría que cambiarle la nacionalidad al jugador primero o eliminar directamente al jugador.

Sintaxis:

DELETE FROM <nombre_de_tabla> where <condición, en este caso el id_jugador> = <número de id>



Eliminar datos

Ej: Queremos eliminar a Agüero porque ya no forma parte de la plantilla

id_jugador	nombre	apellido	equipo_id_equipo	posicion_id_posicion	nacionalidad_id_nacionalidad
8	Sergio Leonel	Agüero	7	5	1
NULL	NULL	NULL	NULL	NULL	NULL

DELETE FROM **jugador** WHERE id_jugador = 8



Eliminar datos

id_jugador	nombre	apellido	equipo_id_equipo	posicion_id_posicion	nacionalidad_id_nacionalidad
1	Lionel Andres	Messi	1	1	1
2	Cristiano	Ronaldo	2	2	2
3	Neymar	Junior	3	2	3
4	Jan	Oblak	4	3	4
5	Eden	Hazard	5	2	5
6	Marc-Andre	ter Stegen	1	3	6
7	Virgil	Van Dijk	6	4	7
NULL	NULL	NULL	NULL	NULL	NULL



Modificar el auto_increment

Ahora si insertamos un nuevo jugador se va a insertar con el id_jugador nº 9 porque nosotros pusimos que ese campo sea auto_increment, si analizando el código vemos que no nos va a generar ningún problema podemos modificar ese auto_increment para que empiece desde el id que tenía Agüero para que no tengamos ese salto en los id de 7 a 9, esto lo hacemos con este comando:

ALTER TABLE <nombre_de_la_tabla> AUTO_INCREMENT = <nº_que_queremos_que_tenga_el_proximo_id_jugador>

ALTER TABLE jugador AUTO_INCREMENT = 8

#	Time	Action	Message	Duration / Fetch
✓ 1	02:53:10	ALTER TABLE jugador AUTO_INCREMENT = 8	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0,104 sec



Cargar un nuevo jugador

Ahora nuestro auto_increment empezará desde el nº 8 y seguirá haciendo +1 con cada jugador que se inserte, ej:

Jugador: Paulo Dybala

Nacionalidad: Argentino

Posicion: Medio centro ofensivo

Equipo: Juventus

Lo primero que hay que hacer es ver si tenemos todos los datos

Agregar una posición

Jugador: Paulo Dybala

Nacionalidad: Argentino **SI**

Posicion: Medio centro ofensivo **NO**

Equipo: Juventus **SI**

id_posicion	detalle_posicion
1	Extremo derecho
2	Extremo izquierdo
3	Arquero
4	Defensa central
5	Centro delantero
NULL	NULL

Antes de insertar al jugador tenemos que agregar esa nueva posición para poder tener el id

INSERT INTO posicion (detalle_posicion) VALUES ('Medio centro ofensivo')

#	Time	Action	Message	Duration / Fetch
1	03:02:21	INSERT INTO posicion (detalle_posicion) VALUES ('Medio centr...	1 row(s) affected	0,00052 sec



Ya podemos agregar al jugador

id_posicion	detalle_posicion
1	Extremo derecho
2	Extremo izquierdo
3	Arquero
4	Defensa central
5	Centro delantero
6	Medio centro ofensivo
NULL	NULL



Cargar un nuevo jugador

Jugador: Paulo Dybala

Nacionalidad: Argentino

Posicion: Medio centro ofensivo

Equipo: Juventus

Jugador: Paulo Dybala

Equipo: 2

Posicion: 6

Nacionalidad: 1

```
INSERT INTO jugador (nombre, apellido, equipo_id_equipo, posicion_id_posicion,  
nacionalidad_id_nacionalidad) VALUES ('Paulo', 'Dybala', 2, 6, 1)
```



Cargar un nuevo jugador

id_jugador	nombre	apellido	equipo_id_equipo	posicion_id_posicion	nacionalidad_id_nacionalidad
1	Lionel Andres	Messi	1	1	1
2	Cristiano	Ronaldo	2	2	2
3	Neymar	Junior	3	2	3
4	Jan	Oblak	4	3	4
5	Eden	Hazard	5	2	5
6	Marc-Andre	ter Stegen	1	3	6
7	Virgil	Van Dijk	6	4	7
8	Paulo	Dybala	2	6	1
NULL	NULL	NULL	NULL	NULL	NULL



Agregar y eliminar columnas de una tabla



Agregar columnas

Vamos a agregar dos columnas a la tabla jugador

- 1) sueldo_mensual
- 2) valor_mercado

Ambas columnas van a crearse luego de la columna apellido

id_jugador	nombre	apellido	equipo_id_equipo	posicion_id_posicion	nacionalidad_id_nacionalidad
1	Lionel Andres	Messi	1	1	1
2	Cristiano	Ronaldo	2	2	2
3	Neymar	Junior	3	2	3



Agregar columnas

Sintaxis:

```
ALTER TABLE <nombre_de_tabla> ADD COLUMN <nombre_de_columna> <tipo_de_dato>  
AFTER <nombre_de_la_columna>;
```

```
ALTER TABLE jugador ADD COLUMN sueldo_mensual int(20) AFTER apellido;
```

```
ALTER TABLE jugador ADD COLUMN valor_mercado int(20) AFTER sueldo_mensual;
```


Agregar columnas

Como ven se creo la nueva columna con los valores vacíos, ahora habría que hacer un update por cada id_jugador para agregarle el sueldo a cada uno.

[illegible]



Agregar columnas

```
UPDATE jugador SET sueldo_mensual = 75000 WHERE id_jugador = 1;  
UPDATE jugador SET sueldo_mensual = 70000 WHERE id_jugador = 2;  
UPDATE jugador SET sueldo_mensual = 68000 WHERE id_jugador = 3;  
UPDATE jugador SET sueldo_mensual = 65000 WHERE id_jugador = 4;  
UPDATE jugador SET sueldo_mensual = 63000 WHERE id_jugador = 5;  
UPDATE jugador SET sueldo_mensual = 60000 WHERE id_jugador = 6;  
UPDATE jugador SET sueldo_mensual = 58000 WHERE id_jugador = 7;  
UPDATE jugador SET sueldo_mensual = 55000 WHERE id_jugador = 8;
```

SELECT * FROM jugador

[illegible]



Eliminar columnas

Supongamos que la columna `valor_mercado` no nos sirve, la podemos eliminar de la siguiente forma:

Sintaxis:

```
ALTER TABLE <nombre_de_la_tabla> DROP COLUMN <nombre_de_la_columna>
```

```
ALTER TABLE jugador DROP COLUMN valor_mercado
```

#	Time	Action	Message	Duration / Fetch
✓ 1	03:31:34	ALTER TABLE jugador DROP COLUMN valor_mercado	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	1,912 sec



SELECT * FROM jugador

id_jugador	nombre	apellido	sueldo_mensual	equipo_id_equipo	posicion_id_posicion	nacionalidad_id_nacionalidad
1	Lionel Andres	Messi	75000	1	1	1
2	Cristiano	Ronaldo	70000	2	2	2
3	Neymar	Junior	68000	3	2	3
4	Jan	Oblak	65000	4	3	4
5	Eden	Hazard	63000	5	2	5
6	Marc-Andre	ter Stegen	60000	1	3	6
7	Virgil	Van Dijk	58000	6	4	7
8	Paulo	Dybala	55000	2	6	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL



Para la próxima clase

- Releer la diapositiva y el material
- La próxima clase vamos a sumar cosas a las tablas para poder hacer consultas más complejas

