BASE DE DATOS I

DEFENSA DEL HITO3

POR:MARCO ANTONIO CALLE VAQUIATA

Consigna

Diseñe un sistema de Base de Datos Relacional utilizando el gestor de Base de Datos SQL Server teniendo como premisa el uso de buenas prácticas en diseño de la base de datos aplicados al siguiente escenario. Una comunidad de estudiantes de la nación UNIFRANZ de nombre los UNIFRANZITOS desea implementar un nuevo sistema para poder administrar los CAMPEONATOS DE FÚTBOL de todas las sedes. Es decir crear un campeonato en donde puedan participar todas las sedes, en el campeonato pueden inscribirse tanto categoría varones y mujeres.

DETALLE DEL PROBLEMA

UNIFRANZITOS	
Problema	Se tiene como contexto un **CAMPEONATO DE FÚTBOL** en el cual se tiene 3 entidades principales el **campeonato** como tal, los **equipos** que participaran en el campeonato y en donde cada equipo tendrá una cantidad de **jugadores**. En tal sentido se deberá crear las siguientes **tablas**. • **campeonato** • **equipo** • **jugador** Detalle de las tablas.

campeonato

id_campeonato => cadena de 12 caracteres y ademas llave primaria
nombre_campeonato => una cadena de 30 caracteres que no acepta valores nulos
sede => una cadena de 20 caracteres que no acepta valores nulos

equipo

id_equipo => cadena de 12 caracteres y ademas llave primaria
nombre_equipo => una cadena de 30 caracteres, que no acepta valores nulos
categoria => esta columna recibe valores como (varones o mujeres), que no acepta valores nulos
id_campeonato => llave foreign key relacionado con la tabla **campeonato**

jugador

id_jugador => cadena de 12 caracteres y ademas llave primaria
nombres => una cadena de 30 caracteres, que no acepta valores nulos
apellidos => una cadena de 50 caracteres, que no acepta valores nulos
ci => una cadena de 15 caracteres (ejem: 8997899LP), que no acepta valores nulos
edad => un valor numérico, que no acepta valores nulos
id_equipo => llave foreign key relacionado con la tabla **equipo**

LO PASAMOS A CODIGO

```
create database CAMPEONATO_UNIFRANZ
use CAMPEONATO_UNIFRANZ

CREATE TABLE CAMPEONATO
(
ID_CAMPEONATO VARCHAR(12) PRIMARY KEY NOT NULL,
NOMBRE_DEL_CAMPEONATO VARCHAR (30) NOT NULL,
SEDE VARCHAR (30) NULL
);

CREATE TABLE EQUIPO
(
ID_EQUIPO VARCHAR(12) PRIMARY KEY NOT NULL,
ID_CAMPEONATO VARCHAR (12) NOT NULL,
NOMBRE_DE_EQUIPO VARCHAR (30) NOT NULL,
CATEGORIA VARCHAR (30) NULL,
FOREIGN KEY (ID_CAMPEONATO) REFERENCES CAMPEONATO (ID_CAMPEONATO )
);
```

LO PASAMOS A CODIGO

```
CREATE TABLE JUGADOR
(
ID_JUGADOR VARCHAR (12) PRIMARY KEY NOT NULL,
NOMBRES VARCHAR (30),
APELLIDOS VARCHAR (30),
CI VARCHAR (30),
EDAD INTEGER,
ID_EQUIPO VARCHAR(12) NOT NULL,
FOREIGN KEY (ID_EQUIPO) REFERENCES EQUIPO(ID_EQUIPO)
);
```

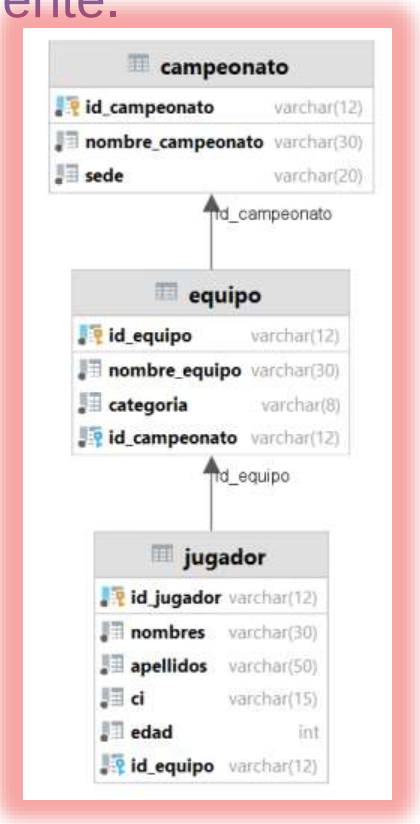
```
INSERT INTO CAMPEONATO(ID_CAMPEONATO,NOMBRE_DEL_CAMPEONATO,SEDE)
VALUES('CAMP-111','CAMPEONATO UNIFRANZ','EL ALTO')
 INSERT INTO CAMPEONATO(ID_CAMPEONATO,NOMBRE_DEL_CAMPEONATO,SEDE)
VALUES('CAMP-222','CAMPEONATO UNIFRANZ','COCHABAMBA')
   INSERT INTO CAMPEONATO(ID_CAMPEONATO,NOMBRE_DEL_CAMPEONATO,SEDE)
VALUES('CAMP-333','CAMPEONATO UNIFRANZ','PANDO')
```

INSERTAMOS LOS VALORES A LAS TABLAS

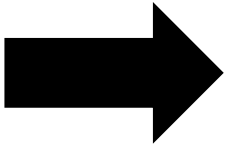
```
INSERT INTO EQUIPO(ID_EQUIPO,NOMBRE_DE_EQUIPO,CATEGORIA,ID_CAMPEONATO)
VALUES('EQUIP-111','GOOGLE','VARONES','CAMP-111')
INSERT INTO EQUIPO(ID_EQUIPO,NOMBRE_DE_EQUIPO,CATEGORIA,ID_CAMPEONATO)
VALUES('EQUIP-222','404 NOT FOUND ','VARONES','CAMP-111')
INSERT INTO EQUIPO(ID_EQUIPO,NOMBRE_DE_EQUIPO,CATEGORIA,ID_CAMPEONATO)
VALUES('EQUIP-333','GIRLS UNIFRANZ ','MUJERES','CAMP-111')


INSERT INTO JUGADOR(ID_JUGADOR,NOMBRES,APELLIDOS,CI,EDAD,ID_EQUIPO)
VALUES('JUG-111','CARLOS','VILLAS','8997811LP',19,'EQUIP-222')
 INSERT INTO JUGADOR(ID_JUGADOR,NOMBRES,APELLIDOS,CI,EDAD,ID_EQUIPO)
VALUES('JUG-222','PEDRO','SALAS','8997822LP',20,'EQUIP-222')
 INSERT INTO JUGADOR(ID_JUGADOR,NOMBRES,APELLIDOS,CI,EDAD,ID_EQUIPO)
VALUES('JUG-333','SAUL','ARAJ','8997833LP',21,'EQUIP-222')
 INSERT INTO JUGADOR(ID_JUGADOR,NOMBRES,APELLIDOS,CI,EDAD,ID_EQUIPO)
VALUES('JUG-444','SANDRA','SOLIS','8997844LP',20,'EQUIP-333')
 INSERT INTO JUGADOR(ID_JUGADOR,NOMBRES,APELLIDOS,CI,EDAD,ID_EQUIPO)
VALUES('JUG-555','ANA','MICA','8997855LP',23,'EQUIP-333')
```

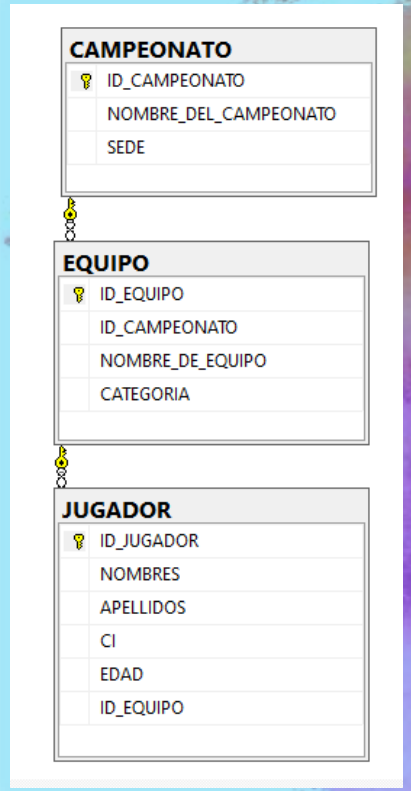
Diseño de base de datos.
1.Dado el detalle explicado en la parte inicial de este documento debería generar una base de datos similar al siguiente.

EJEMPLO



NUESTRO DIAGRAMA EN SQL





Manejo de conceptos
¿Que es DDL? y muestre un ejemplo UNIFRANZITOS

- DLL (definición de datos): Permiten crear y definir nuevas bases de datos, campos e índices.
  - CREATE
  - DROP
  - ALTER

```
CREATE DATABASE UNIFRANZITOS
USE UNIFRANZITOS
create table unifranz
(
id_codigo varchar(30) primary key not null,
nombre varchar(30)not null,
apellido varchar (30) not null,
sede varchar (20)not null
);
drop table unifranz
alter table unifranz
```

Que es DML y muestre un ejemplo aplicado a la base de datos UNIFRANZITOS

- DML (manipulación de datos): Permiten generar consultas para ordenar, filtrar y extraer datos.
  - SELECT
  - INSERT
  - UPDATE
  - DELETE

```
insert into unifranz(id_unifranz,nombre,apellido,sede)
values('123456','MARCO','CALLE','EL ALTO')
select*
from unifranz

delete unifranz

UPDATE unifranz
SET nombre = 'MARCO' , nombre='ANTONIO'
WHERE nombre='ANTONIO'
```

Que significa PRIMARY KEY y FOREIGN KEY

Una llave foránea es un grupo de una o más columnas en una tabla que referencias la llave primaria de otra tabla. No existe un código especial, configuración o definición de tabla que necesites establecer para «designar» oficialmente una llave foránea. Una llave foránea puede también ser parte de una llave primaria.

```
CREATE TABLE table_name (
    pk_column data_type PRIMARY KEY,
    ...
);
```

```
CONSTRAINT fk_grupo FOREIGN KEY (grupo_id)
    REFERENCES procurement.grupo_vendedores (grupo_id)
```

Defina que es una TABLA y que es una VISTA

Una **vista** es una **tabla** virtual cuyo contenido está definido por una consulta. Al igual que una **tabla**, una **vista** consta de un conjunto de columnas y filas de datos con un nombre. Sin embargo, a menos que esté indizada, una **vista** no existe como conjunto de valores de datos almacenados en una base de datos

```
create table unifranz
(
id_unifranz varchar(30) primary key not null,
nombre varchar(30)not null,
apellido varchar (30) not null,
sede varchar (20)not null
);
```

```
CREATE VIEW DetailsView AS
SELECT NAME, ADDRESS
FROM StudentDetails
WHERE S_ID < 5;
```

Cómo funciona LIKE en una consulta SQL. Adjunte un ejemplo

LIKE es un operador lógico de SQL Server que determina si una cadena de caracteres coincide con un patrón especificado. Un patrón puede incluir caracteres regulares y caracteres comodín. El operador LIKE se usa en la cláusula **WHERE** de las instrucciones **SELECT**, **UPDATE** y **DELETE** para filtrar filas en función de la coincidencia de patrones.

```
column | expression LIKE pattern [ESCAPE escape_character]
```

Para que se utiliza la cláusula WHERE

En una instrucción SQL, la **cláusula** WHERE especifica criterios que tienen que cumplir los valores de campo **para** que los registros que contienen los valores **se** incluyan en los resultados de la consulta.

Para los siguientes ejercicios crear 2 tablas cualesquiera.

```sql
CREATE DATABASE UNIFRANZITOS
USE UNIFRANZITOS
create table unifranz
(
id_unifranz varchar(30) primary key not null,
nombre varchar(30)not null,
apellido varchar (30) not null,
sede varchar (20)not null
);
CREATE TABLE AULAS
(
ID_AULAS VARCHAR (30) PRIMARY KEY NOT NULL,
NOM_DEL_DOCENTE VARCHAR (30) NOT NULL,
HORARIO INTEGER not null,
PISO VARCHAR (20) NOT NULL,
id_unifranz varchar(30)not null,
foreign key (id_unifranz) references unifranz(id_unifranz)
);
```

```sql
insert into unifranz(id_unifranz,nombre,apellido,sede)
values('123456','MARCO','LARA','EL ALTO')
insert into unifranz(id_unifranz,nombre,apellido,sede)
values('123457','ANTONIO','CALLE','LA PAZ')
insert into unifranz(id_unifranz,nombre,apellido,sede)
values('123458','FRANCO','SOLES','PANDA')

INSERT INTO  AULAS(ID_AULAS,NOM_DEL_DOCENTE,HORARIO,PISO)
VALUES('1','MATE',12.00,'PISO4')
INSERT INTO  AULAS(ID_AULAS,NOM_DEL_DOCENTE,HORARIO,PISO)
VALUES('2','LEN',13.00,'PISO2')
INSERT INTO  AULAS(ID_AULAS,NOM_DEL_DOCENTE,HORARIO,PISO)
VALUES('3','FISI',14.00,'PISO6')
```

Apoyándonos en el concepto de conjuntos muestre los siguiente: Ejemplo de INNER JOIN
Adjuntar una imagen de conjuntos y la consulta SQL que refleje el INNER JOIN

```
SELECT*
FROM AULAS  AS AU
INNER JOIN unifranz AS un
ON AU.id_unifranz = un.id_unifranz
```

	ID_AULAS	NOM_DEL_DOCENTE	HORARIO	PISO	id_unifranz	id_unifranz	nombre	apellido	sede
1	1	MATE	12	PISO4	123456	123456	MARCO	CALLE	EL ALTO
2	2	LEN	13	PISO2	123456	123456	MARCO	CALLE	EL ALTO
3	3	FISI	14	PISO6	123456	123456	MARCO	CALLE	EL ALTO

Apoyándonos en el concepto de conjuntos muestre los siguiente: Ejemplo de LEFT JOIN
Adjuntar una imagen de conjuntos y la consulta SQL que refleje el LEFT JOIN

```
SELECT*
FROM AULAS  AS AU
LEFT JOIN unifranz AS un
ON AU.ID_AULAS = AU.id_unifranz
```

	ID_AULAS	NOM_DEL_DOCENTE	HORARIO	PISO	id_unifranz	id_unifranz	nombre	apellido	sede
1	1	MATE	12	PISO4	123456	NULL	NULL	NULL	NULL
2	2	LEN	13	PISO2	123456	NULL	NULL	NULL	NULL
3	3	FISI	14	PISO6	123456	NULL	NULL	NULL	NULL

Apoyándonos en el concepto de conjuntos muestre los siguiente:
Ejemplo de RIGHT JOIN
Adjuntar una imagen de conjuntos y la consulta SQL que refleje el RIGHT JOIN

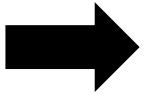
```
SELECT*
FROM AULAS  AS AU
RIGHT JOIN unifranz AS un
ON AU.ID_AULAS = AU.id_unifranz
```

	ID_AULAS	NOM_DEL_DOCENTE	HORARIO	PISO	id_unifranz	id_unifranz	nombre	apellido	sede
1	NULL	NULL	NULL	NULL	NULL	123456	MARCO	CALLE	EL ALTO
2	NULL	NULL	NULL	NULL	NULL	123457	ANTONIO	CALLE	LA PAZ
3	NULL	NULL	NULL	NULL	NULL	123458	FRANCO	SOLES	PANDA

Manejo de consultas

Mostrar que jugadores que formen parte del equipo equ-222

```
SELECT JU.NOMBRES,JU.APELLIDOS
FROM JUGADOR AS JU
INNER JOIN EQUIPO AS EQ
ON JU.ID_EQUIPO=EQ.ID_EQUIPO
WHERE JU.ID_EQUIPO ='EQUIP-222'
```



	NOMBRES	APELLIDOS
1	CARLOS	VILLAS
2	PEDRO	SALAS
3	SAUL	ARAJ

Mostrar que jugadores que formen parte del equipo equ-333

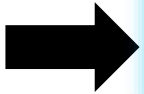
```
SELECT JU.NOMBRES,JU.APELLIDOS
FROM JUGADOR AS JU
INNER JOIN EQUIPO AS EQ
ON JU.ID_EQUIPO=EQ.ID_EQUIPO
WHERE JU.ID_EQUIPO ='EQUIP-333'
```

	NOMBRES	APELLIDOS
1	SANDRA	SOLIS
2	ANA	MICA

Manejo de consultas

Mostrar aquellos jugadores mayores o igual a 21 años

```
SELECT*
FROM JUGADOR AS JU
WHERE JU.EDAD > 21
```



	ID_JUGADOR	NOMBRES	APELLIDOS	CI	EDAD	ID_EQUIPO
1	JUG-555	ANA	MICA	8997855LP	23	EQUIP-333

Mostrar a todos los JUGADORES en donde su apellido empiece con la letra S.

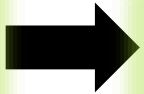
```
SELECT*
FROM JUGADOR AS JU
WHERE JU.APELLIDOS LIKE 'S%'
```

	ID_JUGADOR	NOMBRES	APELLIDOS	CI	EDAD	ID_EQUIPO
1	JUG-222	PEDRO	SALAS	8997822LP	20	EQUIP-222
2	JUG-444	SANDRA	SOLIS	8997844LP	20	EQUIP-333

Manejo de consultas

Mostrar que equipos forman parte del campeonato camp-111 y además sean de la categoría MUJERES

```
SELECT*
FROM EQUIPO AS EQ
INNER JOIN CAMPEONATO AS CA
ON EQ.ID_CAMPEONATO=CA.ID_CAMPEONATO
WHERE CA.ID_CAMPEONATO ='CAMP-111' AND  EQ.CATEGORIA ='MUJERES'
```



	ID_EQUIPO	ID_CAMPEONATO	NOMBRE_DE_EQUIPO	CATEGORIA	ID_CAMPEONATO
1	EQUIP-333	CAMP-111	GIRLS UNIFRANZ	MUJERES	CAMP-111

Mostrar el nombre del equipo del jugador con id_jugador igual a jug-333

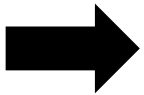
```
SELECT EQ.NOMBRE_DE_EQUIPO
FROM JUGADOR AS JU
INNER JOIN EQUIPO AS EQ
ON JU.ID_EQUIPO=EQ.ID_EQUIPO
WHERE JU.ID_JUGADOR='JUG-333'
```

	NOMBRE_DE_EQUIPO
1	404 NOT FOUND

Manejo de consultas

Mostrar el nombre del campeonato del jugador con id_jugador igual a jug-333

```
SELECT CAM.ID_CAMPEONATO
FROM JUGADOR AS JU
INNER JOIN EQUIPO AS EQ
ON JU.ID_EQUIPO=EQ.ID_EQUIPO
INNER JOIN CAMPEONATO AS CAM
ON EQ.ID_CAMPEONATO=CAM.ID_CAMPEONATO
WHERE JU.ID_JUGADOR='JUG-333'
```



	ID_CAMPEONATO
1	CAMP-111

Crear una consulta SQL que maneje las 3 tablas de la base de datos

```
SELECT CAM.ID_CAMPEONATO
FROM JUGADOR AS JU
INNER JOIN EQUIPO AS EQ
ON JU.ID_EQUIPO=EQ.ID_EQUIPO
INNER JOIN CAMPEONATO AS CAM
ON EQ.ID_CAMPEONATO=CAM.ID_CAMPEONATO
WHERE JU.ID_JUGADOR='JUG-333'
```

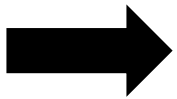
Manejo de consultas

¿Qué estrategia utilizaría para determinar cuántos equipos inscritos hay?

Para poder determinar cuantos equipos hay escritos utilizaremos el comando COUNT

Podría utilizar la función de agregación COUNT

```
SELECT COUNT(*)
FROM EQUIPO AS EQ
```



	(No column name)
1	3

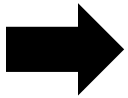
Manejo de consultas

¿Qué estrategia utilizaría para determinar cuántos jugadores pertenecen a la categoría VARONES o Categoria MUJERES?

Para poder determinar cuantos jugadores hay en la categoría tanto varones como mujeres utilizaremos el comando COUNT

Para esto puede utilizar la función de agregación COUNT

```
SELECT COUNT(*) NUMERO_DE_LOS_JUGADORES_VARONES
from JUGADOR AS JU
INNER JOIN EQUIPO AS EQ
ON JU.ID_EQUIPO=EQ.ID_EQUIPO
WHERE EQ.CATEGORIA='VARONES'
SELECT COUNT(*) NUMERO_DE_LOS_JUGADORES_MUJERES
from JUGADOR AS JU
INNER JOIN EQUIPO AS EQ
ON JU.ID_EQUIPO=EQ.ID_EQUIPO
WHERE EQ.CATEGORIA='MUJERES'
```



	NUMERO_DE_LOS_JUGADORES_VARONES
1	3

	NUMERO_DE_LOS_JUGADORES_MUJERES
1	2

GRACIAS!!!