UNIVERSIDAD PRIVADA
UNIFRANZ
FRANZ TAMAYO
Internacionalízate

EVALUACIÓN HITO 3

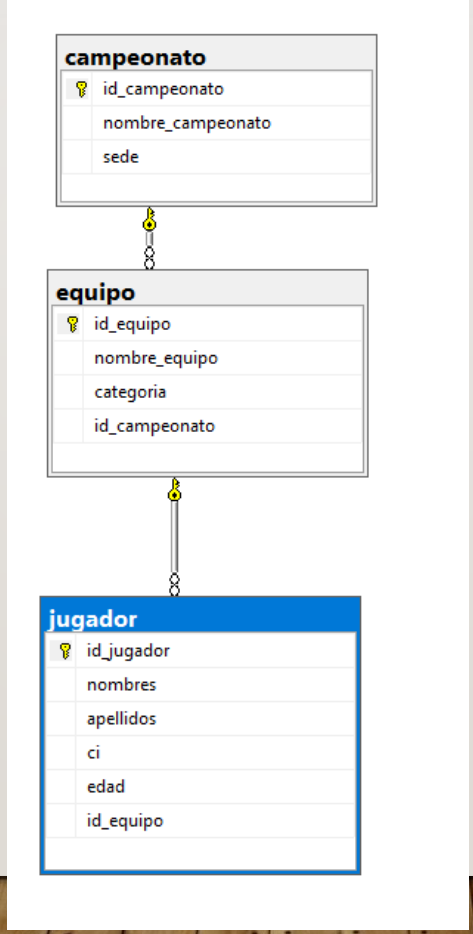
MANEJO DE CONSULTAS

BASE DE DATOS 1-2022

INGENIERÍA DE SISTEMAS

MANEJO DE CONCEPTOS

2.1 Adjuntar el diagrama generado por el editor



2.2¿QUE ES DDL Y DML?

- El lenguaje de definición de datos (DDL), se utilizan para describir una base de datos, para definir su estructura, para crear sus objetos y para crear los subobjetos de la tabla. (CREATE,ALTER, DROP)
- Las sentencias **DML** y lenguaje de manipulación de datos (DML) se utilizan para controlar la información contenida en la base de datos (INSERT, UPDATE).

```
CREATE TABLE campeonato(id_campeonato varchar (12) PRIMARY KEY not null
nombre_campeonato varchar (30) not null,
sede varchar(20) not null);
```

Lenguaje DDL

```
INSERT INTO campeonato(id_campeonato,nombre_campeonato,sede)
values('camp-111','Campeonato Unifranz','El Alto');

INSERT INTO campeonato(id_campeonato,nombre_campeonato,sede)
values('camp-222','Campeonato Unifranz','Cochabamba');
```

Lenguaje DML

2.3¿Qué significa PRIMARY KEY y FOREIGN KEY?

Las llaves primarias (**Primary Key**) son los que identifican de manera única cada fila o registro de una tabla, esto quiere decir que no se puede repetir en una tabla el valor de un campo o columna que se le es asignado como **primary key**.

```
CREATE TABLE equipo(id_equipo varchar (12) PRIMARY KEY not null,
nombre_equipo varchar(30) not null,
categoria varchar (10) not null,
id_campeonato varchar (12),
FOREIGN KEY (id_campeonato) REFERENCES campeonato(id_campeonato));
```

Una llave foránea, externa o ajena (Foreign Key) es un campo de una tabla "X" que sirve para enlazar o relacionar entre sí con otra tabla "Y" en la cual el campo de esta tabla es una llave primaria (Primary Key). Para que sea una llave foránea un campo, esta tiene que ser una llave primaria en otra tabla.

2.4 DEFINA QUE ES UNA TABLA Y EL USO DE IDENTITY

- Las tablas son objetos de base de datos que contienen todos sus datos. En las tablas, los datos se organizan con arreglo a un formato de filas y columnas, similar al de una hoja de cálculo. Cada fila representa un registro único y cada columna un campo dentro del registro.
- Un campo definido como "identity" generalmente se establece como clave primaria. Un campo "identity" no es editable, es decir, no se puede ingresar un valor ni actualizarlo.

```
CREATE TABLE materias(
id_mat integer IDENTITY PRIMARY KEY not null,
nombre_mat varchar(100) not null,
cod_mat varchar(100) not null);
```

	id_mat	nombre_mat	cod_mat
1	1	Introduccion a la Arquitectura	ARQ-101
2	2	Urbanismo y Diseno	ARQ-102
3	3	Dibujo y Pintura Arquitectonico	ARQ-103
4	4	Matematica discreta	ARQ-104
5	5	Fisica Basica	ARQ-105

2.5 Para que se utiliza la clausula WHERE

Se utiliza al momento de utilizar la clausula "Select-From", donde WHERE se utiliza para marcar una condición al momento de querer mostrar una clausula especifica.

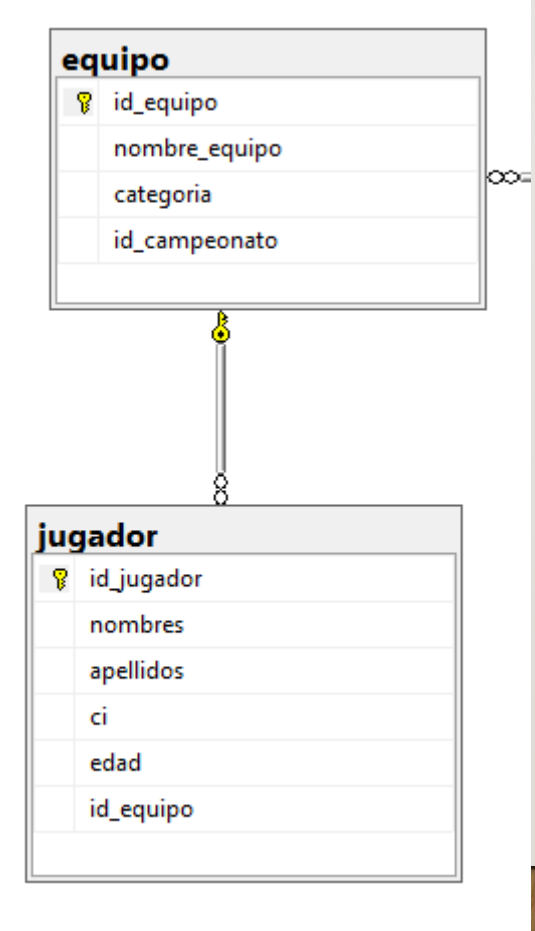
```
SELECT *
FROM jugador as ju
WHERE ju.nombres LIKE 'S%' and ju.apellidos LIKE '%s';
```

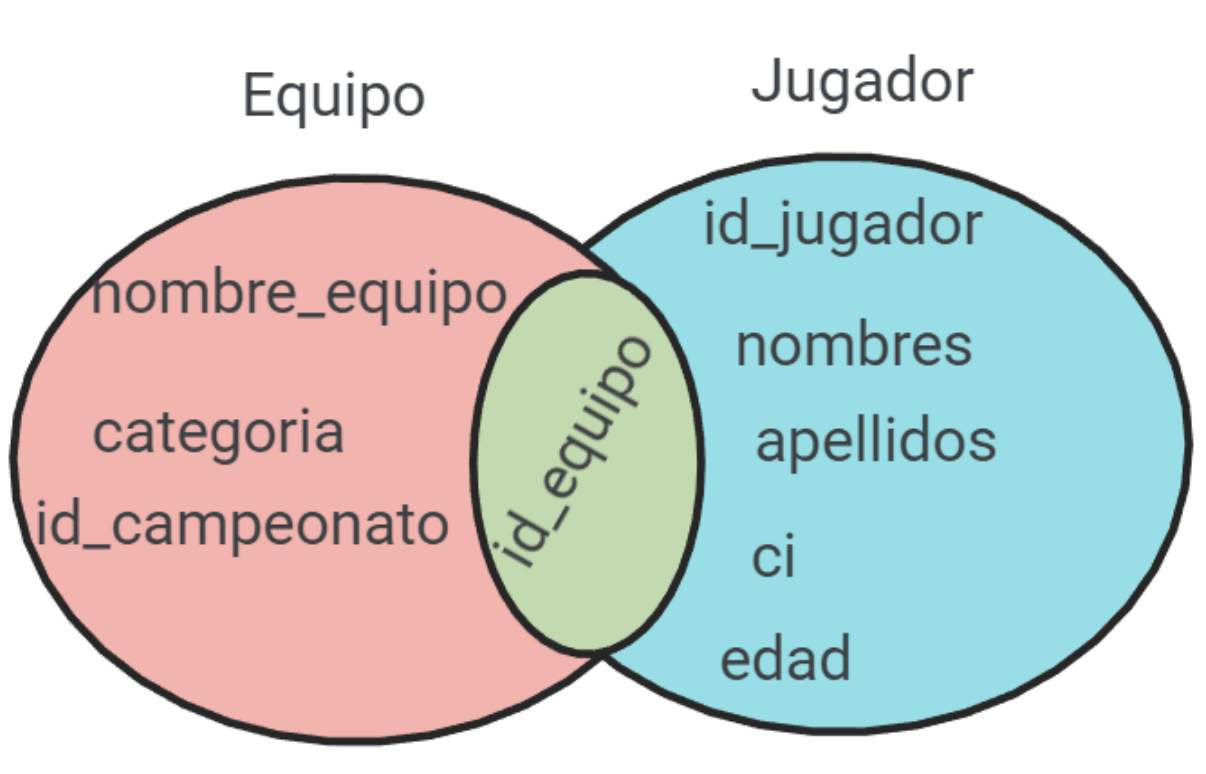
2.6 Para que se utiliza la instrucción INNER JOIN

Para relacionar dos tablas que tengan valores coincidentes en un campo común.

```
INNER JOIN equipo as eq ON eq.id_equipo=ju.id_equipo
```

2.7 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





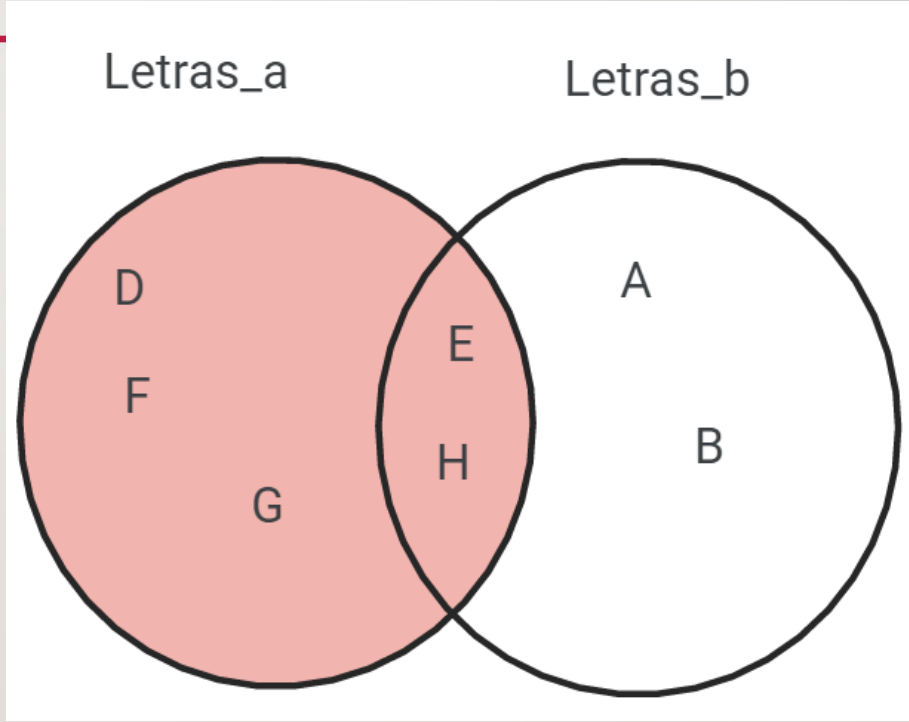

```
SELECT ju.nombres,ju.apellidos,ju.edad
FROM jugador as ju
INNER JOIN equipo as eq ON eq.id_equipo=ju.id_equipo
WHERE ju.edad<=20 AND eq.nombre_equipo='404 Not found';
```

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

```
CREATE TABLE letras_A
(letra char PRIMARY KEY NOT NULL);
CREATE TABLE letras_B
(letras char PRIMARY KEY NOT NULL);
INSERT INTO letras_A(letra)values ('D');
INSERT INTO letras_A(letra)values ('E');
INSERT INTO letras_A(letra)values ('F');
INSERT INTO letras_A(letra)values ('G');
INSERT INTO letras_A(letra)values ('H');
INSERT INTO letras_B(letras)VALUES ('A');
INSERT INTO letras_B(letras)values ('B');
INSERT INTO letras_B(letras)values ('E');
INSERT INTO letras_B(letras)values ('H');
```

```
SELECT ta.letra,tb.letras
FROM letras_A as ta
Left JOIN letras_B AS tb ON ta.letra=tb.letras;
```

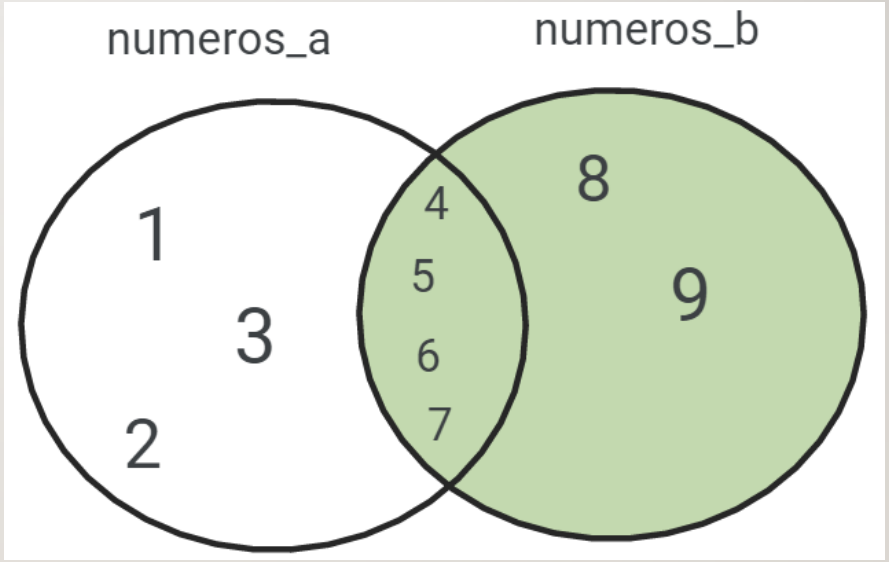
	letra	letras
1	D	NULL
2	E	E
3	F	NULL
4	G	NULL
5	H	H



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

```sql
CREATE TABLE numeros_A (numer int NOT NULL);
INSERT INTO numeros_A(numer)values (1);
INSERT INTO numeros_A(numer)values (2);
INSERT INTO numeros_A(numer)values (3);
INSERT INTO numeros_A(numer)values (4);
INSERT INTO numeros_A(numer)values (5);
INSERT INTO numeros_A(numer)values (6);
INSERT INTO numeros_A(numer)values (7);

CREATE TABLE numeros_B (numero int NOT NULL);
INSERT INTO numeros_B(numero)values (4);
INSERT INTO numeros_B(numero)values (5);
INSERT INTO numeros_B(numero)values (6);
INSERT INTO numeros_B(numero)values (7);
INSERT INTO numeros_B(numero)values (8);
INSERT INTO numeros_B(numero)values (9);
```




```sql
SELECT ta.numer,tb.numero
FROM numeros_A as ta
Right JOIN numeros_B AS tb ON ta.numer=tb.numero;
```

	numer	numero
1	4	4
2	5	5
3	6	6
4	7	7
5	NULL	8
6	NULL	9

2.10 Crear 3 tablas y crear una consulta SQL que muestra el uso de INNER JOIN.

```
CREATE TABLE numeros_A (numer int NOT NULL);
INSERT INTO numeros_A(numer)values (1);
INSERT INTO numeros_A(numer)values (2);
INSERT INTO numeros_A(numer)values (3);
INSERT INTO numeros_A(numer)values (4);
INSERT INTO numeros_A(numer)values (5);
INSERT INTO numeros_A(numer)values (6);
INSERT INTO numeros_A(numer)values (7);

CREATE TABLE numeros_B (numero int NOT NULL);
INSERT INTO numeros_B(numero)values (4);
INSERT INTO numeros_B(numero)values (5);
INSERT INTO numeros_B(numero)values (6);
INSERT INTO numeros_B(numero)values (7);
INSERT INTO numeros_B(numero)values (8);
INSERT INTO numeros_B(numero)values (9);
```

```
CREATE TABLE numeros_C(numeroi int not null);
INSERT INTO numeros_C(numeroi)values(2),
(4),
(6),
(8),
(10);
```

```
SELECT ta.numer,tb.numero,tc.numeroi
FROM numeros_A as ta
Inner JOIN numeros_B AS tb ON ta.numer=tb.numero
Inner join numeros_C AS tc ON tc.numeroi=tb.numero;
```

	numer	numero	numeroi
1	4	4	4
2	6	6	6

Parte Practica

Manejo de consultas

3.1Mostrar que jugadores que son del equipo equ-222

3.2 Mostrar que jugadores(nombres, apellidos) que juegan en la sede de El Alto

3.3. Mostrar aquellos jugadores mayores o igual a 21 años que sean de la categoría VARONES

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

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

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

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

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

3.9. ¿Qué estrategia utilizaría para determinar cuántos equipos inscritos hay?

3.10. ¿Qué estrategia utilizaría para determinar cuántos jugadores pertenecen a la categoría VARONES o Categoría MUJERES.