

# PROCESUAL HJTO 4

## INGENIERIA EN SISTEMAS

**INGENIERO:**

WILLIAM BARRA

**INTEGRANTE:**

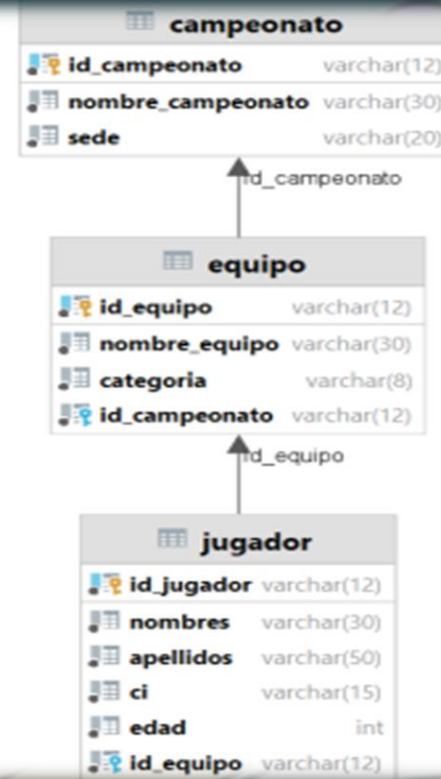
HEBER QUISPE

**GESTION:** 2023



# CONSIGNIA

1. Diseño de base de datos. 1.1. Dado el detalle explicado en la parte inicial de este documento debería generar una base de datos similar al siguiente



1. Los registros de cada tabla deberían quedar de la siguiente forma



# CONSIGNIA

```
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')

INSERT INTO EQUIPO (ID_EQUIPO , NOMBRE_EQUIPO, CATEGORIA, ID_CAMPEONATO)
VALUES ('equ-111', 'Google', 'Varones', 'camp-111')
INSERT INTO EQUIPO (ID_EQUIPO , NOMBRE_EQUIPO, CATEGORIA, ID_CAMPEONATO)
VALUES ('equ-222', '404_Not_Fund', 'Varones', 'camp-111')
INSERT INTO EQUIPO (ID_EQUIPO , NOMBRE_EQUIPO, CATEGORIA, ID_CAMPEONATO)
VALUES ('equ-333', 'Girls_Unifranz', 'Mujeres', 'camp-111')

INSERT INTO JUGADOR (ID_JUGADOR, NOMBRES, APELLIDOS, CI , EDAD , ID_EQUIPO)
VALUES ('jug-111', 'Carlos', 'Villa', '8997811LP', 19, 'equ-222')
INSERT INTO JUGADOR (ID_JUGADOR, NOMBRES, APELLIDOS, CI , EDAD , ID_EQUIPO)
VALUES ('jug-222', 'Pedro', 'Salas', '8997822LP', 20, 'equ-222')
INSERT INTO JUGADOR (ID_JUGADOR, NOMBRES, APELLIDOS, CI , EDAD , ID_EQUIPO)
VALUES ('jug-333', 'Saul', 'Araj', '8997833LP', 21, 'equ-222')
INSERT INTO JUGADOR (ID_JUGADOR, NOMBRES, APELLIDOS, CI , EDAD , ID_EQUIPO)
VALUES ('jug-444', 'Sandra', 'Solis', '8997844LP', 20, 'equ-333')
INSERT INTO JUGADOR (ID_JUGADOR, NOMBRES, APELLIDOS, CI , EDAD , ID_EQUIPO)
VALUES ('jug-555', 'Ana', 'Mica', '8997855LP', 23, 'equ-333')
```

# MANEJO DE CONCEPTOS

## 2.1. Muestra un ejemplo de DDL.

DDL (Lenguaje de Definición de Datos) es un conjunto de comandos SQL utilizados para definir la estructura de una base de datos. Aquí hay un ejemplo de una sentencia DDL:

```
CREATE TABLE CAMPEONATO  
(  
  ID_CAMPEONATO VARCHAR(20) PRIMARY KEY,  
  NOMBRE_CAMPEONATO VARCHAR(30),  
  SEDE VARCHAR(20)  
);
```



# MANEJO DE CONCEPTOS

## 2.2. Muestra un ejemplo de DML

DML (Lenguaje de Manipulación de Datos) es un conjunto de comandos SQL utilizados para manipular los datos en una base de datos relacional. Aquí hay algunos ejemplos de sentencias DML:

```
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')
```



## MANEJO DE CONCEPTOS

### 2.3. Para que Sirve INNER JOIN.

En resumen, el INNER JOIN en SQL se utiliza para combinar filas de dos o más tablas en una sola tabla, basándose en una columna común entre ellas. Solo devuelve las filas que tienen una coincidencia en ambas tablas.

### 2.4. Defina que es una función de agregación

Una función de agregación en SQL es una función que se utiliza para realizar cálculos sobre un conjunto de resultados y devolver un único valor agregado para todos ellos



# MANEJO DE CONCEPTOS

## 2.5. Liste funciones de agregación que conozca.

Estas funciones se utilizan comúnmente en consultas SELECT para realizar cálculos estadísticos y resumir datos. Se pueden utilizar en combinación con otras cláusulas SQL

## 2.6. Mencione algunas funciones propias de SQL-Server. .

SQL Server es un sistema de gestión de bases de datos relacionales desarrollado por Microsoft que cuenta con varias funciones propias





# MANEJO DE CONCEPTOS

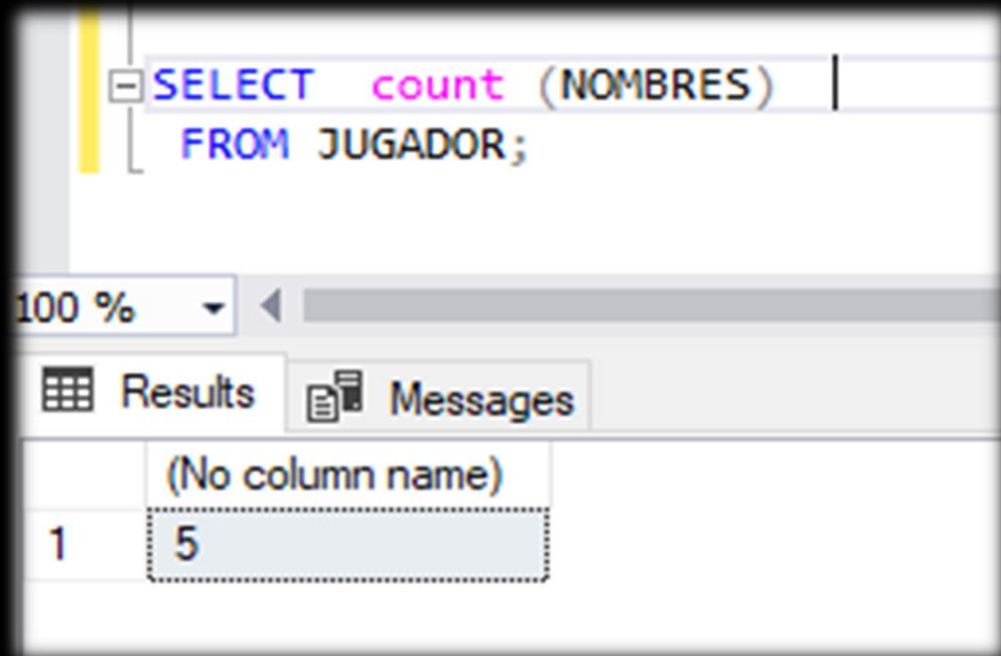
## 2.7. Para qué sirve la función CONCAT en SQL-Server.

En resumen, la función CONCAT en SQL Server se utiliza para concatenar dos o más cadenas de texto en una sola cadena. Se puede utilizar en combinación con otras funciones para convertir valores de diferentes tipos de datos en cadenas de texto antes de concatenarlos.



# MANEJO DE CONCEPTOS

## 2.8. Muestra un ejemplo del uso de COUNT.



The screenshot shows a SQL query editor with the following query:

```
SELECT count (NOMBRES)  
FROM JUGADOR;
```

Below the query, there is a results pane with a zoom level of 100%. The results pane has two tabs: "Results" and "Messages". The "Results" tab is active, showing a single row of data.

	(No column name)
1	5

# MANEJO DE CONCEPTOS

## 2.9. Muestra un ejemplo del usos de AVG

```
SELECT AVG (EDAD) AS promedio_de_edad  
FROM JUGADOR;
```

100 %

Results

Messages

promedio\_de\_edad

1

20

# MANEJO DE CONCEPTOS

## 2.10. Muestra un ejemplo del uso de MIN-MAX

```
SELECT MAX (EDAD)  
FROM JUGADOR;
```

100 %

Results Messages

	(No column name)
1	23

```
SELECT MIN (EDAD)  
FROM JUGADOR;
```

100 %

Results Messages

	(No column name)
1	19



# MANEJO DE CONSULTAS

## 3.1. Mostrar que jugadores que formen parte del equipo equ-333

```
--3.1. Mostrar que jugadores que formen parte del equipo equ-333
SELECT *
FROM JUGADOR
WHERE ID_EQUIPO = 'equ-333'

--3.2. Crear una función que permita saber cuántos jugadores están
```

Results		Messages				
	ID_JUGADOR	NOMBRES	APELLIDOS	CI	EDAD	ID_EQUIPO
1	jug-444	Sandra	Solis	8997844LP	20	equ-333
2	jug-555	Ana	Mica	8997855LP	23	equ-333

# MANEJO DECONSULTAS

3.2. Crear una función que permita saber cuántos jugadores están inscritos.

- La función debe llamarse Crear una función que permita saber cuántos jugadores están inscritos. ■

La función debe llamarse F1\_CantidadJugadores()

```
CREATE FUNCTION F1_CantidadJugadores()  
RETURNS INT  
BEGIN  
    DECLARE @respuesta INT =0;  
  
    SELECT @respuesta = COUNT(*)  
    FROM JUGADOR;  
    RETURN @respuesta;  
END;  
SELECT dbo.F1_CantidadJugadores() AS JUGADORES_INSCRITOS
```

--Crear una función que permita saber cuántos jugadores están

100 %

Results Messages

JUGADORES\_INSCRITOS

1

5

# MANEJO DECONSULTAS

- 3.3. Crear una función que permita saber cuántos jugadores están inscritos y que sean de la categoría varones o mujeres. ■ La función debe llamarse F2\_CantidadJugadoresParam()
- La función debe recibir un parámetro "Varones" o "Mujeres"

```
--Crear una función que permita saber cuántos jugadores están inscritos y que sean de la categoría varones o mujeres
CREATE FUNCTION F2_CantidadJugadoresParam(@CATEGORIA VARCHAR(8))
RETURNS INT
BEGIN
    DECLARE @respuesta INT = 0;
    SELECT @respuesta = COUNT(*)
    FROM JUGADOR AS j
    INNER JOIN EQUIPO AS a ON j.ID_EQUIPO = a.ID_EQUIPO
    WHERE CATEGORIA = @CATEGORIA;
    RETURN @respuesta;
END;

SELECT dbo.F2_CantidadJugadoresParam('Mujeres')
```

100 %

Results Messages

(No column name)

1

2



# MANEJO DECONSULTAS

- 3.4. Crear una función que obtenga el promedio de las edades mayores a una cierta edad. ■ La función debe llamarse F3\_PromedioEdades()
- La función debe recibir como parámetro 2 valores.
    - La categoría. (Varones o Mujeres)
    - La edad con la que se comparara (21 años ejemplo)
  - Es decir mostrar el promedio de edades que sean de una categoría y que sean mayores a 21 años.

# MANEJO DE CONSULTAS

```
CREATE FUNCTION F3_PromedioEdades(@CATEGORIA VARCHAR(10), @EDAD INT)
RETURNS FLOAT
BEGIN
    DECLARE @respuesta FLOAT=0;
    SELECT @respuesta = AVG(EDAD)
    FROM JUGADOR AS j
    INNER JOIN EQUIPO AS a ON j.ID_EQUIPO = a.ID_EQUIPO
    WHERE CATEGORIA = @CATEGORIA AND EDAD > @EDAD;
    RETURN @respuesta;
END;
SELECT dbo.F3_PromedioEdades('Mujeres',20)

--Crear una función que permita concatenar 3 parámetros.
```

100 %

Results Messages

	(No column name)
1	23

# MANEJO DECONSULTAS

**3.5. Crear una función que permita concatenar 3 parámetros.**

- La función debe llamarse `F4_ConcatItems()` 6
- La función debe de recibir 3 parámetros.
- La función debe de concatenar los 3 valores.
- Para verificar la correcta creación de la función debe mostrar lo siguiente.
- Mostrar los nombres de los jugadores, el nombre del equipo y la sede concatenada, utilizando la función que acaba de crear.



# MANEJO DE CONSULTAS

```
CREATE FUNCTION F4_ConcatItems(@param1 VARCHAR(50), @param2 VARCHAR(50), @param3 VARCHAR(50))  
RETURNS VARCHAR(150)  
AS  
BEGIN  
    DECLARE @result VARCHAR(150)  
    SET @result = CONCAT(@param1, ' ', @param2, ' ', @param3)  
    RETURN @result  
END  
SELECT dbo.F4_ConcatItems('Lionel Messi', 'Paris Saint-Germain', 'Paris, France')  
SELECT dbo.F4_ConcatItems('Neymar Jr.', 'Paris Saint-Germain', 'Paris, France')
```

100 %

Results Messages

(No column name)

1 Lionel Messi Paris Saint-Germain Paris, France

(No column name)

1 Neymar Jr. Paris Saint-Germain Paris, France

# MANEJO DECONSULTAS

3.6. Generar la serie fibonacci. ■ El objetivo es generar una función que retorne una cadena con la serie de la fibonacci. ● La función solo recibe el valor N. ● Comportamiento esperado

```
CREATE FUNCTION F5_Fibonacci(@N INT)
RETURNS VARCHAR(MAX)
AS
BEGIN
    DECLARE @Fibonacci VARCHAR(MAX) = '';
    DECLARE @a INT = 0;
    DECLARE @b INT = 1;
    DECLARE @c INT = 1;

    WHILE @c <= @N
    BEGIN
        SET @Fibonacci = @Fibonacci + CAST(@a AS VARCHAR(MAX)) + ', ';
        SET @b = @a + @b;
        SET @a = @b - @a;
        SET @c = @c + 1;
    END

    RETURN LEFT(@Fibonacci, LEN(@Fibonacci) - 2);
END;

SELECT dbo.F5_Fibonacci(6) AS SERIE_DE_FIBONACCI;
```

100 %

Results Messages

SERIE\_DE\_FIBONACCI

1	0, 1, 1, 2, 3,
---	----------------