IIC2413

AYUDANTÍA

6

PSQL y SQL

¿QUÉ VEREMOS?

¿QUE ES SQL?

SQL (Structured Query Language) es un lenguaje estándar definido por ANSI/ISO que se utiliza para gestionar y manipular bases de datos relacionales.

Para el curso se utiliza PostgresSQL que sigue fielmente este estandar, con pequeñas modificaciones

CREAR TABLAS

Sintaxis general:

```
CREATE  TABLE  nombre_tabla  (    name_column1
TIPO_DE_DATO  [CONSTRAINTS],    name_column2
TIPO_DE_DATO [CONSTRAINTS],
    ...
    [CONSTRAINTS]
    ...
);
```

INSERTAR EN TABLAS

Sintaxis general:

```
INSERT INTO nombre_tabla VALUES (
    valor_columna1,
    valor_columna2,
    valor_columna3,
    ...
);
```

CONSULTAR TABLAS

Sintaxis consulta basica:

SELECT columnas **FROM** tablas **WHERE** condicion;

MODIFICAR TABLAS

Eliminar tabla:

```
DROP TABLE nombre_tabla ;
```

Eliminar atributo:

```
ALTER TABLE nombre_tabla DROP COLUMN
nombre_columna;
```

Agregar atributo:

```
ALTER TABLE nombre_tabla ADD COLUMN
nombre_columna TIPO;
```

MODIFICAR DATOS

UPDATE: Actualizar valores de una tabla

Sintaxis general:

```
UPDATE nombre_tabla
SET
    columna1 = nuevo_valor1,
    columna2 = nuevo_valor2,
    ...
WHERE
    condicion;
```

ELIMINAR DATOS

DELETE FROM: Eliminar **filas** de una tabla

Sintaxis general:

DELETE FROM nombre_tabla WHERE condición;

PRODUCTO CRUZ

Nos permite consultar información de mas de una tabla a la vez

cursos

id	sigla	creditos
1	IIC1103	10
2	IIC2133	10
3	IIC1005	5
4	IIC100B	5
5	IIC2413	10

estudiantes

id	nombre	edad
2100	Felipe	22
2101	Alvaro	20
2102	Javiera	21
2103	Matias	19
2104	Laura	22

inscripciones

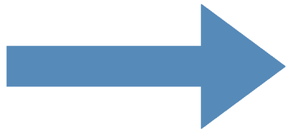
estudiante_id	curso_id	fecha_inscripcion
2100	1	'2025-01-06'
2100	3	'2025-01-05'
2101	5	'2025-01-04'
2102	3	'2025-01-01'
2104	2	'2025-01-03'
2104	1	'2025-01-01'

PRODUCTO CRUZ

id	nombre	edad
2100	Felipe	22
2101	Alvaro	20
2102	Javiera	21
2103	Matias	19
2104	Laura	22

```
SELECT * FROM
estudiantes, inscripciones
WHERE estudiantes.id =
inscripciones.estudiantes_id
;
```

estudiante_id	curso_id	fecha_inscripcion
2100	1	'2025-01-06'
2100	3	'2025-01-05'
2101	5	'2025-01-04'
2102	3	'2025-01-01'
2104	2	'2025-01-03'
2104	1	'2025-01-01'



id	nombre	edad	estudiante_id	curso_id	fecha_inscripcion
2100	Felipe	22	2100	1	'2025-01-06'
2100	Felipe	22	2100	3	'2025-01-05'
2101	Alvaro	20	2101	5	'2025-01-04'
2102	Javiera	21	2102	3	'2025-01-01'
2104	Laura	22	2104	2	'2025-01-03'
2104	Laura	22	2104	1	'2025-01-01'

PRODUCTO CRUZ

```
SELECT * FROMestudiantes,cursos,inscripciones
WHERE estudiantes.id = inscripciones.estudiante_id
AND cursos.id = inscripciones.curso_id;
```

estudiantes.id	nombre	edad	cursos.id	sigla	creditos	estudiante_id	curso_id	fecha_inscripcion
2100	Felipe	22	1	IIC1103	10	2100	1	'2025-01-06'
2100	Felipe	22	3	IIC1005	5	2100	3	'2025-01-05'
2101	Alvaro	20	5	IIC2413	10	2101	5	'2025-01-04'
2102	Javiera	21	3	IIC1005	5	2102	3	'2025-01-01'
2103	Matias	19	2	IIC2133	10	2104	2	'2025-01-03'
2104	Laura	22	1	IIC1103	10	2104	1	'2025-01-01'

PRODUCTO CRUZ

SELECT estudiantes.id, nombre, sigla, fecha_inscripcion **FROM** estudiantes, cursos, inscripciones **WHERE** estudiantes.id = inscripciones.estudiante_id **AND** cursos.id = inscripciones.curso_id;

estudiantes.id	nombre	sigla	fecha_inscripcion
2100	Felipe	IIC1103	'2025-01-06'
2100	Felipe	IIC1005	'2025-01-05'
2101	Alvaro	IIC2413	'2025-01-04'
2102	Javiera	IIC1005	'2025-01-01'
2103	Matias	IIC2133	'2025-01-03'
2104	Laura	IIC1103	'2025-01-01'

PRODUCTO CRUZ (JOIN)

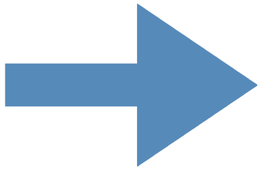
```
SELECT estudiantes.id, nombre, sigla, fecha_inscripcion FROM
inscripciones
JOIN estudiantes ON inscripciones.estudiante_id = estudiantes.id
JOIN cursos ON inscripciones.curso_id = cursos.id ;
```

estudiantes.id	nombre	sigla	fecha_inscripcion
2100	Felipe	IIC1103	'2025-01-06'
2100	Felipe	IIC1005	'2025-01-05'
2101	Alvaro	IIC2413	'2025-01-04'
2102	Javiera	IIC1005	'2025-01-01'
2103	Matias	IIC2133	'2025-01-03'
2104	Laura	IIC1103	'2025-01-01'

GROUP BY

```
SELECT curso, AVG(nota)as promedioFROMevaluaciones WHERE
fecha >'05-03-2025'GROUP BYcurso;
```

id	curso	estudiante	nota	fecha_evaluaciones
1	Álgebra	Ana	5	01-03-2025
2	Álgebra	Tomás	4.5	03-03-2025
3	Programación	Ana	6	05-03-2025
4	Programación	Camila	5.5	10-03-2025
6	Física	Diego	4	15-03-2025
7	Física	Camila	4.5	18-03-2025



curso	promedio
Programación	5.5
Física	4.25

ORDER BY

Ordenar resultadossegúnunaomáscolumnas.

```
SELECT nombre, edad
FROM futbolistas
ORDER BY edad ASC;
```

LIKE

Sirve para comparar strings
Utiliza los siguientes símbolos:

- "%" : Cualquier secuencia de caracteres
- "_" : Cualquier Caracter (uno)

```
SELECT nombre FROM futbolistas
WHERE club LIKE '%Barcelona%' ;
```

HAVING

```
SELECT curso, AVG(nota)as promedio FROM evaluaciones WHERE
fecha >'05-03-2025'GROUP BYcurso HAVING AVG(nota)>5.5;
```

HAVING

Se usa para filtrar resultados despues de agruparlos con **GROUP BY**

- **WHERE** filtra antes del agrupamiento
- **HAVING** filtra despues del agrupamiento (Sobre los grupos ya formados)

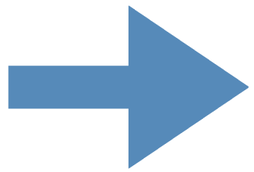
AGREGACION

Funciones queresumeno agrupandatos.

SUM, MIN, MAX, COUNT, AVG, etc

SELECT SUM(cantidad) **FROM** ventas **WHERE** precio >15;

id	producto	cantidad	precio
1	Agua	24	10
2	Bebida	23	20
3	Leche	25	30



sum
48

UNION

Une resultados sin duplicados

Las filas repetidas de “Elena” y “Raimundo” no se toman en cuenta

nombre
María
Francisco
Elena
Raimundo

```
SELECT nombre FROM gimnasta
UNION
SELECT nombre FROM juez;
```

UNION ALL

Une resultados (admite duplicados)

nombre
María
Elena
Raimundo
Francisco
Elena
Raimundo

```
SELECT nombre FROM gimnasta
UNION ALL
SELECT nombre FROM juez;
```

EXCEPT

"Muéstrame la tablaAmenosloquetambién están en la tabla B"

Gimnasta

id	nombre	edad
1	María	24
2	Elena	23
3	Raimundo	25
4	Tomás	26

Juez

id	nombre	edad
1	Francisco	32
2	Elena	40
3	Raimundo	29
4	Agustin	35

```
SELECT nombre FROM gimnasta
EXCEPT
SELECT nombre FROM juez;
```

EXCEPT

"Muéstrame la tablaAmenosloquetambién está en la tabla B"

“María” y “Tomás” son los nombres que estan en la tabla “A” que no estan en “B”

nombre
María
Tomás

SELECT nombre **FROM** gimnasta
EXCEPT
SELECT nombre **FROM** juez;

!! El orden es importante!!

Muestra las filas de la primera consulta que no están en la segunda

INTERSECT

Devuelve solo lasfilas queaparecenenambas consultas, es decir, la intersección entredosconjuntosde resultados.

“Elena” y “Raimundo” aparecen en ambas tablas

nombre
Elena
Raimundo

```
SELECT nombre FROM gimnasta
INTERSECT
SELECT nombre FROM juez;
```

CONSULTAS ANIDADAS

Como Condición

```
SELECT bandas.nombre FROM bandas,
estudiantes WHERE bandas.vocalista =
estudiantes.nombre AND bandas.nombre
IN (
SELECT toco_en.nombre_banda FROM
toco_en WHERE toco_en.nombre_festival =
'Lollapalooza'
);
```

CONSULTAS ANIDADAS

Como Condición

Tipos de condiciones

- s **IN** R
- s > **ALL** R
- s > **ANY** R
- **EXISTS** R

CONSULTAS ANIDADAS

Como Condición

```
SELECT nombre FROM estudiantes WHERE nombre
IN (SELECT vocalista
FROM bandas);
```

Devuelve estudiantes que son vocalistas de alguna banda.

```
SELECT nombre, calificacion FROM pelicula WHERE calificacion >
ALL (SELECT calificacion FROM pelicula
WHERE año = 2020);
```

Películas cuya calificación es mayor a todas las películas del 2020.
(Si se usa ANY, seria mayor que alguna película del 2020)

```
SELECT nombre FROM clientes WHERE
EXISTS (
SELECT * FROM compras
WHERE compras.id_cliente = clientes.id);
```

Para cada cliente de la tabla clientes, busca si hay al menos una fila en compras que tenga el mismo id_cliente
Si existe → el cliente sí ha comprado algo → se incluye en el resultado.

CONSULTAS ANIDADAS

Como Joins

profesores

id	nombre
1	Carolina
2	Javier
3	Francisca
4	Rodrigo

cursos

id	nombre_curso	año
10	Bases de Datos	2021
11	Álgebra	2020
12	Redes	2022
13	Programacion	2020
14	IA	2023

hizo_clase

id_profesor	id_curso
1	10
1	14
2	11
2	12
3	13
4	12

CONSULTAS ANIDADAS

Como Joins

```
SELECT profesores.nombre, maximos.año
FROM profesores, (
        SELECT hizo_clases.id_profesor AS id,
        MAX(curso.año) AS año FROM hizo_clase, cursos
        WHERE hizo_clase.id_curso=cursos.id
        GROUP BY hizo_clase.id_profesor
) AS maximos
WHERE profesores.id=maximos.id
```

CONSULTAS ANIDADAS

Como Joins

SELECT hizo_clases.id_profesor**AS**id,**MAX**(curso.año) **AS** año **FROM** hizo_clase, cursos **WHERE** hizo_clase.id_curso=cursos.id **GROUP BY** hizo_clase.id_profesor

id_profesor	id_curso	año
1	10	2021
1	14	2023
2	11	2020
2	12	2022
3	13	2020
4	12	2022



id	año
1	2023
2	2022
3	2020
4	2022

CONSULTAS ANIDADAS

Como Joins

```
SELECT profesores.nombre, maximos.año
FROM profesores, (
) AS maximos
WHERE profesores.id=maximos.id
```

maximos

id	año
1	2023
2	2022
3	2020
4	2022



nombre	año
Carolina	2023
Javier	2022
Francisca	2020
Rodrigo	2022

NULOS

Cuando no se sabe siel valor existeonoexiste, o simplemente no se tiene lainformacion

id	marca	modelo	patente	dueño_id
1	Toyota	Supra	ABCD12	101
2	Nissan	Sentra	EFGH34	NULL
3	Chevrolet	Silverado	IJKL56	242
4	Tesla	Model 3	NULL	123

NULOS

Condicionparaversielelemento es nulo

SELECT * **FROM** R **WHERE** R.b **IS NULL**

Condicion para ver si el elemento no es nulo

SELECT * **FROM** R **WHERE** R.b **IS NOT NULL**

Las consultas con operaciones (+, -, *, etc) con NULL se vuelven nulas

Las consultas de agregacion (SUM, MIN, COUNT) ignoran los NULL

PSQL

Conexión a Postgres

1. Por defecto para conectarse a postgres se utiliza el siguiente comando

```
psql
```

4. Para ejecutar un archivo SQL

```
psql -f archivo.sql
```

7. Se pueden combinar para que reciba más parámetros

```
psql -U usuario -d nombre_db -h host.cl -f archivo.sql
```

PSQL

Comandos de PSQL:

4. Para ver las tablas de la base de datos actual

```
\dt
```

5. Para ver los atributos que recibe y las IC de las tablas

```
\d nombre_tabla
```

6. Para salir de postgres

```
\q
```

PSQL

Poblar tablas:

Poblar Tablas

```
-- Se crea una tabla temporal
CREATE TEMP TABLE jugadores_tmp (
    id INTEGER,
    nombre TEXT,
    edad INTEGER,
    club TEXT
);

-- Se cargan los datos desde el csv
\copy jugadores_tmp FROM 'csv/players_22.csv' DELIMITER ',' CSV HEADER;

-- Descartamos nulos
\copy (
    SELECT *
    FROM jugadores_tmp
    WHERE id IS NULL OR nombre IS NULL
) TO 'descartados/jugadores_descartados.csv' CSV HEADER;

-- Insertamos los datos validos en una tabla definitiva
INSERT INTO jugadores (id, nombre, edad, club)
SELECT DISTINCT ON (id, nombre)
    id, nombre, edad, club
FROM jugadores_tmp
WHERE id IS NOT NULL
  AND nombre IS NOT NULL;

```

PSQL

Crear y Ejecutar

archivo.sql

Crean archivo.sql con consulta:

Ejecutar un archivo

```sql
SELECT nombre
FROM jugadores
WHERE preferred_foot LIKE '%left%';
```

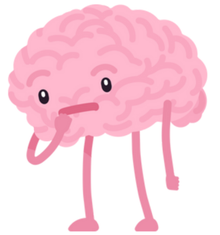
Suben archivo al servidor:

```
scp {nombre_archivo} {usuario_uc}.bdd@stonebraker.ing.uc.cl:{ruta_destino}
```

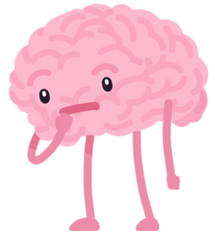
Ejecutan archivo en el servidor:

```
psql -U usuario -d nombre_db -h host.cl -f archivo.sql
```

EJERCICIOS



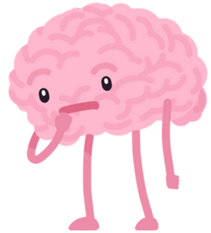
EJERCICIOS



- **jugadores**(sofifa_id, nombre, nombre_completo, precio_eur, edad, nacimiento, altura, peso, club_id, liga_id, numero, fecha_ingreso, nacionalidad_id, pie)
- **ligas**(liga_id, nombre)
- **clubes**(club_id, nombre, liga_id)
- **nacionalidades**(nacionalidad_id, nombre)
- **posiciones**(posicion_id, nombre)
- **jugador_posiciones**(jugador_id, posicion_id)

sofifa_id	nombre	nombre_completo	posiciones	precio_eur	edad	nacimiento	altura	peso	club	liga	numero	fecha_ingreso	nacionalidad	pie
158023	L. Messi	Lionel Andrés Messi Cuccittini	RW, ST, CF	78000000.0	34	1987-06-24	170	72	Paris Saint-Germain	French Ligue 1	30	2021-08-10	Argentina	Left
188545	R. Lewandowski	Robert Lewandowski	ST	119500000.0	32	1988-08-21	185	81	FC Bayern München	German 1. Bundesliga	9	2014-07-01	Poland	Right
20801	Cristiano Ronaldo	Cristiano Ronaldo dos Santos Aveiro	ST, LW	45000000.0	36	1985-02-05	187	83	Manchester United	English Premier League	7	2021-08-27	Portugal	Right
190871	Neymar Jr	Neymar da Silva Santos Júnior	LW, CAM	129000000.0	29	1992-02-05	175	68	Paris Saint-Germain	French Ligue 1	10	2017-08-03	Brazil	Right
192985	K. De Bruyne	Kevin De Bruyne	CM, CAM	125500000.0	30	1991-06-28	181	70	Manchester City	English Premier League	17	2015-08-30	Belgium	Right

CREACION TABLAS



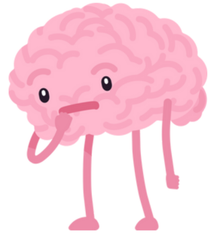
1.Crear tabla ligas con liga_id(PK) y nombre

2.Crear tabla clubes con club_id(PK), nombre y liga_id(FK)

3.Crear tabla nacionalidades con nacionalidad_id (PK) y nombre

4.Crear la tabla posiciones donde un jugador puede jugar en varias posiciones (Este solo explicarlo)

CONSULTAS



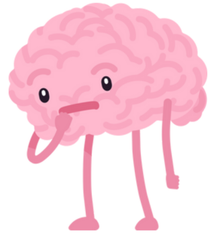
1a.TOP 10 JUGADORES MAS CAROS: muestra en orden descendiente los jugadores en los cuales su precio sea mayor.

1b.TOP 10 CLUBES MAS CAROS: muestra en orden descendiente los clubes en los cuales la sumatoria de los precios de sus jugadores sea mayor.

2.TOP 10 DIAS QUE MAS GANACIA A GENERADO: muestra en orden descendiente los dias en los cuales la suma total de los precios de los jugadores nacidos ese dia sea mayor.

3.TOP 25 JUGADORES QUE JUEGAN MAS DE UNA POSICION: muestra en orden descendiente los 25 mejores jugadores (segun precio de mercado) que jueguen mas de una posicion, en caso de empate compara por edad ascendente.

CONSULTAS



4.TOP 12 MESES CON MAS JUGADORES: muestra en orden descendiente los meses en los cuales mas jugadores han nacido.

5. EQUIPOS CON MAS JUGADORES NACIDOS EL MISMO MES: muestra el top 10 de equipos donde mas jugadores esten de cumpleaños el mismo mes.

6. CLUBES CON MAYOR VALOR Y EDAD PROMEDIO: muestra el top 10 de equipos con mercado y edad mas altos.