



IIC2413

AYUDANTÍA 6

SQL 1





¿QUÉ VEREMOS?



¿Que es SQL?

Tipo de Datos

Administrar Tablas

Manejo de Datos

Consulta de Datos



¿QUE ES SQL?

Lenguaje de Consultas Estructurado:

Es un lenguaje especializado en el manejo de bases de datos relacionales

Nos permite:

- Crear y Administrar Tablas
- Agregar Datos
- Modificar Datos
- Eliminar Datos
- Consultar Datos

¿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 PostgreSQL que sigue fielmente este estandar, con pequeñas modificaciones

TIPOS DE DATOS

Strings

Tipo	Descripcion	Ejemplo
CHAR(n)	Cadena de longitud fija	saludo CHAR(10) → 'Hola '
VARCHAR(n)	Cadena de longitud variable hasta un maximo de n	nuevo_saludo VARCHAR(10) → 'Hola'
TEXT	Texto sin limites	saludo_final TEXT → 'Hola Mundo'

TIPOS DE DATOS

Numericos

Tipo	Descripcion	Ejemplo
INT	Numero entero (4 bytes)	edad INT → 25
BIGINT	Numero entero grande (8 bytes)	visitas BIGINT → 10923874562
FLOAT	Numero decimal no preciso (8 bytes)	temperatura FLOAT→ 36.6
NUMERIC(p,s)	Numero decimal con precision exacta con p digitos en total y s digitos en la parte decimal	precio NUMERIC(6,2) → 4999.99

TIPOS DE DATOS

Otros

Tipo	Descripcion	Ejemplo
BOOLEAN	Valores logicos: TRUE o FALSE	activo BOOLEAN → TRUE
DATE	Fecha (año, mes, dia)	nacimiento DATE → '2025-04-01'
TIME	Hora (hora, minuto, segundo)	hora_inicio TIME → '14:30:00'
TIMESTAMP	Fecha y hora combinadas	creado TIMESTAMP → '2025-04-01 14:30:00'

CREAR TABLAS

Sintaxis general:

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

TIPOS DE CONSTRAINTS

Constrain	Descripcion	Ejemplo
PRIMARY KEY	Columna con PK	id INT PRIMARY KEY
UNIQUE	El dato debe ser único en la columna	nombre VARCHAR(20) UNIQUE
NOT NULL	El dato no puede ser NULL en la columna	edad INT NOT NULL
DEFAULT valor	Se le asigna un valor por default cuando no se indica valor	puntos INT DEFAULT 100
FOREIGN KEY	Establece una relación con una columna de otra tabla (llave foránea) Importante: la columna referenciada debe tener como restricción PRIMARY KEY	dept_id INT REFERENCES departamentos(id)

También se pueden combinar constraints como:
nombre **VARCHAR(20) UNIQUE NOT NULL**

CREAR TABLAS

id	nombre	edad
...

```
CREATE TABLE estudiantes (  
  id INT PRIMARY KEY,  
  nombre VARCHAR(100),  
  edad INTEGER  
);
```

id	sigla	creditos
...

```
CREATE TABLE cursos (  
  id INT,  
  sigla VARCHAR(10),  
  creditos INTEGER,  
  PRIMARY KEY (id)  
);
```

CREAR TABLAS

estudiante_id	curso_id	fecha_inscripcion
...

```
CREATE TABLE inscripciones (  
  estudiante_id INT REFERENCES estudiantes(id),  
  curso_id INT,  
  fecha_inscripcion DATE,  
  PRIMARY KEY (estudiante_id, curso_id),  
  FOREIGN KEY (curso_id) REFERENCES cursos(id)  
);
```

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;
```

INSERTAR EN TABLAS

Sintaxis general:

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

INSERTAR EN TABLAS

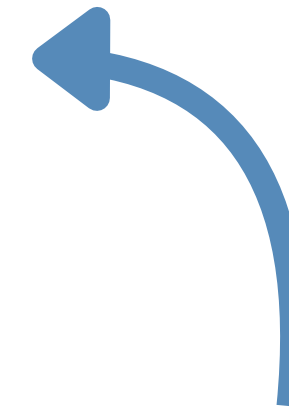
Sintaxis general (segunda versión):

```
INSERT INTO nombre_tabla (  
    columna1,  
    columna2,  
    columna3,  
    ...  
) VALUES (  
    valor1,  
    valor2,  
    valor3,  
    ...  
);
```

INSERTAR EN TABLAS

estudiantes

id	nombre	edad
1	David	20
2	Gabriel	22



INSERT INTO **estudiantes** **VALUES** (2, 'Gabriel', 22);

INSERTAR EN TABLAS

estudiantes

id (PK)	nombre	edad
1	Felipe	19

cursos

id (PK)	sigla	creditos
1	IIC2413	10

Inscripciones

estudiante_id (PK) (FK)	curso_id (PK) (FK)	fecha_inscripcion
1	1	'02-04-2025'
2	1	'22-04-2025'

Qué pasa si hacemos? :

INSERT INTO **inscripciones** **VALUES** (2, 1, '22-04-2025');

INSERTAR EN TABLAS

estudiantes

id (PK)	nombre	edad
1	David	20

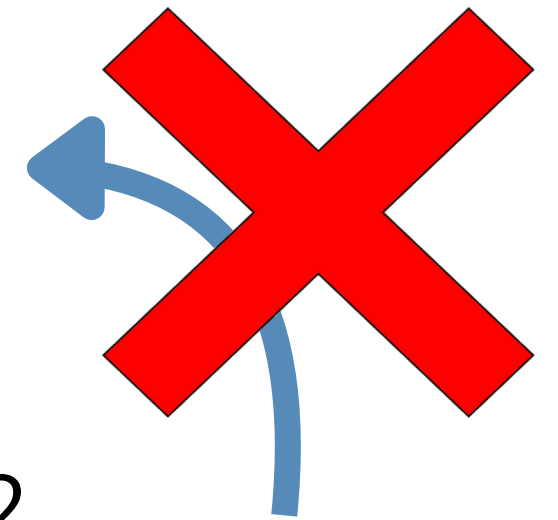
cursos

id (PK)	sigla	creditos
1	IIC2413	10

Inscripciones

estudiante_id (PK) (FK)	curso_id (PK) (FK)	fecha_inscripcion
1	1	'02-04-2025'
2	1	'22-04-2025'

Hay **error** debido a que no hay estudiante con id 2



MODIFICAR DATOS

UPDATE: Actualizar valores de una tabla

Sintaxis general:

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

MODIFICAR DATOS

UPDATE **cursos** SET **sigla** = 'IIC1253' WHERE id = 1;

cursos

id (PK)	sigla	creditos
1	IIC2413	10



cursos

id (PK)	sigla	creditos
1	IIC1253	10

MODIFICAR DATOS

estudiantes

id (PK)	nombre	edad
1	Felipe	19

cursos

id (PK)	sigla	creditos
1	IIC2413	10

Inscripciones

estudiante_id (FK)	curso_id (FK)	fecha_inscripcion
1	1	'02-04-2025'

Qué pasa si hacemos? :

UPDATE **estudiantes** **SET** id = 2 **WHERE** nombre = 'Felipe';

MODIFICAR DATOS

estudiantes

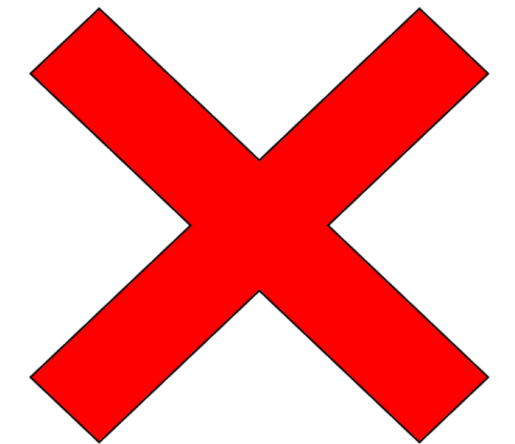
id (PK)	nombre	edad
1	Felipe	19

cursos

id (PK)	sigla	creditos
1	IIC2413	10

Inscripciones

estudiante_id (PK) (FK)	curso_id (PK) (FK)	fecha_inscripcion
1	1	'02-04-2025'



Hay **error** debido a que la llave 1 sigue referenciada en **inscripciones**

MODIFICAR DATOS

Para que las FK cambien si se modifica una PK debemos definir, en la creación de la tabla, una **ACCIÓN** que se hará sobre ella al modificar una FK

Esta acción puede ser **CASCADE** (propagar la modificación)

ANTES:

```
CREATE TABLE inscripciones (  
    estudiante_id INT REFERENCES estudiantes(id),  
    curso_id INT,  
    fecha_inscripcion DATE,  
    PRIMARY KEY (estudiante_id, curso_id),  
    FOREIGN KEY (curso_id) REFERENCES cursos(id)  
);
```

MODIFICAR DATOS

Para que las FK cambien si se modifica una PK debemos definir, en la creación de la tabla, una **ACCIÓN** que se hará sobre ella al modificar una FK

Esta acción puede ser **CASCADE** (propagar la modificación)

DESPUÉS:

```
CREATE TABLE inscripciones (  
  estudiante_id INT REFERENCES estudiantes(id) ON UPDATE CASCADE,  
  curso_id INT,  
  fecha_inscripcion DATE,  
  PRIMARY KEY (estudiante_id, curso_id),  
  FOREIGN KEY (curso_id) REFERENCES cursos(id)  
);
```


MODIFICAR DATOS

estudiantes

id (PK)	nombre	edad
2	Felipe	19

cursos

id (PK)	sigla	creditos
1	IIC2413	10

Inscripciones

estudiante_id (PK) (FK)	curso_id (PK) (FK)	fecha_inscripcion
2	1	'02-04-2025'

Ahora si cambia las FK

UPDATE **estudiantes** **SET** id = 2 **WHERE** nombre = 'Felipe';



ELIMINAR DATOS

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

Sintaxis general:

DELETE FROM **nombre_tabla** **WHERE** condición;

ELIMINAR DATOS

DELETE FROM cursos **WHERE** sigla = 'IIC2413';

cursos

id (PK)	sigla	creditos
1	IIC2413	10
2	IIC2133	10



cursos

id (PK)	sigla	creditos
2	IIC2133	10

ELIMINAR DATOS

estudiantes

id (PK)	nombre	edad
1	Felipe	19

cursos

id (PK)	sigla	creditos
1	IIC2413	10

Inscripciones

estudiante_id (PK) (FK)	curso_id (PK) (FK)	fecha_inscripcion
1	1	'02-04-2025'

Qué pasa si hacemos? :

DELETE FROM **estudiantes** **WHERE** nombre = 'Felipe';

ELIMINAR DATOS

estudiantes

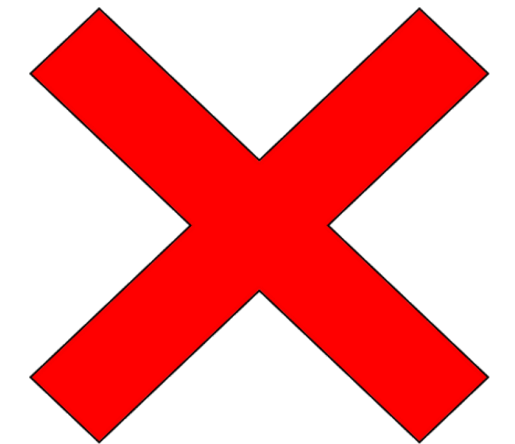
id (PK)	nombre	edad
1	Felipe	19

cursos

id (PK)	sigla	creditos
1	IIC2413	10

Inscripciones

estudiante_id (PK) (FK)	curso_id (PK) (FK)	fecha_inscripcion
1	1	'02-04-2025'



Hay **error** debido a que la llave 1 sigue referenciada en **inscripciones**

ELIMINAR DATOS

Para poder manejar la situación en donde se elimina un elemento que esta referenciado, debemos definir, en la creación de la tabla, una **ACCIÓN** que se hará sobre ella al elimina una FK

Esta acción puede ser **CASCADE** (propagar la eliminacion)

ANTES:

```
CREATE TABLE inscripciones (  
    estudiante_id INT REFERENCES estudiantes(id),  
    curso_id INT,  
    fecha_inscripcion DATE,  
    PRIMARY KEY (estudiante_id, curso_id),  
    FOREIGN KEY (curso_id) REFERENCES cursos(id)  
);
```

ELIMINAR DATOS

Para que las FK cambien si se modifica una PK debemos definir, en la creación de la tabla, una **ACCIÓN** que se hará sobre ella al modificar una FK

Esta acción puede ser **CASCADE** (propagar la modificación)

DESPUÉS:

```
CREATE TABLE inscripciones (  
  estudiante_id INT REFERENCES estudiantes(id) ON DELETE CASCADE,  
  curso_id INT,  
  fecha_inscripcion DATE,  
  PRIMARY KEY (estudiante_id, curso_id),  
  FOREIGN KEY (curso_id) REFERENCES cursos(id)  
);
```

ELIMINAR DATOS

estudiantes

id (PK)	nombre	edad
---------	--------	------

cursos

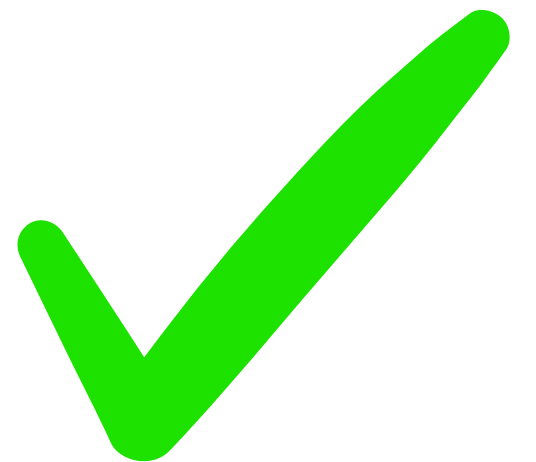
id (PK)	sigla	creditos
1	IIC2413	10

Inscripciones

estudiante_id (PK) (FK)	curso_id (PK) (FK)	fecha_inscripcion
-------------------------	--------------------	-------------------

Ahora si se elimina en ambas

DELETE FROM **estudiantes** **WHERE** nombre = 'Felipe';



CONSULTAR TABLAS

Sintaxis consulta basica:

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

CONSULTAR TABLAS

SELECT * **FROM** cursos **WHERE**
creditos = 5;

cursos

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



id	sigla	creditos
3	IIC1005	5
4	IIC100B	5

CONSULTAR TABLAS

SELECT nombre **FROM**
estudiantes **WHERE** edad > 20;

estudiantes

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



nombre
Felipe
Javiera
Laura

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 * **FROM** estudiantes, 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;

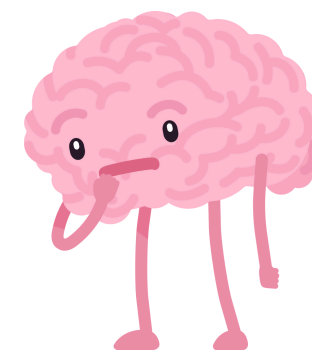
estudiantes.id	nombre	sigla	fecha_inscripcion
2100	Felipe	IIC1103	'2025-01-06'
2100	Felipe	IIC1005	'2025-01-05'
2101	Alvaro	IIC2413	'2025-01-04'
2102	Javiera	IIC1005	'2025-01-01'
2103	Matias	IIC2133	'2025-01-03'
2104	Laura	IIC1103	'2025-01-01'

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 ;

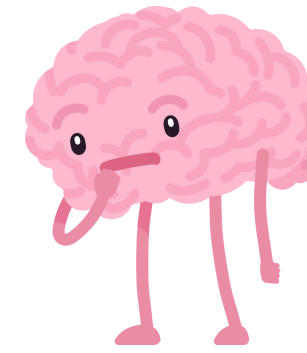
estudiantes.id	nombre	sigla	fecha_inscripcion
2100	Felipe	IIC1103	'2025-01-06'
2100	Felipe	IIC1005	'2025-01-05'
2101	Alvaro	IIC2413	'2025-01-04'
2102	Javiera	IIC1005	'2025-01-01'
2103	Matias	IIC2133	'2025-01-03'
2104	Laura	IIC1103	'2025-01-01'

EJERCICIO



A continuación se describe un esquema de base de datos para gestionar campeonatos de gimnasia rítmica, considerando la participación individual y en conjunto, así como los instrumentos utilizados en las competiciones individuales.

EJERCICIO



Tablas relevantes:

Competencia (id_competencia (PK), nombre, fecha, lugar)

Gimnasta (id_gimnasta (PK), nombre, fecha_nacimiento, país, id_equipo (FK))

Equipo (id_equipo (PK), nombre, país)

Conjunto (id_conjunto (PK), nombre, id_equipo (FK))

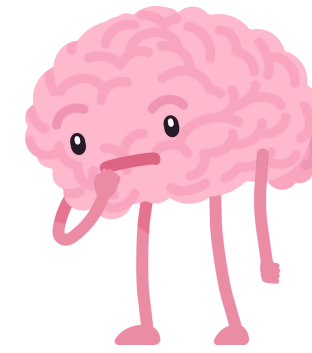
Gimnasta_Conjunto (id_gimnasta, id_conjunto (PK compuesta, ambas FKs))

Participación (id_participacion (PK), id_competencia (FK), id_gimnasta (FK, puede ser NULL), id_conjunto (FK, puede ser NULL), tipo_participacion ('individual' o 'conjunto'), instrumento ('pelota', 'clavas', 'cinta', 'manos libres', puede ser NULL), puntaje)

Juez (id_juez (PK), nombre, país)

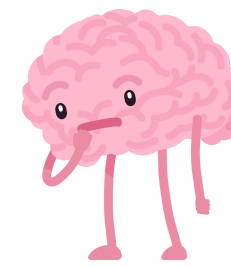
Evaluación (id_evaluacion (PK), id_participacion (FK), id_juez (FK), puntaje, comentarios)

EJERCICIO



a) Cree la tabla "participacion" usando SQL, recuerde incluir la llave primaria, las llaves foraneas y el tipo de dato.

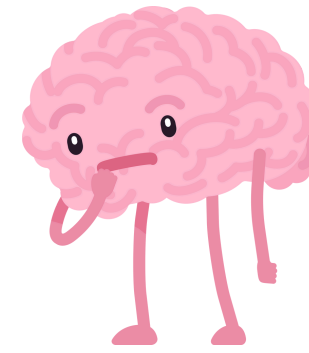
SOLUCION



a) Cree la tabla "participacion" usando SQL, recuerde incluir la llave primaria, las llaves foraneas y el tipo de dato.

```
CREATE TABLE participacion (  
  id_participacion SERIAL PRIMARY KEY,  
  id_competencia INT REFERENCES competencia(id_competencia),  
  id_gimnasta INT REFERENCES gimnasta(id_gimnasta),  
  id_conjunto INT REFERENCES conjunto(id_conjunto),  
  tipo_participacion VARCHAR(20),  
  instrumento VARCHAR(20),  
  puntaje NUMERIC(5,2)  
);
```

EJERCICIO

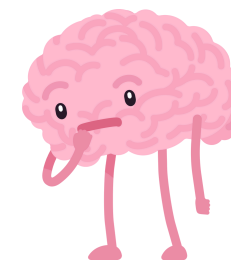


b) Genere una consulta SQL que obtenga el puntaje de cada participación individual en la competencia `id_competencia = 4`.

La consulta debe incluir:

- Nombre del equipo
- Tipo de participación
- Instrumento usado

SOLUCION



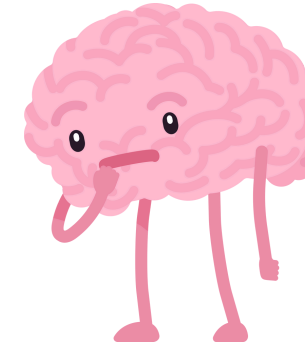
b) Genere una consulta SQL que obtenga el puntaje de cada participación individual en la competencia `id_competencia = 4`.

La consulta debe incluir:

- Nombre del equipo
- Tipo de participación
- Instrumento usado

```
SELECT equipo.nombre,  
        participacion.tipo_participacion,  
        participacion.instrumento,  
        participacion.puntaje  
FROM participacion  
JOIN gimnasta ON participacion.id_gimnasta = gimnasta.id_gimnasta  
JOIN equipo ON gimnasta.id_equipo = equipo.id_equipo  
WHERE participacion.id_competencia = 4  
AND participacion.tipo_participacion = 'individual';
```

EJERCICIO

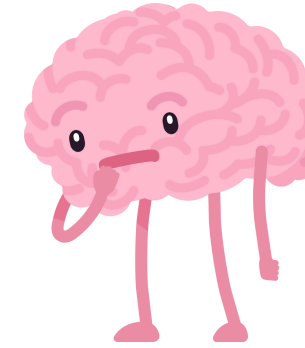


En una empresa de e-commerce tienen el siguiente esquema para la capacidad de despacho de los productos que venden:

- **zona_despacho:** (id_zona: serial, nombre_zona: string, comuna: string, region: string)
- **slot:** (id_slot: serial, dia: string, hora_inicio: time, hora_fin: time)
- **asig_capac:** (id_capac: serial, id_zona: int, id_slot: int, capac_sm: int, capac_med: int, capac_big: int, precio_sm: int, precio_med: int, precio_big: int)

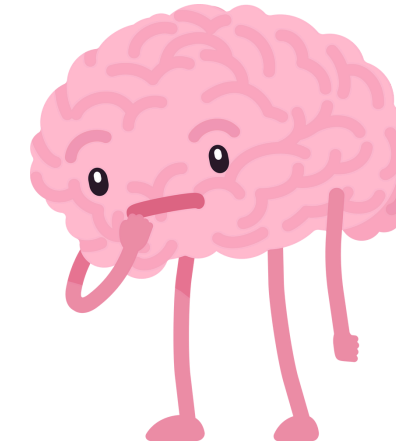
- zona_despacho: corresponde a un área dentro de una comuna y región
- slot: corresponde al rango de tiempo para despachar en un día específico (ej: miércoles de 3 a 6)
- asig_capac: corresponde a la capacidad de entregar productos de tamaño **pequeño, mediano y grande** (**capac_sm, capac_med, capac_big**) junto a sus precios respectivos

EJERCICIO



- a)** Cree las tablas asociada al esquema del enunciado. Especifique las llaves primarias, foraneas y los tipos de datos.
- b)** Escriba una instrucción SQL que inserte un nuevo registro en la tabla "asig_capac", asociando la zona con id 2 y el slot con id 5. Suponga que la capacidad para productos es 10 pequeños, 5 medianos, y 2 grandes, y que los precios son 2000, 3500 y 6000 respectivamente
- c)** Escriba una consulta SQL que entregue los nombres de todas las zonas de despacho que están en la región "Metropolitana".
- d)** Escriba una consulta SQL que entregue, por cada zona de despacho, los rangos horarios (hora_inicio y hora_fin) de los slots que tienen una capacidad mayor a 5 productos grandes ($\text{capac_big} > 5$), junto al nombre de la zona y el día del slot.

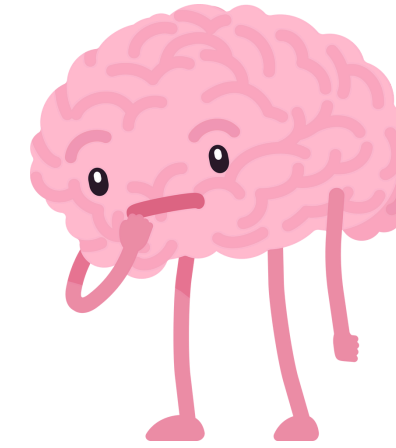
SOLUCION



a) Cree las tablas asociada al esquema del enunciado. Especifique las llaves primarias, foraneas y los tipos de datos.

```
CREATE TABLE zona_despacho(  
    id_zona SERIAL PRIMARY KEY,  
    nombre_zona VARCHAR(50),  
    comuna VARCHAR(50),  
    region VARCHAR(50),  
);
```

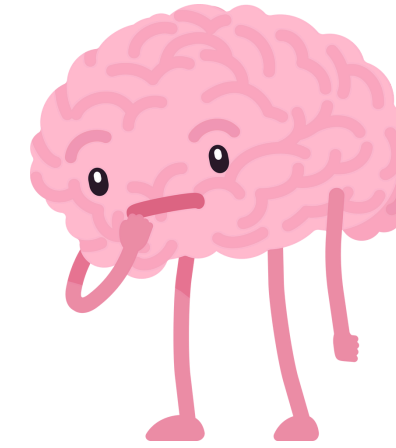
SOLUCION



a) Cree las tablas asociada al esquema del enunciado. Especifique las llaves primarias, foraneas y los tipos de datos.

```
CREATE TABLE slot(  
    id_slot SERIAL PRIMARY KEY,  
    dia VARCHAR(25),  
    hora_inicio TIME,  
    hora_fin TIME,  
);
```

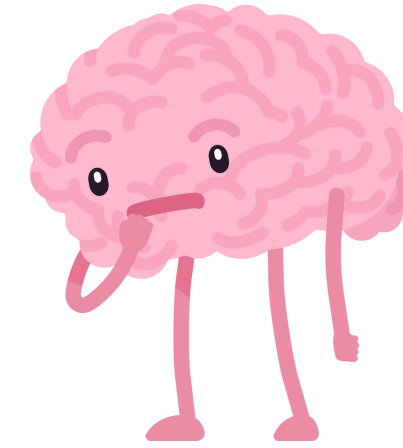
SOLUCION



a) Cree las tablas asociada al esquema del enunciado. Especifique las llaves primarias, foraneas y los tipos de datos.

```
CREATE TABLE asig_capac (  
    id_capac SERIAL PRIMARY KEY,  
    id_zona INT REFERENCES zona_despacho(id_zona),  
    id_slot INT REFERENCES slot(id_slot),  
    capac_sm INT,  
    capac_med INT,  
    capac_big INT,  
    precio_sm INT,  
    precio_med INT,  
    precio_big INT  
);
```

SOLUCION

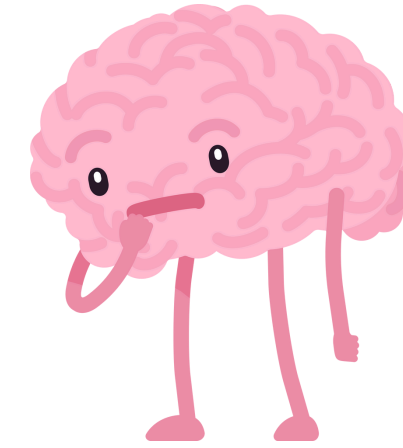


b) Escriba una instrucción SQL que inserte un nuevo registro en la tabla "asig_capac", asociando la zona con id 2 y el slot con id 5. Suponga que la capacidad para productos es 10 pequeños, 5 medianos, y 2 grandes, y que los precios son 2000, 3500 y 6000 respectivamente

```
INSERT INTO asig_capac (id_zona, id_slot, capac_sm, capac_med, capac_big,  
precio_sm, precio_med, precio_big)  
VALUES (2, 5, 10, 5, 2, 2000, 3500, 6000);
```

Recordatorio y aclaración: SERIAL, por defecto, lo que hace es asignarle a la columna el tipo de dato INT y hacerlo autoincremental. De esta forma, no tenemos que especificar un valor manualmente.

SOLUCION



c) Escriba una consulta SQL que entregue los nombres de todas las zonas de despacho que están en la región "Metropolitana".

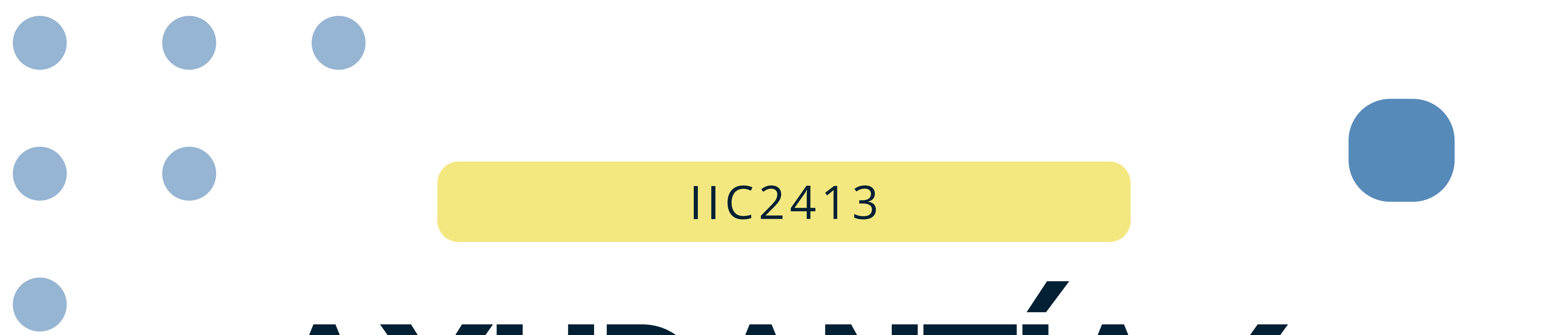
```
SELECT nombre_zona FROM zona_despacho WHERE region = 'Metropolitana';
```

SOLUCION



d) Escriba una consulta SQL que entregue, por cada zona de despacho, los rangos horarios (hora_inicio y hora_fin) de los slots que tienen una capacidad mayor a 5 productos grandes (capac_big > 5), junto al nombre de la zona y el día del slot.

```
SELECT zona_despacho.nombre_zona, slot.dia, slot.hora_inicio, slot.hora_fin  
FROM asig_capac, slot, zona_despacho  
WHERE asig_capac.id_slot = slot.id_slot  
AND asig_capac.id_zona = zona_despacho.id_zona  
AND asig_capac.capac_big>5  
;
```



IIC2413

AYUDANTÍA 6

SQL 1

