



# Consultas en múltiples tablas



**Activen las cámaras los que puedan y  
pasemos asistencia**





# Inicio

**{desafío}**  
latam\_



*/\* Realizar un cruce de tablas con select \*/*

*/\* Realizar cruces de tablas con joins \*/*

INNER JOIN

LEFT JOIN

FULL JOIN

CROSS JOIN

*/\* Diferenciar cuándo utilizar cada tipo de join \*/*

# Objetivos

# Activación de conceptos

*Contesta la pregunta correctamente y gana un punto*

## Instrucciones:

- Se realizará una pregunta, el primero en escribir “YO” por el chat, dará su respuesta al resto de la clase.
- El docente validará la respuesta.
- En caso de que no sea correcta, dará la oportunidad a la segunda persona que dijo “Yo”.
- Cada estudiante podrá participar un máximo de 2 veces.
- Al final, el/la docente indicará el 1º, 2º y 3º lugar.
- Esta actividad no es calificada, es solo una dinámica para recordar los conceptos clave para abordar esta sesión.





Activación de conceptos



¿Cuál de las siguientes no es una función de agregado?

a. COUNT

b. AVG

c. MAX

d. HAVING

{desafío}  
latam\_



Activación de conceptos



¿Cuál es la diferencia entre where y having?



## Activación de conceptos

```
SELECT nombre, count(*)  
FROM monedas
```

nombre
Guarani
Rupiah
Ruble
Peso
Peso
Yuan Renminbi
Yuan Renminbi

¿Qué le falta a la consulta para poder contar  
cuantas veces aparece cada nombre?

```
SELECT *  
FROM ventas  
WHERE monto  
  > (SELECT AVG(monto) FROM ventas);
```

consulta exterior

consulta interior

¿Para qué sirven las subconsultas?



Activación de conceptos



Primer lugar:

\_\_\_\_\_



Segundo lugar:

\_\_\_\_\_



Tercer lugar:

\_\_\_\_\_



# Desarrollo

{desafío}  
latam\_



# Consultas en múltiples tablas

## *Motivación*

En esta unidad estudiaremos consultas sobre múltiples tablas. Esto es importante porque frecuentemente necesitamos juntar información que está en distintas tablas.

# Consultas en múltiples tablas

*Nuestro primer ejemplo*

Para aprender a trabajar con múltiples tablas partiremos creando un modelo para un blog. En este modelo guardaremos posts (artículos) y comments (comentarios).

- Por cada post guardaremos el título y el contenido.
- Por cada comentario guardaremos el contenido y a qué post pertenece.
- Para ambos guardaremos un id autoincremental.

# Consultas en múltiples tablas

## *Ingresando nuestros datos*

```
CREATE TABLE posts(  
  id SERIAL,  
  title VARCHAR,  
  content TEXT  
);  
INSERT INTO posts (title, content)  
VALUES ('Artículo 1', 'LOREM IPSUM ET ...');  
INSERT INTO posts (title, content)  
VALUES ('Artículo 2', 'LOREM IPSUM ET ...');
```

id	title	content
1	Artículo 1	LOREM IPSUM ET ...
2	Artículo 2	LOREM IPSUM ET ...

# Consultas en múltiples tablas

## *Ingresando nuestros datos*

```
CREATE TABLE comments(  
  id SERIAL,  
  content text,  
  post_id integer  
);  
INSERT INTO comments (content, post_id)  
VALUES ('comentario 1', 1);  
INSERT INTO comments (content, post_id)  
VALUES ('comentario 2', 1);  
INSERT INTO comments (content, post_id)  
VALUES ('comentario 1', 2);
```

id	content	post_id
1	comentario 1	1
2	comentario 2	1
3	comentario 1	2



# Consultas en múltiples tablas

*SELECT \* FROM múltiples tablas*



Al utilizar FROM con múltiples tablas todos los datos de ambas tablas se cruzan

Utilizando WHERE se filtran los datos que no corresponden.

```
SELECT * FROM comments,  
posts WHERE post_id = id;
```

content	post_id	id	title	content
comentario 1	1	1	Artículo 1	LOREM IPSUM ET ...
comentario 2	1	1	Artículo 1	LOREM IPSUM ET ...
comentario 1	2	2	Artículo 2	LOREM IPSUM ET ...

# Consultas en múltiples tablas

*SELECT \* FROM múltiples tablas*

También podemos escoger todas las columnas de una tabla utilizando **nombre\_tabla.\***

```
SELECT posts.*, comments.* FROM  
comments, posts WHERE post_id = id;
```

Utilizar el nombre de la tabla será particularmente útil para diferenciar cuando un campo tenga el mismo nombre en ambas tablas.

content	post_id	id	title	content
comentario 1	1	1	Artículo 1	LOREM IPSUM ET ...
comentario 2	1	1	Artículo 1	LOREM IPSUM ET ...
comentario 1	2	2	Artículo 2	LOREM IPSUM ET ...

# Consultas en múltiples tablas

## *SELECT de múltiples tablas sin el where que las juntas*

Probemos la misma instrucción sin la cláusula WHERE.

```
SELECT * FROM comments,  
posts;
```

content	post_id	id	title	content
comentario 1	1	1	Artículo 1	LOREM IPSUM ET ...
comentario 1	1	2	Artículo 2	LOREM IPSUM ET ...
comentario 2	1	1	Artículo 1	LOREM IPSUM ET ...
comentario 2	1	2	Artículo 2	LOREM IPSUM ET ...
comentario 1	2	1	Artículo 1	LOREM IPSUM ET ...
comentario 1	2	2	Artículo 2	LOREM IPSUM ET ...



Obtenemos todos los registros de una contra todos los registros de otra.  
A este resultado se le denomina **producto cartesiano**.

## Ejercicio - Consultando dos tablas


Descarga el archivo usuarios.sql y bitcoins.sql que se encuentran en la plataforma Empieza, e ingrésalos en la base de datos.

Cada uno de los archivos corresponde a una tabla.

- Selecciona los datos de la tabla usuarios y la de bitcoins para mostrar una tabla con los campos nombre, email, dirección y monto de cada usuario.
- Muestra la columna email solo una vez (puedes escoger cualquiera de las dos tablas).

## Ejercicio ¡Manos al teclado!



*/\* Realizar un cruce de tablas con select \*/* 

*/\* Realizar cruces de tablas con joins \*/*

INNER JOIN

LEFT JOIN

FULL JOIN

CROSS JOIN

*/\* Diferenciar cuándo utilizar cada tipo de join \*/*

# Objetivos

# Consultas en múltiples tablas

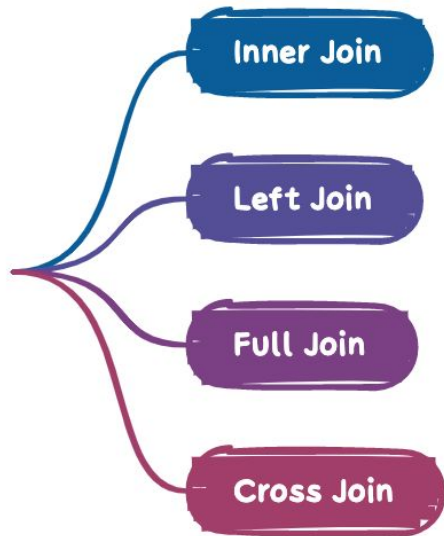
## Joins

Los joins nos dan flexibilidad a la hora de escoger qué registros incluir de cada tabla.

La sintaxis para unir tablas con join es la siguiente:

```
SELECT *  
FROM tabla1  
TIPO-DE-JOIN tabla2  
ON tabla1.campo = tabla2.campo
```

### Tipos de Join



# Consultas en múltiples tablas

## *Preparando nuestros datos*

Los tipos de join difieren especialmente en qué hacer cuando el campo que los une es nulo.  
Para probar esto tendremos que actualizar nuestros datos.

content	post_id	id	title	content
comentario 1	1	1	Artículo 1	LOREM IPSUM ET ...
comentario 2		1	Artículo 1	LOREM IPSUM ET ...
comentario 1		2	Artículo 2	LOREM IPSUM ET ...

Para probarlo borremos los post\_id  
que es donde juntamos los datos.

```
UPDATE comments  
SET post_id = NULL  
WHERE content = 'comentario 2'  
OR post_id = 2
```

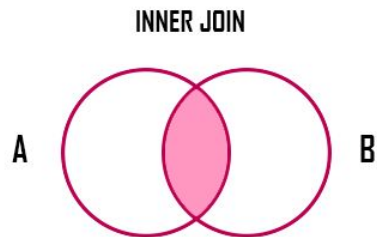
# Consultas en múltiples tablas

## INNER JOIN

INNER JOIN selecciona las filas que hacen match en ambas columnas.

id	title	content
1	Artículo 1	LOREM IPSUM ET ...
2	Artículo 2	LOREM IPSUM ET ...

content	post_id	id	title	content
comentario 1	1	1	Artículo 1	LOREM IPSUM ET ...
comentario 2		1	Artículo 1	LOREM IPSUM ET ...
comentario 1		2	Artículo 2	LOREM IPSUM ET ...



```
SELECT columnas  
FROM A  
INNER JOIN B  
ON A.columna=B.columna
```

El diagrama de VENN nos muestra cómo se realizará el cruce, en el caso del INNER JOIN es la **intersección** de ambos grupos, o sea solo los registros que están en **ambos** grupos.



# Consultas en múltiples tablas

## INNER JOIN

```
SELECT *  
FROM posts  
INNER JOIN comments  
ON posts.id = comments.post_id
```

id	title	content
1	Artículo 1	LOREM IPSUM ET ...
2	Artículo 2	LOREM IPSUM ET ...

content	post_id	id	title	content
comentario 1	1	1	Artículo 1	LOREM IPSUM ET ...
comentario 2		1	Artículo 1	LOREM IPSUM ET ...
comentario 1		2	Artículo 2	LOREM IPSUM ET ...

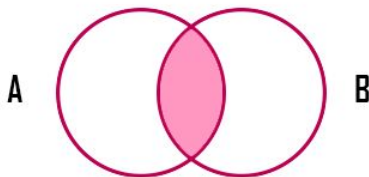
id	title	content	content	post_id
1	Artículo 1	LOREM IPSUM ET ...	comentario 1	1

# Consultas en múltiples tablas

## INNER JOIN

- INNER JOIN es equivalente a `SELECT * FROM tabla1, tabla2 WHERE tabla1.campo = tabla2.campo`
- Es preferible ocupar INNER JOIN para juntar tablas ya que deja claro lo que se intenta lograr.

INNER JOIN



```
SELECT columnas
FROM A
INNER JOIN B
ON A.columna=B.columna
```

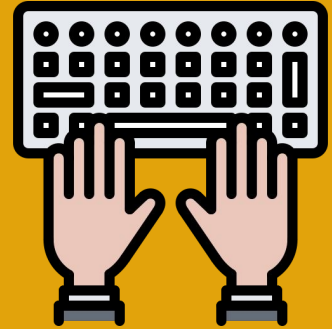
## Ejercicio

Utilizando las tablas `usuarios.sql` y `bitcoins.sql` ingresadas previamente.

- Selecciona los datos de ambas tablas utilizando INNER JOIN en lugar de WHERE

## Ejercicio

¡Manos al teclado!



*/\* Realizar un cruce de tablas con select \*/* ✓

*/\* Realizar cruces de tablas con joins \*/*

INNER JOIN ✓

LEFT JOIN

FULL JOIN

CROSS JOIN

*/\* Diferenciar cuándo utilizar cada tipo de join \*/*

# Objetivos

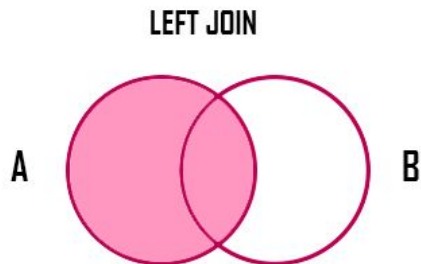
# Consultas en múltiples tablas

## LEFT JOIN

LEFT JOIN selecciona todas las filas de la primera tabla y se unifica con la segunda siempre y cuando haya match

id	title	content
1	Artículo 1	LOREM IPSUM ET ...
2	Artículo 2	LOREM IPSUM ET ...

content	post_id	id	title	content
comentario 1	1	1	Artículo 1	LOREM IPSUM ET ...
comentario 2		1	Artículo 1	LOREM IPSUM ET ...
comentario 1		2	Artículo 2	LOREM IPSUM ET ...



```
SELECT columnas  
FROM A  
LEFT JOIN B  
ON A.columna=B.columna
```

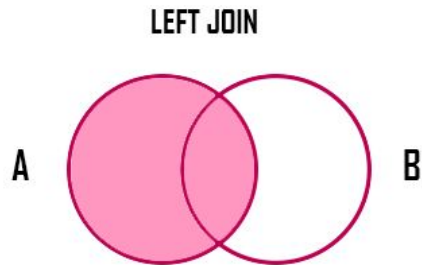
El diagrama de VENN nos muestra cómo se realizará el cruce, en el caso del LEFT JOIN nos muestra que el resultado tendrá todos los datos de la primera tabla y solo los que hagan match de la segunda.

# Consultas en múltiples tablas

## LEFT JOIN

```
SELECT *  
FROM posts  
LEFT JOIN comments  
ON posts.id = comments.post_id
```

id	title	content	content	post_id
1	Artículo 1	LOREM IPSUM ET ...	comentario 1	1
2	Artículo 2	LOREM IPSUM ET ...	NULL	NULL



```
SELECT columnas  
FROM A  
LEFT JOIN B  
ON A.columna=B.columna
```

Todos los resultados de la tabla posts y aquellos que cumplen con la condición fueron unificados. Los que no cumplen, quedan como NULL.

## Ejercicio - Usando LEFT/RIGHT JOIN

Utilizando las tablas usuarios.sql y bitcoins.sql ingresadas previamente.

1. Selecciona los datos de ambas tablas utilizando LEFT JOIN y tomando como primera la tabla de bitcoins.
2. Selecciona los datos de ambas tablas utilizando LEFT JOIN y tomando como primera la tabla de usuarios.
3. Compara las diferencias.
4. Vuelve al punto 1 pero reemplaza LEFT JOIN por RIGHT JOIN.

## Ejercicio ¡Manos al teclado!



*/\* Realizar un cruce de tablas con select \*/* ✓

*/\* Realizar cruces de tablas con joins \*/*

INNER JOIN ✓

LEFT JOIN ✓

FULL JOIN

CROSS JOIN

*/\* Diferenciar cuándo utilizar cada tipo de join \*/*

# Objetivos



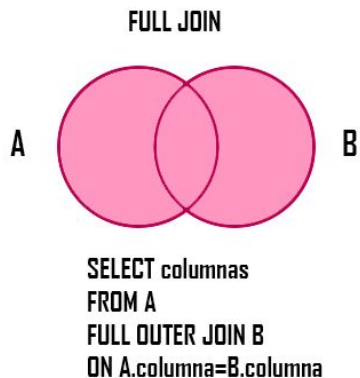
# Consultas en múltiples tablas

## *FULL JOIN*

FULL JOIN une todos los registros de ambas tablas haya match o no.

id	title	content	content	post_id
1	Artículo 1	LOREM IPSUM ET ...	comentario 1	1
null	null	null	comentario 2	null
null	null	null	comentario 1	null
2	Artículo 2	LOREM IPSUM ET ...	null	null

Se puede consultar “proyectos” y “trabajadores” y ver en cuáles no hay trabajadores asignados, y quienes sí tienen un proyecto asignado.



## Ejercicio - Usando FULL JOIN

Utilizando las tablas usuarios.sql y bitcoins.sql ingresadas previamente.

- Selecciona los datos de ambas tablas utilizando FULL JOIN

## Ejercicio ¡Manos al teclado!



/\* Realizar un cruce de tablas con select \*/



/\* Realizar cruces de tablas con joins \*/

INNER JOIN



LEFT JOIN



FULL JOIN



CROSS JOIN

/\* Diferenciar cuándo utilizar cada tipo de join \*/

# Objetivos

# Consultas en múltiples tablas

## CROSS JOIN

CROSS JOIN combina cada uno de los registros de una tabla con los registros de la otra, del mismo modo que `SELECT * FROM posts, comments;`

Con cross join el código quedaría de la siguiente manera:

`SELECT * FROM posts CROSS JOIN comments`

```
posts=# select * from posts, comments;
```

id	title	content	content	post_id
1	Artículo 1	LOREM IPSUM ET ...	comentario 1	1
2	Artículo 2	LOREM IPSUM ET ...	comentario 1	1
1	Artículo 1	LOREM IPSUM ET ...	comentario 2	1
2	Artículo 2	LOREM IPSUM ET ...	comentario 2	1
1	Artículo 1	LOREM IPSUM ET ...	comentario 1	2
2	Artículo 2	LOREM IPSUM ET ...	comentario 1	2

(6 rows)

```
posts=# select * from posts cross join comments;
```

id	title	content	content	post_id
1	Artículo 1	LOREM IPSUM ET ...	comentario 1	1
2	Artículo 2	LOREM IPSUM ET ...	comentario 1	1
1	Artículo 1	LOREM IPSUM ET ...	comentario 2	1
2	Artículo 2	LOREM IPSUM ET ...	comentario 2	1
1	Artículo 1	LOREM IPSUM ET ...	comentario 1	2
2	Artículo 2	LOREM IPSUM ET ...	comentario 1	2

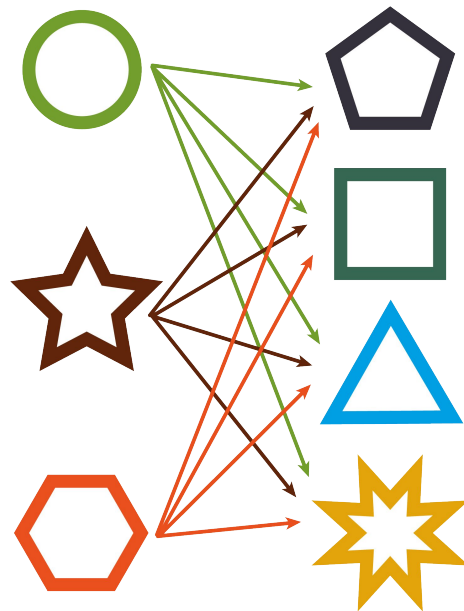
(6 rows)

# Consultas en múltiples tablas

## CROSS JOIN

- Como se muestra en los códigos de ejemplo, ambos códigos realizan y entregan el mismo resultado.
- Los resultados podrían no tener sentido dado que se genera una multiplicidad de información. A nosotros nos interesa saber cuántos comentarios tiene un mismo posts.
- Lo pertinente es agregar una condición donde se evalúen los datos que cumplen una condición, por ejemplo `post_id = id`.

```
select * from posts cross join comments where post_id = id;
```



# Consultas en múltiples tablas

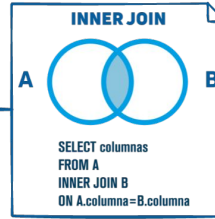
*Un caso útil*

Tenemos una tabla de platos de comida y otra tabla de bebestibles y queremos generar menús automáticamente. En ese caso podríamos hacer un CROSS JOIN.

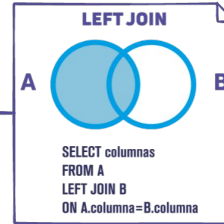


## Tipos de Join

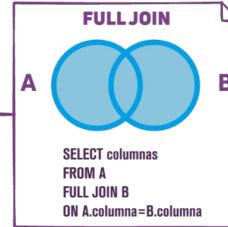
### INNER JOIN



### LEFT JOIN



### FULL JOIN



### CROSS JOIN



*/\* Realizar un cruce de tablas con select \*/* ✓

*/\* Realizar cruces de tablas con joins \*/*

INNER JOIN ✓

LEFT JOIN ✓

FULL JOIN ✓

CROSS JOIN ✓

*/\* Diferenciar cuándo utilizar cada tipo de join \*/* ✓

# Objetivos





Cierre

{desafío}  
latam\_



¿Existe algún concepto que no  
hayas comprendido?

Reflexionemos

- Revisar la guía que trabajarán de forma autónoma.
- Revisar en conjunto el desafío.

¿Qué sigue?

¿Tienen alguna duda respecto al Desafío?



*Academia de  
talentos digitales*

[www.desafiolatam.com](http://www.desafiolatam.com)



/DesafioLatam



/DesafioLatam



/DesafioLatam



/DesafioLatam