



Consultas en SQL (Parte I)

Clase sincrónica

***Generar consultas
agrupadas sobre múltiples
tablas en SQL para extraer
información de bases de
datos.***

- Unidad 1: Introducción al análisis y estructura de datos.
- Unidad 2: Consultas en SQL
 - (Parte I)
 - (Parte II)
 - (Parte III)
 - (Parte IV)



Te encuentras aquí



¿Qué aprenderás en esta sesión?

- Consultar datos de una tabla filtrando por columna o por condición, ordenando los resultados y limitando la cantidad de respuestas.
- Insertar, modificar y eliminar datos de una tabla.
- Crear, modificar y borrar tablas.

¿Por qué necesitamos
base de datos?



/* Introducción a bases de datos */

Introducción a bases de datos

¿Por qué necesitamos bases de datos?

- Guardar información de forma permanente (o por un largo tiempo).
- Permiten guardar millones de registros y recuperar los que necesitamos de forma sencilla.
 - En esta unidad aprenderemos a crear una base de datos, guardar datos en ella y recuperarlos, todo esto utilizando un lenguaje llamado SQL.
 - Esto nos permitirá crear aplicaciones que guarden datos o utilicen datos ya guardados.

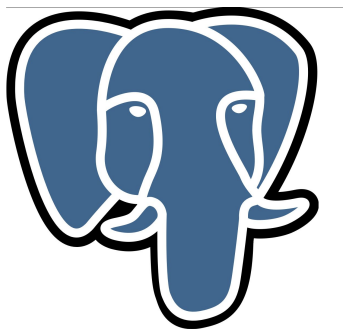


Nuestras herramientas

Entorno de trabajo

PostgreSQL

En este curso ocuparemos el motor de base de datos postgresQL. En la primera clase ocuparemos una versión online, luego tendremos que instalarlo.



{desafío}
latam_

Navegador

Para esta clase utilizaremos sqliteonline por lo que necesitaremos un navegador. Se recomienda Firefox, pero se pueden utilizar otros.



Introducción a bases de datos

Conceptos básicos

- Existen distintos tipos de motores de bases de datos.
 - Utilizaremos postgresQL uno de los más utilizados.
 - Pertenece al tipo de motor conocido como **bases de datos relacionales**.
 - Tiene soporte para datos no relacionales.
- Por ahora, solo tenemos que saber que trabajaremos con **bases de datos relacionales**.
 - En ellas, los datos se organizan en tablas.



Introducción a bases de datos

Conceptos básicos

- Una base de datos relacional es similar a un archivo excel.
- Tenemos múltiples **tablas** que guardan información, cada fila de la tabla es un **registro**.



Introducción a bases de datos

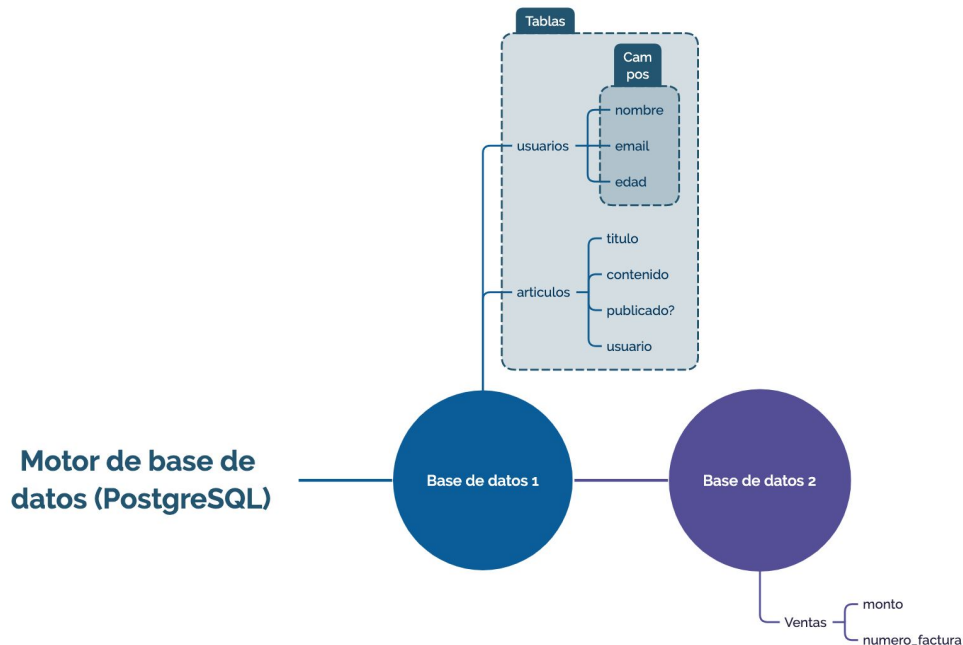
Conceptos básicos

- Por ejemplo, en esta tabla cada registro representa una persona.
- Cada columna guarda un atributo o **campo** (field) del registro. Y lo que guarda en ese campo es un **valor** (value).

1	Id	Nombre	Apellido	Email	Ciudad	Compañía
2	1	Menard	Pozzo	mpozzo0@ask.com	Mitsuke	Jast and Sons
3	2	Elfrida	Caitlin	ecaitlin1@wisc.edu	Ozerne	Schinner-Little
4	3	Basilus	Squire	bsquire2@nifty.com	Pueblo Nuevo	Braun, Kozey and Runolfssdottir
5	4	Merrielle	Ridewood	mridentwood3@cnet.c	Bila Krynytsya	Harber-Fay
6	5	Peggy	Stiell	pstiell4@cbc.ca	Halton	Ratke and Sons
7	6	Almeda	Bosman	abosman5@ebay.co	Ziliang	Bradtke-Labadie
8	7	Ingra	Jee	jjee6@slate.com	Göteborg	Denesik Group
9	8	Wilow	Oddey	woddey7@dot.gov	Amapala	Weber-Haag
10	9	Ofelia	Lismore	olismore8@sphinn.c	Xinglong	Littel, Casper and Deckow
11	10	Evvy	Muzzini	emuzzini9@hao123.c	Puno	Schmitt-Bechtelar
12	11	Katharina	Stutt	kstutta@booking.co	Obo	Sipes LLC
13	12	Janetta	Pietrowicz	jpietrowiczb@cbsloc	Cacaopera	Jenkins LLC
14	13	Lin	Arntzen	larntzenc@github.co	Ingenio La Esperanz	Pfeffer, Koelpin and Satterfield
15	14	Hillie	Muehle	hmuehled@addthis.c	Staroshcherbinovsk	Rohan-Bosco
16	15	Annabel	Bellson	abellsone@ebay.com	Ljukovo	Luetngen-Bahringer
17	16	Dion	Habin	dhabinf@nymag.com	Gällivare	Romaguera Inc
18	17	Rickey	Blandamore	rblandamoreg@sou	Vantaa	Dicki, Kerluke and Carter
19	18	Eran	Hearon	ehearonh@fc2.com	Dujiating	Collins-Hackett
20	19	Caroline	Manilove	cmanilovei@ucoz.co	Huamali	Wilderman, Wiza and Spinka
21	20	Raine	Owtrim	rowtrim@house.gov	Tyszowce	Christiansen Group
22	21	Sheeree	Vinton	svintonk@noaa.gov	Żychlin	Russel, Boyer and Emmerich
23	22	Heda	MacIraith	hmacilraith@cmu.ec	Mpongwe	Howell-Muller
24	23	Cecelia	Delieu	cdelleium@linkedin.c	Shenwan	Schowalter Inc
25	24	Blaire	Middleditch	bmiddleditchn@orac	Odessa	Fadel Group

Introducción a bases de datos

Conceptos básicos



En un **motor de base de datos** podemos tener **múltiples bases**.

- Dentro cada una de estas podemos tener diversas **tablas**.
- Y en cada tabla, puede haber diferentes **registros**.

Introducción a bases de datos

Conceptos básicos

SQL permite hacer consultas a una base de datos como las siguientes:

- “Selecciona todos los registros de la ciudad Odessa”
- “Selecciona todos los registros con id mayor a 100”

Nos permite, además, crear y modificar tablas, e insertar, modificar y borrar registros en una tabla.

{desafío}
latam_

1	Id	Nombre	Apellido	Email	Ciudad	Compañía
2	1	Menard	Pozzo	mpozzo0@ask.com	Mitsuke	Jast and Sons
3	2	Elfrida	Caitlin	ecaitlin1@wisc.edu	Ozerne	Schinner-Little
4	3	Basilius	Squire	bsquire2@nifty.com	Pueblo Nuevo	Braun, Kozey and Runolfsdottir
5	4	Merrielle	Ridewood	mridentwood3@cnet.c	Bila Krynytsya	Harber-Fay
6	5	Peggy	Stiell	pstiell4@cbc.ca	Halton	Ratke and Sons
7	6	Almeda	Bosman	abosman5@ebay.co	Ziliang	Bradtke-Labadie
8	7	Ingra	Jee	ijee6@slate.com	Göteborg	Denesik Group
9	8	Willow	Oddey	woddey7@dot.gov	Amapala	Weber-Haag
10	9	Ofelia	Lismore	olismore8@sphinn.c	Xinglong	Littel, Casper and Deckow
11	10	Evvy	Muzzini	emuzzini9@hao123.c	Puno	Schmitt-Bechtelar
12	11	Katharina	Stutt	kstutta@booking.co	Obo	Sipes LLC
13	12	Janetta	Pietrowicz	jpietrowicz@cbsslo	Cacaopera	Jenkins LLC
14	13	Lin	Arntzen	larnztenc@github.co	Ingenio La Esperanz	Pfeffer, Koelpin and Satterfield
15	14	Hillie	Muehle	hmuehle@addthis.c	Staroshcherbinovsk	Rohan-Bosco
16	15	Annabel	Bellson	abellsone@ebay.com	Ljukovo	Luetgen-Bahringer
17	16	Dion	Habin	dhabinf@nymag.com	Gällivare	Romaguera Inc
18	17	Rickey	Blandamore	rblandamore@soup	Vantaa	Dicki, Kerluke and Carter
19	18	Eran	Hearon	ehearonh@fc2.com	Dujiajing	Collins-Hackett
20	19	Caroline	Manilove	cmanilove@ucoz.co	Huamali	Wilderman, Wiza and Spinka
21	20	Raine	Owttrim	rowttrimi@house.gov	Tyszowce	Christiansen Group
22	21	Sheeree	Vinton	svintonk@noaa.gov	Żychlin	Russel, Boyer and Emmerich
23	22	Heda	MacIlraith	hmacilraith@cmu.ec	Mpongwe	Howell-Muller
24	23	Cecelia	Delieu	cdelieu@linkedin.c	Shenwan	Schowalter Inc
25	24	Blaire	Middleditch	bmiddleditchn@orac	Odessa	Fadel Group

Ejercicio guiado: “Realizando nuestra primera consulta”



Ejercicio guiado

Realizando nuestra primera consulta

- **Paso 1:** Ingresa a sqliteonline.com:
- **Paso 2:** Selecciona PostgreSQL como motor de bases de datos.
- **Paso 3:** Escribe la siguiente consulta `Select * from Demo;`
- **Paso 4:** Presiona el botón RUN.
- **Paso 5:** Por último, observemos el resultado como se muestra en la imagen.

id	name	hint
1	test	1
2	server	postgresql
3	limit time out	3h
4	limit db	70mb
5	limit count name: table, proc, fun, ..	300
6	limit overrun	auto-drop DB
7	valor 1	valor 2



Ejercicio guiado

Entendamos el código

SELECT	Indica que la consulta a realizar será de selección.
*	El asterisco es un comodín para indicar que se deben seleccionar todos los campos, o sea, todas las columnas de la tabla.
FROM	Indica de qué tabla específica se va a seleccionar.
demo	Nombre de la tabla. En este caso, esta viene precargada en sqlliteonline.
;	Una consulta termina con un punto y coma, de esta forma podemos separar varias instrucciones.



¿Cuál es el comodín para
seleccionar todos los
campos de la tabla
demo?



**/* Sintaxis y resultados
de una consulta */**

Insensibilidad a las mayúsculas

Sintaxis SQL

```
SELECT * FROM demo;
```

Es lo mismo que:

```
select * from demo;
```



Resultados de una consulta

¡El resultado de una consulta a una tabla es otra tabla!

Esto quiere decir que cuando consultamos por una tabla, **siempre** obtendremos como resultado otra tabla con los valores consultados.

```
SELECT * FROM tabla;
```

c1	c2	c3	c4
98	23	'a'	True
45	45	'b'	False

Tabla a consultar

c1	c2	c3	c4
98	23	'a'	True
45	45	'b'	False

Resultado de la consulta

Consulta especificando el campo

La información precisa

Tabla a consultar

c1	c2	c3	c4
98	23	'a'	True
45	45	'b'	False
34	76	'c'	True
87	34	'd'	None
82	56	'e'	True
90	10	'f'	False

{desafío}
latam_

```
SELECT c1,c3 FROM tabla;
```

Para especificar
múltiples campos,
separamos los nombres
de los campos con coma
(,) en lugar de utilizar
asterisco (*)

Resultado

c1	c3
98	'a'
45	'b'
34	'c'
87	'd'
82	'e'
90	'f'

Ejercicio: “Consultando campos”



Ejercicio guiado

Contexto

A continuación, realizarás un ejercicio en el cual deberás consultar la tabla demo de [sqliteonline](#), para extraer los id y los names de la tabla.
Para lograrlo, recuerda separar por comas los campos que deseas obtener información.

1. Primero selecciona solo los names.
2. Luego, selecciona el id y los nombres.



c1	c2	c3	c4
98	23	'a'	True
45	45	'b'	False

A partir de la tabla anterior, ¿cuál es la consulta para obtener los valores de c2 y c4?



/* Consultas y condiciones */

Consultando datos

Con condiciones

Tabla a consultar

c1	c2	c3	c4
98	23	'a'	True
45	45	'b'	False
34	76	'c'	True
87	34	'd'	None
82	56	'e'	True
90	10	'f'	False

SELECT *
FROM tabla
where c4 =
True;

Resultado

c1	c2	c3	c4
98	23	'a'	True
34	76	'c'	True
82	56	'e'	True

Ejercicio

“Consulta con condiciones”



Ejercicio guiado

Consulta con condiciones

A partir de la tabla demo de [sqliteonline](https://sqliteonline.com/), utiliza las consultas condicionadas para obtener aquellos datos cuyo id sean mayores que 4.

Recuerda el uso del where.



Consultando datos

Más condiciones

Tabla a consultar

c1	c2	c3	c4
98	23	'a'	True
45	45	'b'	False
34	76	'c'	True
87	34	'd'	None
82	56	'e'	True
90	10	'f'	False

`SELECT *
FROM tabla
LIMIT 2;`

Resultado

c1	c2	c3	c4
98	23	'a'	True
45	45	'b'	False

Consultando datos

Agreguemos orden

Tabla a consultar



```
SELECT * FROM  
tabla  
ORDER BY c1 DESC;
```



Resultado

c1	c2	c3	c4
98	23	'a'	True
45	45	'b'	False
34	76	'c'	True
87	34	'd'	None
82	56	'e'	True
90	10	'f'	False

Con ASC,
ordenamos por la
columna en orden
creciente, con
DESC, en orden
decreciente.

c1	c2	c3	c4
98	23	'a'	True
90	10	'f'	False
87	34	'd'	None
82	56	'e'	True
45	45	'b'	False
34	76	'c'	True

Ejercicio

“Consultas complejas”



Ejercicio guiado

Consultas complejas

Utilizando la tabla demo de [sqliteonline](#):

1. Selecciona todos los registros ordenados alfabéticamente por nombre (al ordenar por una columna de tipo string se entenderá que el orden es alfabético).
2. Selecciona los primeros 3 registros ordenados por id:
 - a. Primero ordena con ORDER BY.
 - b. Luego limita con Limit.

Resultado del segundo requerimiento del ejercicio:

id	name	hint
1	test	1
2	server	postgresql
3	limit time out	3h



`/* Insertando datos*/`

Insertando datos

Nuestra primera inserción

```
INSERT INTO demo (NAME, HINT) VALUES ('Gonzalo', 'Pista 1');
```

INSERT	Indica que vamos a insertar un dato.
INTO	Indica en qué tabla vamos a insertar.
demo	Es el nombre de la tabla en la que se insertarán los datos
(name, hint)	Los nombres de las columnas donde se insertarán datos.
VALUES	Indica que a continuación vamos a insertar valores.
('Gonzalo', 'Pista 1');	Los valores que estamos insertando, están en el mismo orden que las columnas, es decir, Gonzalo corresponde a name y 'pista 1' a Hint.

Insertar datos

Hay dos forma de insertar datos

1. La primera forma es ingresando valores asociados a todas las columnas.

```
INSERT INTO tabla  
VALUES (1, 2);
```



Suponiendo que la tabla solo tiene 2 columnas.

2. La segunda, es ingresando valores asociados a ciertas columnas.

```
INSERT INTO tabla  
  (columna2) VALUES  
  (2);
```

Insertar datos

Otras reglas importantes

- Los strings se ingresan con comillas simples en postgresQL, utilizar comillas dobles nos mostrará error.

```
INSERT INTO tabla(  
    VALUES ('Valor 1', Valor 2')  
);
```



Suponiendo que la tabla solo tiene 2 columnas.

- Más adelante revisaremos los tipos de datos que podemos ingresar.

Insertar datos

Insertando solo algunas columnas

```
INSERT INTO tabla  
  (columna2) VALUES  
  (2);
```

Cuando insertamos los datos indicando las columnas, el resto de los campos para ese registro queda nulo (NULL), excepto en algunos casos como nuestra tabla demo.



En la tabla demo hay una columna de tipo **Serial**, estas son muy frecuentes y tienen valor autoincremental, o sea que se incrementa solo.

Ejercicio guiado “Insertando datos”



Insertando datos

Contexto

A continuación, realizaremos un ejercicio de inserción de datos en la tabla demo de sqlliteonline. Para lograrlo realizaremos los siguientes pasos:

- **Paso 1:** Insertaremos 2 registros nuevos en la tabla demo especificando las columnas name y hint, y asignaremos valores.
- **Paso 2:** Mostraremos todos los registros.
- **Paso 3:** Insertaremos un nuevo registro en la tabla demo, solo con la columna name.
- **Paso 4:** Consultamos la tabla para verificar la inserción de los registros nuevos.



Insertar datos

Nulo no es lo mismo que vacío

Vacío, 0 o false
dependiendo
del tipo de
columna



Nulo

Insertar datos

Nulo no es lo mismo que vacío

Valores null:

- En el contexto de las bases de datos, un valor null se entiende como indefinido o desconocido.

Por ejemplo:

- En la tabla demo de sqlliteonline, un valor null se genera cuando omitimos los datos de uno de sus campos, es decir, no es rellenado con ningún valor.

Valor vacío

- Por su parte, un valor vacío se genera cuando uno de los campos de la tabla no contiene ningún carácter.

Por ejemplo:

```
INSERT into demo(name, hint) values ('nombre 4',  
");
```

*Nótese que el valor a asignar en el campo **hint** las comillas se encuentran vacías.*

Ejercicio guiado “Insertando y ordenando”



Insertando y ordenando

Contexto

A continuación, realizaremos un ejercicio en el cual insertaremos registros en la tabla demo de sqlliteonline. En esta ocasión, además de insertar, ordenaremos la tabla de manera decreciente.

Recordemos que `ORDER BY`, por defecto, ordena los registros de manera ascendente, es decir, de menor a mayor. En este sentido, si luego de `ORDER BY` escribimos `DESC`, entonces SQL interpretará que se quiere ordenar, pero de manera decreciente, es decir, de mayor a menor.

```
select * from demo order by id desc;
```

Sigamos los pasos a continuación...



Ejercicio guiado

Sigue los pasos...

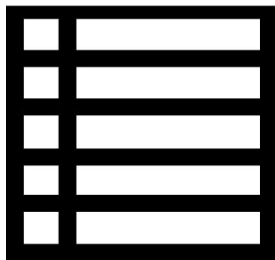
Utilizando la tabla demo de [sqliteonline](https://sqliteonline.com/):

1. Insertamos un nombre y una pista ("Pista" es cualquier frase que quieras ingresar).
2. Inserta el nombre de un compañero y otra pista.
3. Selecciona los últimos dos registros (ordena por id decreciente y limita 2 resultados).



Actualizando datos

No basta con cambiar o agregar




```
UPDATE tabla SET  
columna1=1  
WHERE condicion;
```

```
UPDATE demo SET  
hint ='cambiado'  
WHERE id = 1;
```



Quando actualizamos datos tenemos que agregar a la cláusula el **WHERE**. De lo contrario podríamos afectar todos los otros registros por error.

Ejercicio guiado “Cambiando valores”



Ejercicio guiado

Cambiando valores

Utilizando la tabla demo de [sqliteonline](https://sqliteonline.com/):

1. Seleccionamos el registro que tiene tu nombre y cambia el valor de hint por "Aprendí a cambiar valores de una tabla" (recordemos el uso del **where**).
2. Seleccionamos todos los ids menores que 5 y cambiamos el valor en la columna hint por "Cambio masivo".



Eliminación de registros



La eliminación es muy similar a la modificación, y al igual que en esta, debemos recordar utilizar la cláusula **where** para evitar borrar todos los registros.

Ejercicio guiado “Eliminando registros”



Ejercicio guiado

Eliminando registros

Utilizando la tabla demo de [sqliteonline](https://sqliteonline.com/), vamos a borrar los primeros dos registros que contengan los id 1 y 2.

Recordemos que al igual que el UPDATE debemos utilizar el WHERE para que dicha eliminación cumpla con una condición, dado que si no especificamos podemos eliminar datos de manera incorrecta.



/*Bases de datos*/

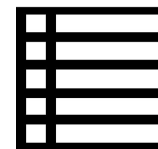
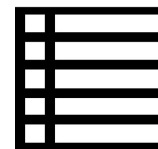
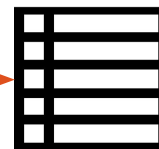
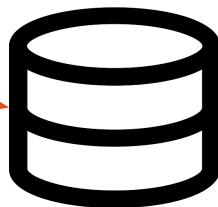
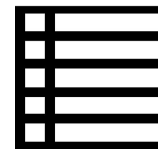
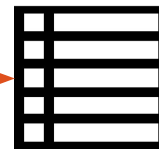
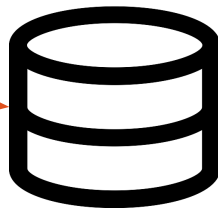
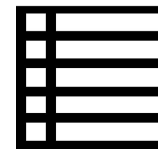
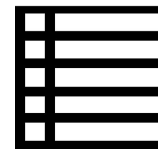
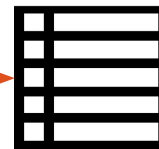
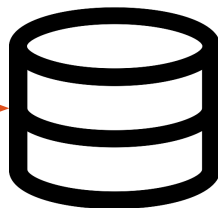
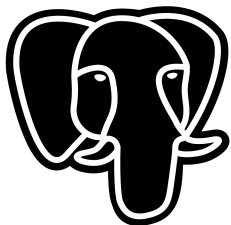
Comprendiendo las bases de datos

Estructura de una base de datos

Motor de Base de Datos

Bases de Datos

Tablas



Trabajando con bases de datos

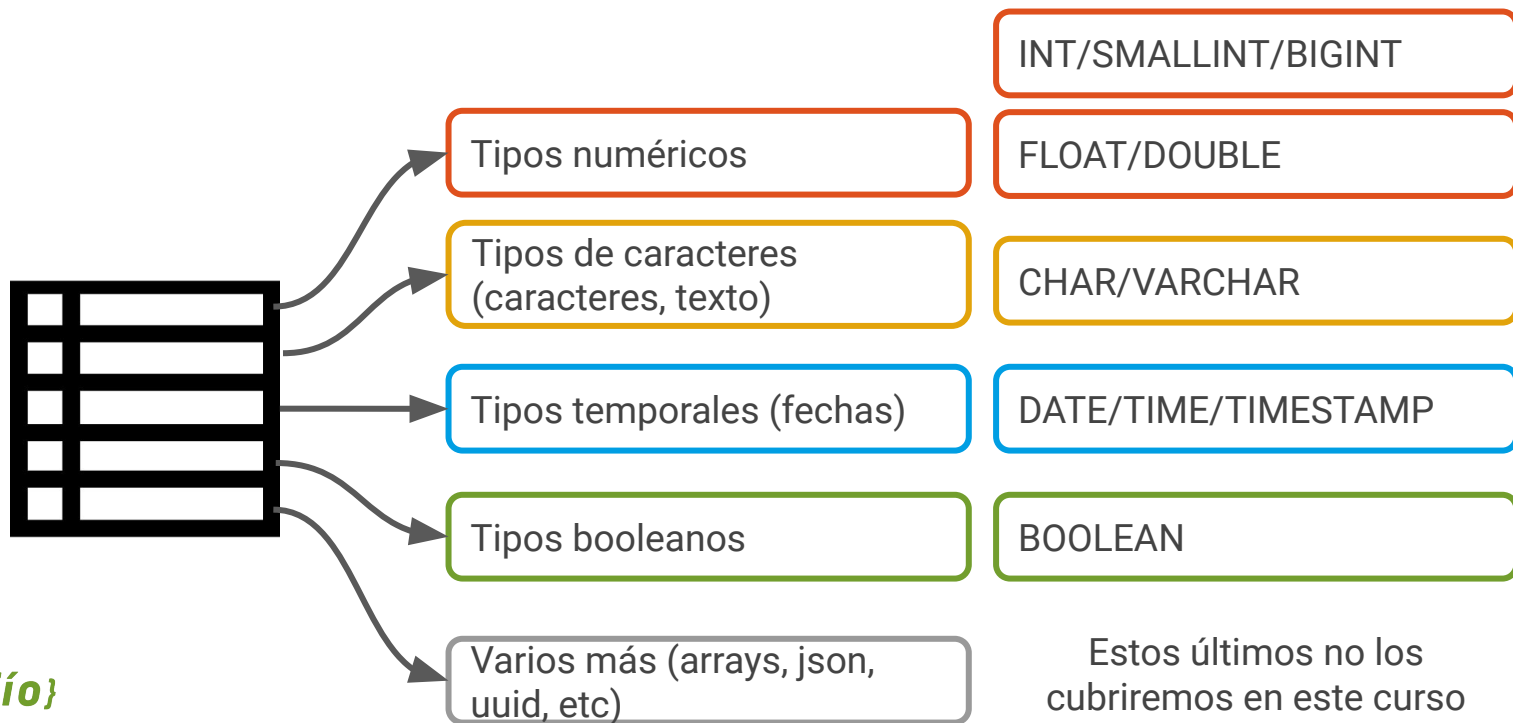
Tablas

Column	
 id	integer
 name	character varying(200)
 hint	text

- Dentro de un motor de base de datos puede haber múltiples bases, y dentro de estas, puede haber diversas tablas.
- Ahora aprenderemos a crear una tabla nueva.
- Una tabla se compone de varias columnas, cada una con un nombre y un tipo de dato.
 - Por ejemplo, podemos ver en la tabla demo de [sqliteonline](#) que las columnas especifican un tipo de dato junto al nombre.

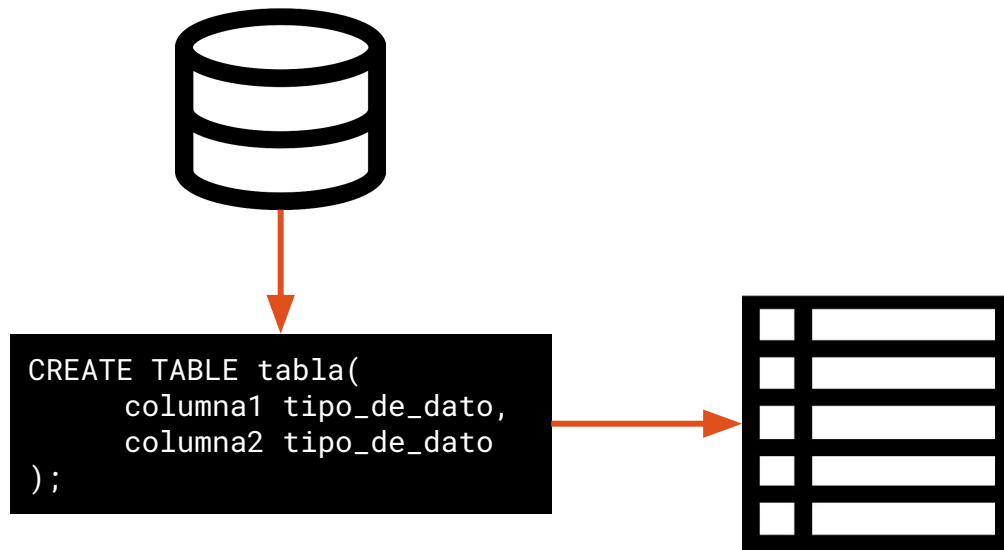
Trabajando con bases de datos

Tablas y tipos de datos



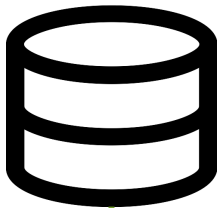
Trabajando con bases de datos

Creando una tabla nueva



Trabajando con bases de datos

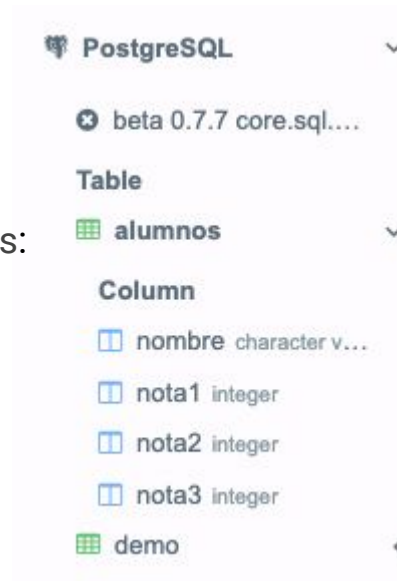
Creando una tabla nueva



Probemos dentro de [sqliteonline](https://sqliteonline.com/) el siguiente código:

```
CREATE TABLE alumnos(  
  nombre varchar,  
  nota1 integer,  
  nota2 integer,  
  nota3 integer  
);
```

Obtendremos:



Trabajando con bases de datos

Insertando datos en nuestra tabla

```
insert into alumnos (Nombre, Nota1, Nota2, Nota3) values ('Elmira', 10, 1, 6);  
insert into alumnos (Nombre, Nota1, Nota2, Nota3) values ('Izak', 1, 6, 5);  
insert into alumnos (Nombre, Nota1, Nota2, Nota3) values ('Claudette', 5, 1, 3);  
insert into alumnos (Nombre, Nota1, Nota2, Nota3) values ('Noreen', 1, 6, 4);  
insert into alumnos (Nombre, Nota1, Nota2, Nota3) values ('Natalina', 8, 5, 3);  
insert into alumnos (Nombre, Nota1, Nota2, Nota3) values ('Jacky', 5, 2, 6);  
insert into alumnos (Nombre, Nota1, Nota2, Nota3) values ('Jilly', 6, 1, 2);  
insert into alumnos (Nombre, Nota1, Nota2, Nota3) values ('Robbie', 4, 8, 5);
```

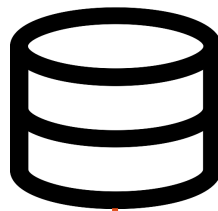
Seleccionamos los datos para probar

```
SELECT nombre * FROM alumnos
```


Trabajando con bases de datos

Borrando una tabla

- Probemos dentro de sqlliteonline el siguiente código:



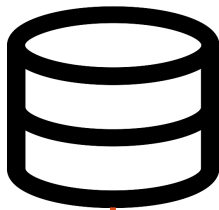
```
Drop TABLE alumnos;
```



Cuidado: ¡No hay vuelta atrás!
Aunque podemos volver a crearla.

Trabajando con bases de datos

Modificando una tabla



```
ALTER TABLE alumnos drop column  
nota3;  
  
ALTER TABLE alumnos add column  
nota3 integer;
```

Al modificar una tabla podemos agregar, borrar y cambiar el tipo de dato de una columna, entre otras acciones.



Al agregar columnas tenemos que especificar el tipo de dato.

Desafío - Introducción a base de datos



Desafío

"Introducción a bases de datos"

- Descarga el archivo "Desafío".
- Tiempo de desarrollo asincrónico: desde 2 horas.
- Tipo de desafío: individual
- Para desarrollar el desafío, deberás antes leer la guía de estudio e instalar Postgre SQL en tu computador.

¡AHORA TE TOCA A TI! 💪



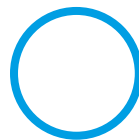
Ideas fuerza



Las **bases de datos** nos permiten almacenar **gran cantidad de datos**, por **largo tiempo**, y con **fácil acceso**.



PostgreSQL nos permite **consultar** datos de tablas, escoger algunos de ellos mediante **condiciones** y **ordenarlos** de acuerdo a requerimientos.



Dentro de una **tabla**, podemos **ingresar datos, cambiarlos, borrarlos** y **actualizar** la información.



Asimismo, en una **base de datos** podemos **crear** y **borrar** **tablas**.

¿Qué contenidos de la clase crees que debes reforzar?



Recursos asincrónicos

¡No olvides revisarlos!

Esta semana contarás con los siguientes recursos:

- Guía de estudio.
- Desafío “Introducción a las bases de datos”.





Próxima sesión...

- *Tutoría.*

{desafío}
latam_

*Academia de
talentos digitales*

