Consultas en SQL (Parte IV)

Clase sincrónica

{desafío} latam_

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

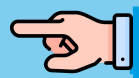
- Unidad 1: Primeros pasos del analista de datos
- Unidad 2: Consultas en SQL

(Parte I)

(Parte II)

(Parte III)

(Parte IV)



Te encuentras aquí

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Agregar una clave primaria.*
- *Agregar una clave foránea.*
- *Convertir un modelo de datos a una base de datos.*

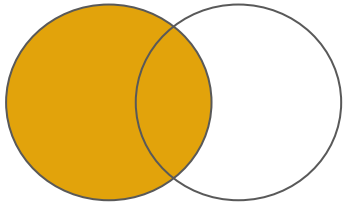
¿Qué recordamos de la consulta a múltiples tablas?

{desafío} latam_

Consultas a múltiples tablas

Recordemos

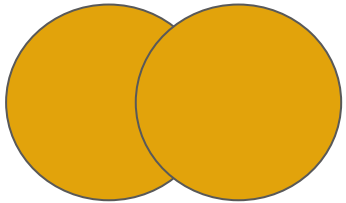
¿A qué tipo de JOIN corresponde el siguiente gráfico?



Consultas a múltiples tablas

Recordemos

¿A qué tipo de JOIN corresponde el siguiente gráfico?



Consultas a múltiples tablas

Recordemos

Si unimos con Inner Join todos los elementos de dos tablas, pero una de ella está vacía, ¿qué sucede?

{desafío} latam_

/* Integridad de datos */

Integridad de datos

¿De qué se trata?

- Como administradores de bases de datos debemos considerar 3 aspectos en la integridad de datos:
  a. Integridad referencial: Generada entre datos que se cruzan entre varias tablas.
  b. Integridad mediante clave primaria: Generada a través de un dato que contiene un valor único, por ejemplo la clave primaria (Primay Key, PK)
  c. Integridad mediante restricciones: Generada a partir de la restricción de los tipos de valores que se pueden almacenar en un campo de una tabla.
     - Por ejemplo, si tenemos un campo que es de tipo numérico, entonces la restricción al configurar el campo y la tabla, es definirlo como numérico. En este sentido, si intentamos ingresar texto, obtendremos un error.

Integridad de datos

Motivación

Integridad de datos = datos correctos + datos completos

- Problemas de diseño de una base de datos pueden causar problemas de integridad.
- En esta clase aprenderemos a evitar algunos de ellos.

Integridad de datos

Falta de identificador único

Nombre	Edad
Consuelo	27
Consuelo	32
Francisco	27

Queremos aumentar la edad de Consuelo, ¿cómo lo hacemos?

- Si actualizamos por edad, modificaremos 2 registros.
- Si actualizamos por nombre, modificaremos 2 registros.
- Es posible modificar por ambos, pero en algunas ocasiones se nos podría olvidar y causar un problema de correctitud de datos.

Integridad de datos

Solución: agregar un valor único

Si agregamos una columna con un valor distinto para cada registro, que no pueda repetirse por ningún motivo y que no pueda ser nulo, entonces tendríamos resuelto el problema.

Id	Nombre	Edad
1	Consuelo	27
2	Consuelo	32
3	Francisco	27

Integridad de datos

Asegurándose que el valor sea único y no nulo

Para asegurarnos que este valor realmente no pueda repetirse, ya que sería una amenaza a la integridad de los datos, especificaremos que esta columna es una **clave primaria (primary key, abreviado PK)**.

Id (PK)	Nombre	Edad
1	Consuelo	27
2	Consuelo	32
3	Francisco	27

Integridad de datos

Agregamos un identificador único

Para crear desde cero una tabla con PK	Para alterar una tabla existente

`CREATE TABLE users(`
`  id int primary key,`
`  nombre varchar,`
`  edad int`
`);`
	`/* Si ya tiene una columna para la PK */`
	`ALTER TABLE users`
	`  ADD PRIMARY KEY (id)`
	
	`/* Si no tiene la columna */`
	`ALTER TABLE users`
	`  ADD COLUMN ID PRIMARY KEY`

Esto no es necesario saberlo de memoria, puedes encontrarlo en Google.

Integridad de datos

Probando la solución

Id	Nombre	Edad
1	Consuelo	27
	Consuelo	32
3	Francisco	27
3	Javier	10

- Al establecer que un campo es clave primaria (Primary key, abreviado **PK**) nos aseguramos de que deba ser **único** y además, que no pueda ser **nulo**.
- Intentemos insertar dos usuarios con el mismo id y un usuario sin id.
- Veamos los errores.

Ejercicio "Claves primarias e integridad"

{desafío} latam_

Claves primarias e integridad

A continuación, realizaremos un ejercicio donde verificaremos los resultados de ejecución cuando insertamos datos en una tabla que contiene clave primaria.

1. Crea una tabla posts con id, título y contenido, dejando id como clave primaria.
2. Intenta ingresar 2 posts con el mismo id.
3. Intenta ingresar un post sin id.

```
create table posts (id int, title varchar(100), content varchar(100), primary key(id));
```

Integridad de datos

Integridad referencial

En otras ocasiones, nos preocupará que no se borre un registro relacionado con otro registro.

Usuarios

Id	Nombre	Edad
1	Consuelo	27
2	Consuelo	32
3	Francisco	27

Pago

Id	Monto	Usuario_id
1	1500	1
2	1300	2
3	2700	3

No queremos que se borre un usuario, porque quedaría un pago “zombie”.

Integridad de datos

Agregando una clave foránea

Una clave foránea (Foreign Key y abreviado FK) nos permite protegernos de esta situación. Una clave foránea debe apuntar a una clave única (y de preferencia primaria).

```
CREATE TABLE pagos (
  id int primary key,
  monto int,
  usuario_id int references users(id)
);
```

- Insertemos un pago asociado al usuario 1 y luego intentemos borrar el usuario 1.

Ejercicio
"Claves foráneas"

{desafío}
latam_

Claves foráneas

A continuación, realizaremos un ejercicio donde se analizará el comportamiento de las claves foráneas para datos que estén relacionados entre dos tablas. Para ello, realiza las siguientes acciones:

1. Crea la tabla comments con id, contenido y una clave foránea llamada post_id apuntando a la PK de posts.
2. Inserta un comentario asociado al post 1.
3. Intenta borrar el post 1.

```
create table comments (id int, content varchar(100), foreign key(id)
references posts(id),  primary key(id));
```

/* Modelos de datos*/

Modelos de datos

Introducción a modelos de datos

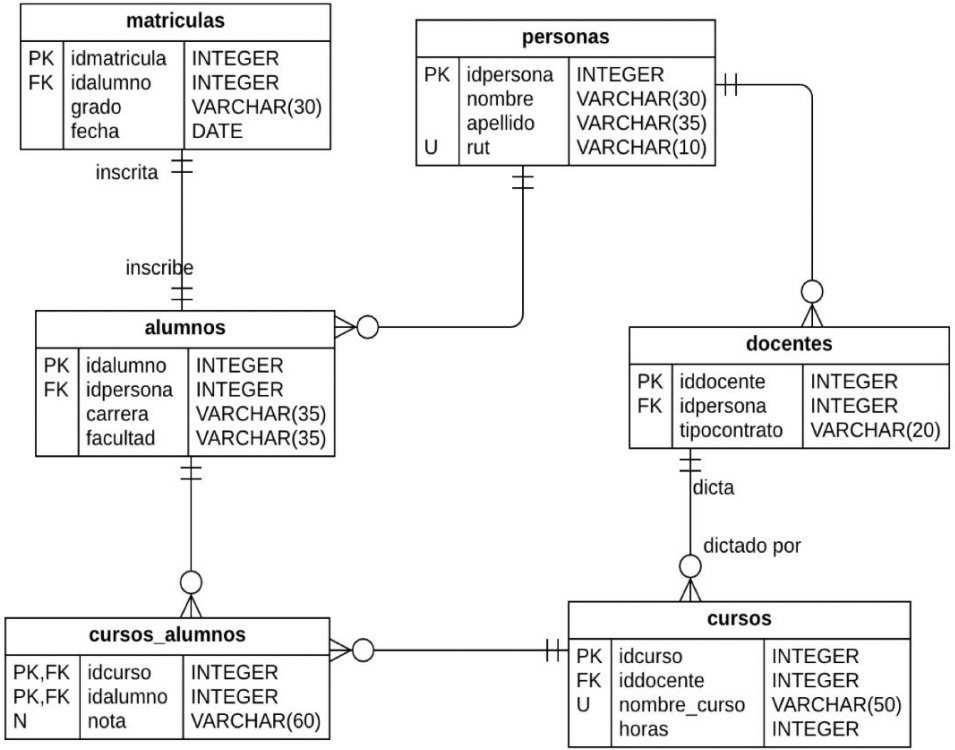
En ocasiones nos entregarán un modelo de datos en lugar de una tabla.
Dentro de los modelos de entidad relación (ER) existen 3 tipos de modelos más conocidos:

- Modelo conceptual
- Modelo lógico
- Modelo físico

Modelos de datos

Modelo físico

- El modelo físico de datos representa cómo se construirá el modelo de la base de datos.
- Aquí se especifican las tablas, columnas, claves primarias y foráneas, así como otras restricciones.

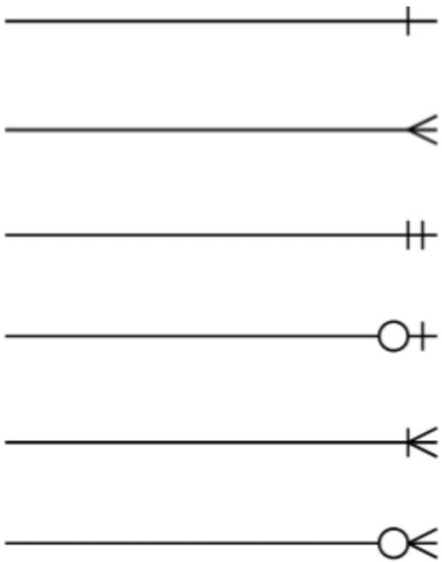


{desafío} latam_

Modelos de datos

Relaciones

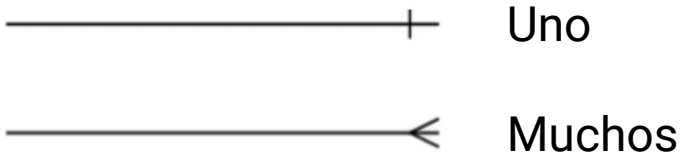
- Las relaciones entre tablas se indican con una línea cuyas terminaciones indican el tipo de relación.
- Como la relación de muchos tiene esa forma, a este tipo de notación de relaciones se le conoce como crow foot (pies de cuervo).



Modelos de datos

Relaciones

Por ahora nos enfocaremos en las relaciones de uno y de muchos.

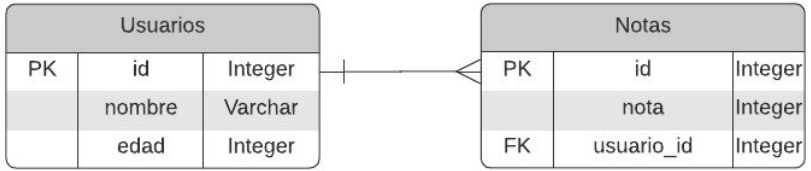


Modelos de datos

Cardinalidad

Se llama cardinalidad al tipo de relación que hay entre dos tablas. Principalmente, hablaremos de 3 casos:

1. 1 : 1 (Uno a uno)
2. 1 : N (Uno a muchos)
3. N : N (Muchos a muchos)



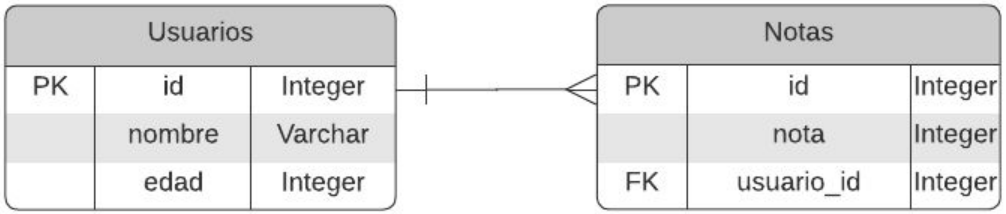
En nuestro ejemplo, tenemos una tabla de uno a muchos:

- Un usuario tiene muchas notas y una nota le pertenece a un único usuario.

Modelos de datos
Cardinalidad

Las tablas más utilizadas son las de 1 a N y las de N a N. Una tabla de 1 a N se implementa directamente tal como hemos hecho hasta ahora.

1. 1 : N (Uno a muchos)

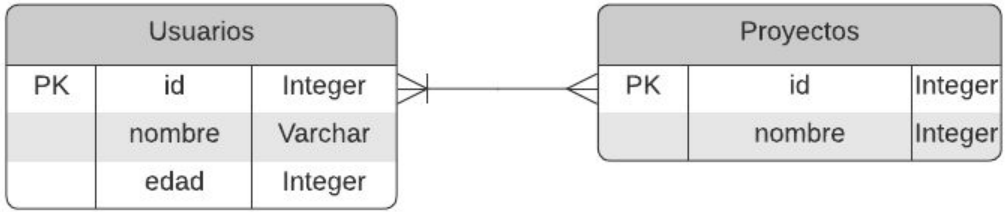


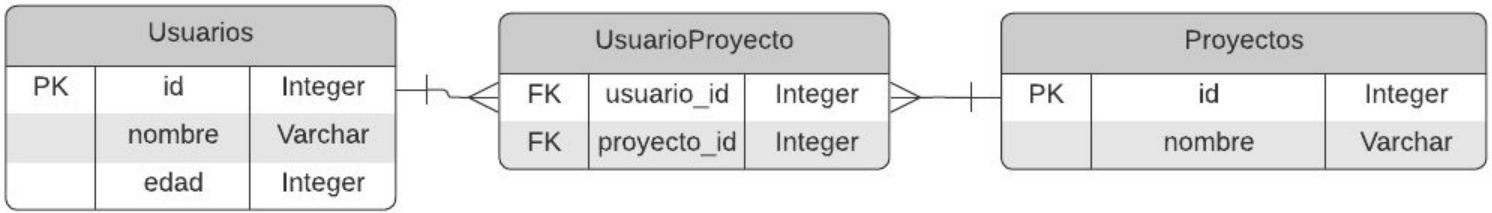
- La línea vertical del lado de la tabla usuario nos indica que necesariamente una nota le debe pertenecer a un usuario, ya que no puede haber una nota sin usuario.

Modelos de datos

Cardinalidad

Las relaciones N a N no pueden implementarse directamente, requieren de crear una tabla intermedia.



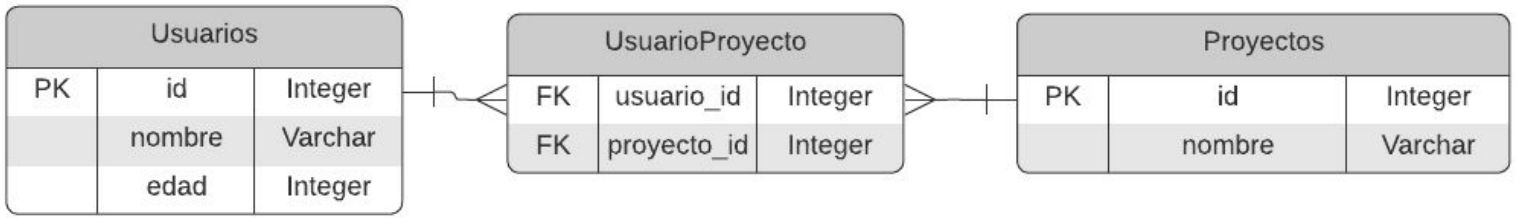


{desafío} latam_

En algunos casos las veremos ya separadas en los modelos.

Modelos de datos

N a N y tabla intermedia

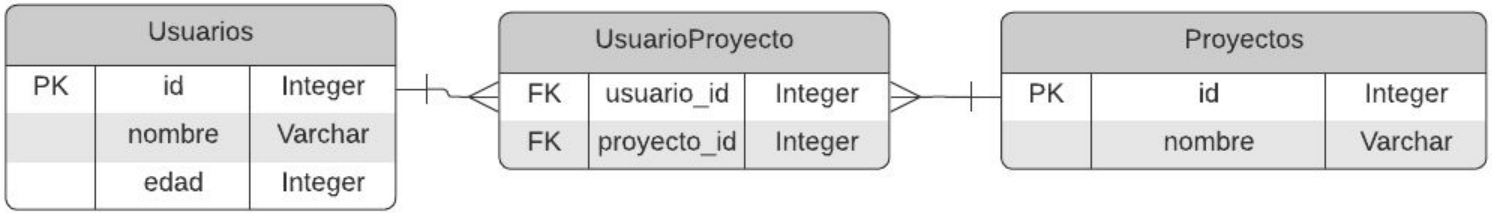


La lógica de la tabla intermedia siempre es igual entre 2 tablas de N a N: contiene las FK de las otras dos tablas y siempre queda del lado de los muchos.

Modelos de datos

N a N y tabla intermedia

Una vez que tenemos el modelo de esta forma podemos implementarlo con SQL.




```
CREATE TABLE
"Usuarios" (
"id" Integer,
"nombre" Varchar,
"edad" Integer,
 PRIMARY KEY ("id")
);
```

```
CREATE TABLE UsuarioProyecto (
usuario_id integer,
proyecto_id integer,
foreign key (usuario_id)
references
"Usuarios" (id),
foreign key (proyecto_id)
references
"Proyectos" (id)
);
```

```
CREATE TABLE "Proyectos" (
"id" Integer,
"nombre" Varchar,
PRIMARY KEY ("id")
);
```

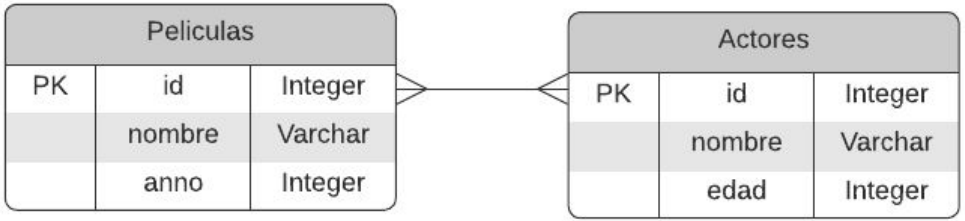
{desafío} latam_

Ejercicio
"Relaciones y tablas"

{desafío}
latam_

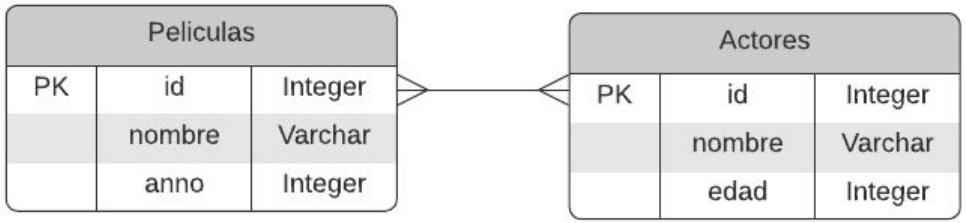
Relaciones y tablas

1. ¿Qué tipo de relación es?



Relaciones y tablas

Utilizando como referencia el modelo de datos N:N, dibuja el diagrama insertando una tercera tabla que sea intermediaria entre películas y actores.



Prueba - SQL

Prueba

SQL

- Descarga el archivo “Prueba”.
- Tiempo de desarrollo asincrónico: desde 4 horas.
- Tipo de desafío: individual.

¡AHORA TE TOCA A TI! 💪

{desafío} latam_

Ideas fuerza

*La **integridad de datos** consiste en contar con datos **correctos** y que además estén completos.*

*Una **clave primaria** puede agregarse para un registro y asegurar que no pueda repetirse, garantizando la corrección de los datos.*

*Una **clave foránea** nos permite evitar el borrado de datos particulares, y garantizar así que permanezcan completos.*

*Un **modelo físico de datos** nos permite representar las relaciones entre los datos y tablas que presentan diferentes **cardinalidades**. Con ello, se puede implementar una **base de datos**.*

¿Qué contenidos de la clase no te quedaron totalmente claros?

{desafío} latam_

Recursos asincrónicos

¡No olvides revisarlos!

Para esta semana encontrarás:

- Guía de estudio.
- Prueba “SQL”.

Próxima sesión...

- *Tutoría.*

{desafío}
latam_

Academia de
talentos digitales