MySQL

Digital House - Curso de Desarrollo Web Full Stack

Diagrama Entidad-Relación (DER)

El DER es un **modelo de datos conceptual** de alto nivel

Permite describir los datos presentes en un sistema de información utilizando los conceptos:

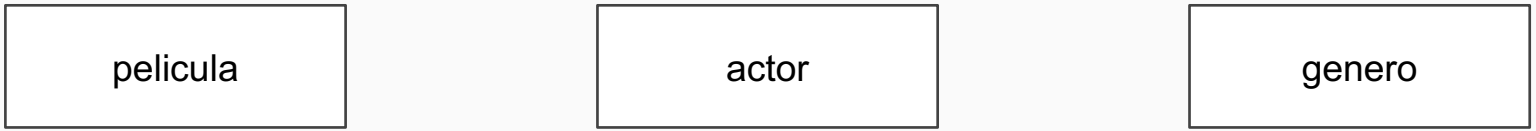
- Entidades
- Relaciones
- Atributos (incluyendo Claves)

DER - Entidad

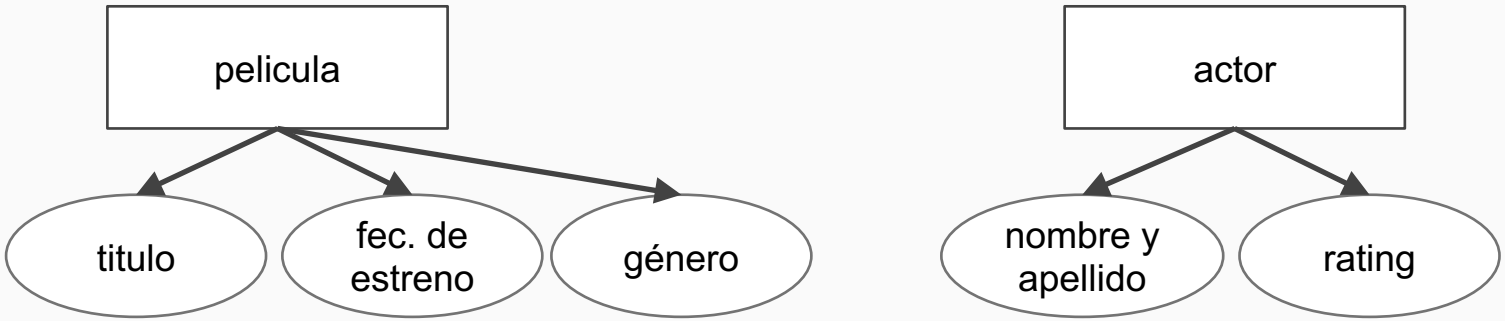
Representa un aspecto del mundo real. Puede ser un objeto concreto/real o abstracto/conceptual.

- Venta
- Animal
- Materia
- Alumno
- Profesor
- Pelicula
- Actor

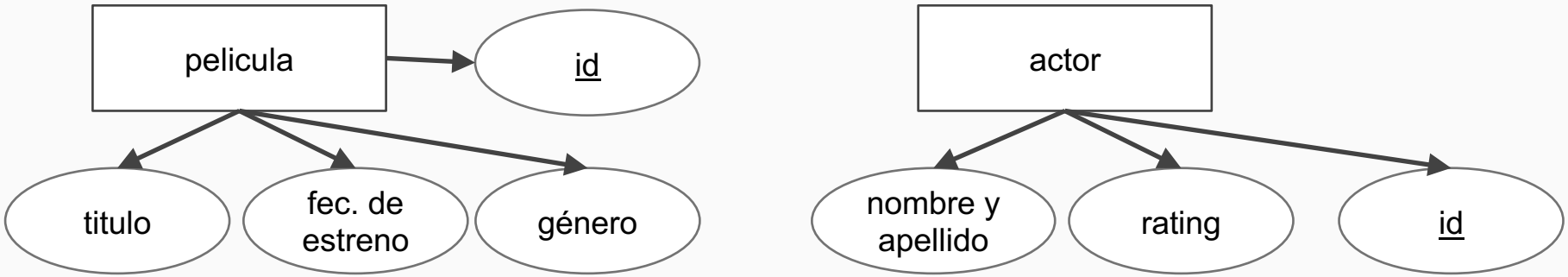
DER - Entidad



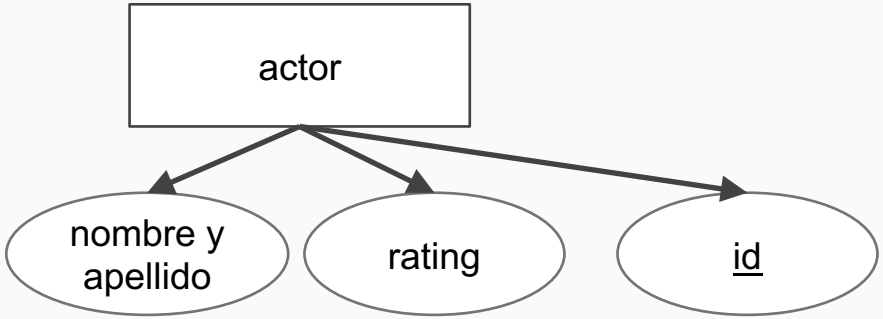
DER - Atributos



DER - Clave


pelicula
id
titulo
fec. de estreno
género
actor
nombre y apellido
rating
id

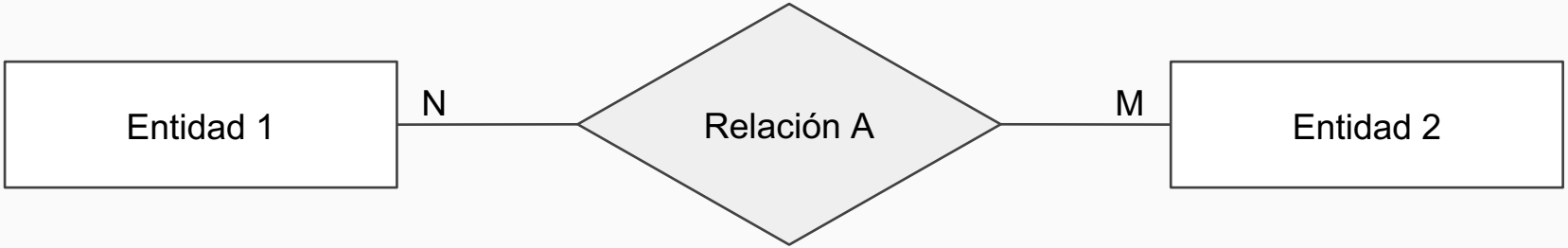
DER - UML



Actor
Id Nombre y apellido Rating

DER - Relación

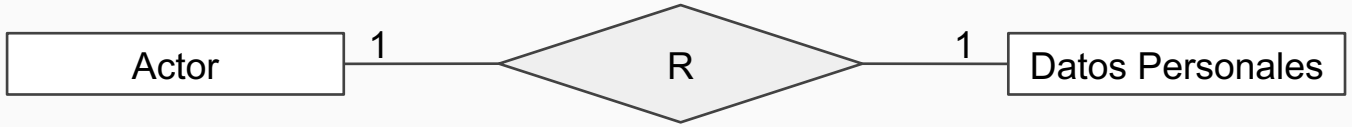
- Conexión lógica entre entidades
- Se representa con un rombo
- Pueden tener atributos propios



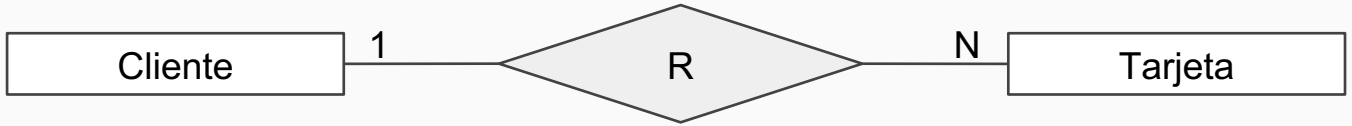
DER - Relación - Cardinaliad

Número de entidades con la cual otra entidad puede asociarse mediante una relación.

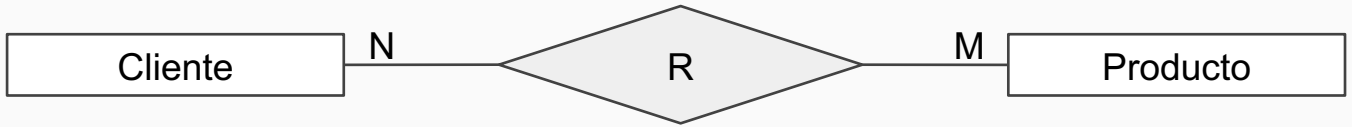
1:1



1:N

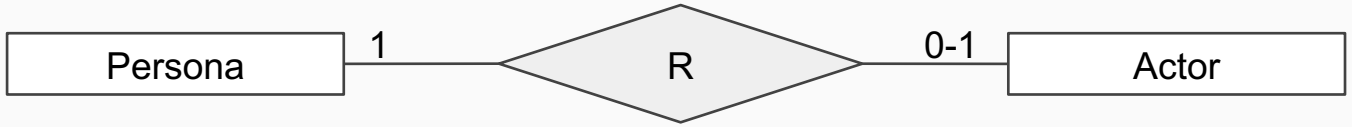


N:M

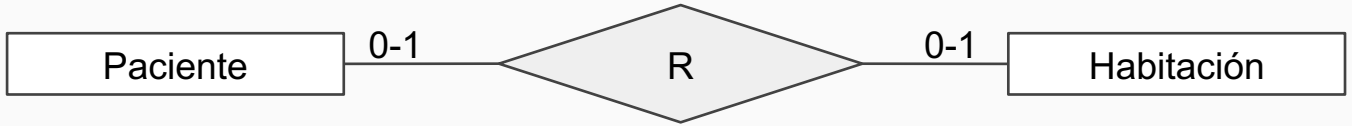


DER - Relación - Cardinaliad

1:0-1

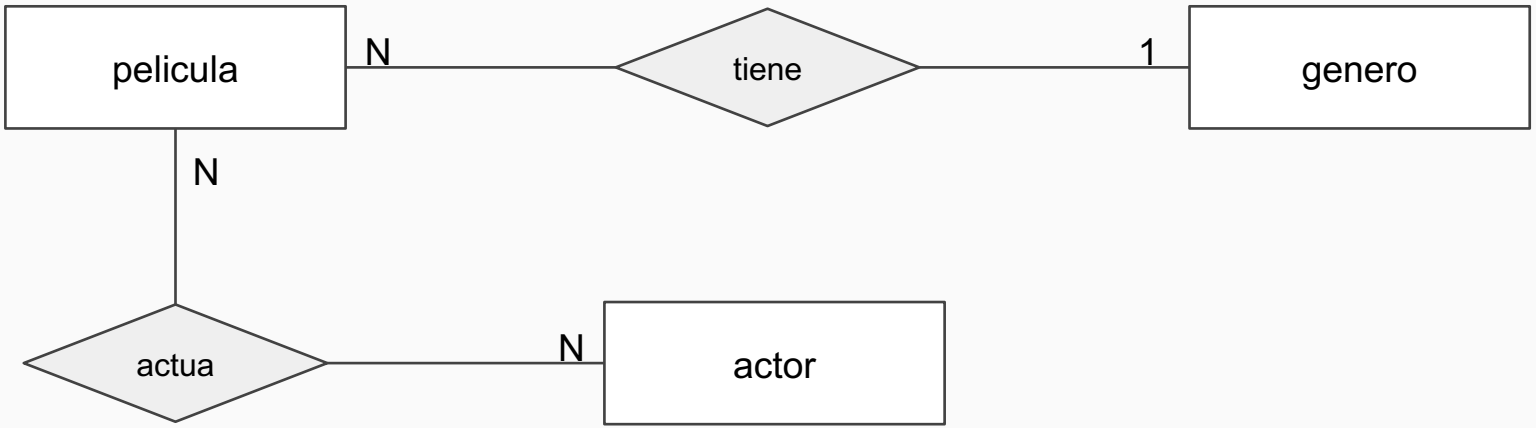


0-1:0-1



¿Cómo se relacionan las entidades?

¿Cómo se relacionan las tablas?



Ejercicio: Crear el DER que responda a los siguientes requerimientos

Un comercio desea implementar una BD para registrar las Ventas de Productos que realiza a sus clientes. A cada Cliente se lo identifica por un Número de Cliente y de ellos se guardan su Apellido, Nombre, Teléfono, Email, Fecha de Nacimiento y Barrio en donde se encuentra. De cada Barrio se sabe el País al que pertenece.

De los Productos que vende el comercio se conoce el Código que lo identifica, la cantidad de Unidades de dicho producto que se encuentran en stock, su Precio Unitario, el Nombre con el que se lo conoce y el País de Origen de dicho producto.

Cuando se realiza una Venta a un Cliente se registran su Fecha y Hora y la Forma de Pago utilizada por el cliente. La misma tendrá un detalle que incluirá de cada Producto vendido la cantidad de unidades que el cliente compró.

FOREIGN KEY (FK) - Clave Foránea

Una FK, asociada a un campo de una tabla, hace referencia a un campo de otra tabla, que suele ser la PK de la misma.

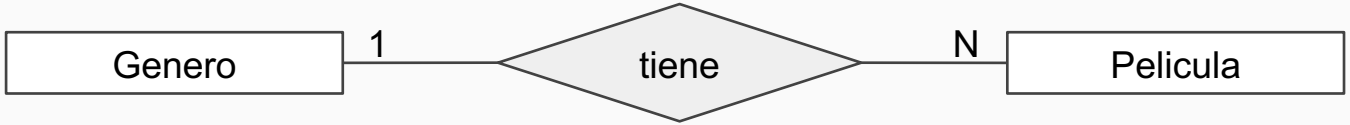
Permite garantizar la **Integridad Referencial** en una BD relacional → consistencia en la relación entre tablas

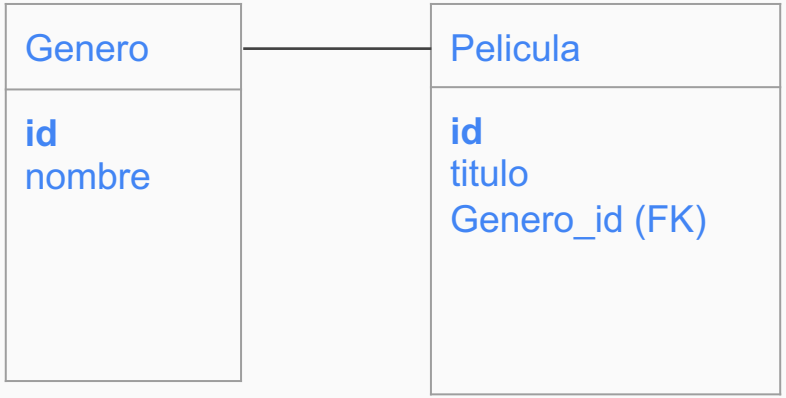
Identificación de Claves Foráneas

1:N

La Clave Primaria de la tabla referenciada se copia como Clave Foránea en la tabla que hace la referencia.

Identificación de Claves Foráneas



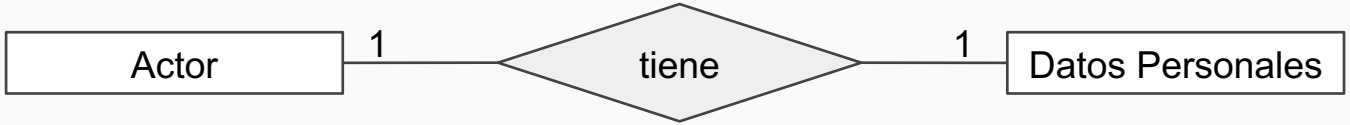


Identificación de Claves Foráneas

1:1

La Clave Primaria de la tabla referenciada se copia en la tabla que hace la referencia, que también será su Clave Primaria.

Identificación de Claves Foráneas



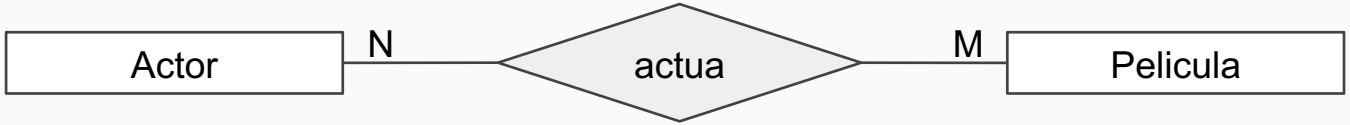
Identificación de Claves Foráneas

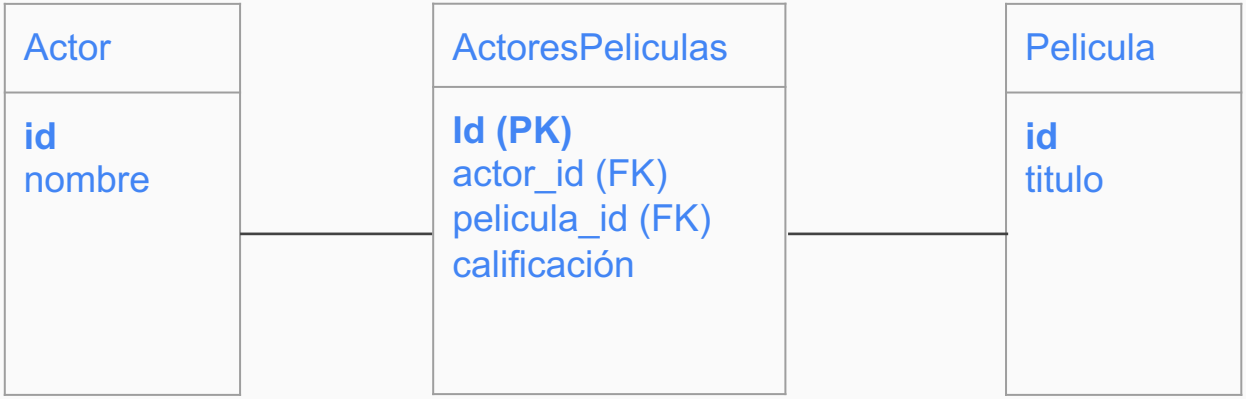
N:M

Se crea una nueva tabla para la Relación del DER.

- Las Claves Primarias de las Entidades (tablas) referenciadas se copian como Claves Foráneas en esta nueva tabla.
- Además, puede contener Atributos (campos) propios

Identificación de Claves Foráneas





FOREIGN KEY - Sintaxis

```
CONSTRAINT constraint_name
FOREIGN KEY (columns)
REFERENCES parent_table(columns)
```

FOREIGN KEY - Sintaxis

```
ALTER TABLE peliculas
    ADD genero_id INT UNSIGNED NULL,
    ADD CONSTRAINT `fk-genero-pelicula` FOREIGN KEY (genero_id) REFERENCES generos(id)
;
```

FOREIGN KEY - Sintaxis

```
CREATE TABLE `peliculas` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `titulo` varchar(500) NOT NULL,
  `rating` decimal(3,1) unsigned NOT NULL,
  `premios` int(10) unsigned NOT NULL DEFAULT '0',
  `fecha_de_estreno` date NOT NULL,
  `duracion` int(10) unsigned DEFAULT NULL,
  `genero_id` int(10) unsigned DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `fk-genero-pelicula` (`genero_id`),
  CONSTRAINT `fk-genero-pelicula` FOREIGN KEY (`genero_id`) REFERENCES `generos` (`id`)
)
```

FOREIGN KEY - Sintaxis

```
ALTER TABLE peliculas
    DROP KEY `fk-genero-pelicula`,
    DROP FOREIGN KEY `fk-genero-pelicula`
;
```

Normalización

Se realiza al **implementar** una BD relacional a partir de su modelo lógico o conceptual (DER). De esta manera se logra:

- ✓ Minimizar la **redundancia**
- ✓ Minimizar las anomalías de inserción, modificación y borrado
- ✓ Diseñar una BD que sea eficiente, cumpla con el modelo relevado y tenga los datos adecuadamente representados

Primera Forma Normal

Una tabla está en Primera Forma Normal (1FN) si:

- Contiene una **Clave Primaria** que identifica cada conjunto de datos relacionados
- La Clave Primaria y Claves Candidatas no tienen partes nulas
- Los atributos son “**atómicos**” y su semántica es simple
- No tiene grupos de datos repetidos
- Cada atributo no-clave **depende funcionalmente** de uno clave

Primera Forma Normal

Cliente

ID Cliente	Nombre	Apellido	Teléfono 1	Teléfono 2	Teléfono 3
123	Rachel	Ingram	555-861-2025		
456	James	Wright	555-403-1659	555-776-4100	
789	Cesar	Dure	555-808-9633		

Cliente

ID Cliente	Nombre	Apellido	Teléfono
123	Rachel	Ingram	555-861-2025
456	James	Wright	555-403-1659 555-776-4100
789	Cesar	Dure	555-808-9633

Cliente

ID Cliente	Nombre	Apellido
123	Rachel	Ingram
456	James	Wright
789	Cesar	Dure

Teléfono del cliente

ID Cliente	Teléfono
123	555-861-2025
456	555-403-1659
456	555-776-4100
789	555-808-9633

Segunda y Tercera Forma Normal

Una tabla está en Segunda Forma Normal (2FN) si:

- ✓ Está en 1FN y
- ✓ Cada atributo depende funcionalmente de **toda la Clave Primaria**

Una tabla está en Tercera Forma Normal (3FN) si:

- ✓ Está en 2FN y
- ✓ Ningún atributo depende **transitativamente** de una clave o parte de ella

Ejemplo para Normalizar

Estudios Realizados

Empleado	**DNI**	**Carrera**	Universidad	Domicilio
Martín Manzo	22222222	Ing. Industrial	Fac. Ingeniería UBA	Av. Paseo Colón 850 (C1063ACV), C.A.B.A.
Martín Manzo	22222222	Ing. Informática	Univ. de Buenos Aires	Av. Paseo Colón 850 (C1063ACV), C.A.B.A.
Pablo Perez	22333333	Ing. Industrial	UBA	Paseo Colón 850
Gregorio García	22444444	Lic. Administración	UADE	Lima 775, C1073AAO, Ciudad de Buenos Aires

Ejemplo para Normalizar

Empleados

Empleado	**DNI**
Martín Manzo	22222222
Pablo Perez	22333333
Gregorio García	22444444

Estudios Universitarios

DNI	**Carrera**	**Id_Universidad**
22222222	Ing. Industrial	1
22222222	Ing. Informática	1
22333333	Ing. Industrial	1
22444444	Lic. Administración	2

Universidades

Id_Universidad	Universidad	Domicilio	Localidad
1	Facultad de Ingeniería, UBA	Av. Paseo Colón 850 (C1063ACV)	C.A.B.A.
2	UADE	Lima 775, C1073AAO	C.A.B.A.

Práctica

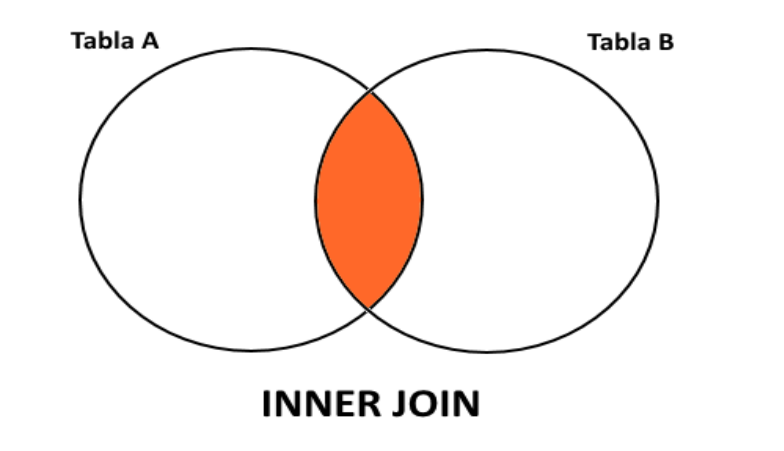
Relaciones en SQL

En una consulta SELECT donde se quieran mostrar campos de distintas tablas se deberán **relacionar** (“joinear”) las tablas involucradas.

- INNER JOIN
- LEFT JOIN
- Producto Cartesiano

INNER JOIN

Especifica una relación de **intersección** de las 2 tablas.



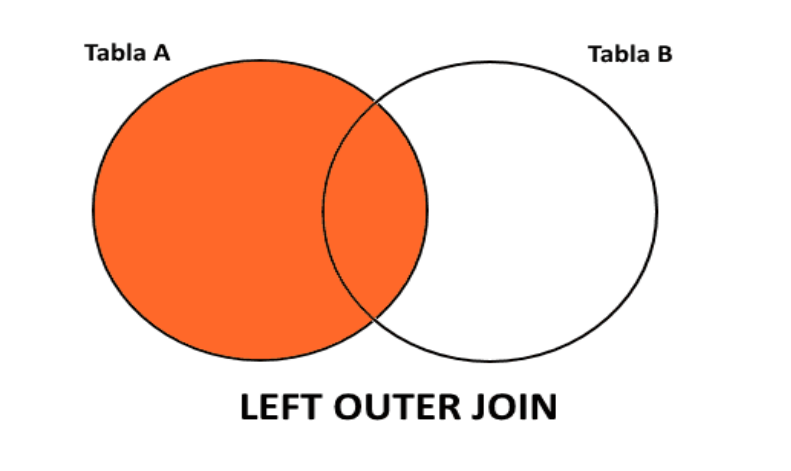
INNER JOIN

```
SELECT
    pelicula.id, pelicula.titulo, genero.nombre
FROM
    pelicula
        INNER JOIN genero ON genero.id = pelicula.id_genero
;
```

- Muestra ID y Titulo de las películas junto a su género.
- Si una pelicula no tiene un género, no la trae en la consulta.
- Tampoco trae un género si no existen películas relacionadas al mismo.

LEFT JOIN

Especifica una relación de **intersección** de las 2 tablas priorizando los datos de una de las tablas (la de la izquierda en la consulta)



LEFT JOIN

```
SELECT genero.id, genero.nombre, pelicula.id
FROM genero
    LEFT OUTER JOIN pelicula ON pelicula.id_genero = genero.id
```

- Muestra el ID y Nombre de todos los géneros junto a los ID de las películas relacionadas a cada uno de ellos.
- A un género lo muestra tantas veces como películas relacionadas tenga.

Producto Cartesiano

```
SELECT genero.id, genero.nombre, pelicula.id, pelicula.titulo
FROM genero, pelicula
```

- Muestra todos los géneros (ID y Nombre) junto a todas las películas (ID y Titulo), sin importar que exista una intersección entre ambas tablas.

- Mediante el WHERE se pueden filtrar las tablas para que muestre lo que se precisa de cada una.

Práctica