

Bases de Datos

Clase 3: Repaso de llaves + SQL

Modelo Relacional

Los datos se almacenan como tablas:

Películas

ID Película	Nombre Película	Año	Categoría	Calificación (IMDB)
1	Interstellar	2014	Fantasía	8.6
2	The Revenant	2015	Drama	8.1
3	The Imitation Game	2014	Biografía	8.1
4	The Theory of Everything	2014	Biografía	7.7

Distinguimos:

- **Relaciones:** a cada tabla le llamamos relación
- **Atributos:** son las columnas de la relación
- **Tuplas:** son las filas de la relación

Modelo Relacional

Una **Relacion** es un ***conjunto*** de tuplas!!!

- Quiere decir; no hay filas repetidas

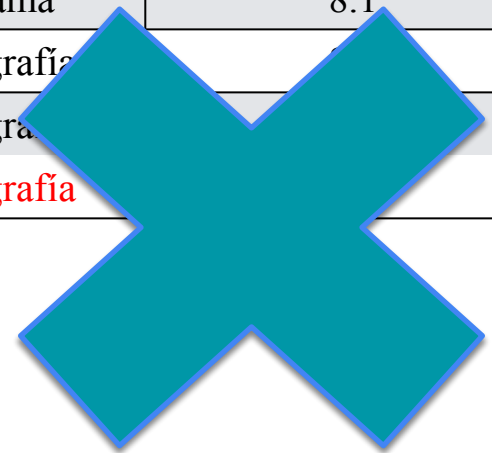
ID Película	Nombre Película	Año	Categoría	Calificación (IMDB)
1	Interstellar	2014	Fantasía	8.6
2	The Revenant	2015	Drama	8.1
3	The Imitation Game	2014	Biografía	8.1
4	The Theory of Everything	2014	Biografía	7.7
4	The Theory of Everything	2014	Biografía	7.7

Modelo Relacional

Una **Relacion** es un ***conjunto*** de tuplas!!!

- Quiere decir; no hay filas repetidas

ID Película	Nombre Película	Año	Categoría	Calificación (IMDB)
1	Interstellar	2014	Fantasia	8.6
2	The Revenant	2015	Drama	8.1
3	The Imitation Game	2014	Biografía	
4	The Theory of Everything	2014	Biografía	
4	The Theory of Everything	2014	Biografía	



Esquema vs Instancia

Esquema

Esquema: especifica nombre de la tabla y sus atributos

Películas(id, nombre, año, categoría, calificación)

Esquema vs Instancia

Instancia

Una **instancia** de un **esquema** es un conjunto de tuplas para cada relación del esquema

Películas(id, nombre, año, categoría, calificación)

ID Película	Nombre Película	Año	Categoría	Calificación (IMDB)
1	Interstellar	2014	Fantasía	8.6
2	The Revenant	2015	Drama	8.1
3	The Imitation Game	2014	Biografía	8.1
4	The Theory of Everything	2014	Biografía	7.7

Modelo Relacional

Restricciones de integridad

Son restricciones que imponemos a un esquema que *todas* las instancias deben satisfacer

La restricción más importante son las **llaves**

Intuitivamente una llave permite identificar, de manera única, a una tupla de nuestra relación

Modelo Relacional

Ejemplo

Una tabla con información de vinos.

Esquema: Vinos(Productor, Cepa, Gama, Año, Grados, Orígen)

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Tarapacá	Merlot	Reserva	2011	14	San Pedro
Viu Manent	Carmenere	Gran Reserva	2014	12	Colchagua
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	14	Maipo
...

Modelo Relacional

Restricciones de integridad

Son restricciones que imponemos a un esquema que *todas* las instancias deben satisfacer

La restricción más importante son las **llaves**

Intuitivamente una llave permite identificar, de manera única, a una tupla de nuestra relación

Hay distintas categorías de llaves!

Modelo Relacional

Super llave

La **super llave** para una relación R es:

un conjunto de atributos de R tal que no pueden existir dos tuplas en R con los mismos valores de estos atributos

Intuitivamente: si conozco los valores de atributos de la super llave, puedo identificar de **forma única** a la tupla de mi relación

Modelo Relacional

Superllave -- ejemplo

El conjunto de todos los atributos de mi relación siempre forman una super llave!

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Tarapacá	Merlot	Reserva	2011	14	San Pedro
Viu Manent	Carmenere	Gran Reserva	2014	12	Colchagua
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	14	Maipo
...

Otras super llaves: (Productor, Cepa, Gama, Año, Grados)
(Productor, Cepa, Gama, Año, Orígen)

Modelo Relacional

Superllave -- ejemplo

Como **(Productor, Cepa, Gama, Año, Origen)** es superllave, lo siguiente no está permitido en una instancia:

Productor	Cepa	Gama	Año	Grados	Origen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Tarapaca	Merlot	Reserva	2011	14	San Pedro
Viu Manent	Carmenere	Gran Reserva	2014	12	Colchagua
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	14	Maipo
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	13	Maipo

Modelo Relacional

Llave (candidata)

La **llave (o llave candidata)** para una relación R es:

un conjunto de atributos de R que es una super llave de R , y no existe un subconjunto propio de estos atributos que sea una super llave

Intuitivamente: una super llave que no se puede achicar!

Modelo Relacional

Llave -- ejemplo

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Tarapacá	Merlot	Reserva	2011	14	San Pedro
Viu Manent	Carmenere	Gran Reserva	2014	12	Colchagua
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	14	Maipo
...

(Productor, Cepa, Gama, Año)

Modelo Relacional

Llave primaria

La **llave primaria** es una llave candidata que queremos destacar, y la subrayamos en el esquema.

Modelo Relacional

Llave primaria -- ejemplo

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Tarapaca	Merlot	Reserva	2011	14	San Pedro
Viu Manent	Carmenere	Gran Reserva	2014	12	Colchagua
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	14	Maipo
...

Vinos(Productor, Cepa, Gama, Año, Grados, Orígen)

Modelo Relacional

Si defino mal la llave primaria habrá problemas!

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2014	14	Maipo
...

Vinos(Productor, Cepa, Gama, Año, Grados, Orígen)

Un productor no puede producir vino de misma cepa y gama en distintos años!

Modelo Relacional

Si defino mal la llave primaria tendré problemas!

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Carmenere	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2014	14	Maipo
...

Vinos(Productor, Cepa, Gama, Año, Grados, Orígen)

Un productor no puede producir vinos de distinta cepa o gama en un año!

Llaves

Terminología

Super llave (superkey): cualquier conjunto de atributos que determina a todo el resto

Llave (candidata): cualquier conjunto de atributos que determina a todo el resto, y ninguno de sus subconjuntos es una super llave

Llave primaria: una llave candidata que queremos destacar (la subrayada en el esquema)

Modelo Relacional

Llaves

Mejor tipo de llave: ID único de la tupla!!!!

En la tabla Vinos no existía!

Surrogate Key

Surrogate key: una llave genérica que simplifica cosas

- **id**

Vinos(id, Productor, Cepa, Gama, Año, Grados, Origen)

Hay que tener cuidado con la lógica de la relación!

Llaves

Ejemplo

Persona(id, rut, nombre)

Llave primaria: id

Llaves

Ejemplo

Persona(id, rut, nombre)

Llave primaria: id

Llaves candidatas:

- id
- rut

Llaves

Ejemplo

Persona(id, rut, nombre)

Llave primaria: id

Llaves candidatas:

- id
- rut

Superllaves:

- id
- rut
- id,rut
- id,nombre
- rut,nombre
- id,rut,nombre

Llaves

Otro ejemplo

Cervezas(Nombre, Tipo, Grados, Ciudad-Origen)

Nombre	Tipo	Grados	Ciudad-Origen
Austral Lager	Lager	4.6	Punta Arenas
Austral Yagan	Ale	5.0	Punta Arenas
Kross 5	Ale	7.2	Curacavi
Kuntsmann Torobayo	Ale	5.0	Valdivia

Vinos(Nombre, Tipo, Año, Grados, Ciudad-Origen)

Nombre	Tipo	Año	Grados	Ciudad-Origen
Tarapacá	Carmenere	2014	13.5	Maipo
Tarapacá	Merlot	2014	13.5	Maipo
Gato	Merlot	2016	14.0	Maule

En-Stock(Nombre, Cantidad, Precio-unitario)

Nombre	Cantidad	Precio unitario
--------	----------	-----------------

¿Cuál es la llave primaria más natural?

Llaves

Otro ejemplo

Cervezas(Nombre, Tipo, Grados, Ciudad-Origen)

Nombre	Tipo	Grados	Ciudad-Origen
Austral Lager	Lager	4.6	Punta Arenas
Austral Yagan	Ale	5.0	Punta Arenas
Kross 5	Ale	7.2	Curacavi
Kuntsmann Torobayo	Ale	5.0	Valdivia

En-Stock(Nombre, Cantidad, Precio-unitario)

Nombre	Cantidad	Precio unitario
--------	----------	-----------------

Vinos(Nombre, Tipo, Año, Grados, Ciudad-Origen)

Nombre	Tipo	Año	Grados	Ciudad-Origen
Tarapacá	Carmenere	2014	13.5	Maipo
Tarapacá	Merlot	2014	13.5	Maipo

Gato	Merlot	2016	14.0	Maule
------	--------	------	------	-------

Llaves

Otro ejemplo -- Alternativa 1: nombre mas especifico para vinos

Cervezas(Nombre, Tipo, Grados, Ciudad-Origen)

Nombre	Tipo	Grados	Ciudad-Origen
Austral Lager	Lager	4.6	Punta Arenas
Austral Yagan	Ale	5.0	Punta Arenas

Vinos(Nombre, Tipo, Año, Grados, Ciudad-Origen)

Nombre	Tipo	Año	Grados	Ciudad-Origen
Tarapacá Carmenere 2014	Carmenere	2014	13.5	Maipo
Tarapacá Merlot 2014	Merlot	2014	13.5	Maipo

En-Stock(Nombre, Cantidad, Precio-unitario)

Nombre	Cantidad	Precio-Unitario
Tarapacá Carmenere 2014	600	2000

Llaves

Otro ejemplo -- Alternativa 2: crear un id

Cervezas(id, Nombre, Tipo, Grados, Ciudad-Origen)

id	Nombre	Tipo	Grados	Ciudad-Origen
CauL00	Austral Lager	Lager	4.6	Punta Arenas
CauY00	Austral Yagan	Ale	5.0	Punta Arenas

Vinos(id, Nombre, Tipo, Año, Grados, Ciudad-Origen)

id	Nombre	Tipo	Año	Grados	Ciudad-Origen
VTTC14	Tarapacá	Carmenere	2014	13.5	Maipo
VTTM14	Tarapacá	Merlot	2014	13.5	Maipo

En-Stock(id, Cantidad, Precio-unitario)

id	Cantidad	Precio-Unitario
CAuL00	600	2000
VTTC14	200	6000

Llaves

Otro ejemplo – Alternativa 3: tablas En-Stock separadas

Cervezas(Nombre, Tipo, Grados, Ciudad-Origen)

Nombre	Tipo	Grados	Ciudad-Origen
Austral Lager	Lager	4.6	Punta Arenas
Austral Yagan	Ale	5.0	Punta Arenas
...

Cerveza-En-Stock(Nombre, Cantidad, Precio-Unitario)

Nombre	Cantidad	Precio-Unitario
Austral Lager	600	2000

Vinos(Nombre, Tipo, Año, Grados, Ciudad-Origen)

Nombre	Tipo	Año	Grados	Ciudad-Origen
Tarapacá	Carmenere	2014	4.6	Punta Arenas
Tarapacá	Merlot	2014	5.0	Punta Arenas
Gato	Merlot	2016	14.0	Maule

Vino-En-Stock(Nombre, Tipo, Año, Cantidad, Precio-Unitario)

Nombre	Tipo	Año	Cantidad	Precio-Unitario
Tarapacá	Merlot	2014	200	6500

Llaves

Otro ejemplo – Alternativa 4: combinar las tablas

Cervezas(Nombre, Tipo, Grados, Ciudad-Origen, Cantidad, Precio-Unitario)

Nombre	Tipo	Grados	Ciudad-Origen	Cantidad	Precio-Unitario
Austral Lager	Lager	4.6	Punta Arenas	300	2000
Austral Yagan	Ale	5.0	Punta Arenas	600	?

Vinos(Nombre, Tipo, Año, Grados, Ciudad-Origen, Cantidad, Precio-Unitario)

Nombre	Tipo	Año	Grados	Ciudad-Origen	Cantidad	Precio-Unitario
Tarapacá	Carmenere	2014	4.6	Punta Arenas	200	6000
Tarapacá	Merlot	2014	5.0	Punta Arenas	?	6500
Gato	Merlot	2016	14.0	Maule	150	?

Dudas sobre llaves?

Hasta ahora

- Tenemos un lenguaje teórico para realizar consultas a relaciones
- Queremos un programa con tablas y un lenguaje de consultas para utilizar en la práctica.

Relational

Data

Base

Management

System



Cómo funciona un
RDBMS ?

DBMS Relacional

- Un DBMS relacional es un programa que se instala en un computador (**servidor**)
- Este programa se mantiene escuchando conexiones
- El usuario (generalmente otro programa) se conecta (**cliente**) al programa y le puede entregar instrucciones.

DBMS Relacional



DBMS Relacional

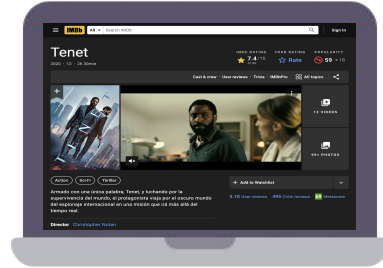


<https://www.imdb.com/title/tt6723592/>

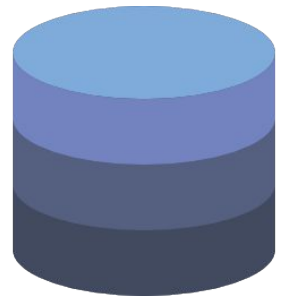
DBMS Relacional



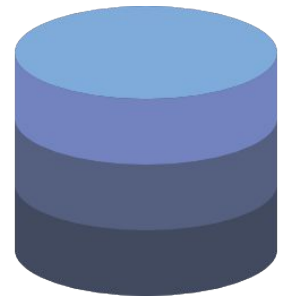
DBMS Relacional



DBMS Relacional

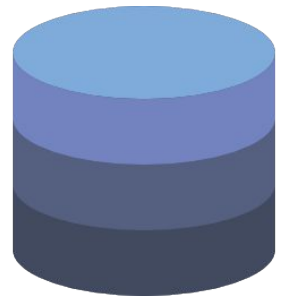


DBMS Relacional



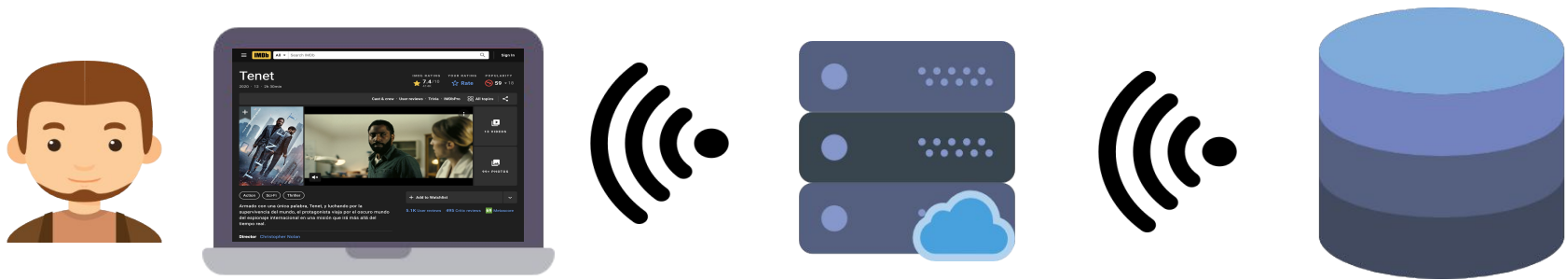
```
SELECT name, score  
FROM titles  
WHERE title.id='tt6723592'
```

DBMS Relacional



name	score
Tenet	7.4

DBMS Relacional



name	score
Tenet	7.4

SQL

Structured Query Language

- Usado para todas las comunicaciones con bases de datos relacionales.
- Aplicaciones web, locales, móviles, análisis de datos, etc.
- Hasta algunas arquitecturas serverless lo requieren o usan [lenguajes basados en SQL](#).

SQL

Structured Query Language

- Último estándar SQL99 (SQL3)
- Softwares implementan “subconjunto” del estándar (cada uno tiene diferencias sutiles)
- Lenguaje declarativo

Declarativo vs. Procedural

- SQL es declarativo, decimos lo que queremos, pero sin dar detalles de cómo lo computamos
- El DBMS transforma la consulta SQL en un algoritmo ejecutado sobre un lenguaje procedural
- Un lenguaje como Java es procedural: para hacer algo debemos indicar paso a paso el procedimiento

SQL

Structured Query Language

- DDL: Lenguaje de definición de datos
 - Crear y modificar tablas, atributos y llaves
- DML: Lenguaje de manipulación de datos
 - Consultar una o más tablas
 - Insertar, eliminar, modificar tuplas

En este curso

Durante el curso vamos a usar un motor RDBMS:

- PostgreSQL (PSQL)

PSQL



PSQL es un sistema relacional *open source*

Tiene varias funcionalidades avanzadas, como por ejemplo el uso de procedimientos almacenados o el almacenamiento de JSON

PSQL



PSQL va a ser usado en el proyecto y parte de la cátedra

Cada grupo tendrá acceso a un servidor en el que dispondrán de una base de datos a la que su aplicación se va a conectar

Los datos son almacenados en múltiples archivos en carpetas ocultas del computador

SQL

SQL

Tipos de datos

- Caracteres (Strings)
 - `char(20)` - Largo fijo
 - `varchar(20)` - Largo variable
- Números
 - `int`, `smallint`, `float`, ...
- Tiempo y fecha
 - `time` - hora formato 24 hrs.
 - `date` - fecha
 - `timestamp` - fecha + hora
- Y varios otros! Dependen del RDBMS que se esté usando.

SQL

Creando un esquema

Consideremos el esquema de ejemplo:

Peliculas(id, nombre, año, categoria, calificacion, director)

Actores(id, nombre, edad)

Actuo_en(id_actor, id_pelicula)

SQL

Crear Tablas

```
CREATE TABLE Peliculas(  
    id int,  
    nombre varchar(30),  
    año int,  
    categoria varchar(30),  
    calificacion float,  
    director varchar(30)  
)
```

SQL

Crear Tablas

```
CREATE TABLE Peliculas(  
    id int,  
    nombre varchar(30),  
    año int,  
    categoria varchar(30),  
    calificacion float,  
    director varchar(30)  
)
```

```
CREATE TABLE Actores(  
    id int,  
    nombre varchar(30),  
    edad int  
)
```

SQL

Crear Tablas

```
CREATE TABLE Peliculas(  
    id int,  
    nombre varchar(30),  
    año int,  
    categoria varchar(30),  
    calificacion float,  
    director varchar(30)  
)
```

```
CREATE TABLE Actores(  
    id int,  
    nombre varchar(30),  
    edad int  
)
```

```
CREATE TABLE Actuo_en(  
    id_actor int,  
    id_pelicula int,  
)
```


SQL

Crear Tablas con llaves

```
CREATE TABLE Peliculas(  
    id int PRIMARY KEY,  
    nombre varchar(30),  
    año int,  
    categoria varchar(30),  
    calificacion float,  
    director varchar(30)  
)
```

```
CREATE TABLE Actores(  
    id int PRIMARY KEY,  
    nombre varchar(30),  
    edad int  
)
```

```
CREATE TABLE Actuo_en(  
    id_actor int,  
    id_pelicula int,  
    PRIMARY KEY (id_pelicula, id_actor)  
)
```

SQL

Valores Default

Sintaxis general:

```
CREATE TABLE <Nombre> (...<atr> tipo DEFAULT <valor>...)
```

Ejemplo:

```
CREATE TABLE Peliculas(  
    id int PRIMARY KEY,  
    nombre varchar(30),  
    año int,  
    categoria varchar(30) DEFAULT 'Acción',  
    calificacion float DEFAULT 0,  
    director varchar(30)  
)
```

SQL

Modificar Tablas

Eliminar tabla:

```
DROP TABLE Peliculas
```

Eliminar atributo:

```
ALTER TABLE Peliculas DROP COLUMN director
```

Agregar atributo:

```
ALTER TABLE Peliculas ADD COLUMN productor varchar(30)
```

SQL

Insertar Datos

Sintaxis general:

```
INSERT INTO R(at_1, ..., at_n) VALUES (v_1, ..., v_n)
```

Ejemplo:

```
INSERT INTO Peliculas(id, nombre, año, categoria, calificacion,  
director) VALUES (321351, 'V for Vendetta', 2005, 'Action', 8.2  
, 'James McTeigue')
```

Ejemplo abreviado (asume orden de creación):

```
INSERT INTO Peliculas VALUES (321351, 'V for Vendetta',  
2005, 'Action', 8.2, 'James McTeigue')
```

Consultando con SQL

SQL

Forma básica

Las consultas en general se ven:

SQL

Forma básica

Las consultas en general se ven:

SELECT atributos

FROM relaciones

WHERE condiciones

SQL

Forma básica

Para ver todo de una tabla (en este caso película):

```
SELECT * FROM Peliculas
```

Para ver nombre y calificación de todas las películas dirigidas por Nolan:

```
SELECT nombre, calificacion  
FROM Peliculas  
WHERE director = 'C. Nolan'
```


SQL

Forma básica

Para las películas estrenadas desde el 2010:

```
SELECT *  
FROM Peliculas  
WHERE año >= 2010
```

El **WHERE** permite =, <>, !=, >, <, <=, >=, AND, OR, NOT, IN, BETWEEN, etc...

SQL

Forma básica

El **WHERE** permite =, <>, !=, >, <, <=, >=, AND, OR, NOT, IN, BETWEEN, etc...

Películas estrenadas entre
1971 y 1978:

```
SELECT *  
FROM Peliculas  
WHERE año BETWEEN 1971 AND 1978
```

Películas estrenadas en
1971, 1973 y 2001:

```
SELECT *  
FROM Peliculas  
WHERE año IN (1971, 1973, 2001)
```

SQL

En General

La consulta:

```
SELECT a_1, ..., a_n  
FROM T_1, ..., T_m  
WHERE <condicion>
```

Se traduce al álgebra relacional como:

$$\pi_{a_1, \dots, a_n}(\sigma_{condiciones}(T_1 \times \dots \times T_m))$$

Update

Para actualizar valores de una tabla:

```
UPDATE Peliculas  
SET calificacion = 0  
WHERE name = 'Sharknado 6'
```

Update

Forma general

UPDATE R

SET <Nuevos valores>

WHERE <Condición sobre R>

<Nuevos valores> \rightarrow (atributo₁ = nuevoValor₁, ..., atributo_n = nuevoValor_n)

Delete

Para borrar filas que cumplan una condición:

```
DELETE FROM R
```

```
WHERE <Condición sobre R>
```

¿Qué pasa si se nos olvida el `WHERE` en un `UPDATE` o `DELETE FROM`?

```
UPDATE Users
```

```
SET password = 'contraseña'
```

```
DELETE FROM Tweets
```

... ¿O si se nos pasa un ';' entre medio?

```
UPDATE Users
```

```
SET password = 'contraseña';
```

```
WHERE email = 'some@email.com'
```

```
DELETE FROM Tweets;
```

```
WHERE user_name = 'realDonaldTrump'
```

¿Qué pasa si se nos olvida el **WHERE** en un **UPDATE** o **DELETE FROM**?
... ¿O si se nos pasa un **;** entre medio?

Se borran / actualizan todas las filas!!



Producto cruz

Si pedimos datos de más de una tabla la base de datos va hacer un producto cruz y entregará **$n \times m$** filas.

```
SELECT *  
FROM Peliculas, Actuo_en
```

SQL

Joins

Podemos hacer un join agregando un **WHERE**

Por ejemplo, para obtener todas las películas junto a los ids de los actores que participaron en ella:

```
SELECT *  
FROM Peliculas, Actuo_en  
WHERE id = id_pelicula
```

Observación: id es atributo de Peliculas, mientras que id_pelicula es atributo de Actuo_en

SQL

Joins - Desambiguando atributos

Entregue todas las películas junto a los id de los actores que participaron en ella:

```
SELECT *  
FROM Peliculas, Actuo_en  
WHERE Peliculas.id = Actuo_en.id_pelicula
```

Sirve cuando tenemos atributos en distintas tablas con el mismo nombre y para agregarle claridad a la consulta.

SQL

Joins

¿Y si queremos los nombres de los actores en vez de los ids?

```
SELECT Peliculas.nombre, Actores.nombre  
FROM Peliculas, Actuo_en, Actores  
WHERE Peliculas.id = Actuo_en.id_pelicula  
AND Actores.id = Actuo_en.id_actor
```

SQL

Alias

Podemos acortar la consulta anterior:

```
SELECT p.nombre, a.nombre  
FROM Peliculas as p, Actuo_en as ae, Actores as a  
WHERE p.id = ae.id_pelicula  
AND a.id = ae.id_actor
```

Ese tipo de alias no es muy recomendable

SQL

Alias

Podemos hacer operaciones y nombrar la columna:

```
SELECT (nombre || ' dirigida por ' || director) as credits, año  
FROM Peliculas
```

credits	año
V for Vendetta dirigida por James McTeigue	2005
Dunkirk dirigida por C. Nolan	2017

SQL

Ordenando

Entregue el nombre y la calificación de todas las películas (orden ascendente):

```
SELECT nombre, calificacion  
FROM Peliculas  
ORDER BY nombre, calificacion
```

El i-ésimo atributo del ORDER BY resuelve un empate en el atributo i-1

SQL

Ordenando

Entregue el nombre y la calificación de todas las películas (orden descendente):

```
SELECT nombre, calificacion  
FROM Peliculas  
ORDER BY nombre DESC, calificacion
```


SQL

Union

Entregue el nombre de todos actores y directores:

```
SELECT nombre  
FROM Actores  
UNION  
SELECT director  
FROM Peliculas
```

SQL

Operadores de conjuntos

- **EXCEPT**: diferencia del álgebra
- **UNION**: unión del álgebra
- **INTERSECT**: intersección del álgebra
- **UNION ALL**: unión que admite duplicados

SQL

Matching de patrones con LIKE

`s LIKE p`: string `s` es como `p`, donde `p` es un patrón definido mediante:

- `%` Cualquier secuencia de caracteres
- `_` Cualquier caracter (solamente uno)

```
SELECT *  
FROM Peliculas  
WHERE name LIKE '%Spiderman%'
```

SQL

Eliminando duplicados

Entregue todos los nombres distintos de las películas:

```
SELECT DISTINCT nombre  
FROM Peliculas
```

OJO: **DISTINCT** es un operador en sí mismo.