



ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

EDUCACIÓN
PROFESIONAL

SQL II

Gestión de Datos

Vicente Calisto

Educación Profesional - Escuela de Ingeniería

Clase diseñada por Matías Toro para GDD, DCDPP 2022

El uso de apuntes de clases estará reservado para finalidades académicas. La reproducción total o parcial de los mismos por cualquier medio, así como su difusión y distribución a terceras personas no está permitida, salvo con autorización del autor.

Sabemos hacer consultas básicas

```
SELECT atributos  
FROM relaciones  
WHERE condiciones
```

Además, existen operadores como `LIKE`,
`DISTINCT`, `ORDER BY`, `UNION`, etc.



- Agregación
- Consultas anidadas
- Valores nulos
- Joins externos
- Limitar resultados
- Redundancia



Agregación



Operadores de agregación

- COUNT
- AVG
- SUM
- MIN
- MAX



Sintaxis

- COUNT ([DISTINCT] A)
- AVG ([DISTINCT] A)
- SUM ([DISTINCT] A)
- MIN (A)
- MAX (A)



Agregación: COUNT

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuántas películas hay?

```
SELECT COUNT (*)  
FROM Peliculas
```

count(*)
4



Agregación: COUNT

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuántas películas hay?

```
SELECT COUNT(*) as conteo  
FROM Peliculas
```

conteo
4



Agregación: COUNT

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuántas categorías hay?

```
SELECT COUNT(categoría) as conteo  
FROM Peliculas
```

conteo
4

¿Cómo contamos las distintas categorías?



Agregación: COUNT

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuántas categorías distintas hay?

```
SELECT DISTINCT COUNT(categoría) as conteo  
FROM Peliculas
```

conteo
4

¡Sigue dándonos el mismo valor!



Agregación: COUNT

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuántas categorías distintas hay?

```
SELECT COUNT(DISTINCT categoria) as conteo  
FROM Peliculas
```

conteo
3

Ahora sí



Agregación: AVG

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuál es el año promedio?

```
SELECT AVG(año) as promedio  
FROM Peliculas
```

promedio
2014.25

(PostgreSQL)

promedio
2014

Depende del sistema



Agregación: AVG DISTINCT

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuál es el año promedio?

```
SELECT AVG (DISTINCT año) as promedio  
FROM Peliculas
```

promedio
2014.5

(PostgreSQL)

promedio
2014

Depende del sistema

promedio
2015



Agregación: AVG CAST

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuál es el año promedio?

```
SELECT AVG (CAST (año) AS FLOAT) AS promedio  
FROM Peliculas
```

promedio
2014.25



Agregación: MAX

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasia	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuál es la mayor calificación?

```
SELECT MAX(calificacion) AS maximo  
FROM Peliculas
```

maximo
8.6



Agregación: MAX

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuáles son los nombres de las películas con máxima calificación?

```
SELECT MAX(calificación) AS maximo, nombre  
FROM Peliculas
```



Error: no existe un operador de agregación.

Volveremos a esto...



Agregación: MIN

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuál es la menor calificación?

```
SELECT MIN(calificación) AS minimo  
FROM Peliculas
```

minimo
7.7



Agregación: SUM

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cuál es la suma de las calificaciones?

```
SELECT SUM(calificación) AS suma  
FROM Películas
```

suma
32.5



Agregación: GROUP BY

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cantidad de películas por categoría?

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
GROUP BY categoria
```



Agregación: GROUP BY

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

COUNT (*)

➡ 1

➡ 1

➡ 2

```
SELECT categoría, COUNT(*) AS conteo
FROM Peliculas
GROUP BY categoría
```

categoría	conteo
Fantasía	1
Drama	1
Biografía	2



Agregación: GROUP BY

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
WHERE calificación>8
GROUP BY categoria
```

1) Se ejecuta el WHERE



Agregación: GROUP BY

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The theory of Everything	2014	Biografía	7.7	J. Marsh

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
WHERE calificación>8
GROUP BY categoria
```

1) Se ejecuta el WHERE



Agregación: GROUP BY

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
WHERE calificación>8
GROUP BY categoria
```

2) Se agrupa según el GROUP BY



Agregación: GROUP BY

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
WHERE calificación>8
GROUP BY categoria
```

2) Se agrupa según el GROUP BY



Agregación: GROUP BY

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum

COUNT (*)

➡ 1

➡ 1

➡ 1

```
SELECT categoria, COUNT (*) AS conteo
FROM Peliculas
WHERE calificación > 8
GROUP BY categoria
```

3) Se aplica la función de agregación



Agregación: GROUP BY

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum

COUNT (*)

➡ 1

➡ 1

➡ 1

```
SELECT categoría, COUNT(*) AS conteo
FROM Peliculas
WHERE calificación > 8
GROUP BY categoría
```

categoría	conteo
Fantasía	1
Drama	1
Biografía	1

4) Se proyectan los atributos



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Cantidad de películas por categoría donde la calificación promedio sea mayor a 8?

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
GROUP BY categoria
HAVING AVG(calificacion) > 8
```



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
GROUP BY categoria
HAVING AVG(calificación) > 8
```

1) Se ejecuta el WHERE ...



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasia	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
GROUP BY categoria
HAVING AVG(calificacion) > 8
```

2) Se agrupa según el GROUP BY



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

AVG(categoría) > 8



AVG(categoría) > 8



AVG(categoría) > 8



```
SELECT categoría, COUNT(*) AS conteo
FROM Peliculas
GROUP BY categoría
HAVING AVG(calificación) > 8
```

3) Se **filtran** los grupos según el HAVING



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

AVG(categoría) > 8



AVG(categoría) > 8



AVG(categoría) > 8



```
SELECT categoría, COUNT(*) AS conteo
FROM Peliculas
GROUP BY categoría
HAVING AVG(calificación) > 8
```

3) Se **filtran** los grupos según el HAVING



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu

COUNT (*)



```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
GROUP BY categoria
HAVING AVG(calificación) > 8
```

4) Se aplica la función de agregación



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu

COUNT (*)



1



1

```
SELECT categoria, COUNT ( * ) AS conteo
FROM Peliculas
GROUP BY categoria
HAVING AVG (calificación) > 8
```

categoría	conteo
Fantasía	1
Drama	1

4) Se proyectan los atributos



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

¿Y ahora?

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
WHERE calificacion>8
GROUP BY categoria
HAVING AVG(calificacion)>8
```



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
WHERE calificacion>8
GROUP BY categoria
HAVING AVG(calificacion)>8
```

1) Se ejecuta el WHERE



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
WHERE calificacion>8
GROUP BY categoria
HAVING AVG(calificacion)>8
```

1) Se ejecuta el WHERE



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
WHERE calificacion>8
GROUP BY categoria
HAVING AVG(calificacion)>8
```

2) Se agrupa según el GROUP BY



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasia	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
WHERE calificacion>8
GROUP BY categoria
HAVING AVG(calificacion)>8
```

2) Se agrupa según el GROUP BY



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum

AVG(categoría) > 8



AVG(categoría) > 8



AVG(categoría) > 8



```
SELECT categoría, COUNT(*) AS conteo  
FROM Peliculas
```

```
WHERE calificación > 8
```

```
GROUP BY categoría
```

```
HAVING AVG(calificación) > 8
```

3) Se **filtran** los grupos según el HAVING



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum

COUNT (*)



```
SELECT categoria, COUNT ( * ) AS conteo  
FROM Peliculas
```

```
WHERE calificación > 8
```

```
GROUP BY categoria
```

```
HAVING AVG (calificación) > 8
```

4) Se aplica la función de agregación



Agregación: GROUP BY / HAVING

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasia	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum

COUNT (*)



```
SELECT categoria, COUNT ( * ) AS conteo  
FROM Peliculas
```

```
WHERE calificación > 8
```

```
GROUP BY categoria
```

```
HAVING AVG ( calificación ) > 8
```

5) Se proyectan los atributos

categoría	conteo
Fantasia	1
Drama	1
Biografía	1



Agregación: HAVING / EVERY

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
GROUP BY categoria
HAVING EVERY(calificacion
                BETWEEN 8.0 AND 9)
```

categoría	conteo
Fantasía	1
Drama	1



Agregación: HAVING / ANY

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

```
SELECT categoria, COUNT(*) AS conteo
FROM Peliculas
GROUP BY categoria
HAVING ANY(calificación
```

categoría	conteo
Fantasía	1
Drama	1
Biografía	2

```
BETWEEN 8.0 AND 9)
```



Agregación: GROUP BY / HAVING

Forma general:

```
SELECT <S>  
FROM R1, ..., Rn  
WHERE <condiciones 1>  
GROUP BY a1, ..., ak  
HAVING <condiciones 2>
```

Puede tener atributos de a_1, \dots, a_k y/o agregados pero ningún otro atributo.

Pueden usar atributos de R_1, \dots, R_n

Condiciones de agregación de los atributos de R_1, \dots, R_n



Consultas Anidadas



Hasta ahora

- Como ya habíamos visto con las operaciones de conjuntos, una consulta puede estar constituida por operaciones entre consultas.
- Pero esa no es la única forma, SQL nos ofrece mucho más.



Consultas Anidadas

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

Actor(id, nombre, edad)

Actuó_en(id_actor, id_película)

Obtengamos los ids de los actores que actuaron en películas con calificación > 8.

```
SELECT id_actor
FROM Actuó_en
WHERE id_película IN
```

```
(
  SELECT id
  FROM Películas
  WHERE calificación>8
)
```

Subconsulta



Consultas Anidadas: NOT/IN

Obtengamos los nombres de los actores que **no** actuaron en películas con calificación > 8.

```
SELECT nombre
FROM Actores
WHERE id NOT IN
(
  SELECT id_actor
  FROM Actuo_en
  WHERE id_pelicula IN
    (
      SELECT id
      FROM Peliculas
      WHERE calificacion>8
    )
)
```



Consultas Anidadas: encontrando el MAX

¿Cuáles son los nombres de las películas con máxima calificación?

```
SELECT *  
FROM Peliculas  
WHERE calificacion =  
(  
    SELECT MAX(calificacion)  
    FROM Peliculas  
)
```

Subconsulta



Consultas Anidadas: EXISTS

Obtengamos los nombres de los actores que han actuado en alguna película.

```
SELECT nombre
FROM Actores
WHERE EXISTS
(
  SELECT *
  FROM Actuo_en
  WHERE id_actor = id
)
```

dependencia



Consultas Anidadas: > ANY

Obtengamos las películas de mayor año respecto a **alguna** película de fantasía.

```
SELECT nombre
FROM Peliculas
WHERE anho > ANY
(
  SELECT anho
  FROM Peliculas
  WHERE categoria = 'Fantasia'
)
```



Consultas Anidadas: > ALL

Obtengamos las películas de mayor año respecto a **todas** las películas de fantasía.

```
SELECT nombre
FROM Peliculas
WHERE anho > ALL
(
  SELECT anho
  FROM Peliculas
  WHERE categoria = 'Fantasia'
)
```



Consultas Anidadas: > ALL

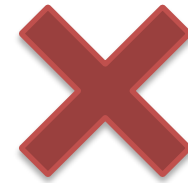
Nota: el resultado de la subconsulta debe retornar **una sola columna** de un **tipo compatible**.

```
SELECT nombre  
FROM Peliculas  
WHERE anho > ALL
```

```
(
```

```
    SELECT nombre, anho  
    FROM Peliculas  
    WHERE categoria = 'Fantasia'
```

```
)
```



Error: la tabla devolvió
más de una columna



Consultas Anidadas: Valor

Nota: el resultado de la subconsulta debe retornar **un solo valor** y **una sola columna** de un **tipo compatible**.

```
SELECT nombre
FROM Peliculas
WHERE anho >
(
  SELECT anho
  FROM Peliculas
  WHERE categoria = 'Fantasia'
)
```



Consultas Anidadas: Valor

Nota: el resultado de la subconsulta debe retornar **un solo valor** y **una sola columna** de un **tipo compatible**.

```
SELECT nombre
FROM Peliculas
WHERE anho >
(
  SELECT anho
  FROM Peliculas
  WHERE categoria = 'Biografía'
)
```



Error: la tabla devolvió
más de una fila



Consultas Anidadas: Valor

Nota: el resultado de la subconsulta debe retornar **un solo valor** y **una sola columna** de un **tipo compatible**.

```
SELECT nombre  
FROM Peliculas  
WHERE anho >
```

```
(
```

```
    SELECT nombre, anho  
    FROM Peliculas
```

```
    WHERE categoria = 'Fantasia'
```

```
)
```



Error: la tabla devolvió
más de una columna



Consultas Anidadas: FROM

El alias en consultas anidadas en el FROM es obligatorio:

```
SELECT nombre
FROM
(
  SELECT P1.id
  FROM Peliculas P1, Peliculas P2
  WHERE P1.director = P2.director
  AND P1.nombre <> P2.nombre
) Multi, Actuo_en, Actores
WHERE Multi.id = Actuo_en.id_pelicula
AND Actuo_en.id_actor = Actores.id
```



Valores nulos



Información Incompleta

- En una base de datos real, muy seguido no tendremos los datos para llenar todas las columnas al agregar una fila.
- También puede ser que por la lógica del problema, que un campo esté vacío tenga una semántica relevante para la aplicación.
- Con SQL podemos modelar la falta de información mediante **nulos (NULL)**.
- Los nulos en las tablas generan ciertos **comportamientos extraños** que es bueno tener en cuenta al trabajar con ellos. Los discutiremos en esta clase.



\perp , \emptyset , NULL, null



id	nombre	año	categoría	calificación	director
1	Interstellar	null	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	null	A. Iñárritu
3	null	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	null

Desconocido o Inaplicable.

id	nombre	año	categoría	calificación	director
1	Interstellar	null	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	null	A. Iñárritu
3	null	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	null

En general los nulos pueden significar:

1. Valor existe, pero no tengo la información
2. Valor no existe
3. Ni siquiera sé si el valor existe o no

Consultando nulos

Sea la relación $R(a, b)$, las consultas:

- `SELECT * FROM R`
- `SELECT * FROM R
WHERE $R.b = 3$ OR $R.b \neq 3$`

¿Son lo mismo?

Si $R.b$ es nulo, $R.b = 3$ y $R.b \neq 3$ evalúan a falso



Consultando nulos

La consulta

```
SELECT * FROM R
```

Equivale a la unión de:

- `SELECT * FROM R WHERE R.b = 3`
- `SELECT * FROM R WHERE R.b <> 3`
- `SELECT * FROM R WHERE R.b IS NULL`

Para ver si un elemento es nulo usamos `IS NULL`

Para ver si un elemento no es nulo usamos `IS NOT NULL`



Operando con nulos

Si algún argumento de una operación aritmética es nulo, el resultado es nulo



Operando con nulos

id	nombre	año	categoría	calificación	director
1	Interstellar	null	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	null	A. Iñárritu
3	null	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	null

```
SELECT nombre  
FROM Peliculas  
WHERE año > 2014
```

nombre
The Revenant



Operando con nulos

id	nombre	año	categoría	calificación	director
1	Interstellar	null	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	null	A. Iñárritu
3	null	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	null

```
SELECT nombre  
FROM Peliculas  
WHERE año = NULL
```

nombre

¡El nulo en la consulta y el nulo en los datos son distintos!



Lógica de tres valores

p	q	p OR q	p AND q	p = q
TRUE	TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	TRUE	FALSE	FALSE
FALSE	TRUE	TRUE	FALSE	FALSE
FALSE	FALSE	FALSE	FALSE	TRUE
TRUE	?	TRUE	?	?
FALSE	?	?	FALSE	?
?	TRUE	TRUE	?	?
?	FALSE	?	FALSE	?
?	?	?	?	?

Cuando no importa el valor del desconocido, el resultado se mantiene.
Cuando importa el valor del desconocido, el resultado es desconocido.



Nulos: Agregación

A
1
null

SELECT COUNT (*) FROM R ➡ 2

SELECT COUNT (R.A) FROM R ➡ 1

SELECT SUM (R.A) FROM R ➡ 1



Nulos: Agregación

Para funciones de **agregación**:

- Se ignoran **todos** los nulos
- Se computa el valor de la agregación
- La **única** excepción es **COUNT(*)**



Joins externos



Inner Joins

Recordemos que podemos hacer `JOINS`, especificando en la sentencia `FROM` de la consulta las tablas que queremos usar y en el `WHERE` las condiciones:

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



Inner Joins

Estas 3 consultas son equivalentes:

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

```
SELECT *  
FROM Peliculas JOIN Actuo_en  
ON id = id_pelicula
```

```
SELECT *  
FROM Peliculas INNER JOIN Actuo_en  
ON id = id_pelicula
```



Outer Joins

Consideremos estas tablas:

Estudio

nombre	Pelicula
Warner	Argo
Warner	El Origen
MGM	El Hobbit

Pelicula

nombre	ingreso
Argo	136
El Origen	292
El Artista	44

Escribamos una consulta que liste los ingresos totales de cada estudio.



Outer Joins

Estudio

nombre	titulo
Warner	Argo
Warner	El Origen
MGM	El Hobbit

Pelicula

titulo	ingreso
Argo	136
El Origen	292
El Artista	44

```
SELECT Estudio.nombre, SUM(Pelicula.ingreso)
FROM Estudio JOIN Pelicula
ON Estudio.titulo = Pelicula.titulo
GROUP BY Estudio.nombre
```

nombre	SUM(...)
Warner	428

¿Algún problema?



Outer Joins

Lo solucionamos con un **Outer Join Izquierdo**, que mantiene las tuplas sin pareja de la primera tabla:

```
SELECT Estudio.nombre, SUM(Pelicula.ingreso)
FROM Estudio LEFT JOIN Pelicula
ON Estudio.titulo = Pelicula.titulo
GROUP BY Estudio.nombre
```

nombre	SUM(...)
Warner	428
MGM	null



Outer Joins

Estudio

nombre	titulo
Warner	Argo
Warner	El Origen
MGM	El Hobbit

Pelicula

titulo	ingreso
Argo	136
El Origen	292
El Artista	44

```
SELECT *  
FROM Estudio LEFT JOIN Pelicula  
ON Estudio.titulo = Pelicula.titulo
```

nombre	titulo	titulo	ingreso
Warner	Argo	Argo	136
Warner	El Origen	El Origen	292
MGM	El Hobbit	null	null



Outer Joins

- **R LEFT JOIN S**: mantenemos las tuplas de **R** que no tienen correspondencia.
- **R RIGHT JOIN S**: mantenemos las tuplas de **S** que no tienen correspondencia.
- **R FULL JOIN S**: mantenemos las tuplas de **R** y **S** que no tienen correspondencia



Outer Joins

1	INNER	A	=	2	A
2	JOIN	B		3	B
3		C			

1	LEFT	A	=	1	A
2	JOIN	B		2	B
3		C		3	

1	RIGHT	A	=	2	A
2	JOIN	B		3	B
3		C			C

1	FULL	A	=	1	A
2	JOIN	B		2	B
3		C		3	C

1	CROSS	A	=	1	A
2	JOIN	B		1	B
3		C		1	C
				2	A
				2	B
				2	C
				3	A
				3	B
				3	C



Outer Joins

1	INNER	A	=	2	A
2	JOIN	B		3	B
3		C			

1	LEFT	A	=	1	null
2	JOIN	B		2	A
3		C		3	B

1	RIGHT	A	=	2	A
2	JOIN	B		3	B
3		C		null	C

1	FULL	A	=	1	null
2	JOIN	B		2	A
3		C		3	B
				null	C

1	CROSS	A	=	1	A
2	JOIN	B		1	B
3		C		1	C
				2	A
				2	B
				2	C
				3	A
				3	B
				3	C



Limitar resultados



Limitar resultados

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

Las primeras 2 películas

```
SELECT * FROM Peliculas
ORDER BY id
LIMIT 2
```

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasía	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu



Limitar resultados + offset

id	nombre	año	categoría	calificación	director
1	Interstellar	2014	Fantasia	8.6	C. Nolan
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum
4	The Theory of Everything	2014	Biografía	7.7	J. Marsh

Las primeras 2 películas partiendo de la 2

```
SELECT * FROM Peliculas  
ORDER BY id  
OFFSET 1 LIMIT 2
```

id	nombre	año	categoría	calificación	director
2	The Revenant	2015	Drama	8.1	A. Iñárritu
3	The Imitation Game	2014	Biografía	8.1	M. Tyldum



Redundancia



Redundancia

¡Son todas equivalentes!

```
SELECT Bandas.nombre
FROM Bandas, Estudiantes_UC
WHERE Bandas.vocalista = Estudiantes_UC.nombre
AND Bandas.nombre IN (
    SELECT Toco_en.nombre_banda
    FROM Toco_en
    WHERE Toco_en.nombre_festival = 'Lollapalooza'
)
```

```
SELECT DISTINCT Bandas.nombre
FROM Bandas, Estudiantes_UC, Toco_en
WHERE Bandas.vocalista = Estudiantes_UC.nombre
AND Banda.nombre = Toco_en.nombre_banda
AND Toco_en.nombre_festival = 'Lollapalooza'
```

```
SELECT Bandas.nombre
FROM Bandas, Estudiantes_UC
WHERE Bandas.vocalista = Estudiantes_UC.nombre
INTERSECT
SELECT Toco_en.nombre_banda
FROM Toco_en
WHERE Toco_en.nombre_festival = 'Lollapalooza'
```

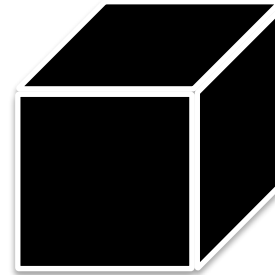
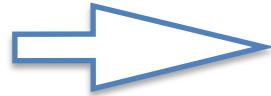


Redundancia

```
SELECT Bandas.nombre
FROM Bandas, Estudiantes_UC
WHERE Bandas.vocalista = Estudiantes_UC.nombre
AND Bandas.nombre IN (
    SELECT Toco_en.nombre_banda
    FROM Toco_en
    WHERE Toco_en.nombre_festival = 'Lollapalooza'
)
```



```
SELECT DISTINCT Bandas.nombre
FROM Bandas, Estudiantes_UC, Toco_en
WHERE Bandas.vocalista =
Estudiantes_UC.nombre
AND Banda.nombre = Toco_en.nombre_banda
AND Toco_en.nombre_festival =
'Lollapalooza'
```



nombre	titulo
Warner	Argo
Warner	El Origen
MGM	El Hobbit

```
SELECT Bandas.nombre
FROM Bandas, Estudiantes_UC
WHERE Bandas.vocalista = Estudiantes_UC.nombre
INTERSECT
SELECT Toco_en.nombre_banda
FROM Toco_en
WHERE Toco_en.nombre_festival = 'Lollapalooza'
```



Uno dice lo que quiere, no
cómo debería ser
computado



¿Preguntas?

