Bases de datos

LMD: Consultas en SQL

SELECT: Sintaxis

SELECT A1, …, An — Describe la salida deseada con:
- Nombres de columnas
- Expresiones aritméticas
- Literales
- Funciones escalares o de columna

FROM T1,… ,Tn — Nombres de las tablas / vistas

WHERE P — Condiciones de selección de filas

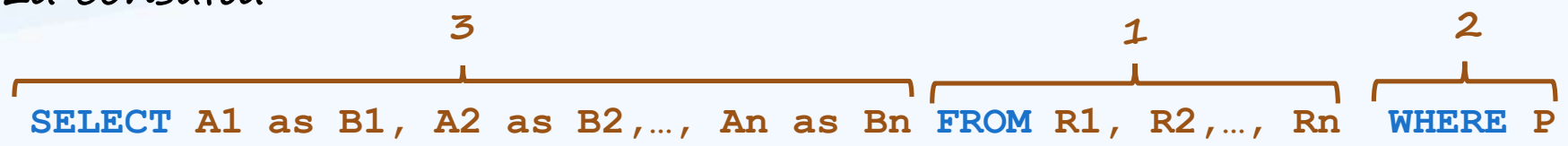
GROUP BY Ai1, …, Ain — Nombre de columnas por las que agrupar

HAVING Q — Condiciones de selección de grupo

ORDER BY Aj1, …, Ajn — Nombre de columnas por las que ordenar (**ASC**, **DESC**)

SELECT: Semántica basada en AR

- La consulta



  equivale en álgebra relacional a

$$\rho_{(B1, B2, \ldots, Bn)} (\Pi_{A1, A2, \ldots, An} (\sigma_P (R1 \times R2 \times \ldots \times Rn)))$$

- La ejecución de la consulta SQL se realiza en el mismo orden que la expresión del AR

- **ADVERTENCIA:** La selección del AR y el `SELECT` de SQL no funcionan igual. Una consulta `SELECT` puede tener tuplas repetidas, por defecto **no se eliminan duplicados**
- **Para eliminar duplicados `SELECT DISTINCT`… pero es costosa**

➢ **`UNION/INTERSECT/MINUS`**

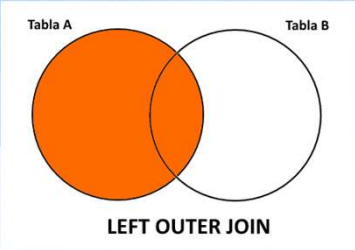
SELECT: Orden de ejecución

1. Se crea la tabla del producto cartesiano/join que hay en el **FROM**
2. Se aplica el predicado **WHERE**: Las tuplas que satisfacen el predicado **WHERE** son colocadas en grupos siguiendo el patrón **GROUP BY**
3. Se ejecuta la cláusula **HAVING** para cada grupo de tuplas anterior
4. Los grupos obtenidos tras aplicar **HAVING** son los que serán procesados por **SELECT**, que calculará, en los casos que se incluyan, las funciones de agregación que le acompañan
5. A las tuplas resultantes de los pasos anteriores se le aplica la ordenación descrita en la cláusula **ORDER BY**
6. Se renombran los atributos a devolver

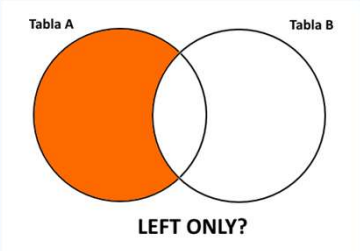
SELECT:

- Atributos:
  - Todos: **SELECT * (todas las columnas de la tabla generada en el FROM)**
  - Algunos: **SELECT nombreColumna_1, …, nombreColumna_n**
- Duplicados:
  - Eliminarlos: **SELECT DISTINCT … (costosa)**
  - Forzar que aparezcan: **SELECT ALL … (por defecto si no hay nada)**
- Predicados:
  - Se utilizan los conectores lógicos **AND**, **OR** y **NOT** (símbolos matemáticos ∧, ∨, ¬)
  - Predicados de comparación en las expresiones:
    - Operadores: **=**, **<>** (es el ≠), **<**, **<=**, **>=**
    - **BETWEEN / NOT BETWEEN** (fechas y valores)
    - **IN / NOT IN** (conjunto o subconsultas)
    - **IS NULL / IS NOT NULL**
    - **LIKE / NOT LIKE** (cadenas de caracteres) '%D_'
    - <Operador> **ALL, SOME/ANY** (subconsultas)
    - **EXISTS / NOT EXISTS** (subconsultas)
- Operadores numéricos:
  - **+**, **-** (números y fechas) y *****, **/** (números)
- Otros operadores (transparencias 29/30)

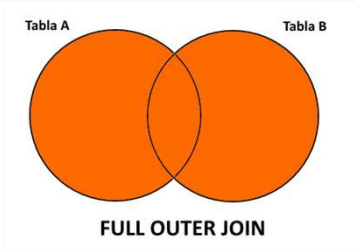
SQL JOINS



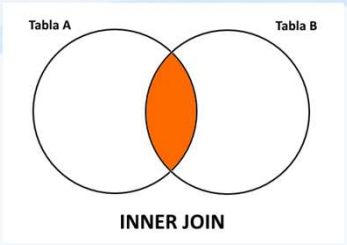

```
SELECT <campos>
FROM TablaA A
        LEFT JOIN TablaB B
                ON A.pk = B.pk
```



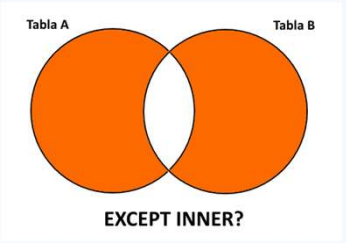

```
SELECT <campos>
FROM TablaA A
        LEFT JOIN TablaB B
                ON A.pk = B.pk
WHERE B.pk IS NULL
```



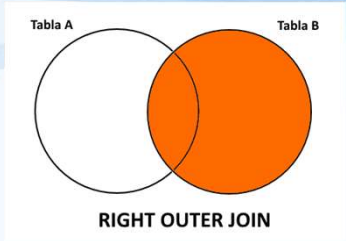

```
SELECT <campos>
FROM TablaA A
        FULL JOIN TablaB B
                ON A.pk = B.pk
```



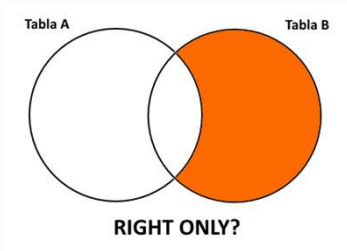

```
SELECT <campos>
FROM TablaA A
        JOIN TablaB B
                ON A.pk = B.pk
```




```
SELECT <campos>
FROM TablaA A
        FULL JOIN TablaB B
                ON A.pk = B.pk
WHERE A.pk IS NULL
        AND B.pk IS NULL
```




```
SELECT <campos>
FROM TablaA A
        RIGHT JOIN TablaB B
                ON A.pk = B.pk
```




```
SELECT <campos>
FROM TablaA A
        RIGHT JOIN TablaB B
                ON A.pk = B.pk
WHERE A.pk IS NULL
```



```
[{NATURAL} {LEFT|RIGHT|FULL} [OUTER]|INNER] JOIN … [ON/USING …]
```

Select. Agrupamientos

```
SELECT A1, …, An
```
Describe la salida deseada con:
- Nombres de columnas
- Expresiones aritméticas
- Literales
- Funciones escalares o de columna

```
FROM T1, …, Tn
```
Nombres de las tablas / vistas

```
WHERE P
```
Condiciones de selección de filas

```
GROUP BY Ai1, …, Ain
```
Nombre de columnas por las que agrupar

```
HAVING Q
```
Condiciones de selección de grupo

```
ORDER BY Aj1, …, Ajn
```
Nombre de columnas por las que ordenar

Agrupamientos: Exposiciones

Asignados(nombreExp, nombrePintor, nombreCuad, póliza)

nombreExp	nombrePintor	nombreCuad	poliza
Art Novo	Martina	El día	123
Art Novo	Pepe	La noche	123
Art Novo2	Pepe	La noche	1234
Art Novo	Mario	Autorretrato	234
Local	Mario	Autorretrato	234
Art Novo2	Picasso	Autorretrato	425

```
SELECT nombreExp, COUNT(*) AS numCuadros
     , COUNT(DISTINC poliza) AS numPolizas
FROM Asignados
GROUP BY nombreExp
HAVING COUNT(*)>=2
```

nombreExp	nombrePintor	nombreCuad	poliza
Art Novo	Martina	El día	123
Art Novo	Pepe	La noche	123
Art Novo	Mario	Autorretrato	234
Local	Mario	Autorretrato	234
Art Novo2	Pepe	La noche	1234
Art Novo2	Picasso	Autorretrato	425

nombreExp	numCuadros
Art Novo	3
Local	1
Art Novo2	2

Agrupamientos: Cuadros

Asignados(nombreExp, nombrePintor, nombreCuad, póliza)

nombreExp	nombrePintor	nombreCuad	poliza
Art Novo	Martina	El día	123
Art Novo	Pepe	La noche	123
Art Novo2	Pepe	La noche	1234
Art Novo	Mario	Autorretrato	234
Local	Mario	Autorretrato	234
Art Novo2	Picasso	Autorretrato	425

```
SELECT nombrePintor, nombreCuad
       , COUNT(*) AS numExp
FROM Asignados
GROUP BY nombrePintor, nombreCuad
HAVING COUNT(DISTINCT póliza) >= 2
```

nombreExp	nombrePintor	nombreCuad	poliza
Art Novo	Martina	El día	123
Art Novo	Pepe	La noche	123
Art Novo2	Pepe	La noche	1234
Art Novo	Mario	Autorretrato	234
Local	Mario	Autorretrato	234
Art Novo2	Picasso	Autorretrato	425

nombrePintor	nombreCuad	numExp
~~Martina~~	~~El día~~	~~1~~
Pepe	La noche	2
~~Mario~~	~~Autorretrato~~	~~2~~
~~Picasso~~	~~Autorretrato~~	~~1~~

Funciones de agregación

Colección valores → Un solo valor

- Las funciones de agregación disponibles son:
  - **AVG**: Cálculo de la media (números)
  - **SUM**: Cálculo del total (números)
  - **MIN**: Obtener el valor mínimo
  - **MAX**: Obtener el valor máximo
  - **COUNT**: Contar el número de filas

COUNT(*)	Cuenta NULL
COUNT(nombreColumna)	Cuenta el total de valores NO NULL
COUNT(DISTINCT nombreColumna)	Cuenta los valores distintos NO NULL

OJO: **AVG, SUM, MIN, MAX NO tienen en cuenta los valores NULL**

Select. Subconsultas

```
SELECT A1, …, An

FROM T1, …, Tn

WHERE P

GROUP BY Ai1, …, Ain
HAVING Q

ORDER BY Aj1, …, Ajn
```

SELECT …
FROM …
WHERE …
GROUP BY …
HAVING …
~~ORDER BY …~~

- *Independientes*: No dependen de las tablas T1, .., Tn. Se calculan SOLO una vez.
- *Correlacionadas*: Dependen de las tablas T1, …, Tn. Se calculan por cada individuo.

VISTAS (LDD)

Consulta como tabla virtual.

Cada vez que se necesita se calcula, actualizada.

1. Creación: **CREATE VIEW** NombreVista **AS** subconsulta;
2. Modificación: **ALTER VIEW** NombreVista **AS** subconsulta;
3. Borrado: **DROP VIEW** NombreVista;

NombreVista = tabla virtual que se calcula cada vez que se utiliza y se puede utilizar como otra tabla más.

UPDATE

- La instrucción **UPDATE** permite actualizar los valores de los campos de una tabla para uno o varios registros
- Sintaxis:

```
UPDATE NombreTabla
SET Campo1 = Valor1, ..., CampoN = ValorN
WHERE Condición;
```

  - NombreTabla: Tabla en la que vamos a actualizar los datos
  - **SET**: Indica los campos que vamos a actualizar y con qué valores lo vamos a hacer
  - **WHERE**: Condiciones que deben de cumplir los registros que se van a actualizar
    - Si no hay WHERE se actualizan todas las tuplas de la tabla

DELETE

- La instrucción DELETE permite eliminar uno o varios registros (incluso todos) de una tabla
- Sintaxis:

```
DELETE [FROM] NombreTabla
WHERE Condición;
```

  - Si no hay WHERE se borran todas las tuplas de la tabla

Guion:

- Renombramiento atributos (AS nombre) y tablas (nuevoNombre). (Tabla.atributo)
- Funciones de agregación pag.79-82
- Group by/Having pag 83
- JOIN
  - INNER pag. 45-53
  - OUTER pag. 54-63
- Subconsultas pag. 96-113
- Vistas pag. 115-118
- UPDATE pag. 119
- DELETE pag. 120

!!Cuidado!! NULL

NULL ≈ Desconocido

- Operaciones: +, -, *, / (3 + NULL → NULL)
- Operaciones booleanas: =, !=, <, <=, >, >= (NULL = 5 → False)
- Funciones de agregación:
  - MIN, MAX, AVG, SUM, COUNT(columna) ignoran nulos
  - COUNT(*) cuenta filas

tablaNULLs

id	a	b
101	5	3
102	5	NULL
103	NULL	9
104	NULL	NULL

¿COUNT(*)?
¿COUNT(a)?
¿COUNT(DISTINCT a)?
¿COUNT(a+b)?

¿SUM(a)?
¿SUM(b)?
¿SUM(a+b)?
¿AVG(b)?

INSERT

- Simples:

```
INSERT INTO NombreTabla (columna1, columna2, ... columna_n )
VALUES (expresión1, expresión2, ... expresión_n );
```

- Múltiples:

```
INSERT INTO NombreTabla (columna1, columna2, ... columna_n )
SELECT expresión1, expresión2, ... expresión_n
FROM TablaOrigen
[WHERE condiciones];
```

```
INSERT ALL
        INTO Enero17_Pintores VALUES ('Marisa', '07/11/1932', 'Pepe')
        INTO Enero17_Pintores VALUES ('Eloisa', '07/11/1942', 'Pepe')
        INTO Enero17_Pintores VALUES ('Sandra', '07/11/1952', 'Eloisa')
SELECT * FROM dual;
```