

3.2 Llenguatges: SQL

- Introducció
- Base de dades exemple
- Sentències SQL
 - Creació d'una taula
 - Inserció / modificació / esborrat de files d'una taula
 - Consultes

Introducció

- Llenguatge estructurat de definició, actualització i consulta de bases de dades
- Proposat per un departament d'investigació d'IBM
- Adoptat com a estàndard per al Model Relacional de bases de dades en els anys 1986-87 (ANSI/X3H2/RDL).
- Des d'aleshores se n'han fet varies versions: SQL-89, SQL-92, SQL:1999, SQL:2003, SQL:2006, SQL:2008, SQL:2011, SQL:2016, SQL:2023.
- Malgrat l'estandarització, els diferents constructors de SGBDR (Sistemes de Gestió de Bases de Dades Relacionals) ofereixen variants de SQL.
- Nosaltres veurem el nucli comú de SQL per a tots els SGBDR, tenint en compte que per alguns la sintaxi de les sentències pot variar.
- Pot ser utilitzat de manera interactiva o de manera hostatjada entre les sentències d'un programa.

Base de dades exemple

- Una base de dades relacional està composta de **Taules (Relacions)** amb un conjunt de **Columnes (Atributs)** i un conjunt **Files (Tuples)**.

departaments(num_dpt, nom_dpt, planta, edifici, ciutat_dpt)					
	1	DIRECCIO	10	PAU CLARIS	BARCELONA
	2	DIRECCIO	8	RIOS ROSAS	MADRID
	3	MARQUETING	1	PAU CLARIS	BARCELONA
projectes(num_proj, nom_proj, producte, pressupost)					
	1	IBDTEL	TELEVISIO	1000000	
	2	IBDVID	VIDEO	500000	
empleats(num_empl, nom_empl, sou, ciutat_empl, num_dpt, num_proj)					
	1	CARME	400000	MATARO	1 1
	2	EUGENIA	350000	TOLEDO	2 2
	3	JOSEP	250000	SITGES	3 1

Base de dades exemple

- **Clau primària:** Cada taula té una clau primària que permet identificar les files de la taula. Per ex. **num_dpt** és la clau primària de la taula departaments. Això vol dir que cada departament té un num_dpt que ha de ser únic entre tots els departaments, és a dir mai hi haurà dos departaments amb el mateix número de departament.
- **Clau forana.** Una clau forana permet relacionar les files de dues taules. Per ex. **num_dpt** és una clau forana de la taula empleats que referencia la taula departaments. Això vol dir que entre les dades d'un empleat hi haurà també el departament al que pertany, i això ens permetrà saber el departament on treballa un empleat, i també els empleats que treballen a un departament.

departaments(<u>num_dpt</u> , nom_dpt, planta, edifici, ciutat_dpt)					
1	DIRECCIO	10	PAU CLARIS	BARCELONA	
2	DIRECCIO	8	RIOS ROSAS	MADRID	
3	MARQUETING	1	PAU CLARIS	BARCELONA	
projectes(<u>num_proj</u> , nom_proj, producte, pressupost)					
1	IBDTEL	TELEVISIO	1000000		
2	IBDVID	VIDEO	500000		
empleats(<u>num_empl</u> , nom_empl, sou, ciutat_empl, num_dpt, num_proj)					
1	CARME	400000	MATARO	1	1
2	EUGENIA	350000	TOLEDO	2	2
3	JOSEP	250000	SITGES	3	1

Creació d'una taula

```
CREATE TABLE <nom_taula>  
    (<nom_columna> <tipus_dades> [<restriccions_col>] [<val_per_defecte>]  
    [, <nom_columna> <tipus_dades> [<restriccions_col>] [<val_per_defecte>]...]  
    [<restriccions_taula>]);
```

- **tipus_dades:** INTEGER, FLOAT(precisió), REAL, CHAR(n), NUMERIC(precisió,escala), DECIMAL(precisió,escala), SMALLINT, DOUBLE PRECISION, VARCHAR(n), DATE,....
- **val_per_defecte:** Valor per defecte d'una columna per a una fila que s'insereix a la taula.

DEFAULT { <literal> | NULL }.

Creació d'una taula: Restriccions de columna

■ restriccions_col:

UNIQUE	La columna no pot tenir valors repetits
PRIMARY KEY	La columna és clau primària de la taula
REFERENCES <taula> [<col>]	La columna és clau forana que referencia la taula indicada.
CHECK (<condicions>)	La columna ha de complir les condicions especificades. Només pot referir-se a la columna per la que es defineix.
NOT NULL	La columna no pot tenir valors nuls

Les **restriccions de columna** es refereixen a una única columna.

Per exemple, en la condició d'un CHECK de columna no es pot fer referència a més d'una columna.

Creació d'una taula: Restriccions de taula

■ restriccions_taula:

UNIQUE (<cols>)	El conjunt de les columnes especificades han de tenir valors únics entre les files de la taula
PRIMARY KEY (<cols>)	El conjunt de les columnes especificades formen la clau primària
FOREIGN KEY (<cols>) REFERENCES <taula> [<cols>]	El conjunt de columnes especificades formen una clau forana que referencia la taula indicada.
CHECK (<condicions>)	La taula ha de complir les condicions especificades. La condició pot referir-se a una o més columnes de la taula.

Les **restriccions de taula** poden referir-se a una o més columnes de la taula. Així, **en cas de restriccions que tenen a veure amb més d'una columna cal usar una restricció de taula.**

Per exemple, en cas de claus primàries compostes per més d'un columna o condició (CHECK) que tenen a veure amb més d'una columna.

Creació d'una taula: Exemple

```
CREATE TABLE empleats
    ( num_empl      INTEGER,
      nom_empl      CHAR(30) NOT NULL,
      sou           INTEGER DEFAULT 100000
                        CHECK (sou>80000),
      ciutat_empl   CHAR(30),
      num_dpt        INTEGER,
      num_proj       INTEGER,
      PRIMARY KEY (num_empl),
      FOREIGN KEY (num_dpt) REFERENCES
      departaments(num_dpt),
      FOREIGN KEY (num_proj) REFERENCES projectes(num_proj));
```


Inserció de files en una taula

```
INSERT INTO <nom_taula> [(<columnes>)]  
( VALUES {<valor1> | NULL}, ..., {<valorn> | NULL} ) | <consulta> ;
```

- En cas de no posar les **columnes** a continuació del **nom_taula**, els valors han de correspondre exactament als valors de les columnes en el **CREATE TABLE** i en el mateix ordre.
- En cas de posar les **columnes**, els valors han de correspondre als valors de les columnes explicitades i en el mateix ordre.
- Els valors de les columnes de la fila o files a inserir es poden obtenir també com a resultat d'una **consulta** (veure subconsultes).

Inserció de files en una taula: Exemples

INSERT INTO empleats

VALUES (4, 'RICARDO', 400000, 'BARCELONA', 1, 1);

INSERT INTO empleats (num_empl, num_dpt, num_proj,
nom_empl)

VALUES (11, 3, 2, 'NURIA');

empleats(<u>num_empl</u> , nom_empl, sou, ciutat_empl, num_dpt, num_proj)					
1	CARME	400000	MATARO	1	1
2	EUGENIA	350000	TOLEDO	2	2
3	JOSEP	250000	SITGES	3	1
4	RICARDO	400000	BARCELONA	1	1
11	NURIA	100000	NULL	3	2

- Files inserides per la primera sentència
- Files inserides per la segona sentència

Esborrat de files d'una taula

```
DELETE FROM <taula>  
WHERE <condicions>;
```

- S'eliminen de la **taula** les files que compleixen les **condicions** especificades a la clàusula **WHERE**.

Esborrat de files d'una taula: Exemples

DELETE FROM empleats
WHERE num_dpt=2;

DELETE FROM empleats
WHERE sou <= 250000;

empleats(<u>num_empl</u> , nom_empl, sou, ciutat_empl, num_dpt, num_proj)						
1	CARME	400000	MATARO	1	1	
2	EUGENIA	350000	TOLEDO	2	2	
3	JOSEP	250000	SITGES	3	1	
4	RICARDO	400000	BARCELONA	1	1	
11	NURIA	100000	NULL	3	2	

- Files esborrades per la primera sentència
- Files esborrades per la segona sentència

Modificació de files d'una taula

```
UPDATE <taula>
SET <col> = {expressió/ NULL} [,<col> = {expressió/ NULL}...]
WHERE <condicions> ;
```

- Es modifiquen de la manera indicada a la clàusula **SET** les columnes de les files de la **taula** que compleixen les **condicions** especificades a la clàusula **WHERE**.

Modificació de files d'una taula: Exemples

```
UPDATE empleats
SET sou = sou + 10000
WHERE num_dpt = 1;
```

```
UPDATE empleats
SET sou = sou + 50000, ciutat_empl = 'VIC'
WHERE num_empl = 11;
```

empleats(num_empl, nom_empl, sou, ciutat_empl, num_dpt, num_proj)

1	CARME	410000	MATARO	1	1
2	EUGENIA	350000	TOLEDO	2	2
3	JOSEP	250000	SITGES	3	1
4	RICARDO	410000	BARCELONA	1	1
11	NURIA	150000	VIC	3	2

- Files modificades per la primera sentència
- Files modificades per la segona sentència

Consultes sobre una taula: Format bàsic

```
SELECT <columnes_a_seleccionar> | *  
FROM <taula_a_consultar>  
[ WHERE <condicions> ] ;
```

- El resultat de la consulta és el valor de les **columnes_a_seleccionar** de la **taula_a_consultar** únicament per a la fila o files que aconsegueixen les **condicions** especificades a la clàusula **WHERE**.
- En el cas de no posar la clàusula **WHERE**, el resultat és el valor de les **columnes_a_seleccionar** per totes les files de la **taula_a_consultar**.
- Si posem un * en lloc de **columnes_a_seleccionar** indica que estem interessats en totes les columnes de la **taula_a_consultar**.

Consultes sobre una taula: Format bàsic - Exemple 1

```
SELECT *  
FROM empleats;
```

empleats(num_empl, nom_empl, sou, ciutat_empl, num_dpt, num_proj)

1	CARME	410000	MATARO	1	1
2	EUGENIA	350000	TOLEDO	2	2
3	JOSEP	250000	SITGES	3	1
4	RICARDO	410000	BARCELONA	1	1
11	NURIA	150000	VIC	3	2

■ Dades obtingudes com a resultat de la consulta

Consultes sobre una taula: Format bàsic - Exemple 2

```
SELECT num_empl, nom_empl, sou  
FROM empleats;
```

empleats(<u>num_empl</u> , nom_empl, sou, ciutat_empl, num_dpt, num_proj)					
1	CARME	410000	MATARO	1	1
2	EUGENIA	350000	TOLEDO	2	2
3	JOSEP	250000	SITGES	3	1
4	RICARDO	410000	BARCELONA	1	1
11	NURIA	150000	VIC	3	2

 Dades obtingudes com a resultat de la consulta

Consultes sobre una taula: Format bàsic - Exemple 3

```
SELECT num_empl, nom_empl, sou  
FROM empleats  
WHERE num_dpt = 3;
```

empleats(<u>num_empl</u> , nom_empl, sou, ciutat_empl, num_dpt, num_proj)					
1	CARME	410000	MATARO	1	1
2	EUGENIA	350000	TOLEDO	2	2
3	JOSEP	250000	SITGES	3	1
4	RICARDO	410000	BARCELONA	1	1
11	NURIA	150000	VIC	3	2

■ Dades obtingudes com a resultat de la consulta

Operadors en les condicions

■ operadors

- **aritmètics:** *, +, -, /
- **de comparació:** =, <, >, <=, >=, <>
- **lògics:** NOT, AND, OR
- **altres:**
 - <columna> **BETWEEN** <límit₁> **AND** <límit₂>
 - <columna> **IN** (<valor₁>, <valor₂> [...,<valor_N>])
 - <columna> **LIKE** <característica>
 - <columna> **IS [NOT] NULL**

Aquests operadors poden sortir a les **condicions**

- En la clàusula **WHERE** de les sentències d'esborrat (**DELETE**), modificació (**UPDATE**) i consulta (**SELECT**)
- En la clàusula **CHECK** de les sentències de creació d'una taula (**CREATE TABLE**).

Operadors en les condicions: Exemple

```

SELECT  num_empl, nom_empl
FROM    empleats
WHERE NOT(num_dpt = 2) AND
        ( ciutat_empl IN ('MATARO', 'SITGES', 'BARCELONA') OR
          ciutat_empl LIKE 'V%') AND
        num_proj IS NOT NULL AND
        sou BETWEEN 400000 AND 500000;

```

empleats(num_empl, nom_empl, sou, ciutat_empl, num_dpt, num_proj)						
1	CARME	410000	MATARO	1	1	
2	EUGENIA	350000	TOLEDO	2	2	
3	JOSEP	250000	SITGES	3	1	
4	RICARDO	410000	BARCELONA	1	1	
11	NURIA	150000	VIC	3	2	

■ Dades obtingudes com a resultat de la consulta

Consultes sobre una taula: Ordenació

```
SELECT <columnes_a_seleccionar> | *  
FROM <taula_a_consultar>  
[ WHERE <condicions> ]  
ORDER BY <columna> [DESC | ASC],.... ;
```

- En el resultat s'obté les dades ordenades segons les columnes que s'explicitin a la clàusula **ORDER BY**.
- Si per a una columna no es posa **DESC** s'enten que la classificació, segons els seus valors, es vol que sigui ascendent. Cosa que també es pot demanar explícitament amb la paraula clau **ASC**.

Consultes sobre una taula: Ordenació - Exemple

```
SELECT num_empl, nom_empl, sou
FROM empleats
WHERE num_dpt IN (1,2)
ORDER BY sou DESC, nom_empl;
```

resultat →

num_empl nom_empl sou

1	CARME	410000
4	RICARDO	410000
2	EUGENIA	350000
12	NURIA	150000

empleats(num_empl, nom_empl, sou, ciutat_empl, num_dpt, num_proj)

1	CARME	410000	MATARO	1	1
2	EUGENIA	350000	TOLEDO	2	2
3	JOSEP	250000	SITGES	3	1
4	RICARDO	410000	BARCELONA	1	1
11	NURIA	150000	VIC	3	2
12	NURIA	150000	MATARO	1	5

■ Files que compleixen la condició del WHERE

■ Dades tal com s'obtenen (ordenades) com a resultat de la consulta

Consultes sobre una taula: Resultats sense repeticions

```
SELECT [ DISTINCT | ALL] <columnes_a_seleccionar>  
FROM <taula_a_consultar>  
[ WHERE <condicions> ] ;
```

- Si volem que el resultat d'una consulta se'ns doni sense repeticions, cal utilitzar la paraula clau **DISTINCT**.
- Si no es posa res se'ns donarà el resultat amb repeticions (en cas de que n'hi hagin). Cosa que també es pot demanar explícitament amb la paraula clau **ALL**.

Consultes sobre una taula: Resultats sense repeticions

```
SELECT DISTINCT nom_empl, sou
FROM empleats
WHERE num_dpt IN (1,3);
```

resultat →

nom_empl	sou
CARME	410000
JOSEP	250000
RICARDO	410000
NURIA	150000

empleats(num_empl, nom_empl, sou, ciutat_empl, **num_dpt**, **num_proj**)

1	CARME	410000	MATARO	1	1
2	EUGENIA	350000	TOLEDO	2	2
3	JOSEP	250000	SITGES	3	1
4	RICARDO	410000	BARCELONA	1	1
11	NURIA	150000	VIC	3	2
12	NURIA	150000	MATARO	1	5

- Files que compleixen la condició del WHERE
- Dades obtingudes com a resultat de la consulta

Consultes sobre una taula: Funcions d'agregació

```
SELECT <funcions_d'agregació>  
FROM <taula_a_consultar>  
[ WHERE <condicions> ] ;
```

- Són funcions que s'apliquen sobre el conjunt de files de la **taula_a_consultar** que aconsegueixen les **condicions** especificades a la clàusula **WHERE**.
 - **COUNT**:
 - **COUNT(*)** - número de files que compleixen la condició del where
 - **COUNT(DISTINCT <columna>)** – número de valors diferents de la columna, sense comptar valors NULL, per a les files que compleixen la condició del where.
 - **COUNT(<columna>)** – número de valors de la columna, sense comptar valors NULL, per a les files que compleixen la condició del where.
 - **SUM (expressió), MIN(expressió), MAX(expressió), AVG(expressió)**:
 - **expressió** pot ser simplement una columna, o pot ser una càlcul a partir del valor de diferents columnes i constants.
 - **SUM**: dóna la suma dels valors resultants de calcular l'expressió per a les files que compleixen la condició del WHERE
 - **MIN**: dóna el valor mínim dels resultats de calcular l'expressió per a les files que compleixen la condició del WHERE
 - **MAX**: dóna el valor màxim dels resultats de calcular l'expressió per a les files que compleixen la condició del WHERE
 - **AVG**: dóna el valor promig dels resultats de calcular l'expressió per a les files que compleixen la condició del WHERE

Consultes sobre una taula: Funcions d'agregació - Exemple

```
SELECT COUNT(*) AS quantEmpl ,
       COUNT(DISTINCT nom_empl) AS
       quantNoms,
       SUM(sou*0.1) AS partSou
FROM empleats
WHERE num_dpt IN (1,3);
```

resultat →

quantEmpl	quantNoms	partSou
5	4	137000

empleats(num_empl, nom_empl, sou, ciutat_empl, num_dpt, num_proj)

1	CARME	410000	MATARO	1	1
2	EUGENIA	350000	TOLEDO	2	2
3	JOSEP	250000	SITGES	3	1
4	RICARDO	410000	BARCELONA	1	1
11	NURIA	150000	VIC	3	2
12	NURIA	150000	MATARO	1	5

- Files que compleixen la condició del WHERE
- Resultat de la consulta

Consultes sobre una taula: Agrupació de files

```
SELECT <columnes_a_seleccionar> [,<funcions_d'agregació>]  
FROM <taula_a_consultar>  
[ WHERE <condicions> ]  
GROUP BY <columnes_segons_les_que_agrupar>;
```

- S'organitza en grups les files de la **taula_a_consultar** que aconsegueixen les **condicions** especificades a la clàusula **WHERE**, segons el seu valor per les **columnes_segons_les_que_agrupar**.
- El resultat de la consulta és el valor de les **columnes_a_seleccionar** per cadascun dels grups de files obtinguts.
- Les **columnes_a_seleccionar** han de ser columnes que continguin el mateix valor per totes les files dins d'un grup.
- En el resultat es pot demanar també el valor de **funcions_d'agregació** que es calculen per cadascun dels grups de files obtinguts.


Consultes sobre una taula: Agrupació de files - Exemple

```
SELECT num_dpt,
       COUNT(*) AS quantEmpl
FROM empleats
WHERE num_dpt IN (1,3)
GROUP BY num_dpt;
```

resultat →

num_dpt	quantEmpl
1	3
3	2

	empleats(<u>num_empl</u> , nom_empl, sou, ciutat_empl, num_dpt, num_proj)					
grup1	1	CARME	410000	MATARO	1	1
	2	EUGENIA	350000	TOLEDO	2	2
	3	JOSEP	250000	SITGES	3	1
grup2	4	RICARDO	410000	BARCELONA	1	1
	11	NURIA	150000	VIC	3	2
	12	NURIA	150000	MATARO	1	5

-   Files que compleixen la condició del WHERE, agrupades segons indica la clàusula GROUP BY
-  Dades obtingudes com a resultat de la consulta. Hi ha un resultat per cada grup

Consultes sobre una taula: Condicions sobre grups

```
SELECT <columnes_a_seleccionar> [,<funcions_d'agregació>]  
FROM <taula_a_consultar>  
[ WHERE <condicions> ]  
GROUP BY <columnes_segons_les_que_agrupar>  
HAVING <condicions_per_grups>;
```

- En el cas de posar la clàusula **HAVING**, només apareix un resultat per cada grup que compleix les **condicions_per_grups**.
- Les **condicions_per_grups** seran comparacions entre constants, columnes amb un valor únic per cada grup i valors de funcions d'agregació (també amb un únic valor per cada grup). *Per exemple: si s'agrupa els empleats per departament, el sou de cada empleat d'un departament pot ser diferent, però el MAX del sou dels empleats d'un departament té valor únic.*
- Les **funcions d'agregació** té sentit aplicar-les a columnes que poden tenir valors diferents entre les files que componen els grups.





Consultes sobre una taula: Condicions sobre grups - Exemple 1

```
SELECT num_dpt, SUM(sou) AS sumaSous
FROM empleats
GROUP BY num_dpt
HAVING COUNT(*) >= 3;
```

resultat →

num_dpt	sumaSous
1	970000

empleats(<u>num_empl</u> , nom_empl, sou, ciutat_empl, num_dpt, num_proj)						
grup1	1	CARME	410000	MATARO	1	1
	2	EUGENIA	350000	TOLEDO	2	2
	3	JOSEP	250000	SITGES	3	1
	4	RICARDO	410000	BARCELONA	1	1
	11	NURIA	150000	VIC	3	2
	12	NURIA	150000	MATARO	1	5




 En no haver-hi WHERE, totes les files agrupades segons indica la clàusula GROUP BY

 Dades obtingudes com a resultat de la consulta, un resultat per cada grup que compleix la condició del HAVING.

Consultes sobre una taula: Condicions sobre grups - Exemple 2





```
SELECT DISTINCT num_dpt
FROM empleats
GROUP BY num_dpt, ciutat_empl
HAVING COUNT(*) >= 2;
```

resultat

num_dpt

1

empleats(num_empl, nom_empl, sou, ciutat_empl, num_dpt, num_proj)						
grup1	1	CARME	410000	MATARO	1	1
grup2	2	EUGENIA	350000	TOLEDO	2	2
grup3	3	JOSEP	250000	SITGES	3	1
grup4	4	RICARDO	410000	BARCELONA	1	1
	11	NURIA	150000	VIC	3	2
grup5	12	NURIA	150000	MATARO	1	5
	13	ALBERT	150000	BARCELONA	1	5




 En no haver-hi WHERE, són totes les files les que s'agrupen segons indica la clàusula GROUP BY

 Dades obtingudes com a resultat de la consulta. Un resultat per cada grup que compleix la condició del HAVING, sense resultats repetits degut a la clàusula DISTINCT.

Consultes sobre més d'una taula: Format bàsic

```
SELECT <columnes_a_seleccionar> | *  
FROM <taules_a_consultar>  
[ WHERE <condicions> ] ;
```

- El resultat de la consulta és el valor de les **columnes_a_seleccionar** de les **taules_a_consultar** únicament per a la fila o files que compleixen les **condicions** especificades a la clàusula **WHERE**.
- En el cas de no posar la clàusula **WHERE**, el resultat és el valor de les **columnes_a_seleccionar** per a les files obtingudes del producte cartesià de les files a les **taules_a_consultar**.
- Si posem un * en lloc de les **columnes_a_seleccionar** indica que estem interessats en totes les columnes de les **taules_a_consultar**.

Consultes sobre més d'una taula: Format bàsic – Exemple 1

SELECT *
FROM empleats e, projectes p;

resultat →

e.num_empl	e.nom_empl	e.num_proj	p.num_proj	p.nom_proj
1	CARME	1	1	IBDTEL
1	CARME	1	2	IBDVID
1	CARME	1	3	IBDTEF
3	JOSEP	1	1	IBDTEL
3	JOSEP	1	2	IBDVID
3	JOSEP	1	3	IBDTEF

empleats(num_empl, nom_empl, num_proj) **projectes(num_proj, nom_proj)**

1	CARME	1	1	IBDTEL
3	JOSEP	1	2	IBDVID
			3	IBDTEF

- Dades obtingudes com a resultat de la consulta. Cal notar que, de totes les files, les que segurament podem estar interessats, són aquelles marcades en negreta (són aquelles en què el número de projecte en què treballa l'empleat coincideix amb el número de projecte del projecte).

Consultes sobre més d'una taula: Format bàsic – Exemple 2

```

SELECT  e.num_empl, p.num_proj,
          p.nom_proj
FROM    empleats e, projectes p
WHERE    e.num_proj = p.num_proj;
  
```

resultat →

e.num_empl	p.num_proj	p.nom_proj
1	1	IBDTEL
3	1	IBDTEL

empleats(<u>num_empl</u> , nom_empl, <u>num_proj</u>)				projectes(<u>num_proj</u> , nom_proj)	
1	CARME	1		1	IBDTEL
3	JOSEP	1		2	IBDVID
				3	IBDTEF

- Dades obtingudes com a resultat de la consulta. Cal notar que només apareixen aquelles combinacions en les que coincideix el número de projecte en què treballa l'empleat, amb el número de projecte del projecte.

Consultes sobre més d'una taula

Sintaxis alternativa– Clàusula Inner Join - Exemple 3

La condició de combinació de les taules es pot escriure:

- O bé en la clàusula WHERE
- O bé usant la clàusula JOIN en el FROM
 - INNER JOIN requereix la condició de combinació de manera explícita
 - NATURAL INNER JOIN fa la combinació per les columnes amb el mateix nom en les taules implicades.

```
SELECT e.num_empl, p.num_proj, p.nom_proj  
FROM empleats e, projectes p  
WHERE e.num_proj = p.num_proj;
```

```
SELECT e.num_empl, p.num_proj, p.nom_proj  
FROM empleats e INNER JOIN projectes p ON e.num_proj = p.num_proj;
```

```
SELECT e.num_empl, p.num_proj, p.nom_proj  
FROM empleats e NATURAL INNER JOIN projectes p;
```

Les tres sentències anteriors són totalment equivalents.

Consultes sobre més d'una taula: General

- En les consultes sobre més d'una taula es pot aplicar totes les variants que es mostren en aquest capítol per a les consultes sobre una única taula. És a dir:
 - resultats sense repeticions
 - funcions d'agregació
 - ordenació de les dades
 - agrupació de files
 - ...



Consultes sobre vàries taules: Exemple amb agrupació de files

```
SELECT p.num_proj, p.nom_proj
FROM projectes p, empleats e
WHERE p.num_proj = e.num_proj
GROUP BY p.num_proj
HAVING p.pressupost < SUM(e.sou);
```

resultat →

p.num_proj	p.nom_proj
1	IBDTEL

	e.num_empl	e.nom_empl	e.sou	e.num_proj	p.num_proj	p.nom_proj	p.pressupost
grup1	1	CARME	410000	1	1	IBDTEL	1000000
	2	EUGENIA	350000	2	2	IBDVID	500000
grup2	3	JOSEP	250000	1	1	IBDTEL	1000000
	4	RICARDO	410000	1	1	IBDTEL	1000000
	11	NURIA	150000	2	2	IBDVID	500000

-  Files resultants de la combinació dels empleats amb els projectes, agrupades segons la clàusula GROUP BY
-  Dades obtingudes com a resultat de la consulta. Un resultat per cada grup que compleix la condició del HAVING. Només hi ha un projecte tal que el seu pressupost és inferior a la suma del sou dels empleats que hi treballen.

Cal notar que l'atribut p.nom_proj pot estar entre els atributs del select perquè té un valor únic per cada grup. Igualment, l'atribut p.pressupost pot estar en la condició del having perquè té un valor únic per cada grup.

Consultes: Unió

```
SELECT <columnes_a_seleccionar> | *  
FROM <taules_a_consultar>  
[ WHERE <condicions> ]  
UNION  
SELECT <columnes_a_seleccionar> | *  
FROM <taules_a_consultar>  
[ WHERE <condicions> ]  
[ORDER BY <columna> [DESC|ASC],...] ;
```

- El resultat és la unió del resultat obtingut de les dues sentències **SELECT**.
- Les **columnes_a_seleccionar** en les dues sentències **SELECT** han de ser semànticament compatibles
- Sempre s'obté resultats sense repeticions (en molts SGBDR ja surten ordenats).
- Les columnes que apareixen a clàusula **ORDER_BY** han de ser un subconjunt de les **columnes_a_seleccionar** del primer **SELECT**.

Consultes: Unió - Exemple

```
SELECT ciutat_empl
FROM empleats
UNION
SELECT ciutat_dpt
FROM departaments
ORDER BY ciutat_empl DESC;
```

resultat →

ciutat_empl

SITGES
MATARO
MADRID
BARCELONA

empleats(<u>num_empl</u> , nom_empl, ciutat_empl)			departaments(<u>num_dpt</u> , ciutat_dpt)	
1	CARME	MATARO	1	BARCELONA
3	JOSEP	SITGES	2	MADRID
			3	BARCELONA

■ Dades obtingudes com a resultat de la consulta.

Consultes: Diferència

```
SELECT <columnes_a_seleccionar> | *  
FROM <taules_a_consultar>  
WHERE <columna_taula> NOT IN ( SELECT <columna_a_seleccionar>  
                                FROM <taules_a_consultar>  
                                [ WHERE <condicions> ] );  
  
SELECT <columnes_a_seleccionar> | *  
FROM <taules_a_consultar>  
WHERE NOT EXISTS ( SELECT *  
                    FROM <taules_a_consultar>  
                    WHERE <condicions> ) ;
```

- Hi ha dos formes alternatives de fer una diferència amb NOT IN o bé amb NOT EXISTS.
- Un NOT IN és cert si el valor de la columna *columna_taula* no està en el resultat de la subconsulta.
- Un NOT EXISTS és cert si la subconsulta no dona cap resultat.
- Hi ha altres maneres de fer una diferència (operador Except en SQL estàndard), encara que hi ha diferents noms de l'operador en diferents sistemes.

Consultes: Diferència – Exemples

```
SELECT p.num_proj, p.nom_proj
FROM projectes p
WHERE p.num_proj NOT IN (SELECT e.num_proj
                        FROM empleats e);
```

```
SELECT p.num_proj, p.nom_proj
FROM projectes p
WHERE NOT EXISTS (SELECT * FROM empleats e
                 WHERE p.num_proj = e.num_proj);
```

resultat →

p.num_proj	p.nom_proj
2	IBDVID
3	IBDTEF
4	IBDCOM

empleats(<u>num_empl</u> , nom_empl, <u>num_proj</u>)			projectes(<u>num_proj</u> , nom_proj)	
1	CARME	1	1	IBDTEL
3	JOSEP	1	2	IBDVID
			3	IBDTEF
			4	IBDCOM



Dades obtingudes com a resultat de la consulta, en qualsevol de les dues alternatives.

En qualsevol cas, la consulta dona aquells projectes que no tenen cap empleat assignat.

NOT IN: Un projecte és al resultat de la consulta si el seu número no està entre els números de projecte dels empleats.

NOT EXISTS: Un projecte és al resultat de la consulta si no existeix cap empleat que tingui el seu número de projecte.

Subconsultes en sentències Delete, Update i Select

- Poden apareixer en aquelles sentències on hi ha la clàusula **WHERE**:

- Esborrat de files d'una taula

```
DELETE FROM <taula>  
WHERE .... (SELECT ..... );
```

- Modificació de files d'una taula

```
UPDATE <taula>  
SET ....  
WHERE ..... (SELECT ..... );
```

- Consultes una taula o més taules

```
SELECT <columnes_a_seleccionar>  
FROM <taules_a_consultar>  
WHERE ..... (SELECT ..... );
```

Subconsultes en sentències Delete - Exemple

```
DELETE FROM projectes
WHERE NOT EXISTS (SELECT *
                  FROM empleats e
                  WHERE e.num_proj = projectes.num_proj);
```

empleats(<u>num_empl</u> , nom_empl, <u>num_proj</u>)			projectes(<u>num_proj</u> , nom_proj)	
1	CARME	1	1	IBDTEL
3	JOSEP	1	2	IBDVID
			3	IBDTEF
			4	IBDCOM

- Files esborrades com a resultat de la sentència. La sentència elimina aquells projectes que no tenen cap empleat assignat. Cal notar que la subconsulta obté resultats per aquells projectes que tenen com a mínim un empleat.

Subconsultes en sentències Update - Exemple

```

UPDATE projectes
SET pressupost = pressupost + (pressupost * 0,1)
WHERE 2 <= (SELECT COUNT(*)
            FROM empleats e
            WHERE projectes.num_proj = e.num_proj);

```

empleats(<u>num_empl</u> , nom_empl, num_proj)			projectes(<u>num_proj</u> , pressupost)	
1	CARME	1	1	1100000
3	JOSEP	1	2	500000
			3	4500000
			4	2000000

- Files modificades com a resultat de la sentència. La sentència puja el pressupost d'aquells projectes que tenen dos o més empleats assignats. Cal notar que la subconsulta el que fa és calcular el nombre d'empleats del projecte que s'està considerant si cal modificar o no.

Subconsultes a la clàusula SET de sentències UPDATE

- És possible utilitzar el resultat d'una subconsulta per assignar un nou valor a una columna dins d'una sentència **UPDATE**.
- La subconsulta ha retornar un únic valor per a cada fila que s'està modificant.



```
UPDATE <taula>  
SET <col> = (SELECT <operació_columna> FROM ...)  
WHERE <condicions> ;
```

Subconsultes a la clàusula SET de sentències UPDATE: Exemple

```
UPDATE empleats
SET sou = (SELECT AVG(e2.sou)
           FROM empleats e2
           WHERE e2.num_dpt = 1)
WHERE num_dpt = 3;
```

empleats(num_empl, nom_empl, sou, ciutat_empl, num_dpt num_proj)

1	CARME	200000	MATARO	1	1
2	EUGENIA	350000	TOLEDO	2	2
3	JOSEP	150000	SITGES	3	1
4	RICARDO	100000	BARCELONA	1	1
11	NURIA	150000	VIC	3	2

-  Files seleccionades en la subconsulta
-  Files actualitzades en l'UPDATE

Subconsultes en sentències Select – Exemple 1

```

SELECT e.num_empl, e.nom_empl
FROM empleats e
WHERE e.sou > ( SELECT AVG(e1.sou)
                  FROM empleats e1);

```

empleats(num_empl, nom_empl, sou, ciutat_empl, num_dpt, num_proj)						
1	CARME	410000	MATARO	1	1	
2	EUGENIA	350000	TOLEDO	2	2	
3	JOSEP	250000	SITGES	3	1	
4	RICARDO	410000	BARCELONA	1	1	
11	NURIA	150000	VIC	3	2	

- Dades obtingudes com a resultat de la sentència. La sentència dona els empleats que tenen un sou superior a la mitjana del sou de tots els empleats. Cal notar que la subconsulta retorna la mitjana del sou de tots els empleats de la taula empleats.

Subconsultes en sentències Select – Exemple 2

```

SELECT count(*) as quantsProjSobrePress
FROM projectes p
WHERE p.pressupost < (SELECT SUM(e.sou)
                      FROM empleats e
                      WHERE e.num_proj=p.num_proj);

```

resultat →




quantsProjSobrePress

1

projectes(num_proj,nom_proj,p.pressupost) empleats(num_empl, nom_empl, sou, num_proj)

1	IBDTEL	1000000
2	IBDVID	500000

1	CARME	410000	1
2	EUGENIA	350000	2
3	JOSEP	250000	1
4	RICARDO	410000	1
11	NURIA	150000	2

-  Dades obtingudes com a resultat de la sentència. La sentència dona la quantitat de projectes que tenen un pressupost inferior a la suma del sou dels empleats que hi estan assignats. Cal notar que la subconsulta retorna la suma del sou dels empleats assignats al projecte que s'està considerant.
-  Files seleccionades en la subconsulta quan es considera el projecte número 1. La suma dels sous és 1070000.
-  Files seleccionades en la subconsulta quan es considera el projecte número 2. La suma dels sous és 500000

Subconsultes en sentències Insert i clàusules Having

- En aquelles sentències on hi ha la clàusula **HAVING**:

```
SELECT <columnes_a_seleccionar>  
FROM <taules_a_consultar>  
WHERE <condicions>  
GROUP BY <columnes_agrupació>  
HAVING ..... (SELECT ..... );
```

- Finalment, en sentències d'inserció de files a una taula (en aquest cas el resultat de la subconsulta és un conjunt de files que ha de ser compatible amb la definició de la taula en el **CREATE TABLE**)

```
INSERT INTO <taula>  
(SELECT ..... );
```

Subconsultes en clàusules Having – Exemple

```

SELECT d.num_dpt, d.nom_dpt, 100*SUM(e.sou)/d.pressupost AS percSous
FROM departaments d, empleats e
WHERE d.num_dpt = e.num_dpt
GROUP BY d.num_dpt
HAVING SUM(e.sou) > ( SELECT SUM(e1.sou)
                      FROM empleats e1
                      WHERE e1.num_dpt = 3);

```

resultat

d.num_dpt	d.nom_dpt	percSous
1	DIRECCIO	82



empleats(num_empl, nom_empl, sou, num_dpt)				departaments(num_dpt, nom_dpt, pressupost)		
1	CARME	410000	1	1	DIRECCIO	1000000
2	EUGENIA	350000	2	2	DIRECCIO	2000000
3	JOSEP	350000	3	3	MARQUETING	2500000
4	RICARDO	410000	1			
11	NURIA	150000	3			

- Dades obtingudes com a resultat de la sentència. La sentència dona els departaments tals que la suma del sou dels seus empleats és superior a la suma del sou dels empleats del departament número 3. Cal notar que la subconsulta retorna la suma del sou dels empleats del departament número 3.
- Files seleccionades en la subconsulta sigui quin sigui el departament que es considera. La suma dels sous és 500000.

Subconsultes en sentències Insert – Exemple

```
INSERT INTO clients
(SELECT num_empl,nom_empl,200000
FROM empleats
WHERE num_dpt IN (2,3));
```

clients(<u>num_client</u> , nom, credit)			empleats(<u>num_empl</u> , nom_empl, num_dpt)		
2	EUGENIA	200000	1	CARME	1
3	JOSEP	200000	2	EUGENIA	2
			3	JOSEP	3
			4	RICARDO	1

-  Files seleccionades a la subconsulta.
-  Files inserides a la taula clients.

Tractament de dades temporals: Tipus de dades

- Aquests tipus de dades permeten realitzar operacions aritmètiques i comparacions de manera intuïtiva

DATE	Emmagatzema únicament la data (any, mes, dia)	'2025-09-15'
TIME	Emmagatzema l'hora del dia. Pot incloure o no la zona horària (WITH / WITHOUT TIME ZONE)	'14:30:00'
TIMESTAMP	Emmagatzema la data i l'hora. Pot incloure o no la zona horària (WITH / WITHOUT TIME ZONE)	'2025-09-15 14:30:00'
INTERVAL	Emmagatzema un període de temps o duració	'2 days' o '3 hours 30 minutes'

Tractament de dades temporals: Creació i Inserció

```
CREATE TABLE vols (  
    codi_vol                VARCHAR(10) PRIMARY KEY,  
    origen                  VARCHAR(50),  
    destí                   VARCHAR(50),  
    data_sortida            DATE NOT NULL,  
    hora_sortida_previsita  TIME NOT NULL,  
    moment_enlairament     TIMESTAMP WITH TIME ZONE );
```

INSERT INTO vols **VALUES**

```
('IB2811', 'BARCELONA', 'MADRID', '2025-08-20', '10:00:00', '2025-08-20 10:12:00+02'),  
( 'RY1020', 'BARCELONA', 'ROMA',   '2025-08-20', '11:30:00', '2025-08-20 11:35:00+02'),  
( 'VY1530', 'GIRONA',   'LONDRES', '2025-08-22', '09:15:00', NULL);
```

Funcions i Operadors Temporals

- SQL proporciona funcions per obtenir informació del sistema i extreure parts de les dates

NOW() / CURRENT_TIMESTAMP	Retorna la data i hora actuals amb zona horària
CURRENT_DATE	Retorna la data actual del sistema
CURRENT_TIME	Retorna l'hora actual del sistema
EXTRACT (<i>part FROM data</i>)	Extreu una part específica d'una data/hora. Ex <i>part</i> : YEAR , MONTH , DAY , HOURL , DOW (dia de la setmana), DOY (dia de l'any).
TO_CHAR (<i>data, format</i>)	Converteix una data a una cadena de text amb un format específic. Ex. TO_CHAR(NOW(), 'DD/MM/YYYY') .
+ / - amb INTERVAL	Permet sumar o restar intervals de temps. Ex. CURRENT_DATE + INTERVAL '1 month'

Consultes amb dades temporals: Exemples

- Obtenir els vols que surten un dia concret

```
SELECT codi_vol, destí  
FROM vols  
WHERE data_sortida = '2025-08-20;
```

- Trobar vols que s'han enlairat amb més de 10 minuts de retard

```
SELECT codi_vol, moment_enlairament  
FROM vols  
WHERE moment_enlairament > (data_sortida + hora_sortida_prevista +  
                           INTERVAL '10 minutes');
```

- Mostrar els vols del cap de setmana (DOW: 0 = Diumenge, 6 = Dissabte).

```
SELECT codi_vol, TO_CHAR(data_sortida, 'Day') AS dia_setmana  
FROM vols  
WHERE EXTRACT(DOW FROM data_sortida) IN (6, 0);
```