Aprenent SQL

TEORIA I EXERCICIS BÀSICS

Institut Sabadell

Els inicis de tot plegat

El model relacional en què es basen el SGBD actuals va aparèixer al 1970 fruit del desenvolupament de Edgar Frank Codd, un matemàtic que treballava per a IBM.

Un dels primers SGBD va ser **System R** d'IBM. El prototip de SGBD anava acompanyat d'un llenguatge de "programació", anomenat SEQUEL, que es va condensar posteriorment en SQL (Structured Query Language)

Institut Sabadell

Cronologia

1995	Primer lanzamiento internacional del DBMS de código abierto MySQL por parte de MySQL AB
2008	Venta de MySQL a Sun Microsystems
2009	Lanzamiento de la bifurcación de MySQL, MariaDB, por parte de Monty Program AB
2010	Adquisición de Sun Microsystems por parte de Oracle
2012	Creación de la MariaDB Foundation
2014	Fusión de Monty Program AB con SkySQL, que pasan a llamarse MariaDB Corporation.

Institut Sabadell

Evolució de SQL

Un cop es va comprovar l'eficàcia del model relacional, les diferents marques comercials van competir entre elles, cada una amb les seves versions:

- IBM
- Relational Software (actualment Oracle Corporation)

Fins aconseguir un estàndard (al 1986), que va permetre una versió comuna:

SLQ-86

Institut Sabadell

SQL

- Structured Query Language.
- Llenguatge estàndard pel model de base de dades relacional.
- Llenguatge declaratiu.
- Ens permet crear, modificar, consultar i eliminar base de dades.
- Molts SGBD utilitzen SQL. (Oracle, Microsoft SQL Server, MySQL, MariaDB,...)

Institut Sabadell

Tipus de sentències SQL

SQL permet realitzar diferents tasques sobre el SGBD i per això hi ha diferents tipus de sentències, que agrupem en:

a) Sentències destinades a la definició de dades (**LDD-DDL**), que permeten definir objectes com a taules, camps, valors possibles, regles d'integritat referencial, restriccions....

b) Sentències destinades al control sobre les dades (**LCD-DCL**), que permeten concedir i retirar permisos sobre els diferents objectes de la base de dades

Institut Sabadell

Tipus de sentències SQL

c) Sentències destinades a la consulta de les dades (**LC- TCL**), que permeten accedir a les dades en mode consulta

d) Sentències destinades a la manipulació de les dades (**LMD-DML**), que permeten actualitzar la base de dades (altes, baixes i modificacions)

Institut Sabadell

Tipus de dades

- Un tipus de dades defineix els valors que pot prendre un camp i a més les operacions que podran fer-se amb aquests valors.

- L'evolució anàrquica del SQL ha permès que hi hagin molts tipus de dades que costa d'unificar. Són similars, però cada SGBD incorpora tipus de dades que haurem de clarificar en la documentació que cada SGBD incorpora.

- Per treballar amb un SGBD hem de conèixer els principals tipus de dades que facilita (numèriques, alfanumèriques...) i ho hem de fer centrant-nos en un SGBD concret (Oracle, MySQL, PostgreSQL, etc)

Institut Sabadell

Tipus de dades

Quan es crea una taula (o clúster) hem d'especificar un tipus de dada de cada columna. Això és per:

- Els tipus de dades definiran el domini de valors que cada columna podrà contenir i les operacions que es podran fer.

- Cada valor en una columna determinada assumirà el tipus de dada de la columna (i donarà error si no s'introdueix coherentment)

Institut Sabadell

Tipus de dades incorporades

Podem distingir:

- Tipus de dades per gestionar informació alfanumèrica
- Tipus de dades per gestionar informació numérica
- Tipus de dades per gestionar el temps (dates i temps)
- Tipus de dades per gestionar gran volum d'informació (binaris)

....

  - https://mariadb.com/kb/en/data-types/

Institut Sabadell

Tipus de dades

Una possibilitat de classificació molt simplificada seria aquesta:

Alfanumèrics	Numèrics	Data
CHAR VARCHAR	INT FLOAT DOUBLE	DATE TIME

Institut Sabadell

Llenguatge DML

- Les sentències DML (data manipulation language) del llenguatge SQL són:
  - *SELECT*. S'utilitza per extreure informació de la base de dades
  - *INSERT*. Insereix un o diversos registres a alguna taula.
  - *DELETE*. Esborrar registres d'una taula
  - *UPDATE*. Actualitza els registres d'una taula.

Institut Sabadell

Normes d'escriptura

- SQL no distingeix majúscules i minúscules (només en les dades).
- Les instruccions finalitzen amb punt i coma.
- Qualsevol comanda SQL pot ser partida amb espais en blancs o salts per millorar la seva lectura. Permet també les tabulacions.
- Per afegir comentaris: /* això és un comentari
  amb dos línies */
  -- comentari d'una línia

Institut Sabadell

SELECT

- La sentència **select** s'utilitza per consultar informació d'una base de dades i recuperar dades seleccionades que compleixen els criteris especificats.
- El format d'una sentència de selecció simple és:

  select "columna1" [,"columna2",etc,...]

  from "nom de la taula"

  [**where** "condició"];
- L'ordre de les clàusules és important.
- Els claudàtors indiquen opcionalitat en la sentència.

Institut Sabadell

WHERE (filtre)

- Condicions que qualsevol gestor de base de dades interpreta per seleccionar registres i mostrar.
- Els operadors que afecten a la clausula WHERE són:
  - Operadors aritmètics
  - De dates
  - De text
  - De comparació

Institut Sabadell

Operadors aritmètics

- Són els típics operadors:

+, -, * , /

- Es fan servir per formar expressions amb constants, valors de columnes i funcions de valors de columnes.

Institut Sabadell

Operadors de dates

- Podem sumar dies a una data, per obtenir una nova data (+).
- Podem restar un nombre determinar de dies a una data i obtenir una nova data (-).
- Podem restar dues dates i obtenir el nombres de dies que les separen (-).

Institut Sabadell

Operadors de text

- Bàsicament l'operador || per concatenar cadenes de caràcters.

Institut Sabadell

Operadors de comparació

- Els típics per efectuar comparacions que ens donarà un resultat booleà (true or false) :

  =, !=, >, >=, <, <=

- L'operador **LIKE** (també tenim un NOT LIKE)
- Un conjunt d'operadors lògics:
  - (**NOT**) **BETWEEN** valor1 AND valor2
  - (**NOT**) **IN** (llista de valors separats per comes)
  - **IS** (**NOT**) **NULL**
  - I després una sèrie d'ajuts als operadors de comparació (**ANY**, **ALL**)

Institut Sabadell

Operadors de comparació

- **LIKE**:
  - L'operador de coincidència LIKE de patrons similars. Es tracta d'un operador que li permet seleccionar només les files que són "com" el que especifiqueu.
- **Els comodins ("%" i "_")**
  - "%" : pot ser utilitzat com a comodí per a que coincideixi amb qualsevol caràcter possible que pugui aparèixer abans o després dels caràcters especificats.

Exemple:

```
select first, last
from empinfo
where last LIKE '%s';
```

Aquesta sentència selecciona els noms i cognoms de la taula empinfo que el seu cognom acaba en 's'

Institut Sabadell

Operadors de comparació

- “_" : El signe guió baix pot ser utilitzat com a comodí per a que coincideixi amb un únic caràcter.

Exemple:

https://www.techonthenet.com/mariadb/comparison_operators.php

```
select first, last
from empinfo
where last LIKE 's_';
```

Aquesta sentència ens selecciona els noms i cognoms d’aquells treballadors que el seu cognom té dues lletres i la primera és s.

Institut Sabadell

Exemples

```
SELECT emp_no  AS “codi”, cognom AS “Empleat”, salari*14 AS “salari anual”
FROM emple
WHERE salari>=2000;
```

```
SELECT cognom AS “Empleat”
FROM emp
WHERE UPPER (cognom) NOT LIKE ‘S%’
```

Institut Sabadell

Exemple.- between

Els dos exemples són iguals:

```
SELECT cognom, salari
FROM emp
WHERE salari>=10000 AND salari<=20000
```

```
SELECT cognom, salari
FROM emp
WHERE salari BETWEEN 10000 AND 20000
```

Institut Sabadell

Exemple.- IN, ANY

Tres maneres de fer el mateix:

```
SELECT cognom, dep
FROM emp
WHERE dep_no=10 OR dep_no=30;
```

```
SELECT cognom, dep
FROM emp
WHERE dep_no IN (10, 30);
```

```
SELECT cognom, dep
FROM emp
WHERE dep_no=ANY(10, 30);
```

Institut Sabadell

ORDER BY

- A la clausula WHERE li podem afegir un **ORDER BY**, indicant el camp pel qual volem organitzar les dades filtrades.
- També podem indicar si el volem ascendent o descendent.

```
SELECT * FROM emp
WHERE…
ORDER BY dept_no ASC, last DESC;
```

Institut Sabadell

Exercici

- A partir de la taula següent Suposem que la taula es diu *emp*:

first	last	id	age	city	state
John	Jones	99980	45	Payson	Arizona
Mary	Jones	99982	25	Payson	Arizona
Eric	Edwards	88232	32	San Diego	California
Mary Ann	Edwards	88233	32	Phoenix	Arizona
Ginger	Howell	98002	42	Cottonwood	Arizona
Sebastian	Smith	92001	23	Gila Bend	Arizona
Gus	Gray	22322	35	Bagdad	Arizona
Mary Ann	May	32326	52	Tucson	Arizona
Erica	Williams	32327	60	Show Low	Arizona
Leroy	Brown	32380	22	Pinetop	Arizona
Elroy	Cleaver	32382	22	Globe	Arizona

Institut Sabadell

Exercici

- **Indica quins serien els resultats de les sentències següents:**

a) select first, last, city from emp;
b) select last, city, age from emp where age > 30;
c) select first, last, city, state from emp where first LIKE 'J%';
d) select * from emp;
e) select first, last, from emp where last LIKE '%s';
f) select first, last, age from emp where last LIKE '%illia%';
g) select * from emp where first = 'Eric';

Institut Sabadell

Consultes sobre més d'una taula

- SQL permet efectuar operacions sobre els resultats de les sentències SELECT per tal d'obtenir un nou resultat.
- Tenim tres **operacions** possibles:
  - **Unió**
  - **Intersecció**
  - **Diferència**.
- Cal tenir present que els conjunts sobre els que operem han de ser compatibles, és a dir, igual nombre de columnes i dades compatibles

Institut Sabadell

Unió de sentències SELECT

- SQL ens proporciona l'operador **UNION** per combinar totes les files del resultat d'una sentència SELECT amb totes les files del resultat d'una altra sentència SELECT (eliminant duplicació entre files).

sentència_select_sense_order_by

union

sentència_select_sense_order_by

[order by ...]

Institut Sabadell

Exemple UNION

Suposem que tenim les següents taules:

Taula Vendes_BOTIGUES

NOM_Botiga	Vendes	Data
Barcelona	1500	1/ene/2015
Sabadell	300	2/ene/2015
Terrassa	340	5/ene/2015
Manresa	260	9/ene/2015

Taula Vendes_Internet

Data	VENDES
5/ene/2015	3000
7/ene/2015	2100
9/ene/2015	300

Volem obtenir les dades en que s'ha fet una venda

Institut Sabadell

Exemple UNION

```
SELECT data

FROM Vendes_Botiga

UNION

SELECT data FROM Vendes_Internet;
```

Data
1/ene/2015
2/ene/2015
5/ene/2015
7/ene/2015
9/ene/2015

UNION o UNION ALL

- Si fem servir UNION no es repeteix
- Si fem servir UNION ALL surten totes les escollides i si es repeteixen tornen a sortir.

Institut Sabadell

Intersecció de sentències SELECT

- SQL ens dona la opció de presentar les files que són simultàniament de dos conjunts amb l'operador **INTERSECT**

sentència_select_sense_order_by

intersect

sentència_select_sense_order_by

[order by ...]

Institut Sabadell

Exemple INTERSECT

En el mateix exemple anterior, si demanem:

SELECT data FROM Vendes_Botiga

INTERSECT

SELECT data FROM Vendes_Internet;

El que ens donarà és els dies en que les vendes s'han produït en la botiga i per internet.

5/ene/2015

Institut Sabadell

Diferència de sentències SELECT

- Tenim també l'operador **MINUS** per tal de presentar les files que es troben en el primer i en canvi no en el segon.

sentència_select_sense_order_by
minus
sentència_select_sense_order_by
[order by ...]

Institut Sabadell

Exemple MINUS

En el mateix exemple anterior, si demanem:

SELECT data FROM Vendes_Botiga

MINUS

SELECT data FROM Vendes_Internet;

Només ens apareixeria l'1 de gener, que és l'únic dia en que hem fet vendes a la botiga i no hem fet cap venda per internet

Institut Sabadell

JOIN

- S'utilitza per fer combinacions entre taules, producte de l'evolució dels estàndards SQL i dels diversos SGBD comercials existents.
- Fem servir SQL Joins per combinar files de dos o més taules.
- El tipus més comú és **SQL INNER JOIN** (o simple JOIN), on fem la selecció a partir d'un camp comú.

Institut Sabadell

Exemple JOIN

Suposem que tenim dues taules: una de **comandes** i altra de **clients** (en la taula de comandes tenim el ID del client que ha fet la comanda)

IDCom	IDClient	DataCom
10308	2	18/9/2015
10309	37	19/9/2015
10310	38	20/9/2015
10311	1	25/9/2015
10312	20	25/9/2015

IDClient	Nom	Contacte	Ciutat
1	Fruites Lluis	Lluis Camps	Terrassa
2	Superverd	Gerard Lopez	Sabadell

Institut Sabadell

Exemple JOIN

Volem que ens mostri de cada comanda, l'identificador, el nom del client que ha fet la comanda i quin dia ho ha fet.

```
SELECT comandes.IDCom, clients.Nom, Comandes. Data
FROM comandes (INNER) JOIN clients
ON comandes.IDclient=clients.IDClient
```

IDCom	Nom	DataCom
10308	Superverd	18/9/2015
10309	Mercat Parada2	19/9/2015
10310	Esladas &Fruts	20/9/2015
10311	Fruites Lluis	25/9/2015
10312	Vegetalia	25/9/2015

Institut Sabadell

Exemple JOIN

Suposem que tenim dues taules: una de **departament** i altra **d'empleat**.

CodiDept	Departament
1	Qualitat
2	Informàtica
3	Vendes
4	Producció

CodEmpleat	Nom	Cognom	CodiDept	edat
1	Pep	Crespo	2	54
2	Artur	Lopez	4	40
3	Pau	Diaz	1	35
4	Maria	Sans		50

Institut Sabadell

Exemple JOIN

Volem que ens mostri el nom i cognom dels empleats i el departament que tenen associat.

```
SELECT empleat. Nom, empleat.Cognom, departament.departament
FROM empleat JOIN departament
ON empleat.CodiDept=departament.CodiDept
```

Nom	Cognom	Departament
Pep	Crespo	Informàtica
Artur	Lopez	Producció
Pau	Diaz	Qualitat

Només mostra els que tenen departament associat.

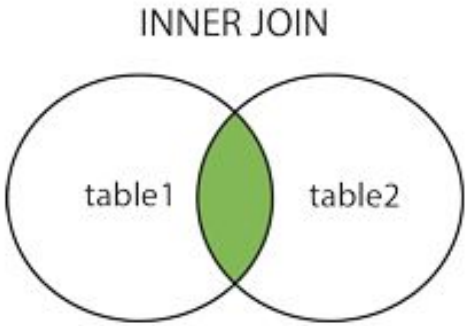
I si volem que surtin tots els empleats encara que no tinguin departament associat?

Institut Sabadell

Tipus de JOIN

- **INNER JOIN=JOIN**: Retorna totes les files que tenen un punt en comú en les dues taules.

```
SELECT column_name(s)
FROM table1
INNER JOIN table2 ON table1.column_name = table2.column_name;
```

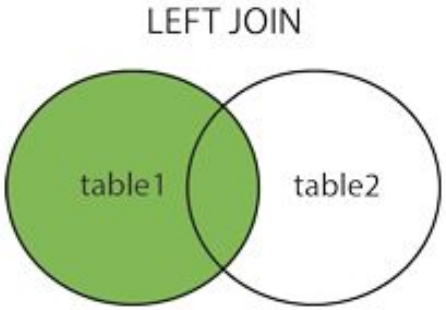


Institut Sabadell

Tipus de JOIN

- **LEFT JOIN**: Retorna totes les files de la taula de l'esquerra i enllaça les files de la taula de la dreta.

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2 ON table1.column_name = table2.column_name;
```

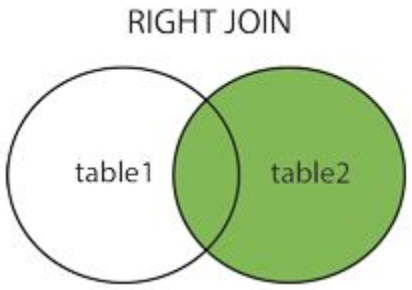


Institut Sabadell

Tipus de JOIN

- **RIGHT JOIN**: Retorna totes les files de la taula de la dreta i enllaça les files de la taula de l'esquerra.

```
SELECT column_name(s)
FROM table1
RIGHT JOIN table2 ON table1.column_name = table2.column_name;
```



Institut Sabadell

Exemples Tipus de JOIN

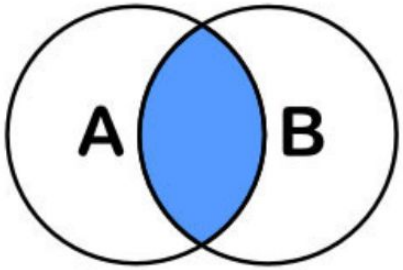
Suposem les següents taules:

Taula A

Id	Nom
1	Roberto
2	Juan
3	Ruben
4	Carlos

Taula B

Id	Nom
1	Alex
2	Carlos
3	Juan
4	Saúl



(INNER) JOIN: El resultat és els registres que coincideixen a les dues taules.

```
SELECT *
FROM TableA INNER JOIN TableB ON
TableA.name = TableB.name
```

Id	Nom	Id	Nom
2	Juan	2	Carlos
4	Carlos	3	Juan

Institut Sabadell

Exemples Tipus de JOIN

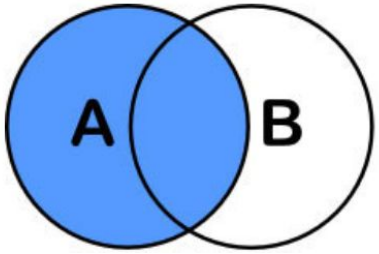
Suposem les següents taules:

Taula A

Id	Nom
1	Roberto
2	Juan
3	Ruben
4	Carlos

Taula B

Id	Nom
1	Alex
2	Carlos
3	Juan
4	Saúl



LEFT JOIN: El resultat és tot els registres de A i els que coincideixen amb B. Pot donar nulls.

```
SELECT *
FROM TableA LEFT OUTER JOIN TableB ON
TableA.name = TableB.name
```

Id	Nom	Id	Nom
1	Roberto	2	Carlos
2	Juan	3	Juan
3	Ruben	Null	null
4	Carlos	Null	null

Institut Sabadell

Exemples Tipus de JOIN

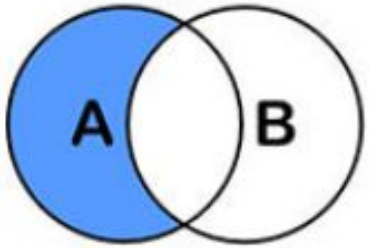
Suposem les següents taules:

Taula A

Id	Nom
1	Roberto
2	Juan
3	Ruben
4	Carlos

Taula B

Id	Nom
1	Alex
2	Carlos
3	Juan
4	Saúl



LEFT JOIN: El resultat és tot els registres de A que no apareixen a la B.

```
SELECT *
FROM TableA LEFT OUTER JOIN TableB ON
TableA.name = TableB.name
WHERE TableB.id IS null
```

Id	Nom	Id	Nom
1	Roberto	Null	null
4	Ruben	Null	null

Institut Sabadell

Exemples Tipus de JOIN

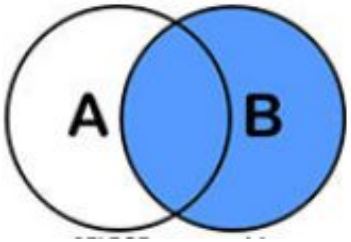
Suposem les següents taules:

Taula A

Id	Nom
1	Roberto
2	Juan
3	Ruben
4	Carlos

Taula B

Id	Nom
1	Alex
2	Carlos
3	Juan
4	Saúl



RIGHT JOIN: El resultat és tot els registres de B i els que coincideixen amb A. Pot donar nulls.

```
SELECT *
FROM TableA RIGHT OUTER JOIN TableB ON
TableA.name = TableB.name
```

Id	Nom	Id	Nom
2	Juan	1	Alex
4	Carlos	2	Carlos
Null	Null	3	Juan
Null	Null	4	Saúl

Institut Sabadell

Exemples Tipus de JOIN

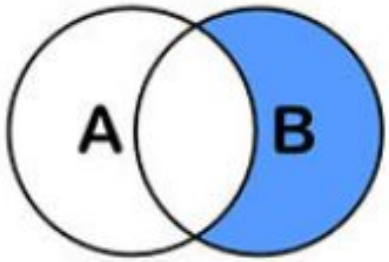
Suposem les següents taules:

Taula A

Id	Nom
1	Roberto
2	Juan
3	Ruben
4	Carlos

Taula B

Id	Nom
1	Alex
2	Carlos
3	Juan
4	Saúl



RIGHT JOIN: El resultat és tot els registres de A i els que coincideixen amb B. Pot donar nulls.

```
SELECT *
FROM TableA RIGHT OUTER JOIN TableB ON
TableA.name = TableB.name
WHERE TableB.id IS null
```

Id	Nom	Id	Nom
1	Alex	Null	null
4	Saúl	Null	null

Institut Sabadell

Group by

- **P**ermet agrupar les files resultat de les clàusules select, from i where segons una o més de les columnes seleccionades.

- La clàusula **having** permet especificar condicions de filtratge sobre els grups assolits per la clàusula group by.

Institut Sabadell

Group by

SELECT[distinct] <expressió/columna>,<expressió/columna>,...
FROM<taula>, <taula>,...
[**WHERE**<condició_de_recerca>]
[**GROUP BY**<àlies/columna>, <àlies/columna>,...]
[**HAVING**<condició_sobre_grups>]
[**ORDER BY**<expressió/columna> [asc|desc], <expressió/columna> [asc|desc],...];

Institut Sabadell

Funcions d'agrupament

Funció	Descripció	Exemples
AVG (n)	Retorna el valor mitjà de la columna n ignorant els valors nuls.	AVG (salari) retorna el salari mitjà de tots els empleats seleccionats que tenen salari (els nuls s'ignoren).
COUNT ([*|expr])	Retorna el nombre de vegades que expr avalua alguna dada amb valor no nul. L'opció * comptabilitza totes les files seleccionades.	COUNT (dept_no) (sobre la taula d'empleats) compta quants empleats estan assignats a algun departament.
MAX (expr)	Retorna el valor màxim de expr.	MAX (salari) retorna el salari més alt.
MIN (expr)	Retorna el valor mínim de expr.	MIN (salari) retorna el salari més baix.
STDDEV (expr)	Retorna la desviació típica de expr sense tenir en compte els valors nuls.	STDDEV (salari) retorna la desviació típica dels salaris.
SUM (expr)	Retorna la suma dels valors de expr sense tenir en compte els valors nuls.	SUM (salari) retorna la suma de tots els salaris.
VARIANCE (expr)	Retorna la variància de expr sense tenir en compte els valors nuls.	VARIANCE (salari) retorna la variància dels salaris.

Institut Sabadell

Exemples

Exemple 1. Utilització de la funció **count()** sobre tota la consulta.

En l'esquema empresa, es vol comptar quants empleats hi ha:

```
SELECT count(*) as "Quants empleats"
FROM emp;
```

Aquesta sentència SELECT és una sentència de selecció de conjunts malgrat que no aparegui la clàusula group by. En aquest cas, l'SGBD ha agrupat totes les files en un únic conjunt per tal de poder-les comptar.

Institut Sabadell

Exemples

Exemple 2. Utilització de l'opció **distinct** en una funció d'agrupament.

En l'esquema empresa, es vol comptar quants oficis diferents hi ha:

```
SELECT count (distinct ofici) as "Quants oficis"
FROM emp;
```

En aquest cas és necessari indicar l'opció distinct, ja que altrament comptaria totes les files que tenen algun valor en la columna ofici, sense descartar els valors repetits.

Institut Sabadell

Exemples

Exemple 3. Utilització de la funció **count**() sobre una consulta amb grups.

En l'esquema empresa, es vol mostrar quants empleats hi ha de cada ofici:

```
SELECT ofici as "Ofici", count (*) as "Quants empleats"
FROM emp
GROUP BY ofici;
```

Institut Sabadell

Exemples

El resultat obtingut és aquest:

```
Ofici     Quants empleats
--------- ---------------------
EMPLEAT   4
VENEDOR   4
ANALISTA  2
PRESIDENT      1
DIRECTOR  3
5 rows selected
```

Institut Sabadell

Exemples

Exemple 4. Coexistència de les clàusules group by i order by.

En l'esquema empresa, es volen mostrar els departaments que tenen empleats, acompanyats del salari més alt dels seus empleats i ordenats de manera ascendent pel salari màxim.

En aquest cas tenim vàries opcions.

Opció 1.

```
select dept_no, max (salari)
from emp
group by dept_no
order by max(salari);
```

Institut Sabadell

Exemples

Opció 2.

```
select dept_no as "Codi", max (salari) as
"Màxim salari"
from emp
group by dept_no
order by "Màxim salari";
```

Opció 3.

```
select dept_no as "Codi", max (all salari) as
"Màxim salari"
from emp
group by dept_no
order by "Màxim salari";
```

Institut Sabadell

Exemples

Exemple 5. Coexistència de les clàusules group by i order by

En l'esquema empresa, es vol comptar quants empleats de cada ofici hi ha en cada departament, i veure el resultats ordenats per departament de manera ascendent i per nombre d'empleats de manera descendent.

```
select dept_no as "Codi", ofici as "Ofici",
count (*) as "Quants empleats"
from emp
group by dept_no, ofici
order by dept_no, 3 desc;
```

Institut Sabadell

Exemples

Exemple 6. Coexistència de les clàusules group by i where.

En l'esquema empresa, es vol mostrar quants empleats de cada ofici hi ha en el departament 20:

```
select ofici as "Ofici", count (*) as "Quants empleats"
from emp
where dept_no=20
group by ofici;
```

Institut Sabadell

Exemples

Exemple 7. Utilització de la clàusula having

En l'esquema empresa, es vol mostrar el nombre d'empleats de cada ofici que hi ha per als oficis

que tenen més d'un empleat. La instrucció per assolir l'objectiu és aquesta:

```
select ofici as "Ofici", count (*) as "Quants empleats"
from emp
group by ofici
having count(*)>1;
```