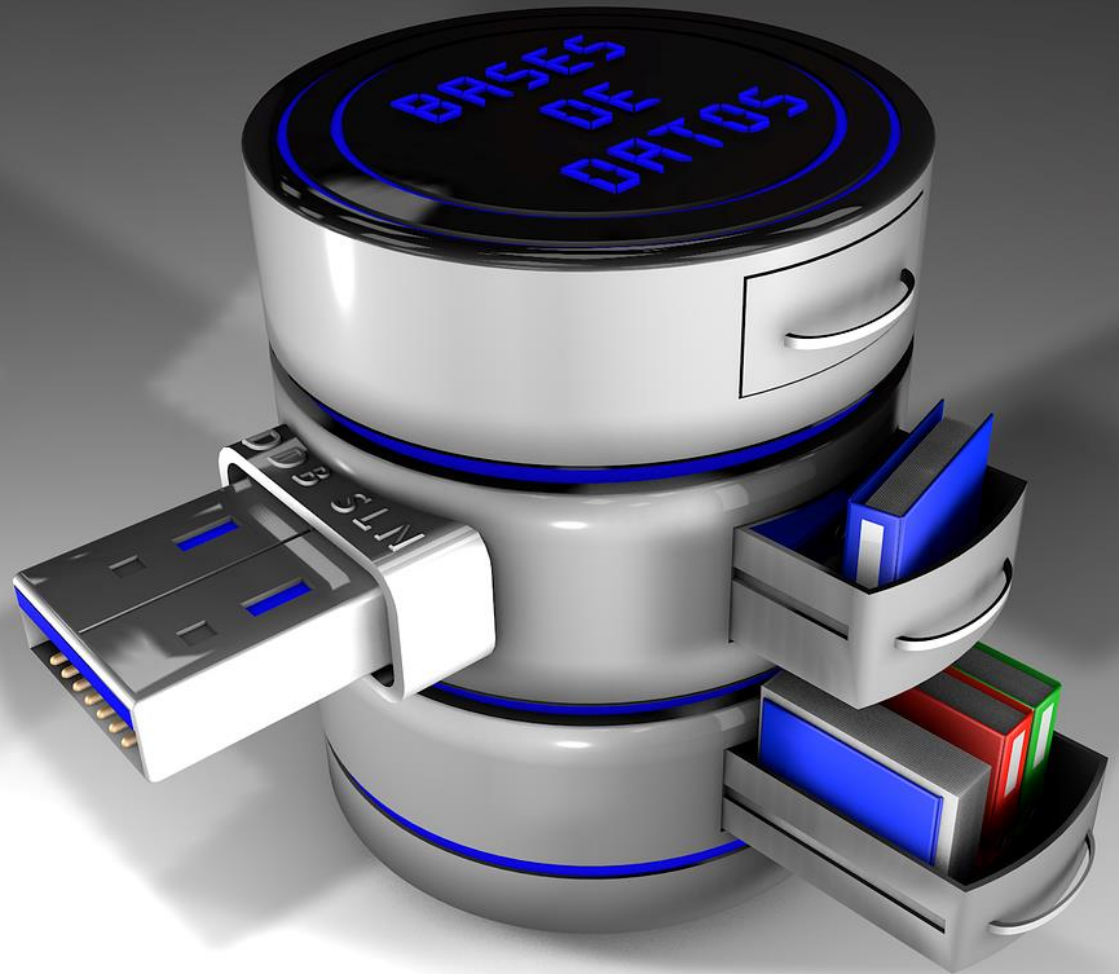


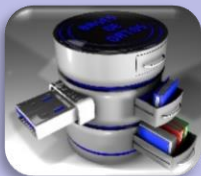
UNITAT DIDÀCTICA 9

SQL. Consultes sobre varies tables i agrupacions



Índex

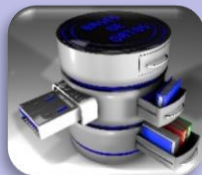
1. Funcions de grup
2. Agrupació de files
3. Anidació de funcions
4. Unions de tables. JOIN



Funcions de grup

¿Qué son les funcions de grup?


A diferència de les funcions d'una sola fila, les funcions de grup funcionen en jocs de files per a proporcionar un resultat per grup. Aquests jocs poden formar la taula completa o la taula dividida en grups.



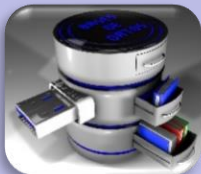
Funcions de grup

¿Qué son les funcions de grup?

SALARY	
24000	
17000	
17000	
9000	
6000	
4800	
4800	
4200	
12008	
9000	



Salario Máximo
24000

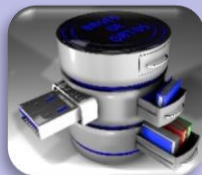


Funcions de grup

¿Qué son les funcions de grup?

Funcions de grup que anem a treballar:

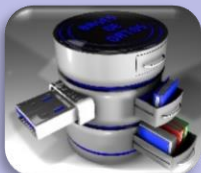
- ✓ AVG: Calcula la mitjana d'uns valors
- ✓ COUNT: Compta el nombre de files (no nules) que retorna la consulta
- ✓ MAX: Retorna el valor màxim
- ✓ MIN: Retorna el valor mínim
- ✓ SUM: Retorna la suma dels valors



Funcions de grup

Sintaxis

```
SELECT      group_function(column), ...  
FROM        table  
[WHERE      condition]  
[ORDER BY   column];
```

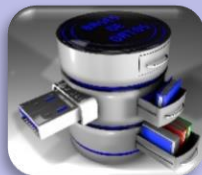


Funcions de grup

La funció de grup es col·loca després de la paraula clau SELECT, encara que no sempre. Pot ser que tinga diverses funcions de grup separades per comes.

Instruccions per a utilitzar les funcions de grup:

- DISTINCT fa que la funció considere només els valors no duplicats; ALL fa que considere cada valor, incloent els duplicats. El valor per defecte és ALL i, per tant, no és necessari especificar-ho.
- Totes les funcions de grup ignoren els valors nuls. Per a substituir un valor per a valors nuls, utilitze les funcions NVL, NVL2, COALESCE



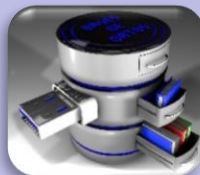
Funcions de grup

AVG – SUM – MIN – MAX

Pot utilitzar les funcions AVG, SUM, MIN i MAX en les columnes que poden emmagatzemar dades numèriques.

L'exemple mostra la suma dels salaris, la mitjana, el valor més alt i el més baix dels salaris mensuals de tots els empleats.

⚡ SUM(SALARY)	⚡ AVG(salary)	⚡ MAX(SALARY)	⚡ MIN(SALARY)
691416	6462	24000	2100



Funcions de grup

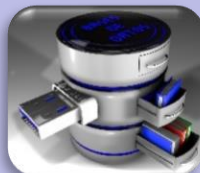
MIN – MAX

Pot utilitzar les funcions MAX i MIN per a tipus de dada numèrics, de caràcters i de data. L'exemple mostra els empleats més i menys experimentats.

MAX(HIRE_DATE)	MIN(HIRE_DATE)
21/04/08	13/01/01

El següent exemple mostra el cognom de l'empleat que està en primer lloc i del qual està en últim lloc en una llista de tots els empleats ordenada alfabèticament:

MIN(LAST_NAME)	MAX(LAST_NAME)
Abel	Zlotkey



Funcions de grup

COUNT

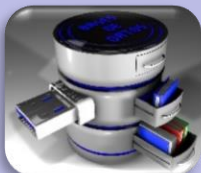
La funció COUNT té tres formats:

- ✓ COUNT(*)
- ✓ COUNT(expr)
- ✓ COUNT(DISTINCT expr)

COUNT(*) retorna el nombre de files en una taula que complisquen amb el criteri de la sentència SELECT, incloent les files duplicades i les files que continguin valors nuls en qualsevol de les columnes. Si s'inclou una clàusula WHERE en la sentència SELECT, COUNT(*) retorna el nombre de files que complisca amb la condició de la clàusula WHERE.

Per contra, COUNT(expr) retorna el nombre de valors no nuls que estan en la columna identificada amb expr.

COUNT(DISTINCT expr) retorna el nombre de valors únics no nuls que estan en la columna identificada amb expr.



Funcions de grup

COUNT

```
SELECT COUNT (*)  
FROM employees  
WHERE department_id=30;
```



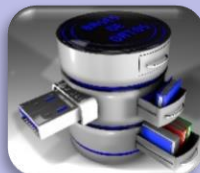
COUNT(*)
6

```
SELECT *  
FROM employees  
WHERE department_id=30;
```



	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL
1	114	Den	Raphaely	DRAP
2	115	Alexander	Khoo	AKHO
3	116	Shelli	Baida	SBAI
4	117	Sigal	Tobias	STOB
5	118	Guy	Himuro	GHIM
6	119	Karen	Colmenares	KCOL

COUNT(*) compta les files que retornaria SELECT *



Funcions de grup

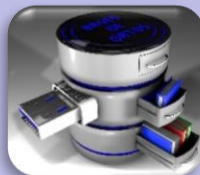
COUNT

```
SELECT COUNT(commission_pct)
FROM employees;
```



COUNT(COMMISSION_PCT)
35

COUNT(expr) compta els valors no nuls de *commission_pct*



Funcions de grup

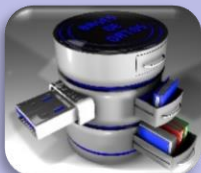
COUNT

```
SELECT COUNT(DISTINCT(department_id))  
FROM employees;
```



COUNT(DISTINCT(DEPARTMENT_ID))
11

COUNT(DISTINCT(expr)) compta els diferents departaments que hi ha a la tabla employees



Funcions de grup

Funcions de grup i valors nuls

Totes les funcions de grup ignoren els valors nuls de la columna.

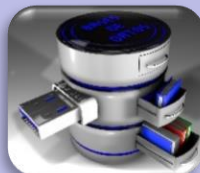
No obstant això, NVL força les funcions de grup perquè incloguen valors nuls.

Exemple 1:

La mitjana es calcula únicament sobre la base de les files de la taula en les quals s'emmagatzema un valor vàlid en la columna COMMISSION_PCT. La mitjana es calcula amb la comissió total pagada a tots els empleats dividida entre el nombre d'empleats que perceben una comissió

```
SELECT AVG(commission_pct)
FROM employees;
```

AVG(commission_pct)
0,2229



Funcions de grup

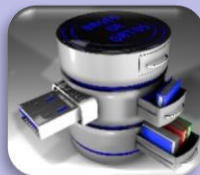
Funcions de grup i valors nuls

Exemple 2:

La mitjana es calcula sobre la base de totes les files de la taula, independentment de si els valors nuls s'emmagatzemen en la columna COMMISSION_PCT. La mitjana es calcula amb la comissió total pagada a tots els empleats dividida entre el nombre d'empleats de la companyia

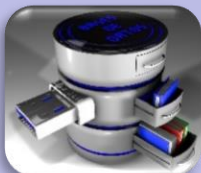
```
SELECT AVG(NVL(commission_pct,0))  
FROM employees;
```

AVG(NVL(commission_pct,0))
0,0729



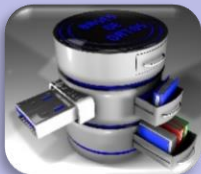
Agrupació de files

Fins a aquest punt, totes les funcions de grup han considerat la taula com un grup d'informació de gran grandària. No obstant això, a vegades és necessari dividir la taula d'informació en grups més xicotets. Per a això, cal utilitzar la clàusula GROUP BY.



Agrupació de files

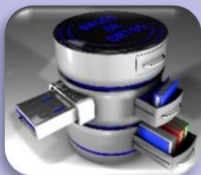
	DEPARTMENT_ID	SALARY			DEPARTMENT_ID	AVG(SALARY)
1	10	4400	4400		1	(null)
2	20	13000			2	20
3	20	6000	9500		3	90
4	50	2500			4	110
5	50	2600			5	50
6	50	3100	3500		6	80
7	50	3500			7	10
8	50	5800			8	60
9	60	9000	6400			
10	60	6000				
11	60	4200				
12	80	11000	10033			
13	80	8600				
...						
18	110	8300				
19	110	12000				
20	(null)	7000				



Agrupació de files

Pot utilitzar la clàusula GROUP BY per a dividir les files de la taula en grups

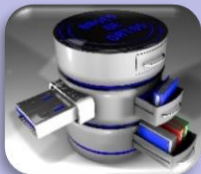
```
SELECT    column, group_function(column)
FROM      table
[WHERE    condition]
[GROUP BY group_by_expression]
[ORDER BY column];
```



Agrupació de files

Instruccions

- ✓ Si inclou una funció de grup en una clàusula SELECT, no pot seleccionar també resultats individuals llevat que la columna individual aparega en la clàusula GROUP BY. Rebrà un missatge d'error si no pot incloure la llista de columnes en la clàusula GROUP BY.
- ✓ En utilitzar la clàusula WHERE, pot excloure les files abans de dividir-les en grups.
- ✓ No pot utilitzar un àlies de columna en la clàusula GROUP BY.



Agrupació de files

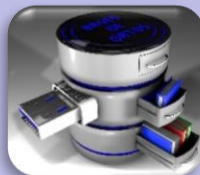
Exemples

- ✓ Mostrar la mitjana salarial de cada departament

```
SELECT department_id, AVG(salary)
FROM employees
GROUP BY department_id;
```

DEPARTMENT_ID	AVG(salary)
100	8601
30	4150
(null)	7000
90	19333
20	9500
70	10000
110	10154
50	3476

Tots els atributs de SELECT (department_id) están a GROUP BY. Salary está dins d'una funció i no compta com atribut



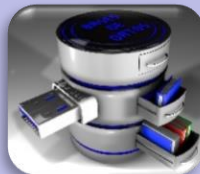
Agrupació de files

Exemples

No és necessari que la columna que apareix al GROUP BY estiga en la clàusula SELECT

```
SELECT SUM(salary)
FROM employees
GROUP BY department_id;
```

SUM(SALARY)
51608
24900
7000
58000
19000
10000
20308
156400



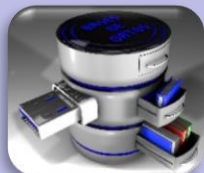
Agrupació de files

Exemples

A vegades necessitarà veure els resultats de grups dins de grups. La diapositiva mostra un informe que mostra el salari total pagat a cada lloc de cada departament. Agrupem per més d'una columna

```
SELECT department_id, job_id, SUM(salary)
FROM employees
GROUP BY department_id, job_id;
```

DEPARTMENT_ID	JOB_ID	SUM(SALARY)
110	AC ACCOUNT	8300
90	AD VP	34000
50	ST CLERK	55700
80	SA REP	243500
50	ST MAN	36400
80	SA MAN	61000
110	AC MGR	12008

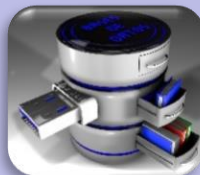


Agrupació de files

Exemples

```
SELECT department_id, job_id, SUM(salary)
FROM employees
WHERE department_id > 60
GROUP BY department_id, job_id;
```

DEPARTMENT_ID	JOB_ID	SUM(SALARY)
110	AC ACCOUNT	8300
90	AD VP	34000
80	SA REP	243500
80	SA MAN	61000
110	AC MGR	12008
90	AD PRES	24000
100	FI MGR	12008

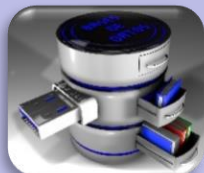


Agrupació de files

Exemples

```
SELECT TO_CHAR(hire_date, 'Month'), COUNT(*)  
FROM employees  
GROUP BY TO_CHAR(hire_date, 'Month');
```

TO_CHAR(HIRE_DATE,'MONTH')	COUNT(*)
Marzo	17
Enero	14
Mayo	6
Febrero	13
Agosto	9
Octubre	6
Noviembre	5
Junio	11
Abril	7
Septiembre	5
Diciembre	7
Julio	7



Agrupació de files

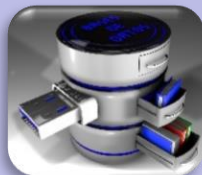
HAVING

No es pot utilitzar la clàusula WHERE per a restringir grups.

Utilitzarem la clàusula HAVING per a restringir grups de la mateixa forma que utilitza la clàusula WHERE per a restringir les files seleccionades.

Per tant:

- Condicions d'atributs → WHERE
- Condicions de grups → HAVING



Agrupació de files

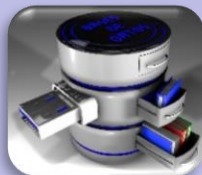
HAVING

No es pot utilitzar la clàusula WHERE per a restringir grups.

Utilitzarem la clàusula HAVING per a restringir grups de la mateixa forma que utilitza la clàusula WHERE per a restringir les files seleccionades.

Per tant:

- Condicions d'atributs → WHERE
- Condicions de grups → HAVING

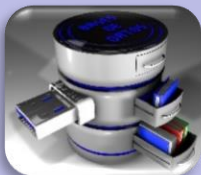


Agrupació de files

HAVING

Sintaxis

```
SELECT    column, group_function
FROM      table
[WHERE    condition]
[GROUP BY group_by_expression]
[HAVING   group_condition]
[ORDER BY column];
```



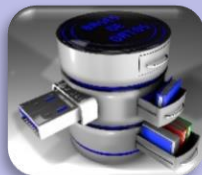
Agrupació de files

HAVING

- ✓ Mostrar el salari màxim per departament sempre que siga major a 10000

Per a buscar el salari màxim de cadascun dels departaments que tenen un salari màxim superior a 10.000, necessitarà realitzar les següents accions:

1. Buscar el salari mitjà de cada departament realitzant una agrupació per número de departament.
2. Restringir els grups als departaments amb un salari màxim superior a 10.000.



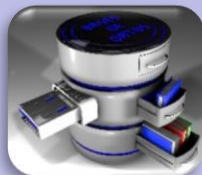
Agrupació de files

HAVING

- ✓ Mostrar el salari màxim per departament sempre que siga major a 10000

```
SELECT MAX(salary), department_id
FROM employees
GROUP BY department_id
HAVING MAX(salary) > 10000;
```

MAX(SALARY)	DEPARTMENT_ID
12008	100
11000	30
24000	90
13000	20
12008	110
14000	80

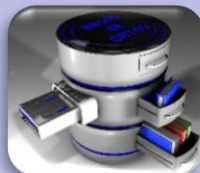


Agrupació de files

HAVING

```
SELECT job_id, SUM(salary)
FROM employees
WHERE job_id NOT LIKE '%REP%'
GROUP BY job_id
HAVING SUM(salary) > 13000
ORDER BY SUM(salary);
```

JOB_ID	SUM(SALARY)
PU_CLERK	13900
AD_PRES	24000
IT_PROG	28800
AD_VP	34000
ST_MAN	36400
FI_ACCOUNT	39600
ST_CLERK	55700
SA_MAN	61000
SH_CLERK	64300



Anidació de funcions

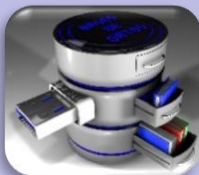
Les funcions de grup es poden anidar en una profunditat de dos. L'exemple calcula el salari mitjà per a cada department_id i, a continuació, mostra el salari màxim mitjà.

Tinga en compte que la clàusula GROUP BY és obligatòria en anidar funcions de grup.

Primer s'executa AVG i després es calcula el màxim. Mostra el salary mitjà més alt.

```
SELECT MAX(AVG(salary))  
FROM employees  
GROUP BY department_id;
```

MAX(AVG(salary))
19333



Unions de tables

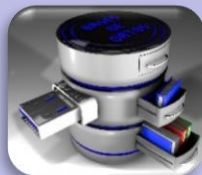
Una unió s'utilitza per a veure informació de diverses taules. Per tant, es poden unir taules per a veure informació de més d'una taula.

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID
166	Ande	Sundar	80
130	Atkinson	Mozhe	50
105	Austin	David	60
204	Baer	Hermann	70
116	Baida	Shelli	30
167	Banda	Amit	80

DEPARTMENTS

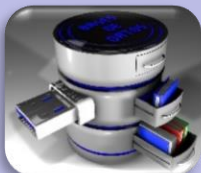
DEPARTMENT_ID	DEPARTMENT_NAME
30	Purchasing
40	Human Resources
50	Shipping
60	IT
70	Public Relations
80	Sales
90	Executive



Unions de tables. JOIN

UNIÓ TAULES EMPLOYEES I DEPARTMENTS

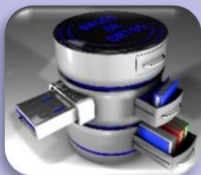
EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
166	Ande	Sundar	80	Sales
130	Atkinson	Mozhe	50	Shipping
105	Austin	David	60	IT
204	Baer	Hermann	70	Public Relations
116	Baida	Shelli	30	Purchasing
167	Banda	Amit	80	Sales



Unions de tables. JOIN

A vegades necessita utilitzar dades de més d'una taula. En l'exemple, l'informe mostra dades de dos taules independents:

- ✓ La taula EMPLOYEES conté els ID d'empleat.
- ✓ Les taules EMPLOYEES i DEPARTMENTS contenen els ID de departament.
- ✓ La taula DEPARTMENTS conté els noms de departament.



Unions de tables. JOIN

EMPLOYEES

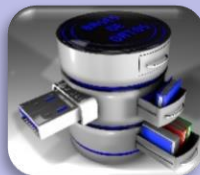
EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID
166	Ande	Sundar	80
130	Atkinson	Mozhe	50
105	Austin	David	60
204	Baer	Hermann	70
116	Baida	Shellli	30
167	Banda	Amit	80

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME
30	Purchasing
40	Human Resources
50	Shipping
60	IT
70	Public Relations
80	Sales
90	Executive

Unim les taules utilitzant l'atribut comú (department_id)

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
167	Banda	Amit	80	Sales

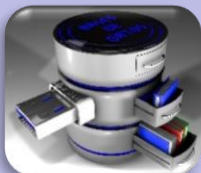


Unions de tables. JOIN

Sintaxis:

```
SELECT    table1.column, table2.column
FROM      table1
[NATURAL JOIN table2] |
[JOIN table2 USING (column_name)] |
[JOIN table2
  ON (table1.column_name = table2.column_name)] |
[LEFT|RIGHT|FULL OUTER JOIN table2
  ON (table1.column_name = table2.column_name)] |
[CROSS JOIN table2];
```

- *table1.column* indica la taula i la columna des de les quals es recuperen les dades
- *NATURAL JOIN* unió dues taules basant-se en el mateix nom de la columna
- *JOIN table2 USING column_name* realitza una unió igualitària basant-se en el nom de la columna

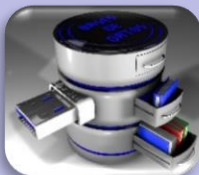


Unions de tables. JOIN

Sintaxis:

```
SELECT    table1.column, table2.column
FROM      table1
[NATURAL JOIN table2] |
[JOIN table2 USING (column_name)] |
[JOIN table2
  ON (table1.column_name = table2.column_name)] |
[LEFT|RIGHT|FULL OUTER JOIN table2
  ON (table1.column_name = table2.column_name)] |
[CROSS JOIN table2];
```

- *JOIN table2 ON table1.column_name = table2.column_name* performs realitza una unió igualitària basant-se en la condició de la clàusula ON
- *LEFT/RIGHT/FULL OUTER* s'utilitza per a realitzar unions OUTER
- *CROSS JOIN* retorna un producte cartesià de les dues taules



Unions de tables. JOIN

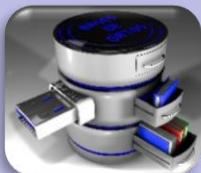
NATURAL JOIN

Permet unir taules automàticament basant-se en les columnes de les dos taules que tenen el mateix nom i els mateixos tipus de dada.

Si les columnes tenen el mateix nom però tipus de dada diferents, la sintaxi NATURAL JOIN produeix un error.

```
SELECT department_id, department_name, location_id, city
FROM departments NATURAL JOIN locations;
```

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID	CITY
60	IT	1400	Southlake
50	Shipping	1500	South San Francisco
10	Administration	1700	Seattle
30	Purchasing	1700	Seattle



Unions de tables. JOIN

NATURAL JOIN

```
SELECT department_id, department_name, location_id, city
FROM departments NATURAL JOIN locations;
```

DESCRIBE deptmens

Nombre	Nulo	Tipo
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)

DESCRIBE locations

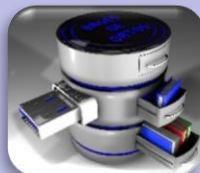
Nombre	Nulo	Tipo
LOCATION_ID	NOT NULL	NUMBER(4)
STREET_ADDRESS		VARCHAR2(40)
POSTAL_CODE		VARCHAR2(12)
CITY	NOT NULL	VARCHAR2(30)
STATE_PROVINCE		VARCHAR2(25)
COUNTRY_ID		CHAR(2)

Atributs amb el mateix nom?

Atributs amb la mateixa mena de dades?



LOCATION_ID (fem la unió per eixe atribut)



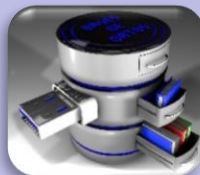
Unions de tables. JOIN

NATURAL JOIN

```
SELECT department_id, department_name, location_id, city
FROM departments NATURAL JOIN locations;
```

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID	LOCATION_ID	CITY
60	IT	103	1400	1300	Hiroshima
50	Shipping	121	1500	1400	Southlake
30	Purchasing	114	1700	1500	South San Francisco
10	Administration	200	1700	1600	South Brunswick
				1700	Seattle

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID	CITY
60	IT	1400	Southlake
50	Shipping	1500	South San Francisco
10	Administration	1700	Seattle
30	Purchasing	1700	Seattle



Unions de tables. JOIN

JOIN USING

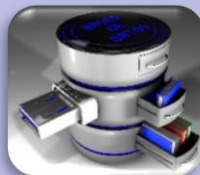
La clàusula USING es pot utilitzar per a especificar només les columnes que s'han d'utilitzar per a realitzar la unió.

Les taules employees i departments tenen dos atributs amb el mateix nom i amb la mateixa mena de dades:

- department_id
- manager_id

El primer s'utilitza per a saber que un empleat pertany a un departament, mentre que el segon l'usarem per a saber el nom, cognom, etc del manager d'un departament.

Per eixe motiu, no és recomanable utilitzar NATURAL JOIN entre aquestes dues taules, ja que farà la unió pels dos camps. Per tant, si volem saber quin és el nom de departament de cada empleat utilitzarem només el camp department_id i així ho especificarem en la clàusula USING



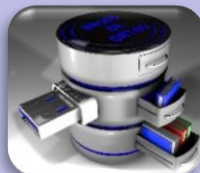
Unions de tables. JOIN

JOIN USING

La clàusula USING es pot utilitzar per a especificar només les columnes que s'han d'utilitzar per a realitzar la unió.

```
SELECT employee_id, last_name, first_name, department_id, department_name
FROM employees JOIN departments USING(department_id);
```

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
200	Whalen	Jennifer	10	Administration
201	Hartstein	Michael	20	Marketing
202	Fay	Pat	20	Marketing
114	Raphaely	Den	30	Purchasing
115	Khoo	Alexander	30	Purchasing



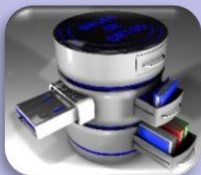
Unions de tables. JOIN

JOIN ON

Podem utilitzar la clàusula ON quan

- ✓ unim columnes que tenen noms diferents.
- ✓ fem unions no igualitaries
- ✓ Especifiquem condicions arbitràries

Segurament necessitem canviar de nom els atributs per a evitar les columnes ambigües.



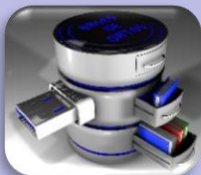
Unions de tables. JOIN

JOIN ON. Columnes ambigües

En unir dos o més taules, ha de qualificar els noms de les columnes amb el nom de la taula per a evitar ambigüitat.

Sense els prefixos de taula, la columna DEPARTMENT_ID de la llista SELECT pot provindre de la taula DEPARTMENTS o de la taula EMPLOYEES.

És necessari agregar el prefix de taula per a executar la consulta. Si no existeixen noms de columna iguals entre les dues taules, no és necessari qualificar les columnes. No obstant això, l'ús del prefix de taula millora el rendiment, ja que indica al servidor de Oracle on trobar exactament les columnes.



Unions de tables. JOIN

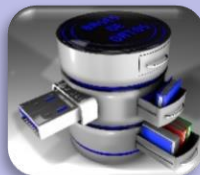
JOIN ON. Columnes ambigües

No obstant això, la qualificació de noms de columna amb noms de taula pot portar bastant temps, especialment si els noms de taula són llargs.

En el seu lloc, pot utilitzar àlies de taula. Igual que un àlies de columna proporciona un altre nom a una columna, un àlies de taula proporciona un altre nom a una taula.

Els àlies de taula ajuden a mantindre el codi SQL més xicotet i, per tant, menys ús de memòria.

El nom de taula s'especifica per complet, seguit d'un espai i de l'àlies de taula. Per exemple, a la taula EMPLOYEES se li pot proporcionar l'àlies *e* i, a la taula DEPARTMENTS l'àlies *d*.

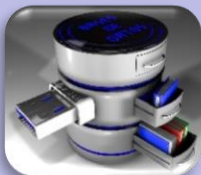


Unions de tables. JOIN

JOIN ON. Columnes ambigües

Instruccions

- ✓ Els àlies de taula poden tindre fins a 30 caràcters de longitud, però els àlies més curts són millors que els llargs.
- ✓ Si s'utilitza un àlies de taula per a un nom de taula determinat en la clàusula FROM, l'àlies de taula s'haurà de substituir pel nom de taula mitjançant la sentència SELECT.
- ✓ Els àlies de taula han de ser significatius.
- ✓ L'àlies de taula és vàlid només per a la sentència actual SELECT.



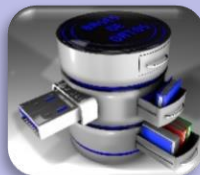
Unions de tables. JOIN

JOIN ON

```
SELECT e.employee_id, e.last_name, e.department_id,  
d.department_name, d.location_id  
FROM employees e JOIN departments d  
ON e.department_id=d.department_id;
```

àlies

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
200	Whalen	10	Administration	1700
201	Hartstein	20	Marketing	1800
202	Fay	20	Marketing	1800
114	Raphaely	30	Purchasing	1700
115	Khoo	30	Purchasing	1700



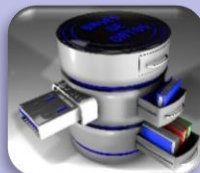
Unions de tables. JOIN

JOIN ON

```
SELECT e.employee_id, e.last_name, department_id,  
d.department_name, d.location_id  
FROM employees e JOIN departments d  
ON e.department_id=d.department_id;
```

```
ORA-00918: column ambiguously defined  
00918. 00000 - "column ambiguously defined"  
*Cause:  
*Action:  
Error en la línea: 1, columna: 36
```

la columna DEPARTMENT_ID de la llista SELECT pot provindre de la taula DEPARTMENTS o de la taula EMPLOYEES. → Ambigüitat

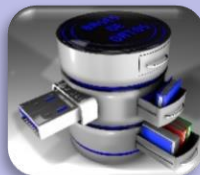


Unions de tables. JOIN

JOIN ON

```
SELECT e.employee_id, e.last_name, d.department_id,  
d.department_name, d.location_id, l.city  
FROM employees e JOIN departments d  
ON e.department_id=d.department_id  
JOIN locations l  
ON d.location_id=l.location_id;
```

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID	CITY
103	Hunold	60	IT	1400	Roma
107	Lorentz	60	IT	1400	Sao Paulo
106	Pataballa	60	IT	1400	Sao Paulo
105	Austin	60	IT	1400	Sao Paulo
104	Ernst	60	IT	1400	Sao Paulo
103	Hunold	60	IT	1400	Sao Paulo
107	Lorentz	60	IT	1400	Seattle

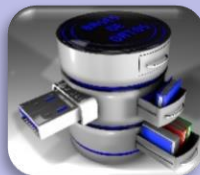


Unions de tables. JOIN

JOIN ON. Exemples

```
SELECT e.employee_id, e.last_name, d.department_id,  
d.department_name, d.location_id  
FROM employees e JOIN departments d  
ON e.department_id=d.department_id  
WHERE e.manager_id=149;
```

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
174	Abel	80	Sales	2500
175	Hutton	80	Sales	2500
179	Johnson	80	Sales	2500
177	Livingston	80	Sales	2500
176	Taylor	80	Sales	2500

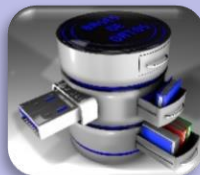


Unions de tables. JOIN

JOIN ON. Exemples

```
SELECT l.city, COUNT(*) total_dptos
FROM locations l JOIN departments d
ON l.location_id=d.location_id
GROUP BY l.city
ORDER BY 2 desc;
```

CITY	TOTAL_DPTOS
Seattle	21
Munich	1
Oxford	1
Toronto	1

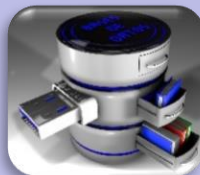


Unions de tables. JOIN

JOIN ON. Exemples

```
SELECT city, COUNT(*) total_dptos
FROM locations JOIN departments
USING(location_id)
GROUP BY city
ORDER BY 2 desc;
```

CITY	TOTAL_DPTOS
Seattle	21
Munich	1
Oxford	1
Toronto	1



Unions de tables. JOIN

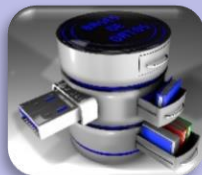
Autounió

Pot ser que a vegades necessite unir una taula amb si mateixa. Per a buscar el nom de cada gestor de l'empleat, necessita unir la taula EMPLOYEES amb si mateixa o realitzar una **autounió**. Per exemple, per a buscar el nom del gestor de Lorentz, necessita:

Buscar a Lorentz en la taula EMPLOYEES buscant en la columna LAST_NAME.

Busque el número de gestor de Lorentz consultant la columna MANAGER_ID. El número de gestor de Lorentz és 103.

Busque el nom del gestor amb un EMPLOYEE_ID de 103 consultant la columna LAST_NAME. El número d'empleat de Hunold és 103, per la qual cosa Hunold en el gestor de Lorentz.

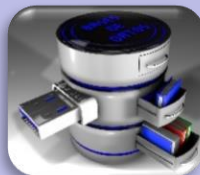
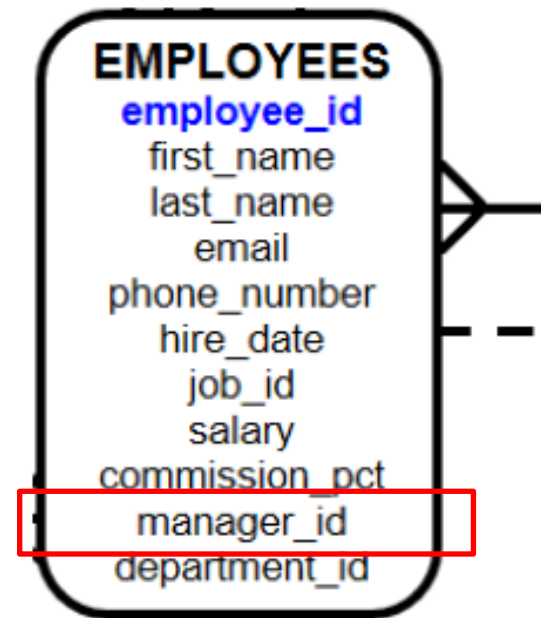


Unions de tables. JOIN

Autounió

Pot ser que a vegades necessite unir una taula amb si mateixa.

Manager_id és una clau aliena. Les claus alienes SEMPRE fan referència a una clau principal

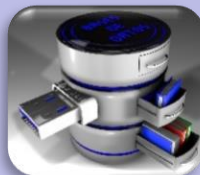


Unions de tables. JOIN

Autounió

Qui és el manager de David Austin?


EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MANAGER_ID
100	King	Steven	(null)
101	Kochhar	Neena	100
102	De Haan	Lex	100
103	Hunold	Alexander	102
104	Ernst	Bruce	103
105	Austin	David	103
106	Pataballa	Valli	103
107	Lorentz	Diana	103
108	Greenberg	Nancy	101
109	Faviet	Daniel	108



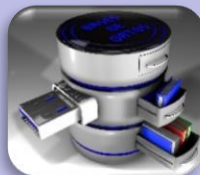
Unions de tables. JOIN

Autounió

MANAGER_id és fa referència a EMPLOYEE_ID



EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MANAGER_ID
100	King	Steven	(null)
101	Kochhar	Neena	100
102	De Haan	Lex	100
103	Hunold	Alexander	102
104	Ernst	Bruce	103
105	Austin	David	103
106	Pataballa	Valli	103
107	Torentz	Diana	103
108	Greenberg	Nancy	101
109	Faviet	Daniel	108



Unions de tables. JOIN

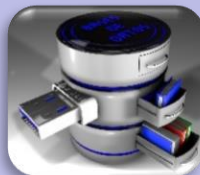
Autounió

Buscar el nom del gestor de Lorentz

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	MANAGER_ID
107	Lorentz	Diana	60	103

```
SELECT employee_id, last_name, first_name,  
department_id, manager_id  
FROM employees  
WHERE employee_id=103;
```

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	MANAGER_ID
103	Hunold	Alexander	60	102



Unions de tables. JOIN

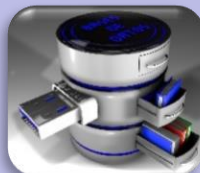
Autounió

EMPLOYEES (WORKER)

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MANAGER_ID
100	King	Steven	(null)
101	Kochhar	Neena	100
102	De Haan	Lex	100
103	Hunold	Alexander	102
104	Ernst	Bruce	103
105	Austin	David	103
106	Pataballa	Valli	103
107	Lorentz	Diana	103
108	Greenberg	Nancy	101
109	Faviet	Daniel	108

EMPLOYEES (MANAGER)

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MANAGER_ID
100	King	Steven	(null)
101	Kochhar	Neena	100
102	De Haan	Lex	100
103	Hunold	Alexander	102
104	Ernst	Bruce	103
105	Austin	David	103
106	Pataballa	Valli	103
107	Lorentz	Diana	103
108	Greenberg	Nancy	101
109	Faviet	Daniel	108

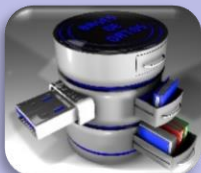


Unions de tables. JOIN

Autounió

```
SELECT worker.last_name "Cognom_EMP", worker.first_name "Nom_EMP",  
manager.last_name "Cognom_MAN", manager.first_name "Nom_MAN"  
FROM employees worker JOIN employees manager  
ON worker.manager_id=manager.employee_id;
```

⚡ Cognom_EMP	⚡ Nom_EMP	⚡ Cognom_MAN	⚡ Nom_MAN
De Haan	Lex	King	Steven
Cambrault	Gerald	King	Steven
Whalen	Jennifer	Kochhar	Neena
Mavris	Susan	Kochhar	Neena
Higgins	Shelley	Kochhar	Neena
Greenberg	Nancy	Kochhar	Neena
Baer	Hermann	Kochhar	Neena
Hunold	Alexander	De Haan	Lex
Pataballa	Valli	Hunold	Alexander
Lorentz	Diana	Hunold	Alexander

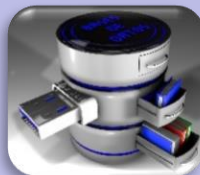


Unions de tables. JOIN

Autounió

Podem fer la mateixa consulta utilitzant àlies

```
SELECT w.last_name "Cognom_EMP", w.first_name "Nom_EMP",  
m.last_name "Cognom_MAN", m.first_name "Nom_MAN"  
FROM employees w JOIN employees m  
ON w.manager_id=m.employee_id;
```

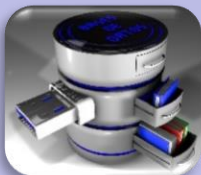


Unions de tables. JOIN

Unions NO igualitàries

Una unió no igualitària és una condició d'unió que conté algun operador diferent de l'operador d'igualtat.

Suposem que volem comparar els salaris entre els treballadors i volem mostrar aquells treballadors que tinguen un salari major que el d'un altre treballador. En aquest cas treballarem només amb la taula employees encara que tindrà dos rols, un el del treballador el salari del qual serà el de referència (en el nostre cas David Lee) i l'altre rol el de la resta de treballadors

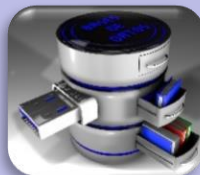


Unions de tables. JOIN

Unions NO igualitàries

```
SELECT e.last_name, e.first_name  
FROM employees e JOIN employees david  
ON e.salary > david.salary  
WHERE david.first_name LIKE 'David' AND david.last_name LIKE 'Lee';
```

LAST_NAME	FIRST_NAME
King	Steven
Kochhar	Neena
De Haan	Lex
Hunold	Alexander
Greenberg	Nancy



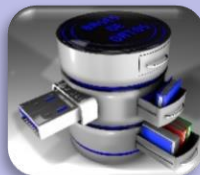
Unions de tables. JOIN

Unions OUTER

Si una fila no compleix una condició d'unió, la fila no apareix en el resultat de consultes.

Amb la clàusula **OUTER** s'aconsegueix mostrar aquelles files de dades que es queden "fora" de la unió de les taules.

Per exemple, suposem que volem mostrar tots els departaments (nom) i d'aquells que tinguen empleats, els noms dels treballadors. Per a això necessitem unir les taules. Es realitzarà la unió a través de l'atribut `department_id`. El problema sorgeix quan el departament no té empleats, ja que en fer la unió per eixe atribut, no trobarà coincidència en la taula `employees`, i els departaments sense empleats no apareixeran.



Unions de tables. JOIN

Unions OUTER

Aquestos són els diferents departaments que hi ha a la taula employees.

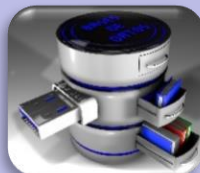
```
SELECT DISTINCT(department_id)
FROM employees
ORDER BY 1;
```

DEPARTMENT_ID
10
20
30
40
50
60
70
80
90
100
110
(null)

aquests departaments no
tenen correspondència
en la taula employees,
perquè no tenen
empleats

```
SELECT department_id, department_name
FROM departments;
```

DEPARTMENT_ID	DEPARTMENT_NAME
70	Public Relations
80	Sales
90	Executive
100	Finance
110	Accounting
120	Treasury
130	Corporate Tax
140	Control And Credit
150	Shareholder Services



Unions de tables. JOIN

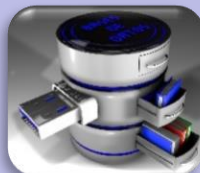
Unions OUTER

Aquests són els diferents departaments que hi ha a la taula employees.

```
SELECT employee_id, last_name, first_name,  
department_id, department_name  
FROM employees JOIN departments  
USING (department_id)  
ORDER BY 4 desc;
```

Els departaments, 120,
130, etc, no apareixen
perque no tenen
empleats

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
205	Higgins	Shelley	110	Accounting
206	Gietz	William	110	Accounting
109	Faviet	Daniel	100	Finance
111	Sciarra	Ismael	100	Finance

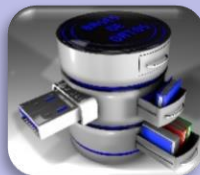


Unions de tables. JOIN

Unions OUTER

Com podem fer perquè apareguen tots els departaments?

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
(null)	(null)	(null)	150	Shareholder Services
(null)	(null)	(null)	140	Control And Credit
(null)	(null)	(null)	130	Corporate Tax
(null)	(null)	(null)	120	Treasury
206	Gietz	William	110	Accounting
205	Higgins	Shelley	110	Accounting
108	Greenberg	Nancy	100	Finance
109	Faviet	Daniel	100	Finance
111	Sciarra	Ismael	100	Finance

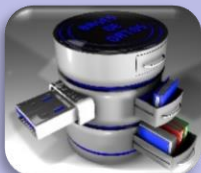


Unions de tables. JOIN

Unions OUTER

Utilitzant la clàusula OUTER en la unió, ja que permet que es mostren les files que es queden fora de la unió

```
SELECT employee_id, last_name, first_name,  
department_id, department_name  
FROM employees RIGHT OUTER JOIN departments  
USING (department_id)  
ORDER BY 4 desc;
```



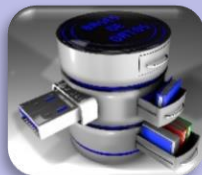
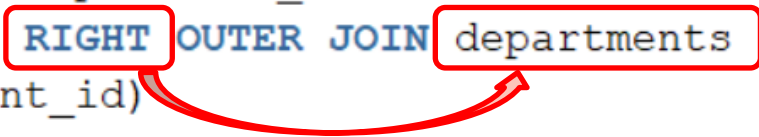
Unions de tables. JOIN

Unions OUTER

La clàusula RIGHT indica que la taula que està escrita a la dreta (RIGHT) de la clàusula JOIN mostrarà també les files de dades que estiguen fora (OUT) de la unió.

És a dir, la taula de la dreta departments en aquest cas, mostrarà les dades comunes i no comunes a la taula employees.

```
SELECT employee_id, last_name, first_name,  
department_id, department_name  
FROM employees RIGHT OUTER JOIN departments  
USING (department_id)  
ORDER BY 4 desc;
```



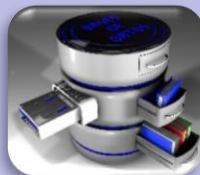
Unions de tables. JOIN

Unions OUTER

Si en la sentència escrivim LEFT OUTER JOIN, mostrarà de la taula que està situada a l'esquerra (employees) tots els empleats (tinguen o no departament)

```
SELECT employee_id, last_name, first_name,  
department_id, department_name  
FROM employees LEFT OUTER JOIN departments  
USING (department_id)  
ORDER BY 4 desc;
```

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
178	Grant	Kimberely	(null)	(null)
205	Higgins	Shelley	110	Accounting
206	Gietz	William	110	Accounting
111	Sciarra	Ismael	100	Finance



Unions de tables. JOIN

Unions OUTER

Si utilitzem FULL OUTER JOIN mostrarà totes les files de dades comunes i no comunes de totes dues taules.

```
SELECT employee_id, last_name, first_name,  
department_id, department_name  
FROM employees FULL OUTER JOIN departments  
USING (department_id)  
ORDER BY 4 desc;
```

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
178	Grant	Kimberely	(null)	(null)
(null)	(null)	(null)	270	Payroll
(null)	(null)	(null)	260	Recruiting
(null)	(null)	(null)	250	Retail Sales
206	Gietz	William	110	Accounting
205	Higgins	Shelley	110	Accounting
108	Greenberg	Nancy	100	Finance
109	Faviet	Daniel	100	Finance

Es mostren empleats amb i sense departament

I es mostren departaments amb i sense empleats

