

Unitat Didàctica 2: El llenguatge SQL

Part 1:

El llenguatge SQL: consultes i actualitzacions (DML)

(Doc. UD2.1)

Objectius

- L'alumne ha de ser capaç de descriure l'estructura bàsica i les clàusules de la instrucció del llenguatge SQL que ens permet fer consultes a una BD: sentència SELECT.
- Donat l'Esquema Lògic d'una BDR i coneixent la sentència SELECT del llenguatge de manipulació del SQL l'alumne ha de ser capaç de:
 - ✓ Formular en el llenguatge SQL consultes senzilles i complexes.
 - ✓ Formular en el llenguatge SQL operacions d'inserció, esborrament i modificació de la informació.

UD2.1 El llenguatge SQL: consultes i actualització (DML)

- ➔ 1 Introducció a SQL
- 2 Consultes senzilles sobre una taula
- 3 Consultes simples sobre varies taules
- 4 Consultes complexes: Subconsultes
- 5 Consultes complexes: Agrupació (GROUP BY)
- 6 Operadors conjuntistes
- 7 Operador JOIN
- 8 Llenguatge de Manipulació de Dades (DML)
- 9 Instrucció INSERT
- 10 Instrucció DELETE
- 11 Instrucció UPDATE

1 Introducció a SQL

Llenguatge SQL:

- El llenguatge SQL (**S**tructured **Q**uery **L**anguage - llenguatge de consulta estructurat) és un llenguatge d'accés a BDR.
- Permet, entre altres coses
 - crear i modificar esquemes de BD, i
 - especificar les operacions sobre BD.
- Junta característiques de l'àlgebra i el càlcul relacional.

1 Introducció a SQL

Subllenguatges de SQL

- **Llenguatge de Definició de Dades (DDL)**
per a crear i modificar esquemes de BD.
- **Llenguatge de Manipulació de Dades (DML)**
per a consulta i actualització de BD.

Consta de las instruccions:

- **SELECT** (consulta)
- **INSERT** (inserir tuples)
- **DELETE** (esborrar tuples)
- **UPDATE** (modificació de tuples)
- Altres instruccions (no s'estudiaran)
- Llenguatge de Control per a canviar dinàmicament propietats de la BD.

1 Introducció a SQL

Llenguatge de Manipulació de Dades (DML)


Es presenten les instruccions que es poden executar des d'un intèrpret de SQL, el que s'anomena *SQL interactiu*.

SQL és un llenguatge molt expressiu i, en general, permet moltes formes d'expressar les mateixes ordres.

Les instruccions que componen el DML són les següents:

- **SELECT:** permet la declaració de consultes per a la recuperació d'informació d'una o més taules d'una base de dades.
- **INSERT:** realitza la inserció d'una o varies files sobre una taula.
- **DELETE:** permet efectuar l'esborrat d'una o varies files d'una taula.
- **UPDATE:** realitza una modificació dels valors d'una o més columnes d'una o varies files d'una taula.

UD2.1 El llenguatge SQL: consultes i actualització (DML)

- 1 Introducció a SQL
-  2 Consultes senzilles sobre una taula.
- 3 Consultes simples sobre varies taules
- 4 Consultes complexes: Subconsultes
- 5 Consultes complexes: Agrupació (GROUP BY)
- 6 Operadors conjuntistes
- 7 Operador JOIN
- 8 Llenguatge de Manipulació de Dades (DML)
- 9 Instrucció INSERT
- 10 Instrucció DELETE
- 11 Instrucció UPDATE

2 SELECT: Consultes senzilles sobre una taula

informació a obtenir

SELECT [ALL | DISTINCT] {expressió₁, expressió₂,..., expressió_n | *}

FROM *taula*

d'on s'obté la informació buscada

[WHERE *condició*]

Condició que han de complir les files de la consulta resultant

[GROUP BY [HAVING *condició*]]

[ORDER BY {columna₁, columna₂,..., columna_n | *}]

ordena per una o varies columnes

2 SELECT: Consultes senzilles sobre una taula

Notació utilitzada per a definir la sintaxi

Cursiva: components de la base de dades

MAJÚSCULES: paraules reservades de SQL

Text normal: elements a definir més endavant

E_1, E_2, \dots, E_n : llista separada per comes de E_i , on $i > 0$

| : separador d'opcions alternatives

[] : contingut opcional

{ } : contingut obligatori

2 SELECT: Consultes senzilles sobre una taula

Ordre d'execució de les distintes clàusules d'una instrucció SELECT:

3 SELECT [ALL|DISTINCT]{expressió₁, expressió₂,..., expressió_n| *}

1 FROM *taula*

2 [WHERE condició]

4 [ORDER BY *columna*₁, *columna*₂,..., *columna*_n]

ALL : Permet l'aparició de files idèntiques (valor per defecte).

DISTINCT: No permet l'aparició de files idèntiques.

Esquema Relacional de la BD de Ciclisme

EQUIPO(**nomeq**: d_eq, director: d_nom)

CP: {nomeq}

CICLISTA(**dorsal**: d_dor, nombre: d_nom, edad: d_edad, nomeq: d_eq)

CP: {dorsal}

CA: {nomeq} → EQUIPO

VNN: {nomeq}

ETAPA(**netapa**: d_nº, km: d_km, salida: d_ciu, llegada: d_ciu, dorsal: d_dor)

CP: {netapa}

CA: {dorsal} → CICLISTA

PUERTO(**nompuerto**: d_nom, altura: d_alt, categoria: d_cat, pendiente: d_pen, netapa: d_nº, dorsal: d_dor)

CP: {nompuerto}

CA: {netapa} → ETAPA

CA: {dorsal} → CICLISTA

VNN: {netapa}

MAILLOT(**codigo**: d_cod, tipo: d_tipo, premio: d_pre, color: d_col)

CP: {codigo}

LLEVAR(dorsal: entero, **netapa**: d_nº, **codigo**: d_tipo)

CP: {netapa, codigo}

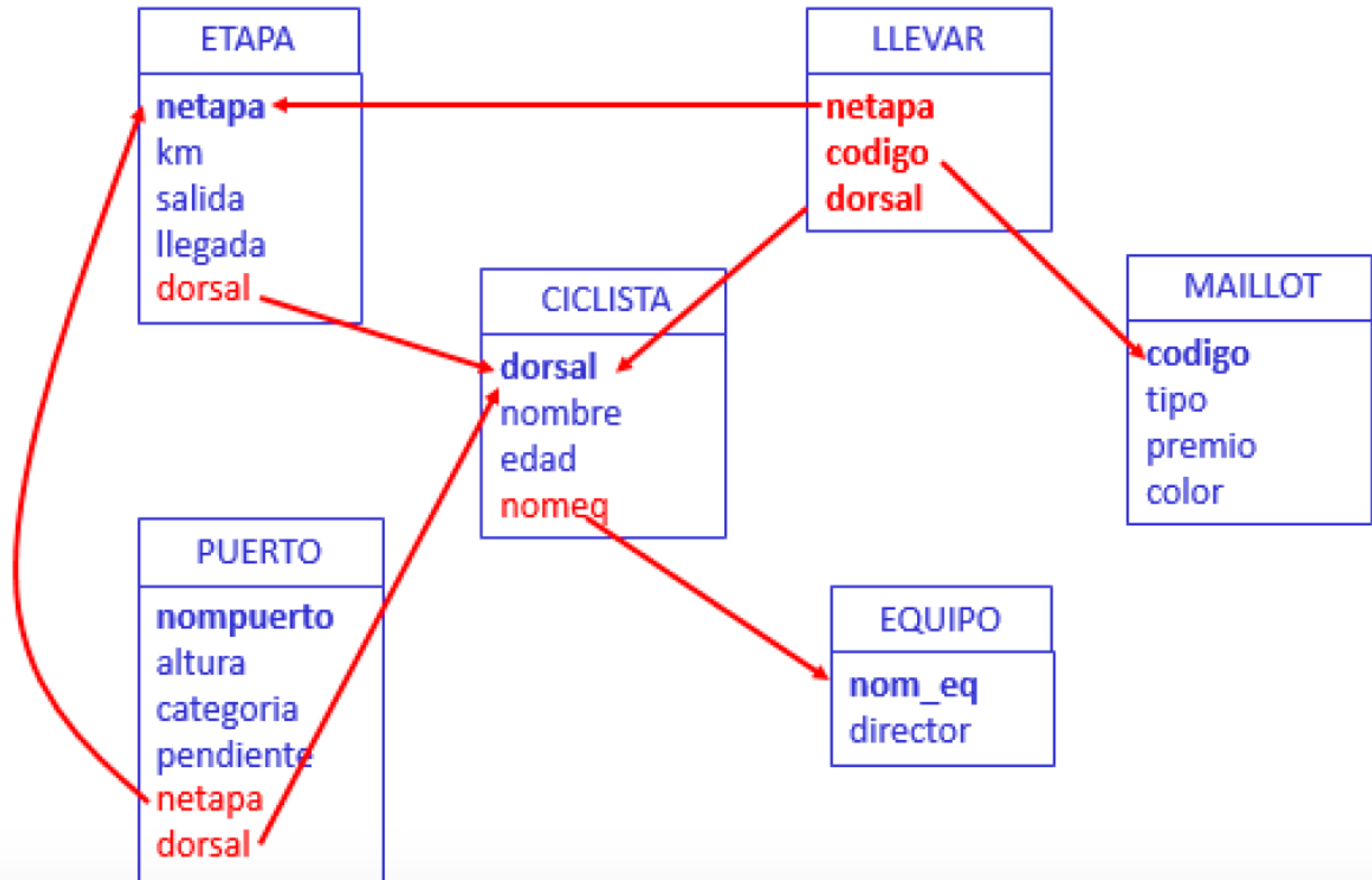
CA: {netapa} → ETAPA

CA: {dorsal} → CICLISTA

CA: {codigo} → MAILLOT

VNN: {dorsal}

Esquema de la BD de Ciclisme



2 SELECT: Consultes senzilles sobre una taula

EXAMPLE: Obtingueu el nom i l'edat de tots els ciclistes.

```
SELECT nombre, edad  
FROM Ciclista;
```

EXAMPLE: Obtingueu el nom i l'altura de tots els ports de 1^a categoria.

```
SELECT nompuerto, altura  
FROM Puerto  
WHERE categoria = '1';
```

2 SELECT: Consultes senzilles sobre una taula

EXAMPLE: Obtingueu sols les diferents edats dels ciclistes.

```
SELECT DISTINCT edad  
FROM Ciclista;
```

EXAMPLE: Obtingueu tota la informació dels equips.

```
SELECT *  
FROM Equipo;
```

2 SELECT: Consultes senzilles sobre una taula

EXEMPLE: Obtingueu el nom, l'altura i la categoria de tots els ports ordenats per altura i categoria.

```
SELECT nompuerto, altura, categoria  
FROM Puerto  
ORDER BY altura, categoria;
```

2 SELECT: Consultes senzilles sobre una taula

Expressió

FUNCIONS AGREGADES EN CONSULTES SENZILLES

```
SELECT 'Núm. de ciclistas =', COUNT(*),  
       'Media Edad =', AVG(edad)  
FROM Ciclista  
WHERE nomeq = 'Banesto';
```

En consultes senzilles (no agrupades), la selecció no podrà incloure al mateix temps referències a funcions agregades o constants i a atributs, ja que les funcions tornen un únic valor i els atributs poden tonar diversos valors.

EXEMPLE INCORRECTE:

```
SELECT nombre, AVG(edad)  
FROM Ciclista  
WHERE nomeq = 'ONCE';
```


2 SELECT: Consultes senzilles sobre una taula

Expressió

CONSULTES DE VALORS AGREGATS

La sintaxi d'ús d'una funció agregada és la següent:

{AVG|MAX|MIN|SUM|COUNT}([ALL|DISTINCT]expressió_escalar)
| COUNT(*)

2 SELECT: Consultes senzilles sobre una taula

Expressió

- Les funcions agregades no es poden niar en les consultes senzilles.
- Per a les funcions SUM i AVG els arguments hauran de ser numèrics.
- **DISTINCT** indica que els valors redundants són eliminats abans del càlcul corresponent.
- La funció especial **COUNT(*)**, en la qual no està permès incloure **DISTINCT** ni **ALL**, dona com a resultat el cardinal del conjunt de files de la selecció.
- Els càlculs es fan després de la selecció i d'aplicar les condicions.
- Els valors nuls són eliminats abans de realitzar els càlculs (inclòs COUNT).
- Si el nombre de files de la selecció es 0, la funció **COUNT** torna el valor 0 i les altres funcions el valor nul.

2 SELECT: Consultes senzilles sobre una taula

WHERE condició

```
3 SELECT[ALL|DISTINCT]{expresión1, expresión2,..., expresiónn| *}
1 FROM tabla
2 [WHERE condició]
4 [ORDER BY columna1, columna2,..., columnan]
```

La *condició* està formada per un conjunt de predicats combinats amb les connectives lògiques AND, OR i NOT.

Els predicats utilitzats que permeten comparar columnes són:

- predicats de **comparació**: =, <>, >, <, >=, <=.
- predicat LIKE: per a comparar una tira de caracters amb un patró.
- predicat BETWEEN: per a comprovar si un escalar està en un rang.
- predicat IN: per a comprovar si el valor està dintre d'un conjunt.
- predicat IS NULL: per a comprovar si el valor és nul.

2 SELECT: Consultes senzilles sobre una taula

WHERE condició

Predicats de comparació: =, <>, >, <, >=, <=.

EXAMPLE: Obtingueu el nom de tots els ciclistes de l'equip 'Banesto' que tinguen 27 anys.

```
SELECT nombre  
FROM Ciclista  
WHERE nomeq= 'Banesto' AND edad = 27;
```

2 SELECT: Consultes senzilles sobre una taula

WHERE condició

Predicats de comparació: =, <>, >, <, >=, <=.

- Cada costat d'un predicat de comparació ha de ser una única expressió del mateix tipus que la de l'altre costat.

E_1 predicat_comparació E_2

el tipus de dades de E_1 ha de ser igual al de E_2

2 SELECT: Consultes senzilles sobre una taula

WHERE condició

EXEMPLE : Obtingueu el nom dels ciclistes on l'edat no està entre 20 i 30 anys.

```
SELECT nombre  
FROM Ciclista  
WHERE edad NOT BETWEEN 20 AND 30;
```

(*) El predicat between és equivalent a una condició amb comparacions de la següent forma: $exp \text{ BETWEEN } exp1 \text{ AND } exp2 \equiv (exp \geq exp1) \text{ AND } (exp \leq exp2)$

2 SELECT: Consultes senzilles sobre una taula

WHERE condició

EXEMPLE : Obtingueu el número de les etapes on el nom de la ciutat d'arribada tinga per segona lletra una "O" o on el nom de la ciutat d'eixida porte dues o més 'A's.

```
SELECT netapa  
FROM Etapa  
WHERE llegada LIKE '_O%' OR salida LIKE '%A%A%';
```

2 SELECT: Consultes senzilles sobre una taula

WHERE condició

MÉS EXEMPLES DE COMPARACIONS

Ús de LIKE en casos especials (si la cadena a buscar conté un caràcter 'comodí')

EXEMPLE : Obtingueu el nom i l'edat dels ciclistes que pertanyen a equips on el nom continga la cadena "100%".

```
SELECT nombre, edad  
FROM Ciclista  
WHERE nomeq LIKE '%100\%%%' ESCAPE '\'
```


2 SELECT: Consultes senzilles sobre una taula

WHERE condició

EXEMPLE : Obtingueu el nom dels ports de 1^a, 2^a o 3^a categoria.

```
SELECT nompuerto  
FROM Puerto  
WHERE categoria IN ( '1', '2', '3' ) ;
```

(*) També el predicat **IN** és equivalent a:

exp **IN** (*exp1*, *exp2*, ..., *expn*) \equiv (*exp*=*exp1*) **OR** (*exp*=*exp2*) **OR**...**OR** (*exp*=*expn*)

EXEMPLE : Obtingueu totes les dades d'aquells ciclistes dels quals es coneix la seua edat.

```
SELECT *  
FROM Ciclista  
WHERE edad IS NOT NULL;
```

2 SELECT: Consultes senzilles sobre una taula

WHERE condició

COMPARACIÓ DE VALORS NULS

Les comparacions entre qualsevol valor i NULL s'avaluen a *indefinit*. EXEMPLE:

```
SELECT *  
FROM Taula  
WHERE atrib1 > atrib2
```

Si, p. ex., atrib1 = 50 i atrib2 fora nul, el resultat de la comparació seria indefinit; la fila no se seleccionaria.

Exemple de consulta incorrecta

```
SELECT nomeq  
FROM Equipo  
WHERE director = null
```

director IS NULL

2 SELECT: Consultes senzilles sobre una taula

WHERE condició


Operadors aritmètics: + (suma), – (diferència), * (producte), / (divisió), etc.

podem trobar-los en condicions i en expressions

EXEMPLE : Obtingueu dels mallots el tipus i el premi en dòlars (suposem que estiga en euros) (\$1 = 0,75 €) d'aquells mallots on el premi supere els 100 dòlars.


```
SELECT tipo, premio/0,75  
FROM Maillot  
WHERE premio/0,75 > 100;
```

UD2.1 El llenguatge SQL: consultes i actualització (DML)

- 1 Introducció a SQL
- 2 Consultes senzilles sobre una taula.
-  3 Consultes simples sobre varies taules
- 4 Consultes complexes: Subconsultes
- 5 Consultes complexes: Agrupació (GROUP BY)
- 6 Operadors conjuntistes
- 7 Operador JOIN
- 8 Llenguatge de Manipulació de Dades (DML)
- 9 Instrucció INSERT
- 10 Instrucció DELETE
- 11 Instrucció UPDATE

3 SELECT: Consultes senzilles sobre varies taules

SINTAXI

SELECT [ALL | DISTINCT] {expressió₁, expressió₂,..., expressió_n|*}
FROM *taula*₁, *taula*₂ ..., *taula*_n  d'on s'obté la informació buscada
[WHERE condició]
[ORDER BY *columna*₁, *columna*₂,..., *columna*_n]

3 SELECT: Consultes senzilles sobre varies taules

CLAUS ALIENES EN CONSULTES DE VARIAS TAULES

La consulta de varies taules correspon al **producte cartesià**.

¡Es important recordar-ho!

Si no es seleccionen bé les condicions, el nombre de files resultants pot ser molt gran.

El més freqüent és una igualtat entre la clau aliena i la clau de la taula a la qual es fa referència (encara que no sempre es així).

Consultes varies taules: producte cartesià

R

| <u>A</u> | B | C |
|----------|----|------|
| a1 | b1 | c1 |
| a2 | b2 | c2 |
| a3 | b3 | NULL |

×

S

| <u>C</u> | D | E |
|----------|----|------|
| c1 | b1 | c1 |
| c2 | b2 | c2 |
| c3 | b3 | NULL |

| R.A | R.B | R.C | S.C | S.D | S.E |
|-----|-----|------|-----|-----|------|
| a1 | b1 | c1 | c1 | b1 | c1 |
| a1 | b1 | c1 | c2 | b2 | c2 |
| a1 | b1 | c1 | c3 | b3 | NULL |
| a2 | b2 | c2 | c1 | b1 | c1 |
| a2 | b2 | c2 | c2 | b2 | c2 |
| a2 | b2 | c2 | c3 | b3 | NULL |
| a3 | b3 | NULL | c1 | b1 | c1 |
| a3 | b3 | NULL | c2 | b2 | c2 |
| a3 | b3 | NULL | c3 | b3 | NULL |

SELECT *
FROM R,S

Consultes varies taules: concatenació

R CA:{C} → S

| <u>A</u> | B | C |
|----------|----|------|
| a1 | b1 | c1 |
| a2 | b2 | c2 |
| a3 | b3 | NULL |

⊗_C

S

| <u>C</u> | D | E |
|----------|----|------|
| c1 | b1 | c1 |
| c2 | b2 | c2 |
| c3 | b3 | NULL |

| R.A | R.B | R.C | S.C | S.D | S.E |
|-----|-----|-----------|-----------|-----|------|
| a1 | b1 | c1 | c1 | b1 | c1 |
| a1 | b1 | c1 | c2 | b2 | c2 |
| a1 | b1 | c1 | c3 | b3 | NULL |
| a2 | b2 | c2 | c1 | b1 | c1 |
| a2 | b2 | c2 | c2 | b2 | c2 |
| a2 | b2 | c2 | c3 | b3 | NULL |
| a3 | b3 | NULL | c1 | b1 | c1 |
| a3 | b3 | NULL | c2 | b2 | c2 |
| a3 | b3 | NULL | c3 | b3 | NULL |

SELECT *
FROM R,S

×

WHERE
R.C=S.C

⊗_C

3 SELECT: Consultes senzilles sobre varies taules

Si la informació que es desitja obtenir està emmagatzemada en varies taules, la consulta ha d'incloure eixes taules en la clàusula FROM.

EXAMPLE: Obtingueu parells de números d'etapes i noms de ports guanyats pel mateix ciclista.

```
SELECT Etapa.netapa, nompuerto  
FROM Etapa, Puerto  
WHERE Etapa.dorsal = Puerto.dorsal;
```

La columna *dorsal* de *Etapa* i *Puerto* ha de qualificar-se amb el nom de la taula; si no, és ambigua.

Sintaxi: [taula | variable_recorregut].columna

3 SELECT: Consultes senzilles sobre varies taules

CLAUS ALIENES EN CONSULTES DE VARIAS TAULES

EXEMPLE: Obtingueu els noms dels ciclistes pertanyents a l'equip dirigit per 'Alvaro Pino'.

```
SELECT C.nombre  
FROM Ciclista C, Equipo E  
WHERE C.nomeq = E.nomeq AND E.director = 'Alvaro Pino';
```

¡És molt important concatenar pels atributs adequats!

Les variables de recorregut permeten donar un altre nom a una taula.

Sintaxis: **FROM** *tabla* [**AS**] *variable_recorregut*

3 SELECT: Consultes senzilles sobre varies taules

CLAUS ALIENES EN CONSULTES DE VARIAS TAULES

EXAMPLE: Obtingueu parells nom de ciclista, número d'etapa, de tal forma que el ciclista hi haja guanyat eixa etapa. A més l'etapa ha de superar els 150 km. de recorregut.

```
SELECT C.nombre, E.netapa  
FROM Ciclista C, Etapa E  
WHERE C.dorsal = E.dorsal AND E.km > 150;
```

3 SELECT: Consultes senzilles sobre varies taules

CLAUS ALIENES EN CONSULTES DE VARIAS TAULES

Obtenció de files repetides.


Al combinar varies taules, una mateixa fila d'una taula R pot aparèixer relacionada amb varies files d'altra taula S.

Si la consulta demana informació d'R, es poden obtenir files repetides, que en la majoria dels casos cal eliminar.

EXEMPLE: Número i longitud de les etapes amb ports

```
SELECT DISTINCT E.netapa, km  
FROM Etapa E, Puerto P  
WHERE E.netapa = P.netapa;
```

UD2.1 El llenguatge SQL: consultes i actualització (DML)

- 1 Introducció a SQL
- 2 Consultes senzilles sobre una taula.
- 3 Consultes simples sobre varies taules
-  4 Consultes complexes: Subconsultes
- 5 Consultes complexes: Agrupació (GROUP BY)
- 6 Operadors conjuntistes
- 7 Operador JOIN
- 8 Llenguatge de Manipulació de Dades (DML)
- 9 Instrucció INSERT
- 10 Instrucció DELETE
- 11 Instrucció UPDATE

4 SELECT. Consultes complexes: Subconsultes

¿Què és una subconsulta?

Una subconsulta és una consulta tancada entre parèntesi, que s'inclou dintre d'altra consulta.

4 SELECT. Consultes complexes: Subconsultes

Quan

informació buscada



una taula

condició de búsqueda



altres taules

EN ALGUNS CASOS es poden utilitzar les

subconsultes

4 SELECT. Consultes complexes: Subconsultes

EXEMPLE: Obtingueu número i longitud de les etapes que tenen ports de muntanya.

Utilitzant igualtats:

```
SELECT DISTINCT E.netapa, km  
FROM Etapa E, Puerto P  
WHERE E.netapa = P.netapa
```

Utilitzant subconsultes:

```
SELECT netapa, km  
FROM Etapa  
WHERE netapa IN (SELECT netapa  
FROM Puerto)
```

Consulta principal

Subconsulta que torna
números d'etapes

4 SELECT. Consultes complexes: Subconsultes

EXEMPLE: Obtingueu els noms dels ciclistes els quals pertanyen a l'equip dirigit per 'Álvaro Pino'

Utilitzant igualtats:

```
SELECT C.nombre  
FROM Ciclista C, Equipo E  
WHERE C.nomeq = E.nomeq AND E.director = 'Álvaro Pino';
```

Utilitzant subconsultes:

```
SELECT C.nombre FROM Ciclista C  
WHERE C.nomeq = (SELECT E.nomeq FROM Equipo E  
                  WHERE E.director = 'Alvaro Pino');
```

Açò és possible perquè la informació que es requereix, nom del ciclista, no està en la taula de la subconsulta (Equipo)

i perquè la subconsulta torna un únic valor

4 SELECT. Consultes complexes: Subconsultes

Les subconsultes poden aparèixer en les condicions de recerca, com arguments d'alguns predicats, tant de la clàusula WHERE com de la HAVING

PREDICATS que poden portar com a arguments subconsultes són:

- predicats de comparació (=, <>, >, <, >=, <=)
- predicats de comparació quantificats (ANY i ALL): permeten comparar un valor amb un conjunt de valors
- IN: comprova si un valor *pertany* a una col·lecció donada mitjançant una subconsulta
- EXISTS: equivalent al quantificador existencial, comprova si una subconsulta torna alguna fila

4 SELECT. Consultes complexes: Subconsultes

PREDICATS DE COMPARACIÓ (=, <>, >, <, >=, <=)

Cadascun dels dos costats d'un predicat de comparació ha de ser una única tupla formada pel mateix nombre de columnes:

(A1, A2, ..., An) predicat_comparació (B1, B2, ..., Bn)

Les subconsultes poden ser arguments, sempre que

- tornen una única fila i
- el nombre de columnes coincideixca en número i tipus amb l'altre costat del predicat de comparació

En cas que la subconsulta estiga buida, es converteix a una fila amb valors nuls en totes les columnes

4 SELECT. Consultes complexes: Subconsultes

EXEMPLE: Obtingueu els noms dels ports l'altura dels quals és major que la mitjana d'altura dels ports de 2a categoria .

1. En quines taules està la informació?

Puerto → FROM Puerto

2. Quina condició han de complir les files resultants?

altura > AVG(altura) dels Ports de segona categoria

→ WHERE altura > (SELECT AVG(altura) FROM Puerto
WHERE categoria = '2');

↗ És un valor - una fila

↙ Compara cada valor d'altura amb el valor obtingut en AVG(altura)

4 SELECT. Consultes complexes: Subconsultes

EXEMPLE: Obtingueu els noms dels ports l'altura dels quals és major que la mitjana d'altura dels ports de 2a categoria .

1. En quines taules està la informació?

Puerto → FROM Puerto

2. Quina condició han de complir les files resultants?

WHERE altura > (SELECT AVG(altura) FROM Puerto
WHERE categoria = '2');

3. Quina informació volem visualitzar?

nompuerto

4 SELECT. Consultes complexes: Subconsultes

EXEMPLE: Obtingueu els noms dels ports l'altura dels quals és major que la mitjana d'altura dels ports de 2a categoria .

```
SELECT nompuerto
FROM Puerto
WHERE altura > (SELECT AVG(altura)
                 FROM Puerto
                 WHERE categoría = '2');
```

INCORRECTE (error d'execució):

```
SELECT nompuerto
FROM Puerto
WHERE altura > AVG (SELECT altura
                       FROM Puerto
                       WHERE categoría = '2');
```

4 SELECT. Consultes complexes: Subconsultes

Què fa el següent exemple?
És correcte?

```
SELECT nompuerto  
FROM Puerto  
WHERE altura > (SELECT altura  
                 FROM Puerto  
                 WHERE categoria = '2' );
```

1 columna amb n filas

És un valor cada vegada

INCORRECTE (error d'execució):

**No es pot fer la comparació
si hi ha més d'un port de 2a categoria**

4 SELECT. Consultes complexes: Subconsultes

Predicado IN

Comprova si un valor *pertany* a una col·lecció donada mitjançant una subconsulta. A la dreta de IN pot aparèixer més d'una fila

constructor_fila [NOT] IN (expressió_taula)

EXAMPLE: Obtingueu el número de les etapes guanyades per ciclistes d'edat superior als 30 anys

```
SELECT netapa  
FROM Etapa  
WHERE dorsal IN (SELECT dorsal FROM Ciclista  
                  WHERE edad > 30);
```


4 SELECT. Consultes complexes: Subconsultes

SUBCONSULTAS ENCADENADAS

EXEMPLE: Obtingueu el número de les etapes guanyades per ciclistes que pertanyen a equips els directors dels quals tinguen un nom que comence per 'A'.

```
SELECT netapa FROM Etapa
WHERE dorsal IN
    (SELECT dorsal FROM Ciclista
    WHERE nomeq IN
        (SELECT nomeq FROM Equipo
        WHERE director LIKE 'A%')));
```

4 SELECT. Consultes complexes: Subconsultes

Predicat EXISTS

EXISTS (expressió_taula)

El predicat EXISTS s'avalua a cert si l'expressió SELECT torna al menys una fila

L'expressió: WHERE EXISTS (SELECT * FROM ...)

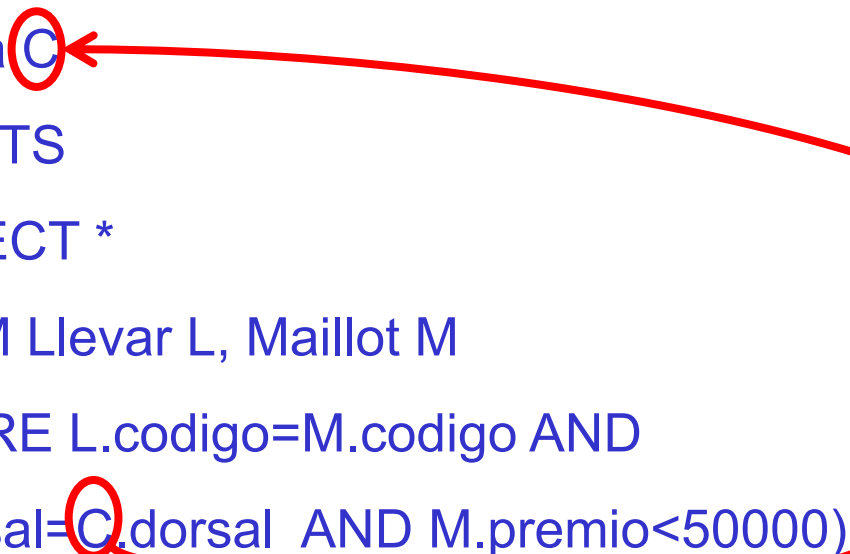
equivale a: WHERE 0 < (SELECT COUNT(*) FROM ...)

4 SELECT. Consultes complexes: Subconsultes

Predicado EXISTS

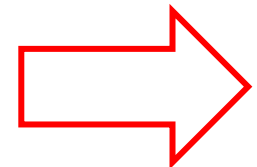
EXAMPLE: Obtingueu el nom d'aquells ciclistes que han dut un mallot d'un premi menor de 50000.

```
SELECT C.nombre  
FROM Ciclista C  
WHERE EXISTS  
    (SELECT *  
     FROM Llevar L, Maillot M  
     WHERE L.codigo=M.codigo AND  
           L.dorsal=C.dorsal AND M.premio<50000)
```



Hi ha una referència des de la subconsulta a la taula C externa

Altra solució amb IN



4 SELECT. Consultes complexes: Subconsultes

Predicado EXISTS

EXEMPLE: Obtingueu el nom d'aquells ciclistes que han dut un mallot d'un premi menor de 50000.

```
SELECT nombre  
FROM Ciclista C  
WHERE dorsal IN (SELECT L.dorsal  
                  FROM Maillot M, Llevar L  
                  WHERE M.codigo = L.codigo  
                        AND M.premio < 50000 );
```

En general, IN i EXISTS son intercanviables i es poden eliminar fent consultes a múltiples taules i igualant per claus alienes.

4 SELECT. Consultes complexes: Subconsultes

Predicado EXISTS

EXEMPLE: Obtingueu el nom d'aquells ciclistes que han dut un mallot d'un premi menor de 50000.

```
SELECT DISTINCT C.nombre  
FROM Ciclista C, Llevar L  
WHERE C.dorsal = L.dorsal AND  
      L.codigo IN (SELECT M.codigo  
                  FROM Maillot M  
                  WHERE M.premio < 50000 );
```

En general, IN i EXISTS son intercanviables i es poden eliminar fent consultes a múltiples taules i igualant per claus alienes.


4 SELECT. Consultes complexes: Subconsultes

EXEMPLE: Obtingueu el nom dels ciclistes que no han guanyat etapes.

```
SELECT nombre  
FROM Ciclista C  
WHERE NOT EXISTS (SELECT * FROM Etapa E  
                  WHERE E.dorsal = C.dorsal);
```

L'expressió: WHERE NOT EXISTS (SELECT * FROM ...)
equivale a: WHERE 0 = (SELECT COUNT(*) FROM ...)

UD2.1 El llenguatge SQL: consultes i actualització (DML)

- 1 Introducció a SQL
- 2 Consultes senzilles sobre una taula.
- 3 Consultes simples sobre varies taules
-  4 Consultes complexes: Subconsultes
- 5 Consultes complexes: Agrupació (GROUP BY)
- 6 Operadors conjuntistes
- 7 Operador JOIN
- 8 Llenguatge de Manipulació de Dades (DML)
- 9 Instrucció INSERT
- 10 Instrucció DELETE
- 11 Instrucció UPDATE

4 SELECT. Consultes complexes: Subconsultes

Predicado IN

Comprova si un valor *pertany* a una col·lecció donada mitjançant una subconsulta. A la dreta de IN pot aparèixer més d'una fila

constructor_fila [NOT] IN (expressió_taula)

4 SELECT. Consultes complexes: Subconsultes

Predicat EXISTS

EXISTS (expressió_taula)

El predicat EXISTS s'avalua a cert si l'expressió SELECT torna al menys una fila

L'expressió: WHERE EXISTS (SELECT * FROM ...)

equivale a: WHERE 0 < (SELECT COUNT(*) FROM ...)

4 SELECT. Consultes complexes: Subconsultes

Predicat NOT EXISTS (expressio_taula)

L'expressió: WHERE NOT EXISTS (SELECT * FROM ...)

equival a: WHERE 0 = (SELECT COUNT(*) FROM ...)

4 SELECT. Consultes complexes: Subconsultes

Avaluació dels predicats amb subconsultes buides

| PREDICAT | EVALUACIÓ |
|--|------------------|
| [expressió subconsulta] α [expressió subconsulta] | INDEFINIT |
| expressió IN (subconsulta) | FALS |
| EXISTS (subconsulta) | FALS |

On α és un operador de comparació: =, <>, >, <, >=, <=.

4 SELECT. Consultes complexes: Subconsultes

Ús de EXISTS per a quantificació universal (NO HI HA EN SQL 92)

$$\forall X F(X) \equiv \neg \exists X \neg F(X)$$

EXAMPLE: Obtingueu el nom del ciclista el qual ha guanyat totes les etapes de més de 200 km.

Per poder expressar aquesta consulta en SQL92 es converteix en:

“Obtingueu el nom del ciclista tal que no existisca una etapa de més de 200 km que ell no haja guanyat”

4 SELECT. Consultes complexes: Subconsultes


“Obtingueu el nom del ciclista tal que no existesca una etapa de més de 200 km que ell no l’haja guanyada” (que l’haja guanyada un altre)

```
SELECT nombre  
FROM Ciclista C  
WHERE NOT EXISTS  
    (SELECT *  
     FROM Etapa E  
     WHERE km > 200 AND C.dorsal <> E.dorsal );
```

4 SELECT. Consultes complexes: Subconsultes

I si no hi ha cap etapa de més de 200 km?

```
SELECT nombre  
FROM Ciclista C  
WHERE NOT EXISTS (SELECT *  
FROM Etapa E  
WHERE km > 200 AND  
C.dorsal <> E.dorsal );
```



CERT per tot
ciclista

FALS per tota
etapa

S'obtenen els noms de tots els ciclistes!!

4 SELECT. Consultes complexes: Subconsultes

EXEMPLE: Obtingueu el nom del ciclista que ha guanyat totes les etapes de més de 200 km

“Obtingueu el nom del ciclista tal que no existesca una etapa de més de 200 km que ell no l’haja guanyada” (que l’haja guanyada un altre)

```
SELECT nombre
FROM Ciclista C
WHERE NOT EXISTS (SELECT *
                  FROM Etapa E
                  WHERE km > 200 AND
                        C.dorsal <> E.dorsal )
AND EXISTS (SELECT *
            FROM Etapa E
            WHERE km > 200);
```

4 SELECT. Consultes complexes: Subconsultes

EXAMPLE: Obtingueu el nom dels equips tals que tots els seus corredors han portat algun mallot o han guanyat algun port.


$$\neg (A \vee B) = \neg (A) \wedge \neg (B)$$

```
SELECT e.nomeq
FROM equipo e
WHERE NOT EXISTS ( SELECT * FROM ciclista c
                    WHERE c.nomeq=e.nomeq
                    AND NOT EXISTS (SELECT * FROM llevar ll
                                     WHERE c.dorsal=ll.dorsal )
                    AND NOT EXISTS (SELECT * FROM puerto p
                                     WHERE p.dorsal=c.dorsal ))
AND EXISTS( SELECT * FROM ciclista c
            WHERE c.nomeq=e.nomeq);
```

```
SELECT e.nomeq
FROM equipo e
WHERE ( SELECT count(*)
        FROM ciclista c
        WHERE c.nomeq=e.nomeq )=
(SELECT count(distinct c.dorsal)
 FROM llevar ll, puerto p, ciclista c
 WHERE c.nomeq= e.nomeq AND
        (c.dorsal=ll.dorsal OR p.dorsal=c.dorsal) )
AND EXISTS( SELECT * FROM ciclista c
            WHERE c.nomeq=e.nomeq);
```

```
SELECT e.nomeq
FROM equipo e
WHERE ( SELECT count(*)
        FROM ciclista c
        WHERE c.nomeq=e.nomeq )=
(SELECT count(*)
 FROM ciclista c
 WHERE EXISTS ( SELECT * FROM llevar ll
                WHERE c.dorsal=ll.dorsal )
        OR
        EXISTS (SELECT * FROM puerto p
                WHERE p.dorsal=c.dorsal ) )
AND EXISTS( SELECT * FROM ciclista c
            WHERE c.nomeq=e.nomeq);
```

UD2.1 El llenguatge SQL: consultes i actualització (DML)

- 1 Introducció a SQL
- 2 Consultes senzilles sobre una taula.
- 3 Consultes simples sobre varies taules
- 4 Consultes complexes: Subconsultes
-  5 Consultes complexes: Agrupació (GROUP BY)
- 6 Operadors conjuntistes
- 7 Operador JOIN
- 8 Llenguatge de Manipulació de Dades (DML)
- 9 Instrucció INSERT
- 10 Instrucció DELETE
- 11 Instrucció UPDATE

5 SELECT. Consultes complexes: Agrupació

Un grup es pot entendre com un conjunt de files amb el mateix valor en les columnes per les quals s'agrupa (les incloses a la clàusula GROUP BY)

EXEMPLE: Obtingueu el nom de cada equip i l'edat mitjana dels seus ciclistes.

| nomeq | edad |
|---------|------|
| Banesto | 22 |
| ONCE | 25 |
| PDM | 32 |
| Banesto | 25 |
| Kelme | 28 |
| ONCE | 30 |
| Kelme | 29 |
| Banesto | 28 |

⇒

| nomeq | edad |
|---------|------|
| Banesto | 22 |
| Banesto | 25 |
| Banesto | 28 |
| ONCE | 25 |
| ONCE | 30 |
| PDM | 32 |
| Kelme | 28 |
| Kelme | 29 |

⇒

| nomeq | AVG(edad) |
|---------|-----------|
| Banesto | 25 |
| ONCE | 27,5 |
| PDM | 32 |
| Kelme | 28,5 |

5 SELECT. Consultes complexes: Agrupació

EXAMPLE: Obtingueu el nom de cada equip i l'edat mitjana dels seus ciclistes.

```
SELECT nomeq, AVG(edad)  
FROM Ciclista  
GROUP BY nomeq;
```

| Nomeq | Edad |
|---------|------|
| Banesto | 22 |
| ONCE | 25 |
| PDM | 32 |
| Banesto | 25 |
| Kelme | 28 |
| ONCE | 30 |
| Kelme | 29 |
| Banesto | 28 |

5 SELECT. Consultes complexes: Agrupació

Les funcions agregades en les consultes agrupades funcionen de forma diferent que en les consultes normals, tornen un únic valor per cada grup format

| Nomeq | Edad |
|---------|------|
| Banesto | 22 |
| Banesto | 25 |
| Banesto | 28 |
| ONCE | 25 |
| ONCE | 30 |
| PDM | 32 |
| Kelme | 29 |
| Kelme | 28 |

Una fila
per Grup

5 SELECT. Consultes complexes: Agrupació

Aleshores, per a

```
SELECT nomeq, AVG(edad)  
FROM Ciclista  
GROUP BY nomeq;
```

El resultat és:

| Nomeq | Edad |
|----------------|-------------|
| Banesto | 25 |
| ONCE | 27,5 |
| PDM | 32 |
| Kelme | 28,5 |

5 SELECT. Consultes complexes: Agrupació

EXEMPLE INCORRECTE

```
SELECT nomeq, nombre, AVG(edad)
FROM Ciclista
GROUP BY nomeq;
```

La regla sintàctica que apliquen els sistemes relacionals per assegurar el correcte funcionament de les consultes agrupades és la següent: **“en la selecció d’una consulta agrupada, sols poden aparèixer**

- referències a columnes per les quals s’agrupa,
- referències a funcions agregades o
- constants”

5 SELECT. Consultes complexes: Agrupació

GROUP BY i WHERE

Si s'inclou la clàusula WHERE, la seua aplicació és prèvia a l'agrupament

```
4      SELECT nomeq, AVG(edad)
1      FROM Ciclista
2      WHERE edad > 25
3      GROUP BY nomeq;
```

5 SELECT. Consultes complexes: Agrupació

GROUP i WHERE

En les consultes agrupades es poden niar les funcions agregades

```
SELECT MAX(AVG(edad)) FROM Ciclista  
GROUP BY nomeq
```

5 SELECT. Consultes complexes: Agrupació

GROUP BY, WHERE i HAVING

La clàusula HAVING sols pot anar en consultes agrupades i és similar a WHERE, però en un ordre diferent:

1r) Condició WHERE (s'usa per a les files)

2n) Agrupament i càlcul de valors agregats

3r) Condició HAVING (s'usa per als grups)

En la clàusula HAVING sols poden aparèixer directament referències a columnes per les quals s'agrupa, funcions agregades i constants

```
SELECT nomeq, avg(edad) FROM ciclista  
GROUP BY nomeq  
HAVING avg(edad)>30 and nomeq like 'B%'
```

5 SELECT. Consultes complexes: Agrupació

EXEMPLE: Obtingueu el nom de cada equip i l'edat mitjana dels seus ciclistes amb més de 25 anys, d'aquells equips amb més de 3 corredors majors de 25 anys

```
SELECT nomeq, AVG(edad)
```

```
FROM Ciclista C
```

```
WHERE edad > 25
```

```
GROUP BY nomeq
```


```
HAVING COUNT(dorsal) > 3;
```

5 SELECT. Consultes complexes: Agrupació

EXAMPLE: Obtingueu el nom del ciclista i el nombre de ports que ha guanyat, sent la mitjana de la pendent d'aquests superior a 10

```
SELECT C.nombre, COUNT(P.nompuerto)
FROM Ciclista C, Puerto P
WHERE C.dorsal = P.dorsal
GROUP BY C.dorsal, C.nombre      /* Agrupar per CP */
HAVING AVG (P.pendiente) >10;
```

UD2.1 El llenguatge SQL: consultes i actualització (DML)

- 1 Introducció a SQL
- 2 Consultes senzilles sobre una taula.
- 3 Consultes simples sobre varies taules
- 4 Consultes complexes: Subconsultes
- 5 Consultes complexes: Agrupació (GROUP BY)
-  6 Operadors conjuntistes
- 7 Operador JOIN
- 8 Llenguatge de Manipulació de Dades (DML)
- 9 Instrucció INSERT
- 10 Instrucció DELETE
- 11 Instrucció UPDATE

6 SELECT. Operadors conjuntistes

Existeixen diverses formes de combinar dues taules en SQL:

- Incloure diverses taules en la clàusula FROM
- Ús de subconsultes en les condicions de les clàusules WHERE o HAVING
- Combinacions conjuntistes de taules: utilitzant operadors de la teoria de conjunts per a combinar les taules
- Concatenacions de taules: utilitzant diferents variants de l'operador concatenació de l'Àlgebra Relacional

6 SELECT. Operadors conjuntistes

Corresponen als operadors unió, diferència i intersecció de l'Àlgebra Relacional

UNION
EXCEPT
INTERSECT

Permeten combinar taules que tinguen esquemes compatibles

6 SELECT. Operadors conjuntistes

UNION

expressió_taula UNION [ALL] terme_taula

- Realitza la unió de les files de les taules que provenen de les dues expressions
- Es permetran duplicats o no en funció de la utilització ALL

6 SELECT. Operadors conjuntistes

EXAMPLE: Obtingueu el nom dels ciclistes que han dut un mallot o han guanyat un port o una etapa.

```
SELECT nombre FROM Ciclista
WHERE dorsal IN
    (SELECT dorsal FROM Llevar
     UNION
     SELECT dorsal FROM Puerto
     UNION
     SELECT dorsal FROM Etapa)
```

6 SELECT. Operadors conjuntistes

EXAMPLE: Obtingueu el nom de tot el personal de la volta

```
(SELECT nombre FROM Ciclista)  
UNION  
(SELECT director FROM Equipo)
```

6 SELECT. Operadors conjuntistes

EXAMPLE: Obtingueu el nom de tots els equips indicant-hi quants ciclistes en té cadascun.

```
(SELECT nomeq, count(*)  
FROM Ciclista  
GROUP BY nomeq)
```

UNION

```
(SELECT nomeq, 0  
FROM Equipo  
WHERE nomeq NOT IN (SELECT nomeq  
FROM Ciclista));
```

6 SELECT. Operadors conjuntistes

expressió_taula INTERSECT terme_taula

Realitza la intersecció de les files de les taules que provenen de les dues expressions

EXAMPLE: Obtingueu els noms de les persones que són tant ciclistes com directors d'equip

```
(SELECT nombre FROM Ciclista)
INTERSECT
(SELECT director FROM Equipo)
```

6 SELECT. Operadors conjuntistes

expressió_taula EXCEPT terme_taula

(MINUS en Oracle10)

Realitzeu la diferència de les files de les taules provinents de les dues expressions


EXAMPLE: Obtingueu els noms que apareixen en la taula de ciclistes i no en la de directors

(SELECT nombre FROM Ciclista)

MINUS

(SELECT director FROM Equipo)

UD2.1 El llenguatge SQL: consultes i actualització (DML)

- 1 Introducció a SQL
- 2 Consultes senzilles sobre una taula.
- 3 Consultes simples sobre varies taules
- 4 Consultes complexes: Subconsultes
- 5 Consultes complexes: Agrupació (GROUP BY)
- 6 Operadors conjuntistes
-  7 Operador JOIN
- 8 Llenguatge de Manipulació de Dades (DML)
- 9 Instrucció INSERT
- 10 Instrucció DELETE
- 11 Instrucció UPDATE

7 SELECT. Operador JOIN

Existeixen varies formes de combinar dues taules en SQL:

- Incloure varies taules en la clàusula FROM
- Ús de subconsultes en les condicions de les clàusules WHERE o HAVING
- Combinacions conjuntistes de taules: utilitzant operadors de la teoria de conjunts per a combinar les taules
- Concatenacions de taules: utilitzant diferents variants de l'operador concatenació de l'Àlgebra Relacional

7 SELECT. Operador JOIN

CONCATENACIONS DE TAULES

Corresponen a variants de l'operador concatenació de l'Àlgebra Relacional.

- **Producte cartesià**
- **Concatenació interna**
- **Concatenació externa**

7 SELECT. Operador JOIN

Producte cartesià (CROSS JOIN)

Concatenació interna

referencia_taula [NATURAL] [INNER] JOIN referencia_taula
[ON condició | USING (*columna₁*, *columna₂*,..., *columna_n*)]

Concatenació externa

referencia_taula [NATURAL]
{LEFT | RIGHT | FULL} [OUTER] JOIN referencia_taula
[ON condició | USING (*columna₁*, *columna₂*,..., *columna_n*)]

7 SELECT. Operador JOIN

PRODUCTE CARTESIÀ

(CROSS JOIN)

```
SELECT *  
FROM referència_taula1 CROSS JOIN referència_taula2
```



```
SELECT *  
FROM referència_taula1, referència_taula2
```

7 SELECT. Operador JOIN

CONCATENACIÓ INTERNA

referencia_taula [NATURAL] [INNER] JOIN referencia_taula
[ON *expressió_condicional* | USING (*comalista_columna*)]

```
SELECT *  
FROM taula1 JOIN taula2 ON  
expressió_condicional
```

≡

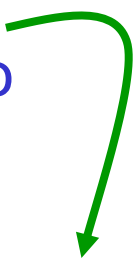
```
SELECT *  
FROM taula1, taula2  
WHERE expressió_condicional
```

7 SELECT. Operador JOIN

EXAMPLE: Obtingueu els noms dels ciclistes que pertanyen a l'equip dirigit per 'Alvaro Pino'.

```
SELECT nombre  
FROM Ciclista C, Equipo E  
WHERE C.nomeq = E.nomeq AND director = 'Alvaro Pino';
```

```
SELECT nombre  
FROM Ciclista NATURAL JOIN Equipo  
WHERE director = 'Alvaro Pino';
```



referencia_taula [NATURAL] [INNER] JOIN referencia_taula

7 SELECT. Operador JOIN

EXAMPLE: Obtingueu els noms dels ports, el nombre de l'etapa en la qual estan i la longitud de l'etapa, per als ports d'altura superior a 800.

```
SELECT nompuerto, netapa, km  
FROM Puerto JOIN Etapa USING (netapa)  
WHERE altura>800
```

SERIA INCORRECTE:

```
SELECT nompuerto, netapa, km  
FROM Puerto NATURAL JOIN Etapa  
WHERE P.altura>800 (concatenaria també per dorsal)
```

```
referència_tabua [natural] [INNER] JOIN referència_taula  
USING (columna1, columna2,..., columnan)
```

7 SELECT. Operador JOIN

EXAMPLE: Obtingueu els noms dels ports i el número de l'etapa en la qual es troben, si l'etapa anterior té més de 200 km.

```
SELECT P.nompuerto, P.netapa  
FROM Puerto P JOIN Etapa E ON P.netapa= E.netapa+1  
WHERE E.km>200
```

**referència_taula [natural] [INNER] JOIN referencia_taula
ON condició**

7 SELECT. Operador JOIN

CONCATENACIÓN EXTERNA

referència_taula [natural]

{LEFT | RIGHT | FULL} [OUTER] JOIN referència_taula
[ON condició| USING (*columna₁*, *columna₂*,..., *columna_n*)]

taula1 LEFT JOIN *taula2* ON *expressió_condicional*

(Concatenació interna de *taula1* i *taula2*)

UNION

(tuples de la *taula1* que no estan en la concatenació interna
amb valors nuls en la resta de columnes)

FULL, es mostren les tuples no concatenades de *taula1* i *taula2*

7 SELECT. Operador JOIN

CONCATENACIÓ EXTERNA

EXEMPLE: Obtingueu per a cada ciclista, el seu dorsal, el seu nom, el codi de cada mallot que ha dut i el número d'etapa en la qual l'ha dut.

```
SELECT C.dorsal, nombre, codigo, netapa  
FROM Ciclista C LEFT JOIN Llevar L ON C.dorsal = L.dorsal
```

referència_taula [natural]

**{LEFT | RIGHT | FULL} [OUTER] JOIN referencia_tabla
ON condició**

7 SELECT. Operador JOIN

EXAMPLE: Obtingueu el nom de tots els ciclistes i la quantitat d'etapes que ha guanyat cadascun.

```
SELECT nombre, COUNT(etapa)  
FROM Ciclista NATURAL LEFT JOIN Etapa  
GROUP BY dorsal, nombre
```

UD2.1 El llenguatge SQL: consultes i actualització (DML)

- 1 Introducció a SQL
- 2 Consultes senzilles sobre una taula.
- 3 Consultes simples sobre varies taules
- 4 Consultes complexes: Subconsultes
- 5 Consultes complexes: Agrupació (GROUP BY)
- 6 Operadors conjuntistes
- 7 Operador JOIN
- 8 Llenguatge de Manipulació de Dades (DML)
- 9 Instrucció INSERT
- 10 Instrucció DELETE
- 11 Instrucció UPDATE

ACTUALIZACIÓ DE LA BASE DE DADES

Les instruccions que modifiquen les dades sols poden aplicar-se a una taula cada vegada.

- **INSERT** (per a la inserció de tuples senceres)
- **DELETE** (per a l'esborrament de tuples senceres)
- **UPDATE** (per a la modificació d'un o més atributs en una o més tuples)

UD2.1 El llenguatge SQL: consultes i actualització (DML)

INTRODUCCIÓ D'INFORMACIÓ

Sintaxi de l'operació INSERT

INSERT INTO taula [(*columna*₁, *columna*₂,..., *columna*_n)]

{DEFAULT VALUES | VALUES(àtom₁, àtom₂,..., àtom_n) |
expressió_taula}

Si no s'inclou la llista de columnes s'hauran de donar valors a tots els atributs de *taula*.

- Si s'inclou l'opció DEFAULT VALUES s'insertarà una única fila en la taula amb els valors per defecte apropiats en cada columna (segons la definició de *taula*).
- En l'opció VALUES(àtom₁, àtom₂,..., àtom_n), un àtom és una expressió escalar del tipus de dades apropiat (text, enter, etc...)
- Amb l'opció *expressió_taula*, s'insertaran les files resultants de l'execució de l'expressió SELECT .

UD2.1 El llenguatge SQL: consultes i actualització (DML)

INTRODUCCIÓ D'INFORMACIÓ:

INSTRUCCIÓ INSERT

EXEMPLE d'inserció d'una tupla completa:

Afegir un ciclista de dorsal 101, nom 'Joan Peris', de l'equip 'Kelme', i de 27 anys.

```
INSERT INTO Ciclista  
VALUES (101, 'Joan Peris', 27, 'Kelme');
```

```
INSERT INTO taula [(columna1, columna2,..., columnan)]  
{DEFAULT VALUES | VALUES (àtom1, àtom2,..., àtomn) |  
expressió_taula}
```

UD2.1 El llenguatge SQL: consultes i actualització (DML)

INTRODUCCIÓ D'INFORMACIÓ:

INSTRUCCIÓ INSERT

EXEMPLE d'inserció d'una tupla incompleta:

Afegir un ciclista de dorsal 101, nom 'Joan Peris', i de l'equip 'Kelme' (no sabem l'edat).

```
INSERT INTO Ciclista (dorsal, nombre, nomeq)  
VALUES (101, 'Joan Peris', 'Kelme');
```

```
INSERT INTO taula [(columna1, columna2,..., columnan)]  
{DEFAULT VALUES | VALUES (àtom1, àtom2,..., àtomn) |  
expressió_taula}
```

UD2.1 El llenguatge SQL: consultes i actualització (DML)

INTRODUCCIÓ D'INFORMACIÓ:

INSTRUCCIÓ INSERT

EXEMPLE d'inserció múltiple:

Afegir a la taua 'Ciclista_ganador', (amb el mateix esquema que Ciclista), la informació dels ciclistes que han guanyat alguna etapa.

```
INSERT INTO Ciclista_ganador
```

```
SELECT * FROM Ciclista
```

```
WHERE dorsal IN (SELECT dorsal FROM etapa)
```

```
INSERT INTO taula [(columna1, columna2,..., columnan)]
```

```
{DEFAULT VALUES | VALUES (àtom1, àtom2,..., àtomn) |  
expressió_taula}
```


UD2.1 El llenguatge SQL: consultes i actualització (DML)

ELIMINACIÓ D'INFORMACIÓ:

INSTRUCCIÓ DELETE

EXEMPLE: Eliminar la informació del ciclista 'M. Indurain' ja que s'ha jubilat.

```
DELETE FROM Ciclista WHERE nombre = 'M. Indurain';
```

SINTAXI:

```
DELETE FROM tabla [WHERE expresión_condicional]
```

Si s'inclou la clàusula WHERE s'eliminaran aquelles tuples que fassen certa la condició; si no, s'eliminaran totes les tuples de la taula.

UD2.1 El llenguatge SQL: consultes i actualització (DML)

MODIFICACIÓ DE LA INFORMACIÓ:

INSTRUCCIÓ UPDATE

EXAMPLE:

Incrementar un 10% els premis dels mallots.

```
UPDATE Maillot SET premio = premio * 1.10
```

SINTAXI:

```
UPDATE taula SET assignació1, assignació2,..., assignación
```

on una *assignació* és de la forma:

```
columna = {DEFAULT | NULL | expressió_escalar}
```

UD2.1 El llenguatge SQL: consultes i actualització (DML)

MODIFICACIÓ DE LA INFORMACIÓ:

INSTRUCCIÓ UPDATE

UPDATE taula SET assignació₁, assignació₂,..., assignació_n
[WHERE *expressió_condicional*]

Si s'inclou la clàusula WHERE sols s'aplicarà a les files que fassen certa la condició.

EXAMPLE:

Els ciclistes del Kelme es canvien tots a l'equip *K10* de nova creació.

```
UPDATE Ciclista SET nomeq = 'K10'  
WHERE nomeq='Kelme'
```

INSTRUCCIONS DE CONTROL DE TRANSACCIONS

INICI DE TRANSACCIÓ: sense instrucció, és implícit en cada sessió i quan acaba altra transacció.

FI DE TRANSACCIÓ: hi ha dues instruccions possibles

- **COMMIT:** indica el fi d'una transacció que es desitja confirmar.
- **ROLLBACK:** indica el fi d'una transacció que es descarta.

UD2.1 El llenguatge SQL: consultes i actualització (DML)

INSTRUCCIONS DE CONTROL DE TRANSACCIONS

Exemple:

```
UPDATE Equipo SET nombre = '¿BanQué?'  
WHERE nomeq = 'Banesto';  
UPDATE Ciclista SET nomeq = '¿BanQué?'  
WHERE nomeq = 'Banesto';  
COMMIT;
```

Nota: Perquè aquesta transacció s'execute, cal indicar al SGBD que no comprove la integritat referencial fins després del COMMIT.

UD2.1 El llenguatge SQL: consultes i actualització (DML)

INSTRUCCIONS DE CONTROL DE TRANSACCIONS

Exemple:

COMMIT;

SET CONSTRAINT ca_ciclista DEFERRED;

UPDATE Equipo SET nomeq = '¿BanQué?'

WHERE nomeq = 'Banesto';

UPDATE Ciclista SET nomeq = '¿BanQué?'

WHERE nomeq = 'Banesto';

COMMIT;

Nota: Perquè aquesta transacció s'execute, cal indicar al SGBD que no comprove la integritat referencial fins després del COMMIT.