

Structured Query Language (SQL)

Ernest Valveny,
Enric Martí

Enginyeria Informàtica
Escola d'Enginyeria (seu Bellaterra)
Universitat Autònoma de Barcelona

Setembre del 2010

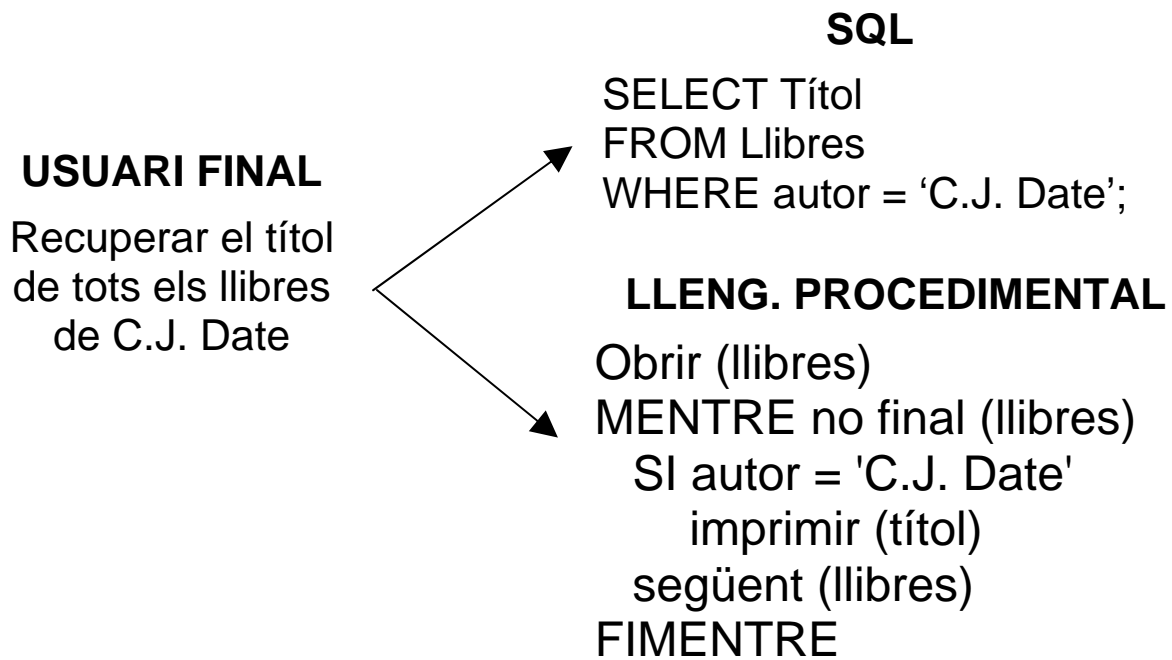
ÍNDEX

1. INTRODUCCIÓ.....	3
2. BASE DE DADES D'EXEMPLE	6
3. CONSULTES	7
3.1 CONSULTES BÀSIQUES	8
3.2 EXPRESSIONS CONDICIONALS.....	11
3.3 CONSULTES AMB VÀRIES TAULES.	13
4. FUNCIONS DE TOTALS	18
5. AGRUPACIÓ DE CONSULTES.....	20
6. SUBCONSULTES.....	24
7. OPERACIONS DE CONJUNTS.....	28
8. ACTUALITZACIONS.....	29
9. SQL IMMERS.....	32
10. DEFINICIÓ DE LA BD.....	41

1. INTRODUCCIÓ (1)

Característiques de SQL

- Conjunt reduït d'instruccions.
- Llenguatge no procedimental:
 - Llenguatge d'alt nivell: proper a l'usuari final.
 - Només s'especifica què es vol fer i no com s'ha de fer.



- Independència de dades.
- Llenguatge estandard d'accés a bases de dades relacionals:
 - Versions diferents de l'standard: SQL/86, SQL2, SQL3.
- Dues formes d'utilització:
 - Interactiva per a usuaris finals.
 - Dins de llenguatges de programació (C, PASCAL, etc.) per programadors d'aplicacions (SQL immers).

1. INTRODUCCIÓ (2)

Subllenguatges de SQL

- **DDL** (*Data Definition Language*): Definició de dades.
 - Quines dades volem guardar: taules i atributs.
 - Mecanismes d'integritat de les dades: claus primàries i externes, restriccions d'usuari (dominis).
- **DML** (*Data Manipulation Language*): Accés i actualització de les dades.
 - Consultes
 - Insercions
 - Actualitzacions
 - Eliminacions
- **DCL** (*Data Control Language*): Control de l'accés a les dades.
 - Control seguretat: definició d'usuaris i privilegis d'accés.
 - Control accés concurrent: gestió de transaccions.

2. BASE DE DADES D'EXEMPLE

Gestió d'una biblioteca. Contingut de la BD

LLIBRES					
codi	títol	editorial	idioma	autor	n_eds
1	Introducción a los sistemas de Bases de Datos	Addison-Wesley	Castellà	Date	5
2	A guide to the SQL standard	Addison-Wesley	Anglès	Date	3
3	Organización de las Bases de Datos	Prentice-Hall	Castellà	Martin	1

TEMES	
codi_llibre	tema
1	Bases de Dades
2	Bases de Dades
2	SQL
3	Bases de Dades

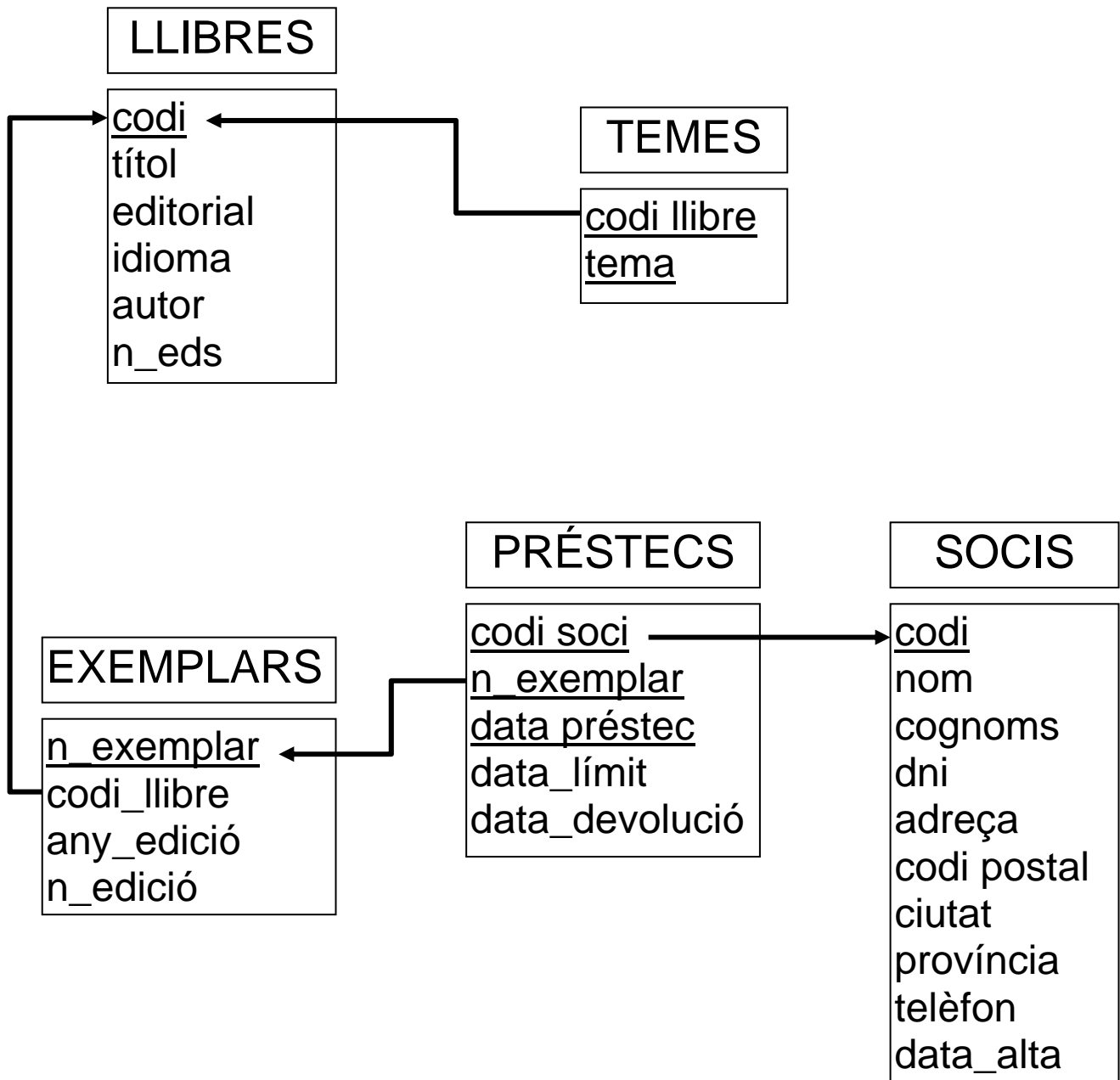
EXEMPLARS			
nº_exemplar	codi_llibre	any	n_edició
1	1	2003	5
2	1	2003	5
3	2	2004	2
4	2	2004	2
5	2	2006	3
6	3	2007	1

SOCIS							
codi	nom	cognoms	dni	adreça	...	telèfon	data_alta
1	Ernest	Valveny	11111111	UAB		111 11 11	1/5/07
2	Enric	Martí	22222222	UAB		222 22 22	15/7/06
3	David	Lloret	33333333	UAB		333 33 33	12/8/08

PRÉSTECES				
codi_soci	n_exemplar	data_préstec	data_límit	data_devolució
1	1	15/10/07	30/10/07	20/10/07
1	3	8/8/08	23/8/08	25/8/08
1	4	5/3/08	5/4/08	8/4/08
2	2	6/9/07	21/9/07	21/9/07
2	2	2/2/08	17/2/08	19/2/08
2	5	9/4/08	9/5/08	7/5/08
3	3	30/9/09	30/9/10	
3	4	8/7/08	15/7/08	15/7/08

2. BASE DE DADES D'EXEMPLE

Gestió d'una biblioteca. Diagrama de la BD



3. CONSULTES

Expressió general

```
SELECT llista d'atributs  
FROM llista de taules [;]  
[WHERE condicions [;] ]  
[GROUP BY llista d'atributs [;] ]  
[HAVING condicions [;] ]  
[ORDER BY llista d'atributs [;] ]
```


- **SELECT**: Atributs que es volen recuperar.
- **FROM**: Taules necessàries per obtenir la informació que es vol recuperar.
- **WHERE**: Condicions que han de complir els registres que es recuperin.
- **GROUP BY**: Atributs pels que s'ha d'agrupar el resultat de la consulta.
- **HAVING**: Condicions que han de complir els grups que es recuperin.
- **ORDER BY**: Atributs que determinen l'ordre de recuperació de les dades.

3.1 CONSULTES BÀSIQUES (1)

Recuperar tota la informació d'una taula

Exemple: Recuperar tots els llibres de la base de dades.

```
SELECT *  
FROM Llibres;
```




codi	títol	editorial	idioma	autor	n_eds
1	Introducción a los sistemas de Bases de Datos	Addison-Wesley	Castellà	Date	5
2	A guide to the SQL standard	Addison-Wesley	Anglès	Date	3
3	Organización de las Bases de Datos	Prentice-Hall	Castellà	Martin	1


Projecció: Seleccionar atributs de la taula

Exemple: Recuperar el títol i autor de tots els llibres.

```
SELECT títol, autor  
FROM Llibres;
```



codi	títol	editorial	idioma	autor	n_eds



títol	autor
Introducción a los sistemas de Bases de Datos	Date
A guide to the SQL standard	Date
Organización de las Bases de Datos	Martin

3.1 CONSULTES BÀSIQUES (2)

Consultes sense duplicats: **DISTINCT**

`SELECT DISTINCT` *llista d'atributs*

- Per defecte, si en el resultat de la consulta hi ha valors duplicats, es visualitzen tants cops com surtin.
- `DISTINCT` elimina del resultat de la consulta les files amb valors duplicats.

Exemple: *Recuperar el nom de tots els autors.*

```
SELECT autor  
FROM Llibres;
```



autor
Date
Date
Martín

```
SELECT DISTINCT autor  
FROM Llibres;
```



autor
Date
Martín

Ordre de les consultes: **ORDER BY**

`ORDER BY` *atribut ASC/DESC, atribut ASC/DESC, ...*

- Per defecte, l'ordre de visualització del resultat d'una consulta no està definit.
- Per defecte, `ORDER BY` ordena de forma ascendent.

Exemple: *Recuperar títol i autor de tots els llibres ordenats per autor (en ordre alfabètic invers) i títol.*

```
SELECT títol, autor  
FROM Llibres  
ORDER BY autor DESC, títol;
```



títol	autor
Organización de las ...	Martín
A guide to the SQL ...	Date
Introducción a los ...	Date

3.1 CONSULTES BÀSIQUES (3)

Ús d'expressions aritmètiques en les consultes

SELECT ***Expressió AS Nom Expressió***

- En el resultat de la consulta es pot incloure qualsevol expressió aritmètica calculada a partir del valor dels atributs de la taula.

Exemple: Per cada préstec, recuperar el nº d'exemplar, el codi del soci, i el nº de dies que s'ha trigat en retornar, ordenat pel codi del soci i el nº de dies.

```
SELECT      n_exemplar, codi_soci,  
            data_devolució - data_préstec AS n_dies  
FROM        Préstecs  
ORDER BY    codi_soci, n_dies;
```

WHERE condició

- La condició indica quins registres es seleccionen.
- Hi ha diferents operadors de condició segons el tipus d'atribut.
- Podem construir expresions condicionals amb els operadors booleans AND, OR, NOT.

Exemple: Recuperar el títol de tots els llibres escrits per Date.

```
SELECT  títol  
FROM    Llibres  
WHERE   autor = 'Date';
```

títol

↓

títol
Introducción a los sistemas de Bases de Datos
A guide to the SQL standard

3.2 EXPRESSIONS CONDICIONALS

1. TIPUS NUMÈRIC I DATA

- Operadors de comparació habituals: <, >, <=, >=, =, <>
- Rangs de valors:

atribut BETWEEN valor1 AND valor2

2. TIPUS ALFANUMÈRIC

- **Llistes de valors:**

atribut IN (valor1, valor2, ..., valorN)

- **Patrons de cadenes de caràcters:** permeten fer comparacions parcials de cadenes de text.

atribut LIKE expressió

expressió és una cadena de caràcters a comparar amb el valor de l'atribut. Pot incloure caràcters especials per poder representar un conjunt de cadenes de caràcters:

- *'_'* representa qualsevol caràcter individual.
- *'%'* representa qualsevol cadena de caràcters (*'*' en ACCESS*)

Exemple: *Si volem els noms que comencen per 'A':*

nom LIKE 'A%' (nom LIKE 'A' en ACCESS)*

3. VALORS NULS

atribut IS [NOT] NULL

- Retorna cert si l'atribut no té valor, té valor nul [o no].
- És l'única forma de saber si un atribut té valor o no.

4. OPERADORS BOOLEANS DE CONDICIÓ

- Combinació de condicions: AND, OR, NOT

Exemple:

(codi_llibre=234) AND (any > 2007) OR (dni IS NOT NULL)

3.2 EXPRESSIONS CONDICIONALS (2)

Exemples

- *Nom i cognoms dels socis que viuen a Barcelona o Girona i que s'han donat d'alta durant aquest any.*

(Socis WHERE ciutat='Barcelona' OR ciutat='Girona' AND data_alta>=#1/1/09#)[nom,cognoms]

```
SELECT  nom, cognoms
FROM    Socis
WHERE   ciutat IN ('Barcelona', 'Girona')
AND     data_alta >= #1/1/09#
```

- *Recuperar el número d'exemplar i el codi del soci de tots els préstecs fets l'any 2008 i que encara no han estat retornats.*

(Préstecs WHERE data_pretec>'1/1/08' AND data_pretec<'31/12/08' AND data_devolucio=NULL)[nº_exemplar, codi_soci]

```
SELECT  n_exemplar, codi_soci
FROM    Préstecs
WHERE   data_préstec BETWEEN #1/1/08# AND #31/12/08#
AND     data_devolució IS NULL;
```

- *Llibres que en el títol contenen l'expressió **Bases de dades***

(Llibres WHERE títol LIKE '%Base de Dades%')

```
SELECT  *
FROM    Llibres
WHERE   títol LIKE '%Bases de dades%';
```

- *Recuperar el número d'exemplar i el codi del soci de tots els préstecs fets aquest any i que o bé han estat retornats fora de plaç o bé encara no s'han retornat i ja estan fora de plaç.*

(Préstecs WHERE (data_préstec>=#1/1/09#) AND ((data_devolució IS NULL) AND (CURRENT_DATE>data_límit)))[nº_exemplar, codi_soci]

```
SELECT  nº_exemplar, codi_soci
FROM    Préstecs
WHERE   (data_préstec >= #1/1/09#) AND
        (data_devolució > data límit) OR
        ((data_devolució IS NULL AND
          CURRENT_DATE > data_límit));
```

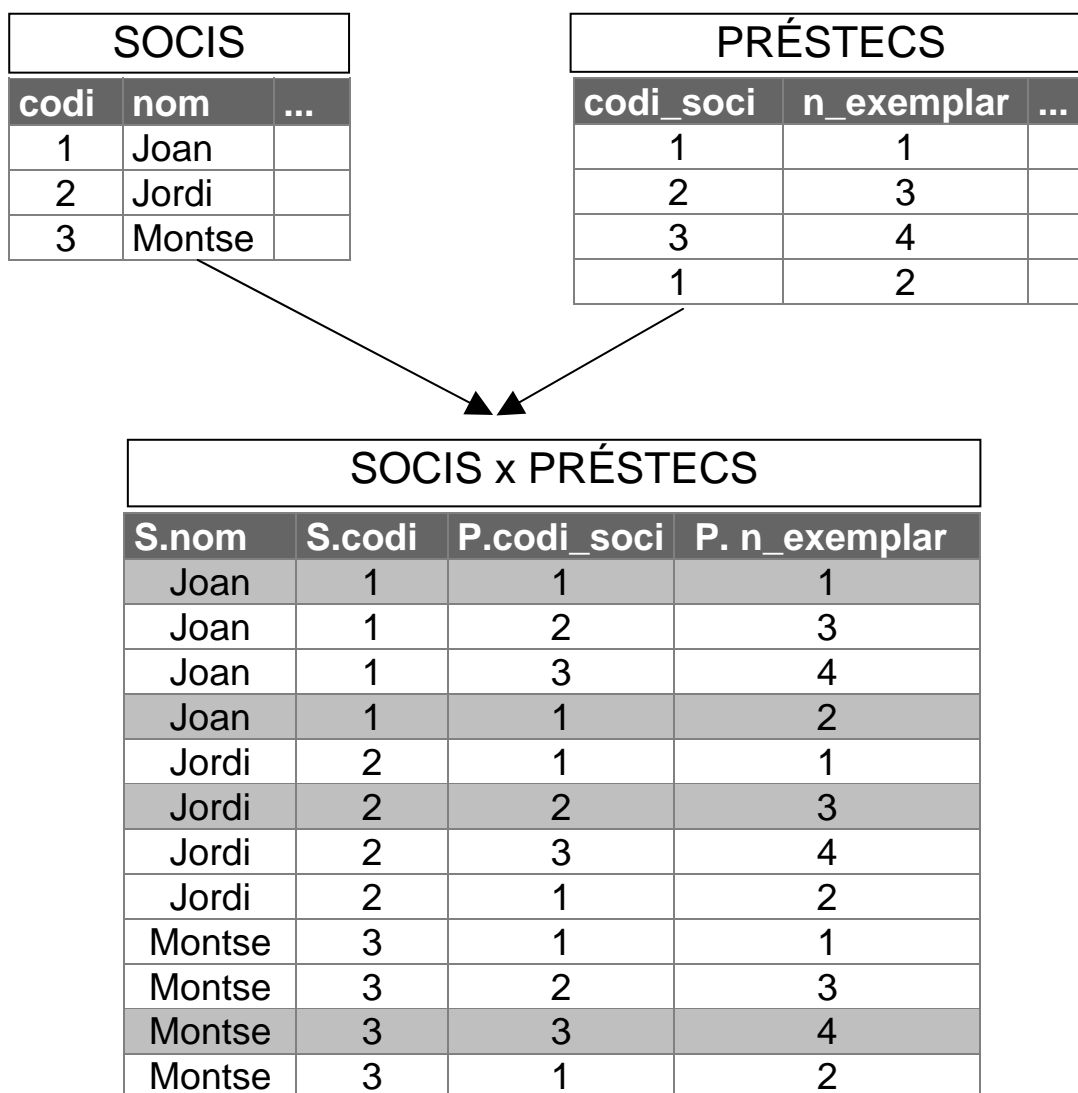
3.3 CONSULTES AMB VÀRIES TAULES (1)

Producte Cartesià

```
SELECT ...  
FROM taula1, taula2
```

- Combinació de tots els registres d'una taula amb tots els de l'altra.

```
SELECT S.nom, S.codi, P.codi_soci, P.n°_exemplar  
FROM Préstecs P, Socis S;
```



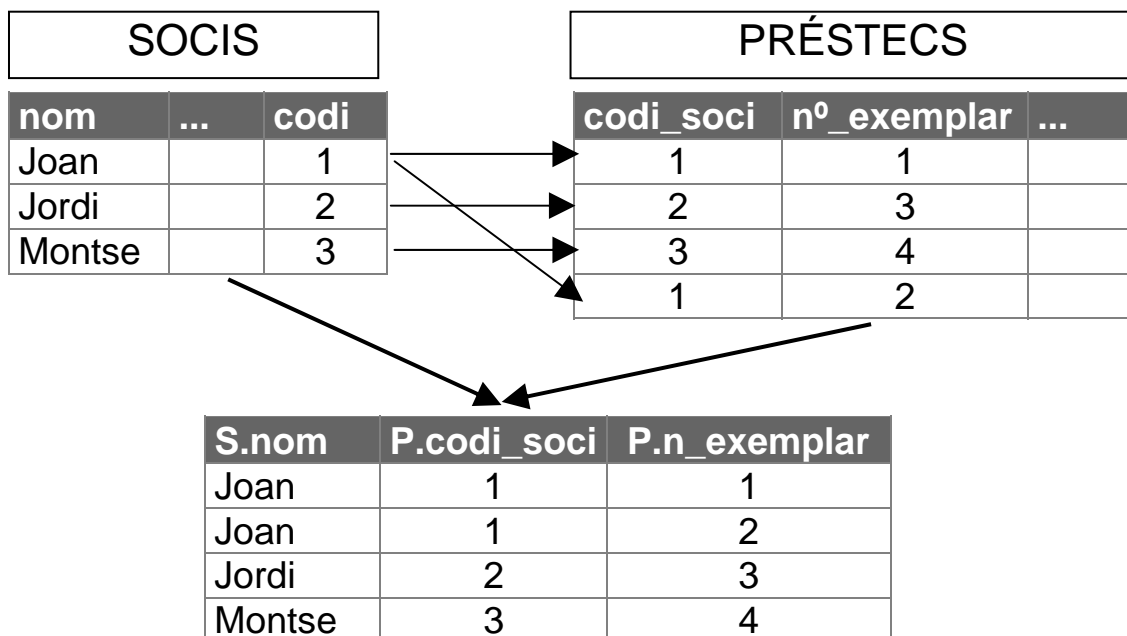
- Només interessa combinar els registres amb el mateix valor als atributs comuns.

3.3 CONSULTES AMB VÀRIES TAULES (2)

Join

- Combinació dels registres de vàries taules, però només dels que estan relacionats per algun atribut comú.
- Normalment, a la clàusula WHERE s'ha d'exigir que els registres tinguin el mateix valor en els atributs comuns.

```
SELECT  S.nom, P.codi_soci, P.n_exemplar
FROM    Socis S, Préstecs P
WHERE   S.codi = P.codi_soci;
```



- FROM: Totes les taules que contenen algun dels atributs que necessitem.
- WHERE: Condicions de combinació de les taules (join) + qualsevol altra condició de selecció de registres.
- Per resoldre ambigüitats en els noms dels atributs:
[Nom taula].[nom atribut]

3.3 CONSULTES AMB VÀRIES TAULES (3)

JOINS: Exemples

- *Temes en els que tenim classificats els llibres escrits per **Date**.*

((Temes x Llibres) WHERE L.Autor='Date')[Tema]

```
SELECT    DISTINCT T.tema
FROM      Temes T, Llibres L
WHERE     T.codi_llibre = L.codi AND
          L.autor = 'Date' ;
```

- *Recuperar els préstecs no retornats. S'ha de veure el codi del llibre i el nom i cognoms del soci, ordenat per cognoms i nom del soci.*

((Exemplars x Prestecs) x Socis) WHERE
P.Data_devolucio=NULL) [codi_llibre,nom,cognoms]

```
SELECT      E.codi_llibre, S.nom, S.cognoms
FROM        Exemplars E, Préstecs P, Socis S
WHERE       E.nº exemplar = P.n_exemplar AND
            S.codi = P.codi_soci AND
            P.data_devolució IS NULL
ORDER BY    S.cognoms, S.nom;
```

- *Nom i cognoms dels socis que durant l'any 2008 han agafat en préstec algun dels llibres escrits per **Date**.*

((((Socis x Prestecs) x Exemplars) x Llibres) WHERE
L.Autor='Date' AND P.Data_Prestec > 1/1/08 AND
P.Data_prestec<#31/12/08#)[nom,cognoms]

```
SELECT      DISTINCT (S.nom, S.cognoms)
FROM        Socis S, Préstecs P, Exemplars E, Llibres L
WHERE       S.codi = P.codi_soci AND
            P.n_exemplar = E.n_exemplar AND
            E.codi_llibre = L.codi AND
            L.autor='Date' AND
            P.data_préstec BETWEEN #1/1/08# AND #31/12/08#;
```

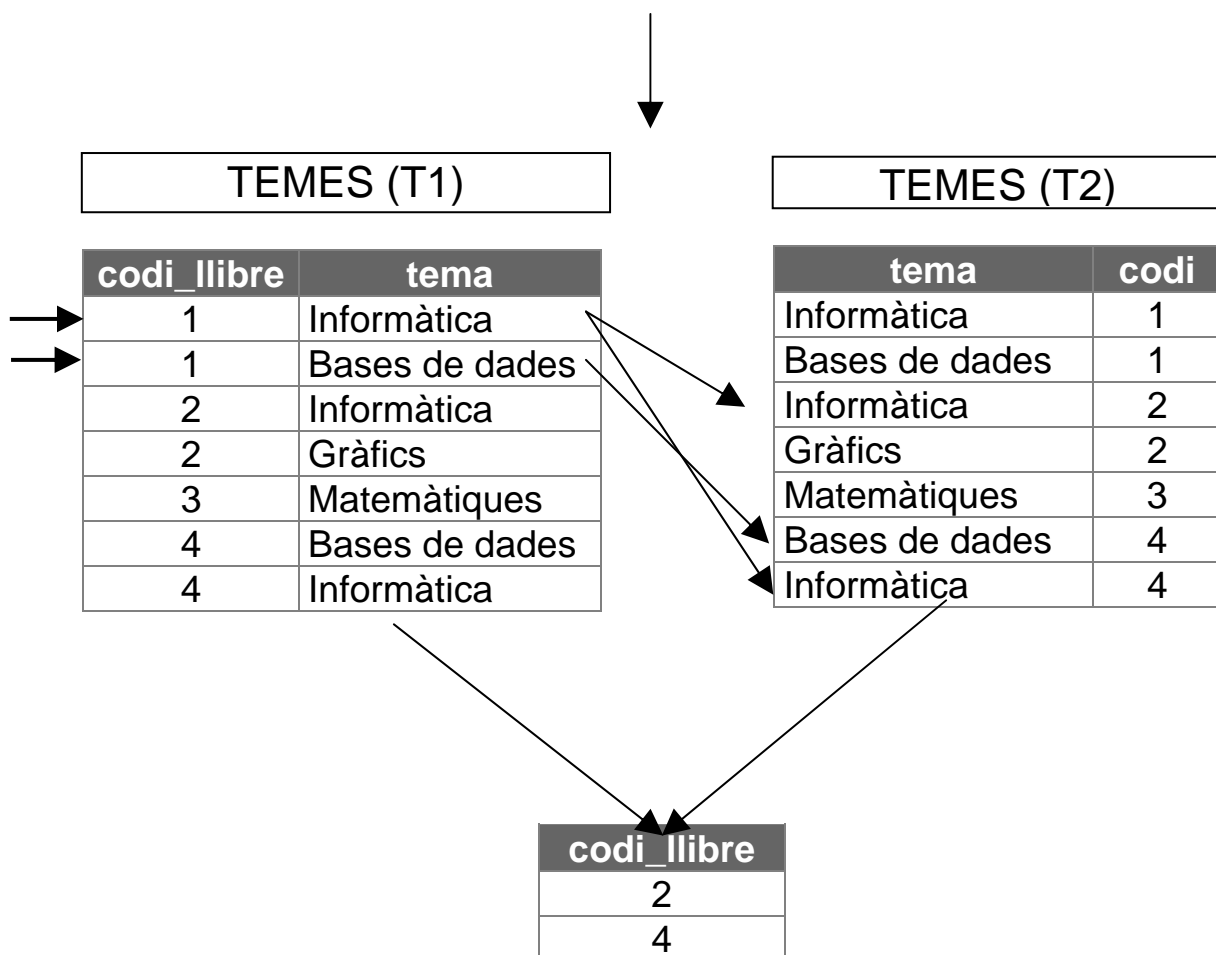
3.3 CONSULTES AMB VÀRIES TAULES (4)

Join d'una taula amb ella mateixa

Exemple: Codis dels llibres que estan classificats en algun dels temes als que pertany el llibre amb codi 1.

```
((Temes T1 x Temes T2) WHERE T1.codi_llibre=1 AND  
T2.codi_llibre<>T1.codi_llibre)[T2.codi_llibre]
```

```
SELECT    DISTINCT T2.codi_llibre  
FROM      Temes T1, Temes T2  
WHERE     T1.codi_llibre = 1 AND  
          T2.tema = T1.tema AND  
          T2.codi_llibre <> T1.codi_llibre;
```



3.3 CONSULTES AMB VÀRIES TAULES (5)

Outer Join

- Permet recuperar registres d'una taula que no estan relacionats amb cap registre de l'altra taula.

taula1 LEFT JOIN taula2 ON atribut1 = atribut2

- Recupera tots els registres de *taula1*, encara que no estiguin relacionats amb cap registre de *taula2*.
- Als atributs de *taula2* pels que no es té valor es posa NULL.

taula1 RIGHT JOIN taula2 ON atribut1 = atribut2

- Recupera tots els registres de *taula2*, encara que no estiguin relacionats amb cap registre de *taula1*.
- Als atributs de *taula1* pels que no es té valor es posa NULL.

SOCIS		
nom	...	codi
Joan		1
Jordi		2
Montse		3

PRÉSTECES		
codi_soci	n_exemplar	...
1	1	
2	2	
4	3	

```
SELECT *
FROM Socis S, Préstecs P
WHERE S.codi = P.codi_soci;
```

codi_soci	nom	n_exemplar	...
1	Joan	1	
2	Jordi	2	

```
SELECT *
FROM Socis LEFT JOIN Préstecs
ON S.codi = P.codi_soci;
```

codi_soci	nom	n_exemplar	...
1	Joan	1	
2	Jordi	2	
3	Montse	NULL	

```
SELECT *
FROM Socis RIGHT JOIN Préstecs
ON S.codi = P.codi_soci;
```

codi_soci	nom	n_exemplar	...
1	Joan	1	
2	Jordi	2	
4	NULL	3	

4. FUNCIONS DE TOTALS (1)

- S'obté un únic valor resultat a partir del valor d'un atribut per tots els registres seleccionats.

LLIBRES		
codi	...	n_eds
1		2
2		5
3		1
4		3



COUNT (codi) = 4
MAX (nº_eds) = 5
AVG (nº_eds) = 2.75

Funcions possibles:

SUM	(atribut)	Suma del valor de l'atribut per tots els registres.
COUNT	(atribut)	Nombre total de registres seleccionats.
COUNT	(*)	
COUNT	(DISTINCT atribut)	Nombre de valors diferents de l'atribut en els registres seleccionats.
MAX	(atribut)	Valor màxim de l'atribut en els registres seleccionats.
MIN	(atribut)	Valor mínim de l'atribut.
AVG	(atribut)	Valor mig de l'atribut.

Exemple: Recuperar el nombre de llibres escrits per Date i el promig d'edicions que s'han fet d'aquests llibres.

```
SELECT    COUNT (codi), AVG (nº_edicions)
FROM      Llibres
WHERE     autor = 'Date' ;
```

Exemple: Recuperar el nombre de dies del préstec més llarg de l'any 2008.

```
SELECT    MAX (data_devolució - data_préstec)
FROM      Préstecs
WHERE     data_préstec BETWEEN #1/1/08# AND #31/12/08#;
```

- Quan s'utilitzen funcions de totals no es poden recuperar alhora atributs individuals, a menys que s'utilitzi GROUP BY.

4. FUNCIONS DE TOTALS (2)

Tractament de valors duplicats:

- Si volem que al calcular la funció, els valors duplicats de l'atribut només es tinguin en compte un cop, hem de posar DISTINCT davant del nom de l'atribut.

Exemple: Recuperar el nombre de llibres que hi ha a la biblioteca, el nº d'exemplars i el promig d'exemplars per llibre.

```
SELECT    COUNT (DISTINCT codi_llibre) AS nllibres,
          COUNT (n_exemplar) AS n_exs,
          num_exs / numllibres AS promig
FROM      Exemplars;
```

- COUNT (DISTINCT <atribut>) no s'implementa en ACCESS.

```
SELECT    COUNT (T1.codi_llibre) AS numllibres,
          COUNT (E.n_exemplar) AS num_exs,
          num_exs / numllibres AS promig
FROM      Exemplars E,
          (SELECT    DISTINCT      codi_llibre      FROM
          exemplars) T1;
```

Tractament de valors nuls i conjunts buits:

- Si el conjunt de registres sobre el que s'aplica la funció és buit, COUNT retorna 0 i la resta de funcions, NULL.
- Al calcular qualsevol d'aquestes funcions els valors nuls són com si no existissin.

Exemple: Recuperar el nombre de préstecs totals fets l'any 2008, el nombre de préstecs retornats i el nombre de préstecs no retornats.

```
SELECT    COUNT (data_préstec) AS total,
          COUNT (data_devolució) AS retornats,
          total - retornats AS no_retornats
FROM      Préstecs
WHERE     data_préstec BETWEEN #1/1/08# AND #31/12/08#;
```

5. AGRUPACIÓ DE CONSULTES (1)

GROUP BY *llista d'atributs*

- Agrupa tots els registres que tenen igual valor en l'atribut o atributs especificats al GROUP BY.
- En el resultat final de la consulta es recupera una única fila per cadascun dels grups que s'han format.
- En el SELECT només es poden recuperar els atributs que defineixen el grup i funcions de total.
- Les funcions de total especificades al SELECT es calculen per cadascun dels grups que s'hagin format i s'apliquen a tots els registres del grup.

Exemple: *Recuperar, per cada soci, el nombre de llibres que ha agafat en préstec.*

```
SELECT      codi_soci, COUNT (n°_exemplar) AS nllibres
FROM        Préstecs
GROUP BY    codi_soci;
```

codi_soci	n_exemplar	...
1	1	
3	3	
2	5	
3	4	
1	3	
2	2	
1	4	

Agrupació
per codi de
soci

codi_soci	n_exemplar	...
2	2	
2	5	
1	1	
1	3	
1	4	
3	3	
3	4	

Càlcul del nombre de llibres
per cada soci (grup)

codi_soci	nllibres
1	3
2	2
3	2

5. AGRUPACIÓ DE CONSULTES (2)

Exemples

- *Nombre de llibres diferents que ha agafat en préstec cada soci.*

```
SELECT      P.codi_soci,  
            COUNT (DISTINCT E.codi_llibre)  
FROM        Préstecs P, exemplars E  
WHERE       P.n_exemplar = E.n_exemplar  
GROUP BY    P.codi_soci;
```

- Primer es seleccionen els registres segons la condició del WHERE, i després s'agrupen pels atributs del GROUP BY i es calculen els totals.

- *Promig de dies que ha estat cada llibre en préstec.*

```
SELECT      E.codi_llibre,  
            AVG (P.data_devolució - P.data_préstec)  
FROM        Préstecs P, Exemplars E  
WHERE       P.n_exemplar = E.n_exemplar AND  
            P.data_devolució IS NOT NULL  
GROUP BY    E.codi_llibre;
```

- *Nombre d'exemplars de cadascun els llibres editats per Addison-Wesley. S'ha de recuperar el títol, idioma, autor i nombre d'exemplars del llibre, ordenat per nombre d'exemplars i autor.*

```
SELECT      L.títol, L.idioma, L.autor,  
            COUNT (n_exemplar) As num_exemplars  
FROM        Llibres L, Exemplars E  
WHERE       L.codi = E.codi_llibre AND  
            L.editorial = 'Addison-Wesley'  
GROUP BY    L.títol, L.idioma, L.autor  
ORDER BY    num_exemplars DESC, L.autor;
```

5. AGRUPACIÓ DE CONSULTES (3)

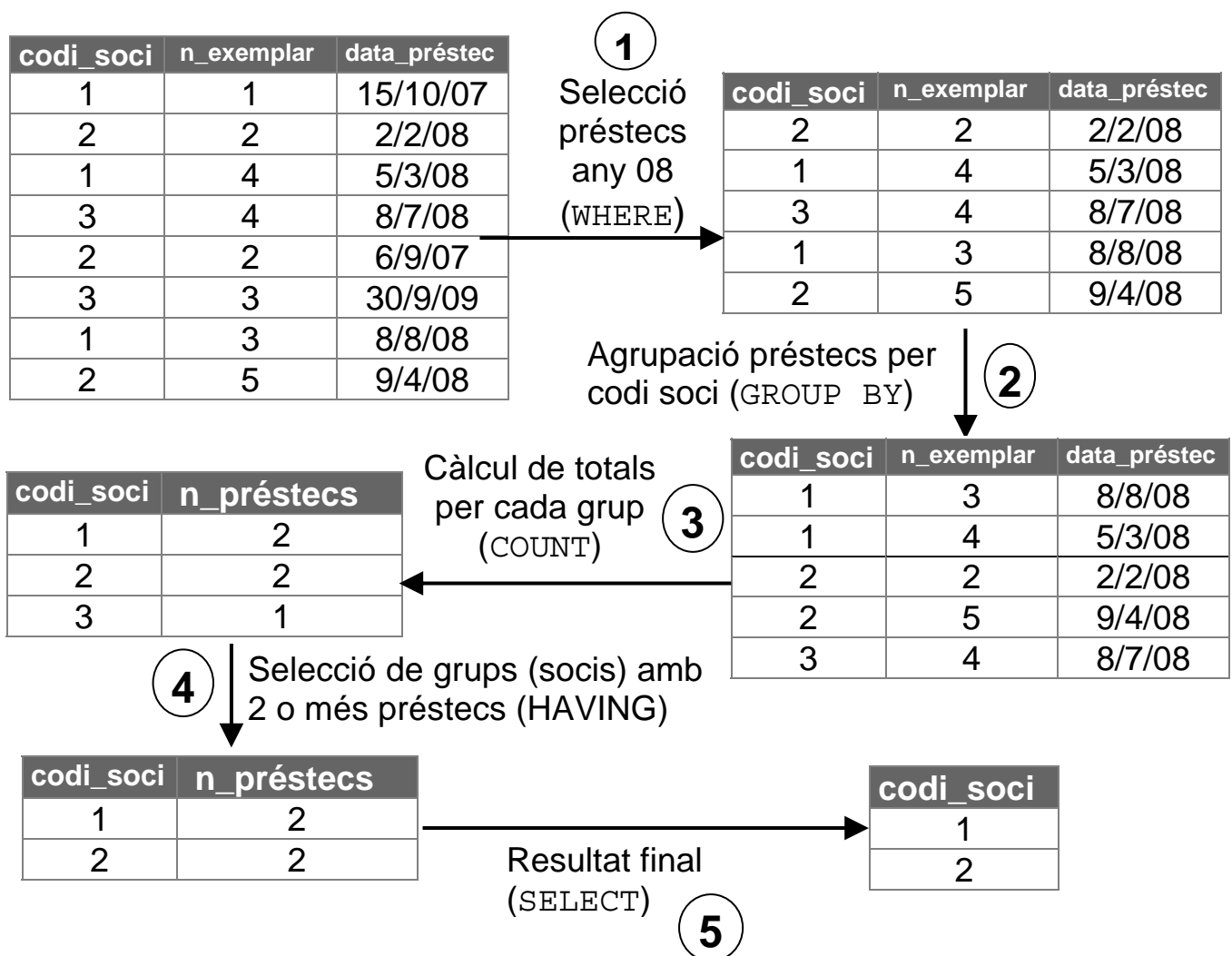
Selecció de grups

HAVING condició

- Després d'agrupar els registres i fer els càlculs per cada grup, aplica una condició als grups per decidir quins es recuperen.

Exemple: Recuperar el codi dels socis que han agafat més d'un llibre en préstec durant l'any 2008.

```
SELECT    codi_soci
FROM      Préstecs
WHERE     data_préstec BETWEEN #1/1/08# AND #31/12/08#
GROUP BY  codi_soci
HAVING    COUNT (*) > 1;
```



5. AGRUPACIÓ DE CONSULTES (4)

Selecció de grups: exemples

- *Recuperar el codi dels llibres que pertanyen a més d'un tema.*

```
SELECT      codi_llibre
FROM        Temes
GROUP BY    codi_llibre
HAVING      COUNT(*) > 1;
```

- *Per tots els autors que han publicat més de cinc llibres a Addison-Wesley, recuperar el nº total d'exemplars que hi ha a la biblioteca de tots els seus llibres, ordenat per nº d'exemplars i autor.*

```
SELECT      L.autor, COUNT (*) AS num_exemplars
FROM        Llibres L, Exemplars E
WHERE       L.editorial = 'Addison-Wesley' AND
            L.codi = E.codi_llibre
GROUP BY    L.Autor
HAVING      COUNT (DISTINCT L.codi) > 5
ORDER BY    num_exemplars DESC, L.autor;
```

- *Nombre total de préstecs que han fet els socis que han tret algun llibre durant l'any 2009.*

```
SELECT      codi_soci, COUNT (*)
FROM        Préstecs
GROUP BY    codi_soci
HAVING      MAX (data préstec) >= #1/1/09#;
```

- Posar la condició de *data préstec* al WHERE suposaria comptar només els préstecs fets aquest any.

6. SUBCONSULTES (1)

Una consulta retorna un conjunt de valors. Aquest conjunt de valors es pot utilitzar en condicions d'altres consultes.

Exemple: *Nom i cognoms del soci més antic de la biblioteca.*

- Amb una consulta podem calcular la data d'inscripció del soci més antic, però no quin és aquest soci.

```
SELECT    MIN (data_alta)
FROM      Socis;
```

- Podem aprofitar el valor que retorna aquesta consulta per recuperar quin és aquest soci.

```
SELECT    nom, cognoms
FROM      Socis
WHERE     data_alta =
```

```
(SELECT    MIN (data_alta)
FROM      socis);
```

Soci amb data d'alta igual
a la més antiga

Data d'alta més
antiga

- Quan una consulta retorna un únic valor, aquest valor es pot fer servir en comparacions com qualsevol altra constant.
- Les subconsultes han de tenir un únic atribut de sortida.

6. SUBCONSULTES (2)

Subconsultes que retornen molts valors

- Necessitem operadors de comparació especials per indicar com s'ha de comparar l'atribut amb el conjunt de valors que retorna la subconsulta.

Operadors especials de comparació: IN, ANY, ALL

atribut IN (SELECT ...)	Retorna CERT si el valor de l'atribut és igual a algun dels valors retornats.
atribut = ANY (SELECT ...) > ANY ...	Retorna CERT si el valor de l'atribut compleix la condició especificada amb algun dels valors retornats.
atribut = ALL (SELECT ...) > ALL ...	Retorna CERT si el valor de l'atribut compleix la condició especificada amb tots els valors retornats.

- IN és equivalent a = ANY
- NOT IN és equivalent a <> ALL

Exemple: *Títol dels llibres deixats en préstec l'any 2009.*

```
SELECT  títol
FROM    Llibres
WHERE   codi IN
```

↑
Títol llibres prestats

Codis dels llibres deixats en préstec

↓

```
(SELECT  codi_llibre
FROM     Préstecs
WHERE    data_préstec > #1/1/09#);
```

Exemple: *Títol del llibre amb més edicions*

```
SELECT  títol
```

Número d'edicions de tots els llibres

```
FROM    Llibres
WHERE   n°_edicions >= ALL
```

↓

```
(SELECT  n°_eds
FROM     Llibres);
```

→
Llibre amb màxim d'edicions

6. SUBCONSULTES (3)

Operador existencial

EXISTS (SELECT ...)

- **CERT** si la subconsulta retorna algun registre.
- **FALS** si la subconsulta retorna el conjunt buit.

Exemple: Nom i cognoms dels socis que tenen algun llibre no retornat.

```
SELECT S.nom, S.cognoms  
FROM socis S  
WHERE EXISTS
```

```
(SELECT *  
FROM Préstecs P  
WHERE data_devolució IS NULL AND  
P.codí_soci = S.codí);
```

Recupera nom i cognoms d'un soci si existeixen préstecs seus no retornats

Préstecs no retornats corresponents al soci de la consulta global

- No importa quins atributs es recuperen amb la subconsulta. Només es comprova si es retorna o no algun registre.
- Són consultes relacionades: la subconsulta fa referència al soci de la consulta global. Per cada soci, s'executa la subconsulta per comprovar si té préstecs no retornats.

6. SUBCONSULTES (4)

Equivalència entre consultes

Exemple: Nom i cognoms dels socis que tenen algun préstec no retornat.

```
SELECT  S.nom, S.cognoms
FROM    Socis S
WHERE   0 <
(       SELECT COUNT (*)
FROM    Préstecs P
WHERE   P.codi_soci = S.codi AND
        P.data_devolució IS NULL);
```

```
SELECT  nom, cognoms
FROM    Socis
WHERE   codi = ANY
(       SELECT codi_soci
FROM    Préstecs
WHERE   data_devolució IS NULL);
```

```
SELECT  nom, cognoms
FROM    Socis S
WHERE   EXISTS
(       SELECT *
FROM    Préstecs P
WHERE   P.codi_soci = S.codi AND
        P.data_devolució IS NULL);
```

```
SELECT  nom, cognoms
FROM    Socis
WHERE   codi IN
(       SELECT codi_soci
FROM    Préstecs
WHERE   data_devolució IS NULL);
```

```
SELECT  DISTINCT S.nom, S.cognoms
FROM    Socis S, Préstecs P
WHERE   S.codi = P.codi_soci AND
        P.data_devolució IS NULL;
```

7. OPERACIONS DE CONJUNTS

UNIÓ:

```
(SELECT ...)  
UNION  
(SELECT ...)
```

INTERSECCIÓ:

```
(SELECT ...)  
INTERSECT  
(SELECT ...)
```

DIFERÈNCIA:

```
(SELECT ...)  
EXCEPT  
(SELECT ...)
```

- En totes aquestes operacions, les dues consultes han de coincidir en número i tipus d'atributs recuperats.

Exemple: *Codis dels socis de Barcelona que no han fet cap préstec durant l'any 2008.*

```
(SELECT codi  
FROM Socis  
WHERE ciutat = 'Barcelona')  
EXCEPT  
(SELECT codi_soci  
FROM Préstecs  
WHERE data_préstec BETWEEN #1/1/08# AND #31/12/08#);
```

8. ACTUALITZACIONS (1)

Insercions

Inserció d'un sol registre:

```
INSERT INTO nom taula (atribut, ..., atribut)  
VALUES (valor. .... valor)
```

Exemple: *Inserció de les dades d'un nou préstec.*

```
INSERT INTO Préstecs (codi_soci, n°_exemplar,  
                     data_préstec, data_límit)  
VALUES (1,1,CURRENT_DATE,CURRENT_DATE+15);
```

Inserció de molts registres:

```
INSERT INTO nom taula (atribut, ..., atribut)  
(SELECT ....)
```

Exemple: *Inserció de tots els llibres de Date com a llibres de Bases de Dades.*

```
INSERT INTO Temes  
(SELECT      L.codi, 'Bases de Dades'  
 FROM        Llibres L  
 WHERE       L.autor = 'Date');
```

- Els valors poden ser qualsevol constant, expressió o funció o bé les paraules clau DEFAULT o NULL.
- S'ha d'especificar valor per tots els atributs de la llista d'atributs, i en el mateix ordre en què apareixen.
- Si hi algun atribut de la taula que no apareix a la llista se li assigna el valor per defecte que tingui i si no en té es deixa a nul.
- Si no s'especifica la llista d'atributs, s'ha de donar valor a tots els atributs de la taula en l'ordre especificat quan es va crear.

8. ACTUALITZACIONS (2)

Modificacions

UPDATE	<i>nom taula</i>
SET	<i>nom atribut = expressió,</i>
	<i>:</i>
	<i>:</i>
WHERE	<i>condició</i>

- Modifica tots els registres de la taula que compleixen la condició.
- Per cada registre, modifica el valor de tots els atributs que s'indiquen, substituïnt-lo pel resultat d'avaluar l'expressió que se'ls hi assigna.
- Només es pot especificar una taula per actualitzar.
- L'expressió pot ser qualsevol constant, expressió o funció, fins i tot una subconsulta que retorni un únic valor.

Exemple: Registrar la informació que el soci amb DNI 33445566 ha retornat avui a la biblioteca el llibre **Introducción a los sistemas de Bases de Datos**.

```
UPDATE Préstecs
SET data_devolució = CURRENT_DATE
WHERE codi_soci = (SELECT codi
                  FROM Socis
                  WHERE DNI = "33445566")
AND
n°_exemplar IN
  (SELECT E.n_exemplar
   FROM Llibres L, Exemplars E
   WHERE L.títol = "Introducción a los Sistemas de
                  Bases de Datos" AND
        L.codi=E.codi_llibre );
```

8. ACTUALITZACIONS (3)

Eliminacions

DELETE	
FROM	<i>nom taula</i>
WHERE	<i>condició</i>

- Elimina tots els registres de la taula que compleixen la condició que s'indica.
- Només es pot especificar una taula.

Exemple: *Eliminar tots els préstecs fets amb anterioritat al primer de gener de l'any 2007 que ja han estat retornats.*

```
DELETE
FROM Préstecs
WHERE data_préstec < #1/1/07# AND
      data_devolució IS NOT NULL;
```

Regles d'integritat

- Totes les actualitzacions que provoquen deixar de complir les regles d'integritat de la base de dades (ja sigui la integritat d'entitats, referencial o definida per l'usuari) produeixen un error i són rebutjades.
- L'eliminació o actualització de registres d'una taula pot provocar també l'actualització o eliminació de registres d'altres taules relacionades segons les regles d'integritat referencial que s'hagin definit (CASCADE, SET NULL, etc.).

9. SQL IMMERS (1)

- Utilització de sentències SQL des de qualsevol altre llenguatge de programació (C, PASCAL, VisualBasic, etc.).
- El llenguatge de programació des del que es crida a SQL s'anomena **llenguatge amfitrió**.
- Hi ha Instruccions SQL que es poden executar directament sense cap modificació:
 - Insercions
 - Modificacions
 - Eliminacions
 - Definició de la base de dades
- Per executar consultes que retornen molts registres: **cursors**
- Per executar les sentències SQL dins del programa:

Instruccions del programa

DoCmd.RunSQL (*Sentència SQL*)

Instruccions del programa

Exemple: *Funció en C que demana el nom i cognoms d'un soci i l'elimina de la base de dades.*

```
Public Function Elimina_Socis()  
Dim nom, cognoms As String  
Dim cad As Variant  
  
nom = InputBox("Nom?")  
cognoms = InputBox("Cognoms?")  
  
cad = "DELETE * FROM Socis WHERE nom = ' " & nom & " ' "  
      AND cognoms = ' " & cognoms & " ' "  
MsgBox (cad)  
DoCmd.RunSQL (cad)  
  
End Function
```


9. SQL IMMERS (2)

Comunicació SQL- Llenguatge amfitrió

- Es poden utilitzar variables del llenguatge amfitrió dins de les sentències SQL:
 - Pas de paràmetres del llenguatge amfitrió a les instruccions SQL.
 - Recuperar la informació que retorna una consulta SQL en variables del llenguatge amfitrió.
- Les referències a les variables del llenguatge amfitrió dins de sentències SQL s'han de concatenar en la cadena que s'executarà.
- Si cal concatenar variables de tipus numèric, es posen directament:

```
cad = " DELETE * FROM Socis WHERE codi = " & nsoci
```

Si nsoci=5 equival a:

```
DELETE * FROM Socis WHERE codi_soci = 5
```

- Si cal concatenar variables de tipus string de text, cal marcar-les amb ' en el text que els envolta:

```
cad = "DELETE * FROM Socis WHERE nom = ' " & nom & " ' "
```

Si nom='Pepe' equival a:

```
DELETE * FROM Socis WHERE nom= 'Pepe'
```

- Per accedir al resultat d'una consulta necessitem el mecanisme del cursor.

9. SQL IMMERS (3)

Exemple

Aplicació: Imprimeix l'adreça del soci que s'especifica mitjançant el nom i els cognoms. Podem distingir quatre blocs en el procediment:

1. Declaració de les variables necessàries.
2. Lectura del número de soci i construcció de la cadena de consulta.
3. Execució de la consulta en SQL que busca el soci especificat.
4. Bucle per totes les tuples que compleixen la condició i impressió en pantalla.

```
Public Function LlegeixAdreça()  
' declaració de variables  
Dim dbs As Database      'objecte base de dades  
Dim rst As Recordset     'cursor  
Dim cad, adreça As Variant  
Dim num As Integer  
  
Set dbs = CurrentDb()    'instància a la BD actual  
  
' lectura i construcció de la consulta  
num = InputBox("Número de soci?")  
cad = "SELECT * FROM Socis WHERE codi = " & num  
  
' execució de la consulta  
Set rst = dbs.OpenRecordset(cad)  
  
' bucle per cada tupla que compleix la condició  
While (Not (rst.EOF))  
    adreça = rst!adreça  
    MsgBox (adreça)  
    rst.MoveNext  
Wend
```

9. SQL IMMERS (4)

Cursors

- Mecanisme que permet:
 - Executar des del llenguatge amfitrió consultes que retornen molts registres.
 - Accedir un a un a tots els registres recuperats amb la consulta.
- Un cursor proporciona:
 - Una estructura per guardar temporalment el resultat de la consulta.
 - Un apuntador al registre actual dins del cursor.
 - Un mètode per recórrer de forma seqüencial tots els registres recuperats per la consulta.
- L'execució d'una consulta amb cursors s'ha de fer en quatre passos:
 1. Declaració del cursor.
 2. Obertura del cursor.
 3. Recorregut de tots els registres recuperats un a un.
 4. Tancament del cursor.

9. SQL IMMERS (5)

Cursors

1. Declaració del cursor:

```
DIM dbs AS Database    'objecte base de dades  
DIM rst AS Recordset  'cursor
```

- És una sentència només declarativa; la consulta encara no s'executa.
- Necessitem un objecte de tipus *Database* per obrir un cursor. Aquest objecte es pot instanciar a la base actual amb:

```
SET dbs = CurrentDb()  'base actual
```

- O bé es pot obrir una base remota:

```
SET dbs = DBEngine.Workspaces(0).OpenDatabase("c:\users\david\albcn.mdb")
```

2. Obertura del cursor:

- Executa la consulta associada amb el cursor.
- Crea l'estructura del cursor i hi emmagatzema el resultat de la consulta.
- Posiciona l'apuntador del cursor just abans del primer registre recuperat.
- Permet accedir als valors de tots els atributs especificats en la consulta.

```
cad = "SELECT ... "  
SET rst = dbs.OpenRecordset(cad)
```

9. SQL IMMERS (6)

Cursors

3. Recorregut del cursor:

- Avança l'apuntador del cursor.
- Recupera el registre actual del cursor posant els valors dels atributs a les variables que s'indiquen.
- Normalment es combina amb un bucle per recórrer i processar tots els registres.

```
rst.MoveNext
```

4. Tancament del cursor i de la base:

```
rst.Close  
dbs.Close
```

- Es poden declarar i obrir varis cursors i bases de dades simultàniament.

9. SQL IMMERS (7)

Cursors: taula resum

Ordre	Significat
<code>SET dbs = CurrentDb()</code>	Obre la base de dades actual.
<code>SET rst = dbs.OpenRecordset(strSQL)</code>	Obre el cursor, instanciant-lo al resultat de la consulta <code>strSQL</code> .
<code>rst.MoveNext</code>	Avança el cursor una tupla.
<code>rst.Atr1</code>	Valor de <code>Atr1</code> en la posició actual del cursor.
<code>rst.AddNew</code>	Afegeix una nova tupla; cal emplenar-ne els camps i després fer un <code>rst.Update</code> .
<code>rst.Update</code>	Graba les modificacions.
<code>rst.Edit</code>	Permet modificar el contingut actual del cursor. Cal fer un <code>rst.Update</code> en acabar.
<code>rst!Camp=Valor</code>	Actualitza un atribut.
<code>rst.Delete</code>	Esborra la tupla actual.
<code>rst.EOF()</code>	Cert quan arriba al final o quan la consulta és buida.

9. SQL IMMERS (8)

Cursors: Exemple

Exemple: Afegir a la taula de Temes tots els llibres de Date classificats com a llibres de Bases de Dades.

- Es declaren totes les variables necessàries.
- Es declara un cursor que recupera tots els llibres de *Date*.
- S'obre el cursor per executar la consulta.
- Per cada llibre s'insereix un nou registre a la taula *Temes*.
- Es tanca el cursor.

```
Public Function ExempleCursors()  
Dim dbs As Database  
Dim rst As Recordset      'cursor  
Dim num As Integer  
Dim cad As Variant  
  
DoCmd.SetWarnings (False) 'desabilita missatges avís  
SET dbs = CurrentDb()     'instancia de la BD actual  
cad = "SELECT codi FROM Llibres WHERE autor='Date'"  
  
'obertura de la consulta  
Set rst = dbs.OpenRecordset(cad)  
  
'bucle per cada tupla que compleix la condició  
num = 0  
While (NOT (rst.EOF))  
    cad = "INSERT INTO TEMES (codi_llibre, tema)  
          VALUES (" & rst!Codi & ", 'Bases de Dades')"  
    DoCmd.RunSQL (cad)  
    rst.MoveNext  
    num = num + 1  
Wend  
MsgBox ("Insertats " & num & " temes")  
rst.Close  
dbs.Close  
End Function
```

9. SQL IMMERS (9)

Cursors: Actualitzacions

- En el recorregut d'un cursor es pot actualitzar o eliminar l'element que ocupa la posició actual, utilitzant les propietats *edit* i *delete* dels recordset.

Exemple: *Eliminar els socis de Barcelona, excepte els que es diuen David, que passaran a viure a Viladecans. Per cada soci a esborrar, primer s'eliminen els seus préstecs abans d'esborrar el soci.*

```
Public Function ExempleCursors2()  
Dim dbs As Database  
Dim rst As Recordset      'cursor  
Dim cad As Variant  
  
DoCmd.SetWarnings (False) 'desabilita missatges avís  
SET dbs = CurrentDb()     'instància de la BD actual  
' obtenir els socis de Barcelona  
cad = "SELECT * FROM Socis WHERE ciutat='Barcelona'"  
  
SET rst = dbs.OpenRecordset(cad)  
While (NOT (rst.EOF))  
    If (rst!nom = "David") Then  
        'actualitzar soci David  
        rst.Edit  
        rst!Ciutat = "Viladecans"  
        rst.Update  
    Else  
        ' esborrar abans registres relacionats de préstecs  
        cad = "DELETE * FROM Préstecs WHERE codi_soci = "  
            & rst!codi_soci  
        DoCmd.RunSQL (cad)  
        ' ara esborrar tupla del cursor  
        rst.Delete  
    End If  
    rst.MoveNext  
Wend  
rst.Close  
dbs.Close  
End Function
```


10. DEFINICIÓ DE LA BD (1)

Definició de dominis i taules

DOMINI:

```
CREATE DOMAIN nom domini  
AS tipus dades  
[DEFAULT expressió | NULL]  
[CHECK condició]
```

- Domini = tipus de dades + valor per defecte + restricció.
- La definició més simple d'un domini consisteix en considerar-lo equivalent a un tipus de dades bàsic.
- Es pot especificar un valor per defecte per tots els atributs que es defineixin a partir del domini.
- Es pot indicar una condició que han de complir els valors de tots els atributs definits a partir del domini.

TAULA:

```
CREATE TABLE nom taula  
(  
    definició d'atribut 1,  
    :  
    definició d'atribut n,  
  
    CONSTRAINT definició de restricció 1,  
    :  
    CONSTRAINT definició de restricció m,  
);
```

- Taula = conjunt d'atributs + conjunt de restriccions que han de complir els valors dels atributs.

10. DEFINICIÓ DE LA BD (2)

Definició d'atributs

nom atribut *tipus_dades* | *domini*
[DEFAULT *expressió* | NULL]
[*restriccions d'atribut*]

- *nom atribut* : Identificador simple (codi_llibre) o identificador compost ([codi llibre]).
- Atribut = tipus de dades + valor per defecte + restriccions
- Sempre s'ha d'especificar el tipus de dades (o domini) del que l'atribut agafa els valors.
- El valor per defecte és opcional (DEFAULT) i pot ser qualsevol expressió o NULL.
- També es poden indicar restriccions que afecten només al valor d'aquest atribut, tot i que és més habitual posar les restriccions d'atribut després de la definició de tots ells.

Restriccions d'atribut:

NOT NULL	L'atribut sempre ha de tenir valor. No es pot deixar en blanc.
UNIQUE	No pot haver dos tuples amb el mateix valor a l'atribut.
CHECK (<i>condició</i>)	Defineix una condició qualsevol que han de complir els valors de l'atribut (ORACLE, no ACCESS).

10. DEFINICIÓ DE LA BD (3)

Definició de restriccions per les taules

• Restricció de Clau Primària:

PRIMARY KEY (*nom atribut*, ...)

- Tota taula ha de tenir sempre una sola definició de clau primària.
- Els atributs de la clau primària han de complir les condicions NOT NULL i UNIQUE.

• Restricció de Clau Externa:

```
FOREIGN KEY (nom atribut, ...)
REFERENCES nom taula (nom atribut, ...)
    ON DELETE {NO ACTION | CASCADE |
               SET DEFAULT | SET NULL}
    ON UPDATE {NO ACTION | CASCADE |
               SET DEFAULT | SET NULL}
```

- Una taula pot tenir varies claus externes.
- Al definir la clau externa s'especifica a quina taula i a quins atributs fan referència.
- Els atributs referenciats han de ser la clau primària de la taula referenciada.
- Podem especificar quines accions s'han de fer si s'elimina o modifica un registre de la taula referencial.
- L'opció per defecte és NO ACTION (opció restringida).

• Restriccions d'usuari (dominis):

CHECK (*condició*)

- Defineix una condició qualsevol que han de complir els valors dels atributs de la taula.
- La condició pot ser qualsevol condició vàlida en SQL en què apareguin els atributs de la taula.

10. DEFINICIÓ DE LA BD (4)

Exemple

- *Definició completa de la taula exemplars i préstecs amb identificadors simples.*

```
CREATE TABLE Exemplars
(
  n°_exemplar      INTEGER,
  codi_llibre      INTEGER,
  any_edició       INTEGER,
  n°_edició        INTEGER,
  CONSTRAINT clau_exemplars PRIMARY KEY (n°_exemplar),
  CONSTRAINT exemplars_externa FOREIGN KEY (codi_llibre)
    REFERENCES Llibres (codi)
    ON DELETE NO ACTION
    ON UPDATE CASCADE
);

CREATE TABLE Préstecs
(
  codi_soci        INTEGER,
  n°_exemplar      INTEGER,
  data_préstec     DATE,
  data_límit       DATE,
  data_devolució   DATE,
  CONSTRAINT clau_préstecs PRIMARY KEY (n°_exemplar,
    data_préstec),
  CONSTRAINT préstecs_externa1 FOREIGN KEY (codi_soci)
    REFERENCES Socis(codi_soci)
  CONSTRAINT préstecs_externa2 FOREIGN KEY (n°_exemplar)
    REFERENCES Exemplars(n°_exemplar)
);
```

CONSTRAINT: Restricció associada a una clau primària o una d'externa. Es Permet canviar la clau primària d'una taula eliminant la constraint.

CONSULTA:

```
SELECT      n°_exemplar, codi_soci,
            data_devolució - data_préstec AS n°_dies
FROM        Préstecs
ORDER BY    codi_soci, n°_dies DESC;
```

10. DEFINICIÓ DE LA BD (5)

Exemple ACCESS

- *Definició completa de la taula exemplars i préstecs amb identificadors compostos.*

```
CREATE TABLE Exemplars
(
    [n° exemplar]    INTEGER,
    [codi llibre]    INTEGER,
    [any edició]     INTEGER,
    [n° edició]      INTEGER,
    CONSTRAINT [clau exemplars] PRIMARY KEY ([n° exemplar]),
    CONSTRAINT [exemplars externa] FOREIGN KEY ([codi
llibre])
        REFERENCES Llibres (codi)
        ON DELETE NO ACTION
        ON UPDATE CASCADE
);

CREATE TABLE Préstecs
(
    [codi soci]      INTEGER,
    [n° exemplar]    INTEGER,
    [data préstec]  DATE,
    [data límit]     DATE,
    [data devolució] DATE,
    CONSTRAINT [clau préstecs] PRIMARY KEY ([n° exemplar],
        [data préstec]),
    CONSTRAINT préstecs_externa1 FOREIGN KEY ([codi soci])
        REFERENCES Sòcis([codi soci])
    CONSTRAINT [préstecs externa2] FOREIGN KEY ([n° exemplar])
        REFERENCES Exemplars([n° exemplar])
);
```

CONSULTA:

```
SELECT    [n° exemplar], [codi soci],
          [data devolució]-[data préstec] AS [n° dies]
FROM      Préstecs
ORDER BY  [codi soci], [n° dies] DESC;
```

10. DEFINICIÓ DE LA BD (6)

Definició de vistes

- Les vistes són taules que no existeixen físicament, sinó que agafen les dades de les taules base (definides amb CREATE TABLE).
- Estan definides a partir d'una consulta sobre les taules base.
- Permeten particularitzar la visió que té cada usuari de la base de dades.

Exemple: Definició d'una vista que contingui els socis que viuen a Barcelona.

```
CREATE VIEW SocisBCN (nom, cognoms, telèfon, adreça)
AS SELECT nom, cognoms, telèfon, adreça
FROM Socis
WHERE ciutat = 'Barcelona';
```

- Podem accedir a la informació d'una vista com a qualsevol altra taula base.
- En l'exemple podríem consultar la vista *SocisBCN* com si fos una taula amb atributs *nom*, *cognoms*, *telèfon* i *adreça*.

Exemple: Definició d'una vista a partir d'una taula.

```
CREATE VIEW Socis
AS SELECT *
FROM Socis;
```

10. DEFINICIÓ DE LA BD (7)

Problema de l'actualització de vistes

- No totes les operacions d'actualització (INSERT, DELETE, UPDATE) es poden fer sobre totes les vistes.

Exemple: Si definim una vista per veure la data dels préstecs, nom i cognoms del soci que l'han fet i el llibre al que correspon

```
CREATE VIEW InfoPréstecs AS
  SELECT S.nom, S.cognoms, L.títol, P.data_préstec
  FROM Socis S, Llibres L, Préstecs P, Exemplars E
  WHERE P.codi_soci = S.codi AND
        P.n_exemplar = E.n_exemplar AND
        E.codi_llibre = L.codi;
```

com s'haurien de tractar aquestes operacions?:

```
INSERT INTO InfoPréstecs
VALUES ('Joan', 'Valls', 'Bases de Dades', #21/4/09#);
```

```
DELETE FROM InfoPréstecs
WHERE data_préstec < #1/1/07#;
```

- En SQL només es poden modificar les vistes en que la consulta que serveix per definir-les compleix aquestes condicions:
 - Té una sola taula en el FROM.
 - No incorpora subconsultes amb atributs relacionats.
 - No fa servir funcions de totals.
 - No fa servir GROUP BY.
 - Recupera tots els atributs definits com a NOT NULL.