UF2
MODIFICACIÓ DADES
I OPTIMITZACIÓ CONSULTES

Claudina Riaza

David Porti

VISTES

Són objectes de les BD (com les taules) que emmagatzemen la definició d'una consulta. No ténen dades pròpies.

- Faciliten la realització de consultes complexes.
- És el resultat d'una consulta SQL
- Ténen mateixa estructura que una taula
- Es una taula virtual(només es guarda la definició, no les dades)
- Es pot consultar com qualsevol taula.

Quan utilizar VISTES

Quan hi han taules amb informació que s'accedeix amb freqüencia:

- Informació derivada de la relació entre varies taules.
- Informació derivada de consultes complexes.

Com a mecanisme de seguretat

- Poden tenir atributs als quals es dessitja permetre accedir determinats usuaris.
  - Ex: dades publiques i privades d'una taula: Treballadors

INSTRUCCIONS PER LES VISTES

CREATE VIEW: Per crear la vista
ALTER VIEW: Per modificar la vista.
DROP VIEW: Per esborrar la vista.
SHOW CREATE VIEW: Per veure la estructura de la vista.

EXEMPLE DE CREACIÓ:

Create view <nameView> as <select sentence> [WITH CHECK OPTION];

CRIDA D'UNA VISTA:

Select <columns> from <nameView>;

VISTES: CREATE

- Exemple: Vista que mostra els treballadors de Barcelona

```
/*Ejemplo 1
Creamos una vista con los trabajadores de Barcelona*/

CREATE VIEW trabajadoresbar
AS SELECT * from trabajadores where ciudad='Barcelona';
Select * from trabajadoresbar;
```

dni	nombre	ciudad	antiguedad	salario	t_numdep
11112222A	Rojo Iglesias, Marta	Barcelona	12	60000.00	2
11112266C	Llamas Rocasolano, Isabel	Barcelona	13	28000.00	3
33112222A	Nualart Vives, Carlos	Barcelona	10	30000.00	4

VISTES: Modificació de camps

- Vista canviant els noms dels camps

```
/*Ejemplo 2 Creamos una vista con los trabajadores de Barcelona
Cambiamos los nombres de los campos*/

CREATE VIEW trabajadoresbar2 (dni_tra,nombre_tra,poblacion)
AS SELECT dni,nombre,ciudad from trabajadores where ciudad='Barcelona';

Select * from trabajadoresbar2;
```

dni_tra	nombre_tra	poblacion
11112222A	Rojo Iglesias, Marta	Barcelona
11112266C	Llamas Rocasolano, Isabel	Barcelona
33112222A	Nualart Vives, Carlos	Barcelona

VISTES: Multiples taules

- Podem utilizar dades de diverses taules o agrupacions.

```
/*Ejemplo 3
Creamos una vista con los datos de varias tablas*/
CREATE VIEW trabajadoresdep
AS SELECT T.*, d.nombredep from trabajadores T
INNER JOIN departamento D
where D.numdep= T.t_numdep
```

dni	nombre	ciudad	antiguedad	salario	t_numdep	nombredep
11112222A	Rojo Iglesias, Marta	Barcelona	12	60000.00	2	Recursos Humanos
11112222C	Gomez Corachán, Manuel	Madrid	15	22000.00	1	Contabilidad
11112233A	Perez Carrillo, Iván	Bilbao	5	48000.00	2	Recursos Humanos
11112244B	Torres Marqués, Fernando	Madrid	11	55000.00	2	Recursos Humanos
11112255B	Rubio Sánchez, María	Sevilla	4	36000.00	3	Informática

```
/*Ejemplo 4 Podemos hacer todo tipo de consultas sobre la vista*/
select nombredep, count(*) as total from trabajadoresdep
group by nombredep;
```

nombredep	total
Comercial	2
Contabilidad	1
Facturación	1
Informática	3
Recursos Humanos	3

VISTES: Altres operacions

- Per obtener información sobre les vistes: SHOW CREATE VIEW

  SHOW CREATE VIEW trabajadoresdep:

	View	Create View	character_set_client	collation_connection
▶	trabajadoresdep	CREATE VIEW "trabajadoresde...	utf8mb4	utf8mb4_general_ci

t "t"."dni" AS "dni","t"."nombre" AS "nombre","t"."ciudad" AS "ciudad","t"."antiguedad" AS "antiguedad","t"."salario" AS "salario","t"."t_numdep" AS "t_nu

- Per obtener eliminar una vista: DROP VIEW

  DROP VIEW trabajadoresdep:

Vistes actualitzables

- Las vistas se pueden actualizar (INSERT, UPDATE, DELETE). En realidad se actualiza el contenido de la tabla o tablas relacionadas.

- Las vistas actualizables han de cumplir algunas restricciones:

    - No usar cláusulas GROUP BY ni HAVING
    - Cualquier modificación (UPDATE, INSERT, DELETE) debe hacer referencia a las columnas de una única tabla.
        - En el caso de INNER JOIN las vistas se pueden actualizar siempre y cuando los campos afectados sean únicamente los de una de las tablas implicadas en el join
    - Los campos que se van a modificar no se pueden obtener con funciones de agregado: AVG, COUNT, SUM, MIN….

VISTES ACTUALITZABLES

- **WITH CHECK OPTION:** Ens permet actualitzar aquells registres que cumpleixen la condició WHERE.

```
/*Ejemplo 6 Creamos una vista con los trabajadores de Contabilidad
y hacemos que solo se puedan insertar los de contabilidad*/
CREATE VIEW trabajadorescont
AS SELECT * from trabajadores where t_numdep=1
WITH CHECK OPTION;

-- Añadimos un registro del departamento 2 y nos da error
INSERT INTO trabajadorescont VALUES ('33332222F','Olalla Burgos, Anna','Madrid',6, 45000, 2);
```

```
57  20:22:10  INSERT INTO trabajadorescont VALUES ('33332222F','Olalla Burgos, Anna','Madrid',6, 45000, 2)   Error Code: 1369. CHECK OPTION failed 'empresa.trabajadorescont'
```

```
-- Añadimos un registro del departamento 1 y funciona
INSERT INTO trabajadorescont VALUES ('33332222F','Olalla Burgos, Anna','Madrid',6, 45000, 1);
-- Comprobamos.
select * from trabajadorescont;
```

dni	nombre	ciudad	antiguedad	salario	t_numdep
11112222C	Gomez Corachán, Manuel	Madrid	15	22000.00	1
33332222F	Olalla Burgos, Anna	Madrid	6	45000.00	1

Vistes amb múltiples taules

Vista con la BD Escuela accediento a las 4 tablas al mismo tiempo.

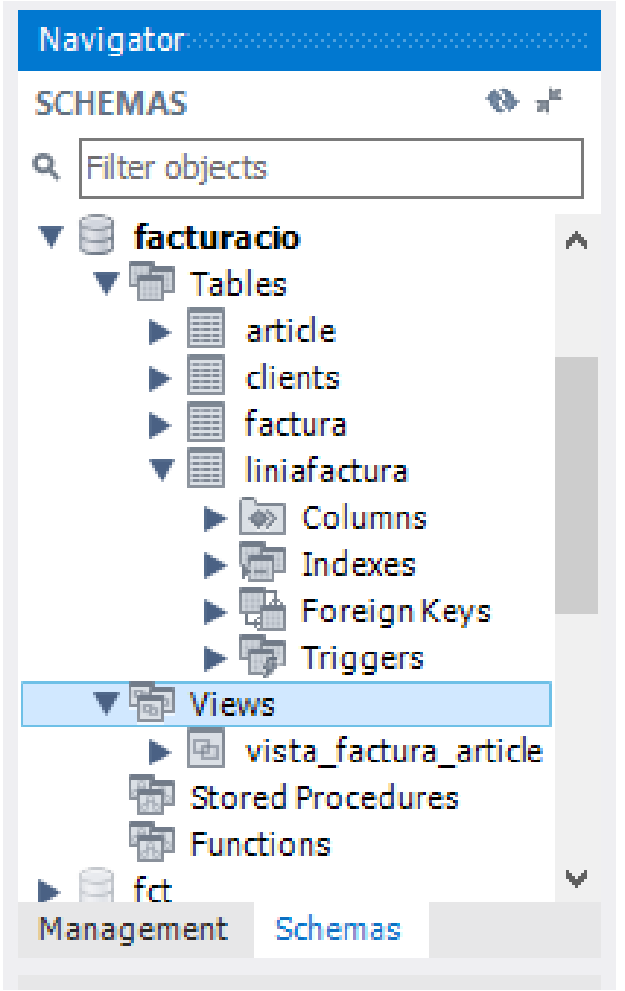
Simplifica las consultas posteriores (sobre todo desde programación)

```
/* Ejemplo escuela
Podemos juntar los datos de varias tablas, en este caso 4
para que luego las consultas sean más fáciles */
use escuela;
drop view alumnosnotas;
create view alumnosnotas as
select A.codigo, A.nombre, A.ciudad, A.edad, N.semestre, N.nota, M.nombre AS NombreModulo,
M.aula, P.dni, P.nombre AS 'Nombre Profesor' FROM Alumno A
INNER JOIN Notas N ON  A.codigo=N.alumnoCodigo
INNER JOIN  Modulo M ON M.codigo=N.moduloCodigo
INNER JOIN  Profesor P ON P.dni = M.profesorDNI;

select * from alumnosnotas;

/* Para hacer esta misma consulta sin la vista necesitamos tres tablas*/
select * from alumnosnotas
where nota<5 and NombreModulo='Base de datos';
```

codigo	nombre	ciudad	edad	semestre	nota	NombreModulo	aula	dni	Nombre Profesor
A0003	Torres Marqués, Fernando	Madrid	19	1617-1	4.20	Base de datos	10	11113333B	Garcia Gomez, Jose
A0005	Llamas Rocasolano, Isabel	Barcelona	25	1617-1	3.00	Base de datos	10	11113333B	Garcia Gomez, Jose
A0009	Villa Bueno, Sandra	Sevilla	32	1617-1	4.30	Base de datos	10	11113333B	Garcia Gomez, Jose

VISTES

- On s'emmagatzeman les vistes?

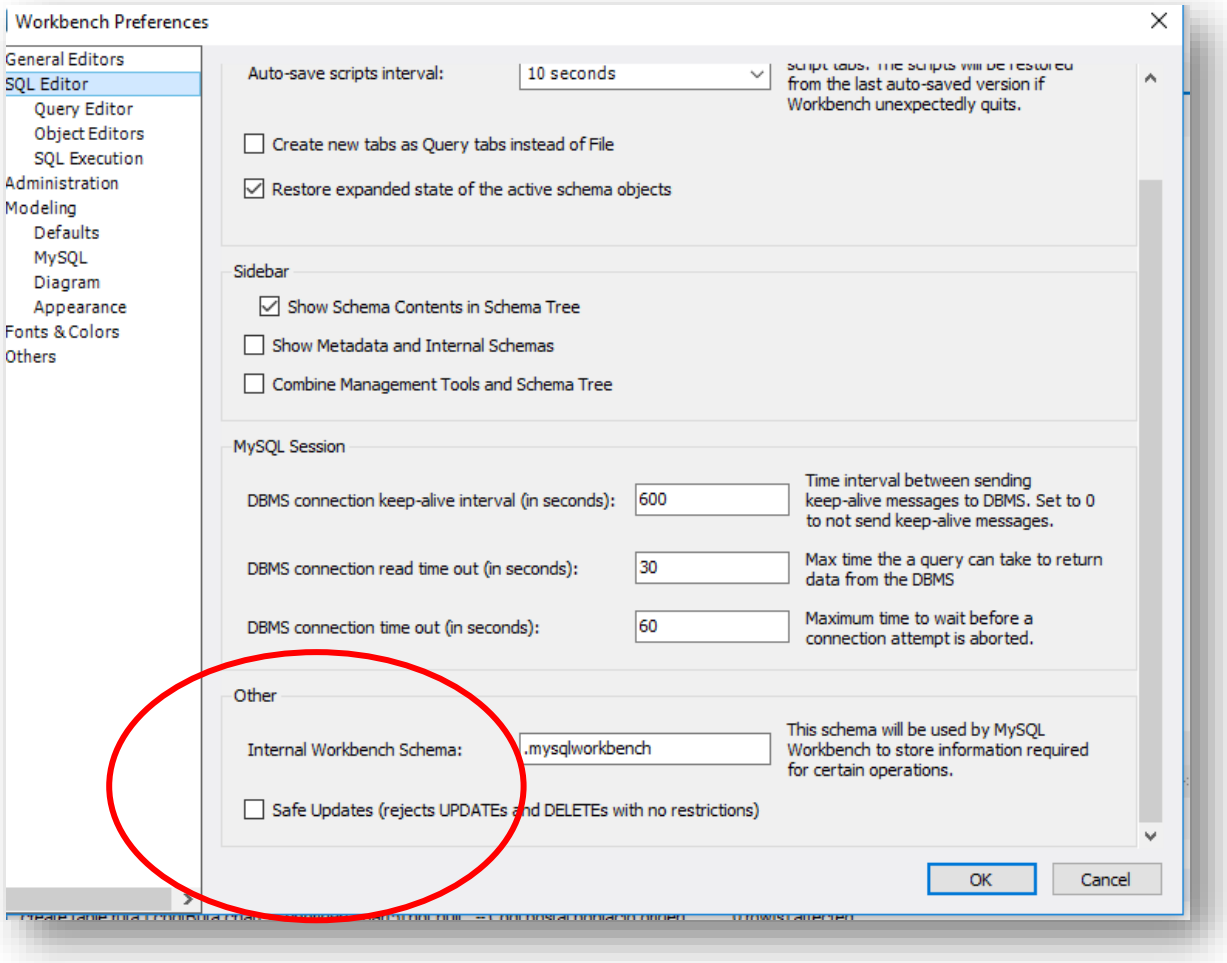


MODIFICACIONS DE DADES

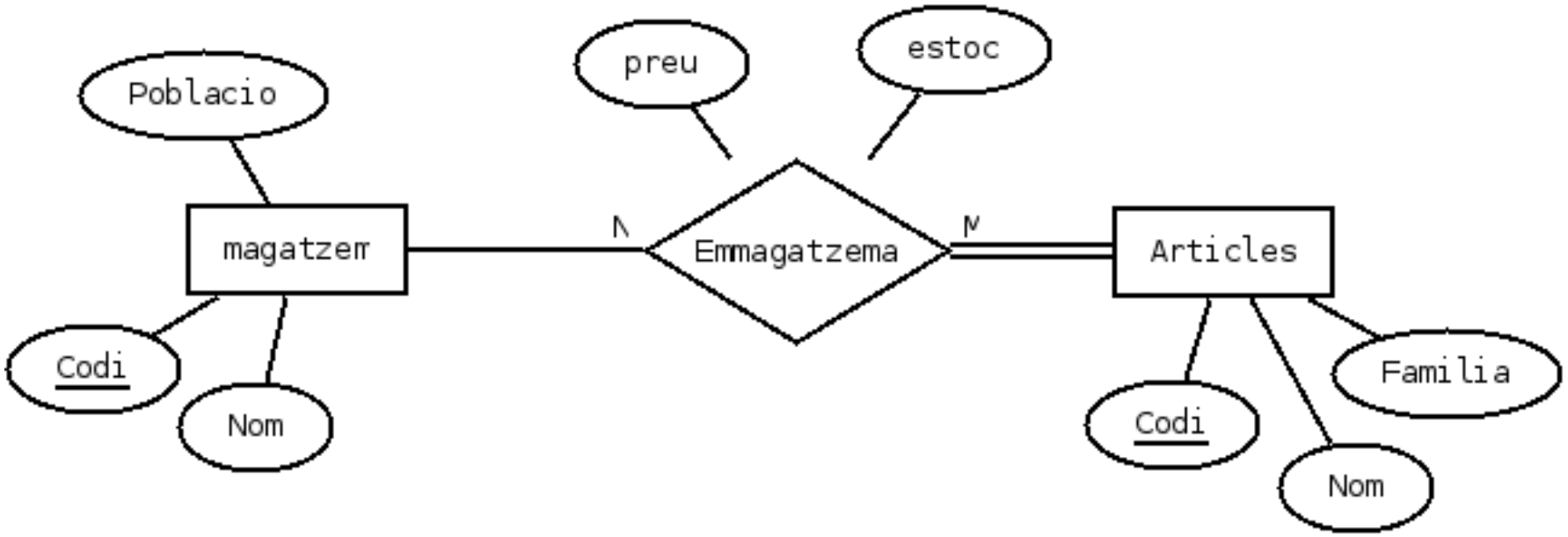
UPDATE / DELETE

DESACTIVA SAFE MODE

Per seguretat no es pot actualizar ni eliminar cap registre sense especificar en la sentencia WHERE cap primary key.



Esquema a fer servir com exemples:



CODI GENERAT

```
create schema if not exists Magatzems;
use Magatzems;

create table if not exists Article (
    codi char(5),
    nom varchar(40) not null,
    familia varchar(30),
    primary key (codi)
)engine=innodb;

create table if not exists Magatzem (
    codi char(5),
    nom varchar(30) not null,
    poblacio varchar(40) not null,
    primary key (codi)
) engine=innodb;
```

```
create table if not exists emmagatzema (
    articleCodi char(8),
    magatzemCodi char(8),
    preu decimal(6,2) not null,
    estoc decimal(6,2) not null,
    primary key(articleCodi,magatzemCodi),
    constraint fk_emmagatzema_article
        foreign key (articleCodi) references Article(codi)
        ON UPDATE CASCADE ON DELETE RESTRICT,
    constraint fk_emmagatzema_magatzem
        foreign key (magatzemCodi) references Magatzem(codi)
        ON UPDATE CASCADE ON DELETE RESTRICT
) engine=innodb;
```

ALERTA AMB EL COMPORTAMENT DE LES RELACIONS:
ON UPDATE – UP DELETE

Valors en les taules

```
insert into article values ("ART1","Lenovo","Portatils"),
        ("ART2","HP","Sobretaula"),    ("ART3","ACER","Sobretaula");

insert into magatzem values ("MAG1","GranBCN","Barcelona"),
        ("MAG2","BigGra","Granollers"),    ("MAG3","ParkBCN","Barcelona");

insert into emmagatzema
values("ART1","MAG1",100,30),("ART1","MAG2",200,30),
        ("ART2","MAG1",300,40),("ART3","MAG1",50,100);
```

UPDATE

Permet actualitzar valors ja introduïts sense la necessitat d'eliminar-los i tornar-los a introduir.

UPDATE table1 [,table2][,table3] ...

SET col_name1={expr1} [, col_name2={expr2}]

... [WHERE where_condition];

***NOTA**: Cal tenir en compte que si s'actualitza un camp que pertany a una relació o FK, hem de tenir en compte el comportament configurat*

ON UPDATE [cascade, restrict, set null]

EXEMPLES UPDATE

1. Incrementar tots els preus un 10%

```
UPDATE emmagatzema SET preu=preu*1.1;
```

2. Canviar el codi de matgatzem de MAG3 per MAG5.

```
UPDATE magatzem
SET codi='MAG3' WHERE codi='MAG5';
```

3. Decrementar el preu un 5% dels articles que el seu estoc sigui superior a 30

```
UPDATE emmagatzema SET preu=preu/1.05 WHERE estoc>30;
```

EXEMPLES UPDATE

4. Incrementar l'estoc amb 5 unitats de tots els productes de la família Portàtils

```
UPDATE emmagatzema E
        inner join Article A on E.articleCodi = A.codi codi
SET  estoc = estoc + 5
WHERE a.familia = 'Portatils'
```

EXEMPLES UPDATE

5. Decrementar amb 3 unitats tots els productes de la familia Sobretaula dels magatzems ubicats a Barcelona.

```
UPDATE Magatzem M
    inner join Emmagatzema E on M.codi = E.magatzemCodi
    inner join Article A on E.articleCodi = A.codi
SET E.estoc = E.estoc-3
WHERE A.familia="Sobretaula" and M.poblacio="Barcelona";
```

EXEMPLES UPDATE

6. Afegir una ‘z’ davant de tots els articles que encara no estan en cap magatzem.

```
UPDATE emmagatzema e left join article a
        on e.articleCodi = a.codi
SET     nom = concat(‘Z’,nom)
WHERE a.familia = 'Portatils’ ;
```

```
-- Ejemplo3 update
-- Añadimos un campo llamado dietas a la tabla trabajadores
-- Ponemos un 2% de su salario
-- A los trabajadores de madrid con una antiguedad superior a 6 años.

select * from trabajadores
where ciudad="Madrid" and antiguedad>6;

alter table trabajadores
add dietas decimal (8,2);

desc trabajadores;

UPDATE trabajadores SET dietas=salario*0.02
where ciudad="Madrid" and antiguedad>6;
```

dni	nombre	ciudad	antiguedad	salario	t_numdep
11112222C	Gomez Corachán, Manuel	Madrid	15	22000.00	1
11112244B	Torres Marqués, Fernando	Madrid	11	60500.00	2

dni	nombre	ciudad	antiguedad	salario	t_numdep	dietas
11112222C	Gomez Corachán, Manuel	Madrid	15	22000.00	1	440.00
11112244B	Torres Marqués, Fernando	Madrid	11	60500.00	2	1210.00

ON UPDATE

En una relació, podem determinar qué pasa quan s'actualitza el valor de la PK el qual fa referencia.

DEMOSTRACIÓ:

- CASCADE
- RESTRICT
- SET NULL

CAL DETERMINAR QUE PASA EN CADA CAS

DELETE

Serveix per eliminar aquells registres o files d'una taula.

DELETE table1. FROM table1 [,table2][,table3] ...*

[WHERE where_condition]

***NOTA**: Cal tenir en compte que si s'actualitza un camp que pertany a una relació o FK, hem de tenir en compte el comportament configurat*

ON DELETE [cascade, restrict]

Exemples DELETE

1. Eliminar l'article amb codi 34.

```
DELETE article.* from article where codi = '34';
```

2. Eliminar l'article que ténen un estoc inferior a 5.

```
DELETE A.* from article A inner join emmagatzema E
        on A.codi = E.articleCodi where E.estoc <5;
```

3. Eliminar tots els articles procedents de magatzems de Granollers.

```
DELETE A.* from article A, emmagatzema E, magatzem M
        where A.codi = E.articleCodi and E.magatzemCodi = M.codi
        and M.poblacio = 'Granollers';
```

Exemples DELETE

4. Eliminar aquells magatzems que no ténen productes

```
DELETE M.* from magatzem M left join estoc E
    on M.codi = E.magatzemCodi
    where E.magatzemCodi is null;
```

5. Inventat una consulta d'eleminació.

TRUNCATE

Permet eliminar tota la información d'una taula d'una forma més rápida pel planificador de consultes.

TRUNCATE TABLE nomDeLaTaula

```
-- Ejemplo Truncate
TRUNCATE TABLE trabajadores;
SELECT * from trabajadores;
```

	dni	nombre	ciudad	antiquedad	salario	t_numdep	dietas

OPTIMITZACIÓ DE CONSULTES

Què pot afectar al rendiment d'una consulta?

- El tamany de les taules consultades (Núm de files)
- El nombre de taules combinades
- El nivell de selectivitat d'una consulta (WHERE...)
- Quantitat de files a ordenar (ORDER BY – GROUP BY
- Existència d'índex o no
- Qüestions físiques.
  - Rendiment de l'equip
  - Rendiment de la connexió

PASOS QUE ES PRODUEIXEN EN UNA SQL

PARSE: Analitzador de sintaxi.

OPTIMIZE: Optimitzar operacions i generar un plà d'execució. (com ha d'executar la consulta (Hi ha subselects, vistes? Indexos?)

EXECUTE: Retornar el resultat segons el plà d'execució.

Base de datos no indexada

Base de datos indexada

INDEX

Permeten optimitzar les consultes ordenant la columna que genera el índex de forma alfabètica. És un PUNTER a una fila determinada.

- Els inserts, updates i deletes siguin algo més lents. (60% són selects)
- La primary key es considera com a INDEX.
- Per cada índex que es crea, es genera una nova taula ordenada per l'index escollit. No s'ha d'abusar dels índex. Ocupen espai de memòria.
- Exemple:

  Sabem que sempre demanarem per nom, caldria crear un índex en el camp nom

```
create table client(
    codi char(2) primary key,
    nom varchar(20)
)engine=innodb;
```

AVANTATGES UTILITZA INDEX

- Disminuir el temps d'execució en consultes amb ordenació (ORDER BY) o agrupament (GROUP BY).
- Si una consulta utilitza una condició simple on la columna és la condició que està indexada, les files seran recuperades directament a partir del índex, sense consultar la taula.
- Disminució dràstica del temps d'execució de consultes de taules de gran tamany. Sinó, NO.

Per què no utilitzem índex a tot?

La creació d'índex també té efectes negatius.

- Els inserts, updates i deletes que es realitzin en alguna taula que tingui índexs, augmentaran el temps d'execució.
- Els índex, generen noves taules i aquestes s'han d'emmagatzemar en algun lloc.

  Per tant, ocupen espai al disc.

CONCLUSIÓ

- En BBDD, sol predominar consultes SELECT de tipus WHERE davant INSERTS, UPDATES o DELETES.
- Les instruccions SELECT comprometen la espera de l'usuari, les altres no.
- Amb INDEX, les instruccions Inserts, Updates o Deletes disminueixen el temps d'execució però no comprometen la usabilitat del sistema.

En general: una bona configuració dels Índex en una base de dades, sense abusar, és essencial per oferir un bon servei.

OPTIMITZACIÓ CONSULTES

- Importancia: (accés seqüencial)

  0,001 seg x 20.000 usuaris = 20 seg.

  0,001 seg x 100.000 usuaris = 100 seg. → 1min i 40 seg.

Imaginem un servidor que solicita a SGBD 30 consultes per segon...

Com veure el que està passant: EXPLAIN

Mostra el plà d'execució de la consulta (WorkBench no es reflexa bé)

Exemple Necessitat Índex

Veiem que la majoria de consultes són per ciutat en la clàusula WHERE.

Primary key

CODI	Nom	…..	Ciutat
COD 1	Maria		Barcelona
COD 2	Marcos		Girona
COD 3	Agustín		Madrid
COD 4	Begoña		Barcelona
COD 5	David		Barcelona
COD 6	Santi		Tarragona
COD 7	Ignasi		Reus
…	…		…
COD 1498	Oriol		Tarragona
COD 1499	Pablo		Reus
COD 1500	Angel		Viladecans

Index

Primary key

Ciutat	CODI	Nom	…..
Barcelona	COD 1	Maria	
Barcelona	COD 4	Begoña	
Barcelona	COD 5	David	
Girona	COD 2	Marcos	
Madrid	COD 3	Agustín	
Reus	COD 7	Ignasi	
Reus	COD 1499	Pablo	
…	…	…	
Tarragona	COD 6	Santi	
Tarragona	COD 1498	Oriol	
Viladecans	COD 1500	Angel	

Amb o sense Index

Tenint en compte que hi han 1500 reg. i busquem el registres de la ciutat de “Barcelona”. Hi han 25. Sabent que cada cerca triga 0.001seg. Quan temps triga en retornar la información:

- Sense INDEX: 1500 reg. (accés seq.) X 0.001 seg/reg = 1,5 seg.
- Amb Index: 25 reg (accés directe) x 0.001 seg/reg = 0,025 seg.

Trigaria 60 cops més.

Tipus d'index

- Per escollir un índex, es important analitzar en les sentencies SELECT quines són les columnes que més utilitzem en la part del WHERE.

- Tipus d'índex en MySql

  - **Primary Key:** Únic i no permet valors NULL
  - **Unique:** valors únics, i permet valors NULL
  - **Index:** Permet valors duplicats i valors NULL

Index: Sintaxi

La sintaxi per crear índex, és la següent:

CREATE INDEX <indexName> ON <tableName> (columnName);

CREATE INDEX IndexClient on client (nom);

CREATE UNIQUE INDEX<indexName> ON <tableName> (columnName);

Eliminar Index

- Per eliminar un índex d'una taula:

  DROP PRIMARY KEY

  DROP INDEX <indexName> on <tableName>

- Per mostrar els index:

  SHOW KEYS from <tableName>