

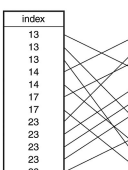
## 21726 - Bases de Dades II

### 6. Indexació

1. Introducció
2. Tipus d'índexs
3. Anàlisi i optimització de consultes
4. Conveniència dels índexs
5. Motors de BD a MySQL

### 1.- Introducció

- INSERT, emmagatzemament seqüencial
- DELETE, deixa registres buits
- Registres buits => endarreriment SELECT
- **Indexació**: mecanisme per accelerar velocitat d'accés a les dades



index	company_num	ad_num	hit_fee
13	14	48	0.01
13	23	49	0.02
13	17	52	0.01
13	13	55	0.03
14	23	62	0.02
14	23	63	0.01
17	23	64	0.02
17	13	77	0.03
23	23	99	0.03
23	14	101	0.01
23	13	102	0.01
23	17	119	0.02

### 1.- Introducció

- **Fitxer d'índexs**: Estructura on s'emmagatzemen els índexs definits
  - Solen tenir una grandària molt menor que el fitxer propi de dades
- **Clau de cerca**: Camp o camps de la taula que són usats per fer la cerca
- Principalment s'usen dos criteris:
  - Índexs ordenats
  - Índexs hash

### 1.- Introducció

#### Criteris d'avaluació

- Temps d'accés
- Temps d'inserció
- Temps d'esborrament
- Espai de sobrecàrrega

## 1.- Introducció

### Sintaxi simplificada:

```
CREATE [UNIQUE|FULLTEXT|SPATIAL]
INDEX index_name
[index_type]
ON tbl_name (index_col_name,...)

index_col_name:
col_name [(length)] [ASC | DESC]

index_type:
USING {BTREE | HASH}
```

## 1.- Introducció

### Borrat dels índexs

```
ALTER TABLE taula DROP PRIMARY KEY
ALTER TABLE taula DROP INDEX nom_index
DROP INDEX nom_index ON taula
```

### Visualització dels índexs

```
SHOW INDEX FROM taula
```

## 6.- Indexació

1. Introducció
2. Tipus d'índexs
3. Anàlisi i optimització de consultes
4. Conveniència dels índexs
5. Motors de BD a MySQL

## 2.- Tipus d'índexs

- Índexs ordenats
  - Primari
  - Ordinari
  - Exclussiu
  - Text
- Per estructura:
  - B-Tree
  - Hash

## 2.- Tipus d'índexs

### Índex ordenat

- El fitxer d'índex està ordenat per clau de cerca
- **Tipus**
  - Índex primari
  - Índex ordinari
  - Índex exclusiu
  - Índex de text

## 2.- Tipus d'índexs

### Índex primari

- Es crea, automàticament, sobre la clau primària
- Procediment de creació: PRIMARY KEY
- Permet claus dobles
- Organització física:
  - Índex: ordenat per clau
  - Taula: no necessàriament ordenada

•Primari  
•Ordinari  
•Exclusiu  
•Text

## 2.- Tipus d'índexs

### Índex primari

Índex	ID	Taula	ID	Nom	Població
1		→	1	Toni	Palma
3		→	4	Joan	Palma
4		→	7	Aina	Inca
7		→	3	Mateu	Palma

•Primari  
•Ordinari  
•Exclusiu  
•Text

## 2.- Tipus d'índexs

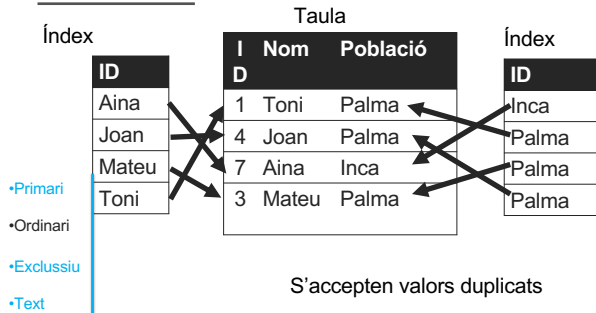
### Índex ordinari

- Índex que es crea sobre atributs no clau i no necessàriament únics
- Procediment de creació: INDEX
- Claus dobles -> entrada conjunta a ÍNDEX
- Organització física:
  - Índex: ordenat
  - Taula: no necessàriament ordenada

•Primari  
•Ordinari  
•Exclusiu  
•Text

## 2.- Tipus d'índexs

### Índex ordinari



## 2.- Tipus d'índexs

### Índex ordinari

```
CREATE TABLE taula (
  camp1 tipus, camp2 tipus, camp3 tipus, ...
  INDEX [nom_index] (nom_camp1, [nom_camp2, etc])
)
```

```
ALTER TABLE taula ADD
INDEX [nom_index] (nom_camp1, [nom_camp2, etc])
```

```
CREATE INDEX nombre-de-indice
ON taula (nom_camp1, [nom_camp2, etc])
```

- Primari
- Ordinari
- Exclusiu
- Text

## 2.- Tipus d'índexs

### Índex exclusiu

- Es crea sobre atributs no clau, necessàriament únics
- Procediment de creació: UNIQUE
- Claus dobles acceptades
- Organització física:
  - Índex: ordenat
  - Taula: no necessàriament ordenada

- Primari
- Ordinari
- Exclusiu
- Text

## 2.- Tipus d'índexs

### Índex exclusiu

```
CREATE TABLE taula (
  camp1 tipus, camp2 tipus, camp3 tipus, ...
  UNIQUE [nom_index] (nom_camp1, [nom_camp2, etc])
)
```

```
ALTER TABLE taula ADD
UNIQUE [nom_index] (nom_camp1, [nom_camp2, etc])
```

```
CREATE UNIQUE INDEX nombre-de-indice
ON taula (nom_camp1, [nom_camp2, etc])
```

- Primari
- Ordinari
- Exclusiu
- Text

## 2.- Tipus d'índexs

### Índex de text

- Són els propis de CHAR, VARCHAR, TEXT
- Procediment de creació:
  - FULLTEXT

```
CREATE TABLE taula (
  camp1 tipus, camp2 tipus, camp3 tipus, ...
  FULLTEXT [nom_index] (nom_camp1, [nom_camp2, etc])
)
```

•Primari

•Ordinari

•Exclusiu

•Text

```
ALTER TABLE taula ADD
FULLTEXT [nom_index] (nom_camp1, [nom_camp2, etc])
```

```
CREATE FULLTEXT INDEX nombre-de-indice
ON taula (nom_camp1, [nom_camp2, etc])
```

## 2.- Tipus d'índexs

### Índex de text

- Utilització

```
SELECT * FROM ...
WHERE MATCH(nom_camp) AGAINST ('text')
```

- Renou si apareix a més del 50% de valors
- Exclou paraules amb <4 caràcters
- Eliminació de renou

•Primari

•Ordinari

•Exclusiu

•Text

```
SELECT *, (MATCH(nom_camp) AGAINST ('text'))
FROM ...
```

## 2.- Tipus d'índexs

### Índex de text

```
SELECT * FROM llistacp
WHERE MATCH (poblacio) AGAINST ('PALMA')

SELECT * FROM llistacp
WHERE MATCH (poblacio) AGAINST ('PORT DE')

SELECT *, MATCH (poblacio) AGAINST ('PORT DE')
FROM llistacp

SELECT *, MATCH (poblacio) AGAINST ('PORT DE') AS prox
FROM llistacp ORDER BY proximitat DESC
```

•Primari

•Ordinari

•Exclusiu

•Text

## 2.- Tipus d'índexs

### Índex de part d'un camp

- Indexen a partir de part d'un camp
- Columnes de tipus:
  - VARCHAR
  - CHAR
  - BLOB
  - TEXT

•Primari

•Ordinari

•Exclusiu

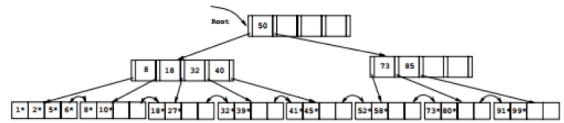
•Text

```
ALTER TABLE taula
ADD INDEX (nom_camp (num_caracters) )
```

## 2.- Tipus d'índexs

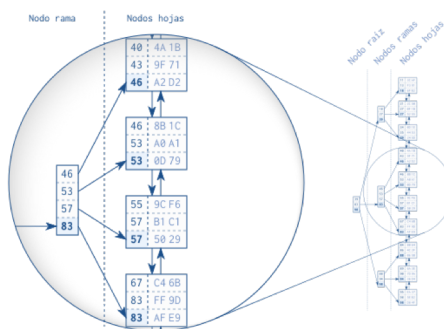
- Índexs ordenats
  - Primari
  - Ordinari
  - Exclusiu
  - Text
- Per estructura:
  - B-Tree
  - Hash

## 2.- Tipus d'índexs



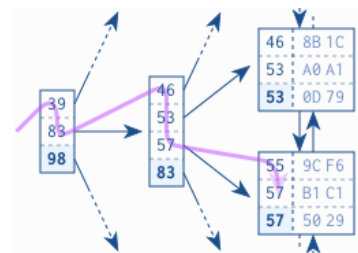
B-Tree  
Hash

## 2.- Tipus d'índexs



B-Tree  
Hash

## 2.- Tipus d'índexs



B-Tree  
Hash

<http://www.cs.usfca.edu/~galles/visualization/BPlusTree.html>

## 2.- Tipus d'índexs

Una consulta per índex pot requerir 3 etapes:

1. Recorregut de l'arbre. **Ràpid**
2. Resseguir la cadena de nodes fulla. **Lent**
3. Accés a les files de la taula apuntades per les fulles escollides. **Molt lent**

B-Tree  
Hash

## 2.- Tipus d'índexs

B-Tree  
Hash

## 2.- Tipus d'índexs

Hash estàtic

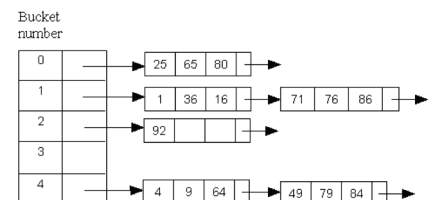
- Bloc: unitat d'emmagatzematge amb 1 o més registres
- En una organització hash obtenim el bloc d'un registre directament del seu valor de clau de cerca utilitzant una funció hash
- h és una funció de hash que obté el bloc que conté tots els valors de claus de cerca
- La funció s'utilitza per localitzar els registres d'accés, inserció i eliminació

B-Tree  
Hash

## 2.- Tipus d'índexs

Hash estàtic

- Els registres amb diferents valors de claus de cerca poden ser assignats al mateix hash, de manera que el valor particular s'ha de cercar seqüencialment per la branca.



B-Tree  
Hash

## 2.- Tipus d'índexs

### Hash estàtic

- Els valors de claus de cerca s'assignen a registres que al llarg del temps creixen o s'encongeixen
  - Si creix, el rendiment es degrada
  - Si es redueix, es perd espai d'emmagatzemament
- Solució: revisió periòdica amb una nova funció hash.
- Car, interromp les operacions normals.
- Millor solució: permetre que el nombre de registres que es modifiqui dinàmicament. Cercar solucions amb hash dinàmic

B-Tree

Hash

## 6.- Indexació

1. Introducció
2. Tipus d'índexs
3. Anàlisi i optimització de consultes
4. Conveniència dels índexs
5. Motors de BD a MySQL

## 3.- Anàlisi i optimització de consultes

Un índex té sentit quan s'empra habitualment a condicions dintre d'un WHERE

- Diferència entre SELECT emprant INDEX o sense INDEX (exemples)
- No crear índexs innecessaris (sobrecàrrega al manteniment)
- EXPLAIN: mostra com es processa un SELECT

**EXPLAIN SELECT** atributs

**FROM** taula  
**WHERE** condició

## 3.- Anàlisi i optimització de consultes

### EXPLAIN (principals arguments)

- Table: taula referenciada
- Possible-Keys: recomanacions sobre índexs que es podrien aplicar
- Key: índex emprat. Si NULL => no s'utilitza cap índex
- Key-len: longitud de l'índex emprat (ha de ser petit)
- Ref: columna de l'índex emprada
- Rows: total de files a examinar per aconseguir els resultats del SELECT



### 3.- Anàlisi i optimització de consultes

#### EXPLAIN (principals arguments)

- **Type:** tipus de combinació que s'empra:
  - **Eq\_ref, Const:** Examina l'índex per arribar a una única fila de la taula (un únic accés a la taula). S'empra quan l'índex és primari o UNIQUE.
  - **Ref, Range:** Fa un recorregut de l'arbre i de les fulles fins que troba totes les entrades desitjades de l'arbre. S'empra quan una clau no és única ni primària.
  - **Index:** S'examina l'índex complet.
  - **ALL:** Examina tota la taula.

### 3.- Anàlisi i optimització de consultes

#### EXPLAIN (principals arguments)

- **Extra:**
  - **Using Index:** No produeix accés a la taula perquè l'índex té tota la informació necessària.
  - **Filesort:** Fa una ordenació explícita.

### 3.- Anàlisi i optimització de consultes

```
EXPLAIN SELECT * FROM
  personesind INNER JOIN llistacp
    ON llistacp.cp = personesind.cp
 WHERE apellido1='MORENO'
```

- *apellido1* i *cp* són índexs. Resultat 15 files de taula analitzades

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	personesind	ref	CP,apellido1	apellido1	26	const	15	Using where
1	SIMPLE	llistacp	eq_ref	PRIMARY	PRIMARY	5	bt2indexos.personesind.CP	1	

### 3.- Anàlisi i optimització de consultes

```
EXPLAIN SELECT * FROM
  personesind INNER JOIN llistacp
    ON llistacp.cp = personesind.cp
 WHERE apellido2='MORENO'
```

- *cp* és un índex. Resultat 2976 files de taula analitzades i 24 entrades d'índex analitzades

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	llistacp	ALL	PRIMARY	NULL	NULL	NULL	124	
1	SIMPLE	personesind	ref	CP	CP	6	bt2indexos.llistacp.CP	24	Using where

### 3.- Anàlisi i optimització de consultes

INSERT INTO taula VALUES (reg1), (reg2), (reg3)

És més ràpid que

INSERT INTO taula VALUES (reg1);  
INSERT INTO taula VALUES (reg2);  
INSERT INTO taula VALUES (reg3);

Per???

### 6.- Indexació

1. Introducció
2. Tipus d'índexs
3. Anàlisi i optimització de consultes
4. Conveniència dels índexs
5. Motors de BD a MySQL

### 4.- Conveniència dels índexs

És convenient definir un índex:

1. Volem garantir la unicitat d'alguns atributs (cal comprovar que l'etiqueta UNIC a un atribut genera un índex).
2. Per forçar l'ordenació de files dins una taula. (Si el SGBD ho té, un *cluster index* força l'ordenació física de la taula segons la clau d'indexació).
3. Per mantenir la integritat referencial si no ho fa el sistema.
4. Optimitzar els SELECT més importants, tenint en compte: WHERE, JOIN, ORDER BY ...

### 4.- Conveniència dels índexs

No es convenient definir un índex:

1. Si la taula té poques files. O, més ben dit, si les files de la taula ocupen poques pàgines.
2. Si discrimina molt poc (el nombre de valors diferents que pren es petit).
3. Si els programes que l'utilitzen ho fan amb predicats no indexables (per exemple NOT EQUAL).
4. Si les aplicacions fan tractaments massius de la taula.

## 4.- Conveniència dels índexs

### Altres consideracions:

1. Índex sobre columnes molt actualitzades.
2. Índexs ascendents o descendents: vigilar l'ordenació segons la seva utilització.
3. Índexs que no es fan servir. Però pensar amb els d'ús futur, segons requisits.
4. El cost de manteniment dels fitxers d'índex és directament proporcional al seu nombre: cal eliminar els superflus i que no son estrictament necessaris.

[What every developer should know about SQL performance](#)

## 6.- Indexació

1. Introducció
2. Tipus d'índexs
3. Anàlisi i optimització de consultes
4. Conveniència dels índexs
5. Motors de BD a MySQL

## 5.- Motors de BD a MySQL

### Motor de base de dades:

És el conjunt d'eines que permeten comunicar-se amb la base de dades física, executen els processos sobre taules i mantenen la integritat de dades. És qui interpreta i executa les consultes, manté els índexs, etc.

MySQL n'empra dos:



## 5.- Motors de BD a MySQL

### MyISAM:

- Basat en el sistema ISAM de IBM.
- Simple i optimitzat per a moltes lectures a taules grans.
- No suporta transaccions (bloquejos) ni manté la integritat referencial → NO ACID.

### InnoDB:

- ACID.
- Alt rendiment per gestionar múltiples usuaris simultanis.

## 5.- Motors de BD a MySQL

### Exercici:

Esbrina les principals característiques i

¿Quan n'empraries un o altre?