# Motors d'emmagatzematge i transaccions en MySQL





**Autor**: Juan José Lopez

# Introducció

- **Què és un motor d'emmagatzematge en un sistema gestor de bases de dades?**

> **Un motor d'emmagatzematge és una part essencial d'un SGDB ja que s'encarrega de crear, recuperar, actualitzar i esborrar les dades d'una base de dades**



**Autor**: Juan José Lopez

# Motors d'emmagatzematge

- **Per què són importants els motors d'emmagatzematge?**
  - Les dades en MySQL poden ser emmagatzemades de diverses formes i amb diferents tècniques que ens aporten diverses funcionalitats.
  - Aquí és on trobem la importància dels motors d'emmagatzematge, ja que, depenent del motor que triem obtindrem diferents avantatges i dependrà de nosaltres saber treure'ls partit d'acord a les nostres necessitats.



**Autor**: Juan José Lopez

# Motors d'emmagatzematge

- MySQL ens proveïx per defecte d'una sèrie de motors que podem usar fàcilment (a part d'altres desenvolupats per terceres parts que també podem instal·lar), de fet aquesta flexibilitat és una de les raons que han jugat a favor de la popularitat de MySQL

- **Com saber quins motors estan disponibles?**
  - És molt senzill, n'hi ha prou amb escriure en una consola MySQL la següent sentència:

    **show engines;**



**Autor**: Juan José Lopez

# Motors d'emmagatzematge

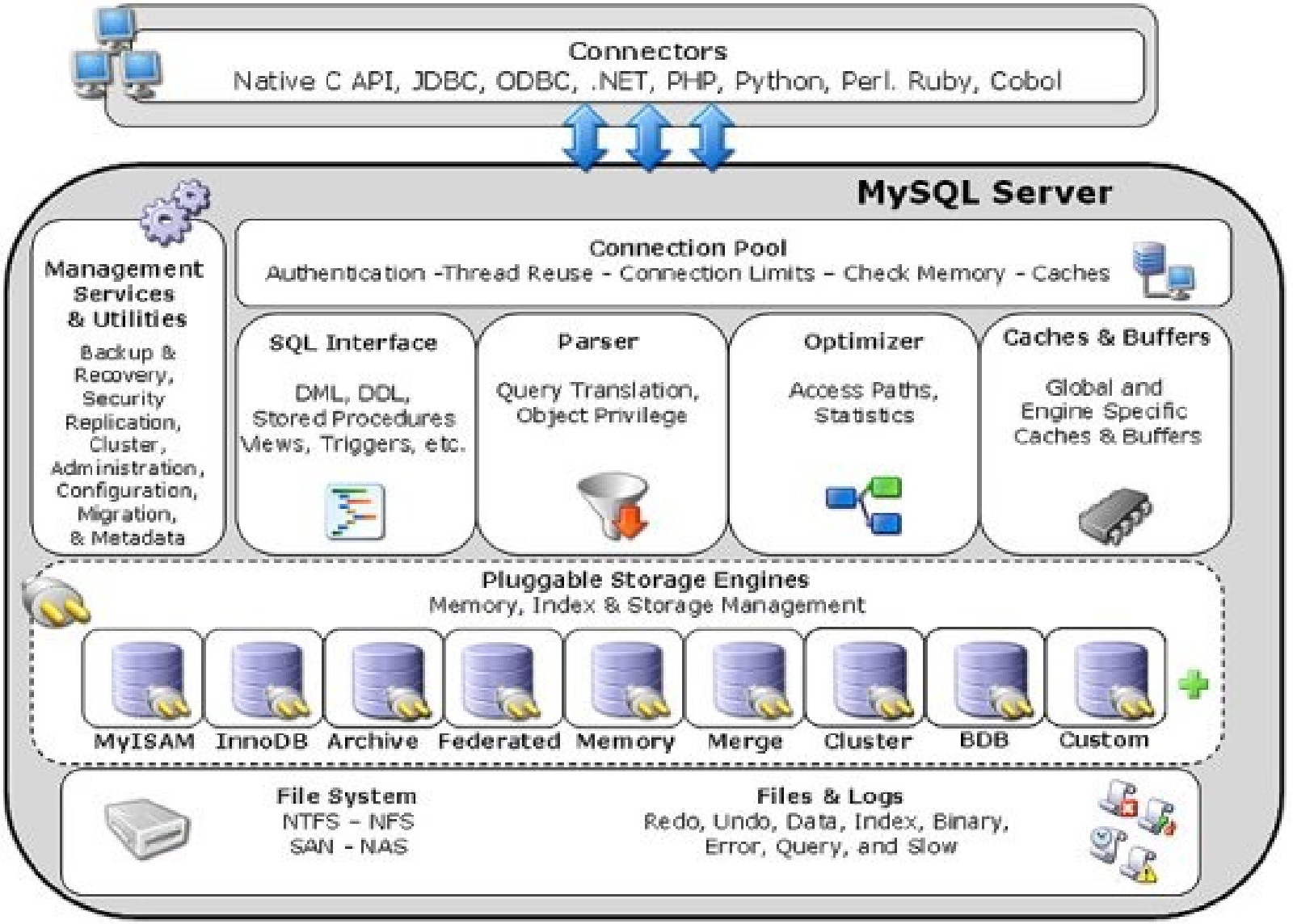
- **I obtindrem això:**

```
mysql> show engines;
+------------+----------+----------------------------------------------------------------+
| Engine     | Support  | Comment                                                        |
+------------+----------+----------------------------------------------------------------+
| MyISAM     | DEFAULT  | Default engine as of MySQL 3.23 with great performance         |
| MEMORY     | YES      | Hash based, stored in memory, useful for temporary tables      |
| InnoDB     | YES      | Supports transactions, row-level locking, and foreign keys     |
| BerkeleyDB | NO       | Supports transactions and page-level locking                   |
| BLACKHOLE  | YES      | /dev/null storage engine (anything you write to it disappears) |
| EXAMPLE    | NO       | Example storage engine                                         |
| ARCHIVE    | YES      | Archive storage engine                                         |
| CSV        | YES      | CSV storage engine                                             |
| ndbcluster | DISABLED | Clustered, fault-tolerant, memory-based tables                 |
| FEDERATED  | DISABLED | Federated MySQL storage engine                                 |
| MRG_MYISAM | YES      | Collection of identical MyISAM tables                          |
| ISAM       | NO       | Obsolete storage engine                                        |
+------------+----------+----------------------------------------------------------------+
12 rows in set (0.00 sec)
```



**Autor**: Juan José Lopez

# Esquema MySQL








**Autor**: Juan José Lopez

# Motors i diferències

## Les comparatives entre motors es solen fer basant-se en quatre funcionalitats clau:

- **Tipus de dades:**
  - Encara que la majoria són comuns hi ha alguns específics que poden ser decisius sota determinades circumstàncies.
- **Bloqueig de dades:**
  - La forma en la qual el motor protegeix una dada que està essent modificada per evitar problemes d'accés concorrent a les dades i mantenir la integritat referencial.



**Autor**: Juan José Lopez

- **Indexat:**
  - Les diferents tècniques d'indexat poden influir dràsticament és el rendiment d'una base de dades.
- **Transaccions:**
  - Les transaccions doten de fiabilitat a les dades mentre es realitzen operacions, permeten utilitzar les dades però només permeten guardar-les quan es comprova que condicions que poguessin requerir-se s'han complert.



**Autor**: Juan José Lopez

# ACID

- **Es denomina ACID a la propietat d'una base de dades per a realitzar transaccions segures.**
  - Així doncs **ACID** defineix a un sistema de gestió de bases de dades que pot realitzar transaccions segures.
  - En concret ACID és un acrònim de Atomicity, Consistency, Isolation and Durability: Atomicitat, Consistència, Aïllament i Durabilitat en català.



**Autor**: Juan José Lopez

# ACID

- **Atomicitat:**
  - És la propietat que assegura que l'operació s'ha realitzat o no, i per tant davant una fallada del sistema no pot quedar a mig fer. Si una operació consisteix en una sèrie de passos, o ocorren tots o cap.
- **Consistència:**
  - És la propietat que assegura que només es comença allò que es pot acabar. Per tant s'executen aquelles operacions que no van a trencar les regles i directrius d'integritat de la base de dades.



**Autor**: Juan José Lopez

# ACID

- **Aïllament:**
  - És la propietat que assegura que una operació no pot afectar altres. Això assegura que la realització de dues transaccions sobre la mateixa informació mai generarà cap tipus d'error.
- **Durabilitat:**
  - És la propietat que assegura que una vegada realitzada l'operació aquesta persistirà i no es podrà desfer encara que falli el sistema.



**Autor**: Juan José Lopez

# Tipus de motors d'emmagatzematge

## MyISAM

- Es basa en l'antic ISAM, al que afegix moltes millores, és el motor que usa MySQL per defecte (fitxer /etc/mysql/my.cnf, paràmetre default-storage-engine)
- És una bona combinació entre funcionalitat i rendiment encara que manca d'algunes característiques interessants per al manteniment de la integritat de la base de dades.
- Cada taula MyISAM s'emmagatzema en disc en tres fitxers, al directori /var/lib/mysql (Ubuntu). Els fitxers tenen noms que comencen amb el nom de taula i tenen una extensió per a indicar el tipus de fitxer. Un fitxer .frm emmagatzema la definició de taula. El fitxer de dades té una extensió .MYD (MYData) . El fitxer índex té una extensió .MYI (MYIndex) .



**Autor**: Juan José Lopez

# MyISAM

- **Característiques més importants:**
  - Límit de 2^32 registres
  - Màxim de 64 índexs per taula
  - Màxim de 16 columnes per índex
  - Les dades són independents de la màquina i el sistema operatiu
  - Permet camps índex com NULL
  - BLOB i TEXT poden ser índexs
  - Permet una gran grandària en les taules (fins a 256TB)



**Autor**: Juan José Lopez

# MyISAM

- **Característiques (cont)**
  - No suporta transaccions
  - Bloqueja les dades a nivell de taula
  - No permet “claus alienes”
  - Suport d'un tipus VARCHAR autèntic; una columna VARCHAR comença amb la longitud emmagatzemada en dues bytes
  - Taules amb VARCHAR poden tenir longitud de registre fixa o dinàmica
  - VARCHAR i CHAR poden ser de fins a 64KB



**Autor**: Juan José Lopez

# MyISAM

- Aquest motor posa especial obstinació en la rapidesa de les operacions de lectura (predomini de SELECT), és una de les raons per les quals MySQL és tan popular en la web, ja que la majoria de les operacions que es realitzen són d'aquest tipus.
- El fet de que no hagi de fer comprovacions d'integritat referencial també influïx en la seva velocitat.
- Exemple de creació d'una taula:

```
CREATE TABLE t (i INT) ENGINE = MYISAM;
```

- En general no farà falta indicar l'ús d'aquest motor doncs és el que s'usa per defecte.



**Autor**: Juan José Lopez

# MERGE

- **Aquest tipus de motor permet combinar diverses taules d'igual estructura en una única taula, podent així realitzar consultes sobre una taula que ens retorna dades de vàries.**
  - Una de les seves funcionalitats pot ser partir una taula molt gran en altres més petites i, a l'unir-les amb MERGE, permetre'ns treballar amb elles com si fossin una sola.



**Autor**: Juan José Lopez

# MERGE

- Quan es crea una taula MERGE , MySQL crea dos fitxers en disc. Els fitxers tenen noms que comencen amb el nom de la taula i tenen una extensió per a indicar el tipus de fitxer.
- Un fitxer .frm emmagatzema la definició de taula , i un fitxer .MRG conté els noms de les taules que han d'usar-se com una.
- Les taules no han d'estar en la mateixa base de dades que la taula MERGE mateixa.
- Es pot usar SELECT, DELETE, UPDATE, i INSERT en la col·lecció de taules però, s'ha de tenir els permisos de SELECT, UPDATE, INSERT i DELETE en les taules que s'afegeixen a una taula MERGE per modificar-les i viceversa.
- Si es fa un DROP de la taula MERGE , només esborra l'especificació MERGE . Les taules subjacents no es veuen afectades.



**Autor**: Juan José Lopez

# MERGE

- Quan es crea una taula MERGE , s'ha d'especificar una clàusula UNION=(list-of-tables) que indica quines taules es volen usar com una.
- Es pot especificar una opció INSERTMETHOD si es vol que les insercions en la taula MERGE es realitzin en la primera o última taula de la llista UNION .
- S'utilitza un valor de FIRST o LAST per a fer que les insercions es facin en la primera o última taula, respectivament. Si no especifica una opció INSERTMETHOD o si l'especifica amb un valor de NO, els intents d'inserir registres en la taula MERGE produïxen un error.



**Autor**: Juan José Lopez

# MERGE

- **Característiques més importants:**
  - Límit de 2^32 registres
  - Les taules “base” han de ser MyISAM
  - Bloqueig a nivell de taula
  - No té índexs, usa els de les taules “base” (excepte FULLTEXT)
  - La lectura és més lenta a l'haver d'anar consultant la clau en cadascuna de les taules subjacents
  - No permet REPLACE
  - No suporta transaccions
  - En la seva creació no comprova que les taules que usa existeixin i tinguin una estructura idèntica



**Autor**: Juan José Lopez

# MERGE

- **Exemple de creació d'una taula a partir d'altres dues:**

```
mysql> CREATE TABLE t1 (
-> a INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
-> message CHAR(20));
mysql> CREATE TABLE t2 (
-> a INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
-> message CHAR(20));
mysql> INSERT INTO t1 (message) VALUES ('Testing'),('table'),('t1');
mysql> INSERT INTO t2 (message) VALUES ('Testing'),('table'),('t2');
mysql> CREATE TABLE total (
-> a INT NOT NULL AUTO_INCREMENT
-> message CHAR(20), INDEX(a))
-> TYPE=MERGE UNION=(t1,t2) INSERT_METHOD=LAST;
```



**Autor**: Juan José Lopez

# MEMORY(HEAP)

- **Aquest tipus de motor guarda totes les dades en memòria, de manera que si cau el servidor o es reinicia MySQL es perden les dades, encara que l'estructura de les taules es guarda.**
  - Aquestes taules resulten útils com taules temporals per a determinades consultes ja que a l'estar en memòria i poder utilitzar índex hash resulten extremadament ràpides, és una bona elecció quan necessitem realitzar operacions molt ràpides sobre conjunts petits de dades.
  - Cada taula MEMORY està associada amb un fitxer de disc. El nom de fitxer comença amb el nom de la taula i té una exensión de .frm per a indicar que emmagatzema la definició de la taula.



**Autor**: Juan José Lopez

# MEMORY(HEAP)

- **Característiques més importants:**
  - Bloqueja les dades a nivell de taula
  - Pot usar índexs HASH.
  - No suporta BLOB ni TEXT
  - No suporta transaccions
  - Resulta extremadament fàcil perdre les dades



**Autor**: Juan José Lopez

# BDB (BerkeleyDB)

- **Aquest motor, independent de MySQL, proveïx altes prestacions. Posseïx un mecanisme d'emmagatzematge basat en hash d'alta eficiència, el que facilita el ràpid accés a les dades de forma directa a costa de la lentitud en l'accés seqüencial.**
  - Per a activar aquest motor, hem d'usar l'opció *--with-berkeley-db* amb *configure* al compilar MySQL.



**Autor**: Juan José Lopez

# BDB(BerkeleyDB)

- **Característiques més importants:**
  - MySQL no ho instal·la per defecte
  - Màxim de 31 índexs per taula
  - Màxim de 16 columnes per índex
  - Fins a 256TB
  - Sí suporta transaccions
  - Usa índexs HASH
  - MySQL necessita una clau primària per cada taula BDB, en cas de no existir crearà una oculta
  - El bloqueig intern de les taules es fa a nivell de pàgina (8192 bytes)
  - Obrir moltes taules és bastant lent
  - Cada taula s'emmagatzema en la ruta de creació definida i no es pot canviar de directori excepte usant mysqldump



**Autor**: Juan José Lopez

# FEDERATED

- **Aquest motor es va incloure en la versió 5.03. La novetat d'aquest motor és que permet l'accés a una base de dades MySQL remota com si fos local.**
  - En realitat tenim una taula local que representa a altra remota, ambdues han de ser idèntiques.
  - Per a activar aquest motor, hem d'usar l'opció *--with-federated-storage-engine* amb *configure* al compilar MySQL.



**Autor**: Juan José Lopez

# FEDERATED

- **Característiques més importants:**
  - Permet accés a BBDD remotes
  - MySQL no instal·la aquest motor per defecte
  - No suporta transaccions
  - No contempla el bloqueig de dades
  - No permet ALTER



**Autor**: Juan José Lopez

# FEDERATED

- **Exemple de creació de taula:**

```
CREATE TABLE federated_table (
id int(20) NOT NULL auto_increment,
name varchar(32) NOT NULL default '',
other int(20) NOT NULL default '0',
PRIMARY KEY (id),
KEY name (name),
KEY other_key (other))
ENGINE=FEDERATED
DEFAULT CHARSET=latin1
COMMENT='mysql://root@remote_host:9306/federated/test_table';
```



**Autor**: Juan José Lopez

# ARCHIVE

- **Aquest motor s'utilitza bàsicament per a emmagatzemar grans quantitats de dades sense índexs en molt poc espai, ja que els comprimeix amb zlib arribant a un nivell d'estalvi d'espai considerable.**
  - Aquest motor resulta especialment útil per a l'emmagatzematge d'històrics o logs, ja que aquestos solen ocupar gran quantitat d'espai i no és necessari modificar-los amb posterioritat.
  - Per activar aquest motor, cal usar l'opció *--with-arxive-storage-engine* amb *configure* al compilar MySQL.
  - Quan es crea una taula ARCHIVE , el servidor crea un fitxer de definició de taula en el directori de base de dades. El fitxer comença amb el nom de taula i té una extensió de .ARZ i .ARM. Un fitxer .ARN pot aparèixer durant operacions d'optimització.



**Autor**: Juan José Lopez

# ARCHIVE

- **Característiques més importants:**
  - Gran compressió de les dades
  - Només permet INSERTS i SELECTS
  - Bloqueja les dades a nivell de registre
  - Emmagatzema les dades en un buffer fins que els comprimeix i insereix
  - No suporta transaccions



**Autor**: Juan José Lopez

# CSV

- **Aquest tipus de motor emmagatzema la informació utilitzant el format de valors separats per comes (comma-separated values), de manera que cada taula és un fitxer que conté les dades.**
  - No suporta indexat i la seva finalitat principal és permetre exportar les dades de manera que puguin ser importades fàcilment per algunes suites ofimàtiques.
- **Característiques més importants:**
  - Útil per a exportar i importar dades
  - No suporta indexació ni transaccions



**Autor**: Juan José Lopez

# BLACKHOLE

- **El sorprenent ús d'aquest motor és no emmagatzemar les dades sinó crear un log amb la consulta SQL utilitzada. Com no emmagatzema cap dada lògicament no suporta índexs, ni transaccions...**
  - La seva principal utilitat és mantenir un servidor esclau que mantingui un log del sistema principal.



**Autor**: Juan José Lopez

# EXAMPLE

- **El motor d'emmagatzematge EXAMPLE és un motor de proves que no fa res.**
  - El seu propòsit és servir com exemple en el codi font MySQL per il·lustrar com començar a escriure nous motors d'emmagatzematge. Com a tal, té interès principalment per a desenvolupadors.
  - Per a activar aquest motor d'emmagatzematge, cal usar l'opció *--with-example-storage-engine* amb *configure* al compilar MySQL.
  - Quan es crea una taula EXAMPLE, el servidor crea un fitxer de definició de la taula en el directori de base de dades. El fitxer comença amb el nom de taula i té una extensió .frm . No es creen més fitxers. No pot emmagatzemar-se ni recuperar-se dades de la taula.



**Autor**: Juan José Lopez

# NDB

- **És el motor d'emmagatzematge del clúster de MySQL, les bases de dades es reparteixen pels diferents nodes d'un clúster.**
  - És una bona elecció quan disposant de diversos servidors necessitem alhora velocitat, transaccions i redundància de dades; replicació síncrona; i resistència a caigudes de servidors.



**Autor**: Juan José Lopez

# NDB

- **Característiques més importants:**
  - Proporciona alta disponibilitat mitjançant redundància.
  - Proporciona alt rendiment mitjançant fragmentació de dades sobre els grups de nodes.
  - Proporciona alta escalabilidad mitjançant la combinació de les dues característiques anteriors.
  - Les dades es mantenen en memòria, però els logs van a disc.



**Autor**: Juan José Lopez

# FALCON

- **És el futur motor d'emmagatzematge per a MySQL 6.0, actualment es troba en estat alpha, i està dissenyat per a entorns de servidors Web d'alt volum de dades.**
- **Característiques més importants:**
  - Compleix ACID
  - Permet recuperació d'estats / dades en cas de caigudes
  - Permet una alta concurrència (bloqueig a nivell de registre)
  - Caché de dades molt ràpida i potent
  - Inclou taules per al monitoratge de rendiment i errors
  - Permet una configuració simple



**Autor**: Juan José Lopez

# InnoDB

**Està considerat com un dels motors més avançats per a l'emmagatzematge de dades en MySQL.**

- Proveïx un motor sòlid amb suport complet de transaccions (és ACID compliant)
- Permet el bloqueig de dades a nivell de registre permetent gran flexibilitat a l'hora d'utilitzar les taules, controla la integritat referencial, permet claus alienes i té un sistema de recuperació de caigudes.
- No obstant això la pedra de toc de InnoDB és el seu mecanisme d'indexació i cache dels registres ja que manté una caché d'índexs i dades en memòria i en disc proporcionant un molt alt rendiment.



**Autor**: Juan José Lopez

# InnoDB

- Dos recursos basats en disc molt importants que gestiona el motor d'emmagatzematge InnoDB són els seus fitxers de dades d'espais de taules i els seus fitxers de registre (log).
- Si no s'especifiquen opcions de configuració per a InnoDB, MySQL 5.0 crea en el directori de dades de MySQL un fitxer de dades amb un espai de taules comú de 10MB (autoextensible) cridat *ibdata1* i dos fitxers de registre (log) de 5MB cridats *iblogfile0* i *iblogfile1*.
- Addicionalment, MySQL crea un fitxer .frm amb el nom de cada taula en el directori de la base de dades en la que es troba la taula, sota el directori designat per a les bases de dades de MySQL. Internament, InnoDB agrega al seu propi diccionari de dades una entrada per a la taula 'nom_bd/nom_taula'. Això significa que pot crear-se una taula amb el mateix nom en altra base de dades, i els noms de les taules no entraran en conflicte dins d'InnoDB.
- És possible agregar l'opció *innodb_file_per_table* a my.cnf, i fer que InnoDB emmagatzemi cada taula en el seu propi fitxer .ibd en un directori de bases de dades de MySQL



**Autor**: Juan José Lopez

# InnoDB

- **Característiques més importants:**
  - ACID compliant
  - Permet claus alienes i transaccions, suport d'integritat referencial
  - Bloqueig de dades a nivell de registre i no bloqueja la lectura durant els selects (millora la concurrència)
  - Sistema de recuperació de caigudes
  - Canviar la ubicació de la base de dades/taula és complicat
  - Una taula no pot tenir més de 1000 columnes
  - La grandària de les seves logs ha de ser inferior a 4GB
  - La grandària màxima per a una taula és de 64TB
  - No permet índexs de FULLTEXT
  - No manté una comptador intern de registres (select count(*) from taula ja que això relentiza el funcionament al tindre que recórrer tot l'índex)



**Autor**: Juan José Lopez

# Transaccions en MySQL

**Una transacció és una unitat lògica de treball. O, informalment, i treballant amb SQL, un conjunt de sentències que s'executen com si fossin una de sola. En general, les sentències que formen part d'una transacció s'in terrelacionen entre si, i no té sentit que se n'executi una sense que s'executin les altres.**



**Autor**: Juan José Lopez

# Transaccions en MySQL

- **Només els motors InnoDB i BDB soporten transaccions a un servidor MySQL**
  - Els altres motors no transaccionals en MySQL Server (com MyISAM) segueixen un paradigma diferent per a garantir la integritat de dades cridat "operacions atòmiques".
  - En termes transaccionals, les taules MyISAM operen en manera AUTOCOMMIT=1.
  - Les operacions atòmiques sovint oferixen integritat comparable amb millor rendiment.
  - MySQL Server suporta ambdós paradigmes, pot decidir si la seva aplicació necessita la velocitat d'operacions atòmiques o l'ús de característiques transaccionals. Aquesta elecció pot fer-se per a cada taula.
  - El compromís entre tipus de taules transaccionals i no transaccionals resideix principalment en el rendiment. Les taules transaccionals tenen requeriments significativament majors per a memòria i espai de disc, i major càrrega de CPU. Per altra banda, tipus de taules transaccionals com InnoDB també ofereixen moltes característiques significatives. El disseny modular de MySQL Server permet l'ús concorrent de diferents motors d'emmagatzematge per a complir diferents requeriments i mostrar-se òptim en totes les situacions.



**Autor**: Juan José Lopez

# Transaccions a MySQL

- En InnoDB, tota l'activitat de l'usuari es produïx dins d'una transacció. Si la manera d'execució automàtica (autocommit) està activat, cada sentència SQL conforma una transacció individual per si mateixa. MySQL sempre comença una nova connexió amb l'execució automàtica habilitada.
- Si la manera d'execució automàtica es deshabilita amb SET AUTOCOMMIT = 0, llavors pot considerar-se que un usuari sempre té una transacció oberta.
- Una sentència SQL COMMIT o ROLLBACK acaba la transacció vigent i comença una nova. Ambdues sentències alliberen tots els bloquejos InnoDB que es van establir durant la transacció vigent. Un COMMIT significa que els canvis fets en la transacció actual es converteixen en permanents i es tornen visibles per als altres usuaris. Per altra banda, una sentència ROLLBACK, cancel·la totes les modificacions produïdes en la transacció actual.
- Si la connexió té l'execució automàtica habilitada, l'usuari pot igualment portar a terme una transacció amb diverses sentències, si la comença explícitament amb START TRANSACTION o BEGIN i l'acaba amb COMMIT o ROLLBACK.

  Mireu:

  http://dev.mysql.com/doc/refman/5.0/es/transactional-commands.html



**Autor**: Juan José Lopez

# Transaccions a MySQL

- **Exemples de transaccions**

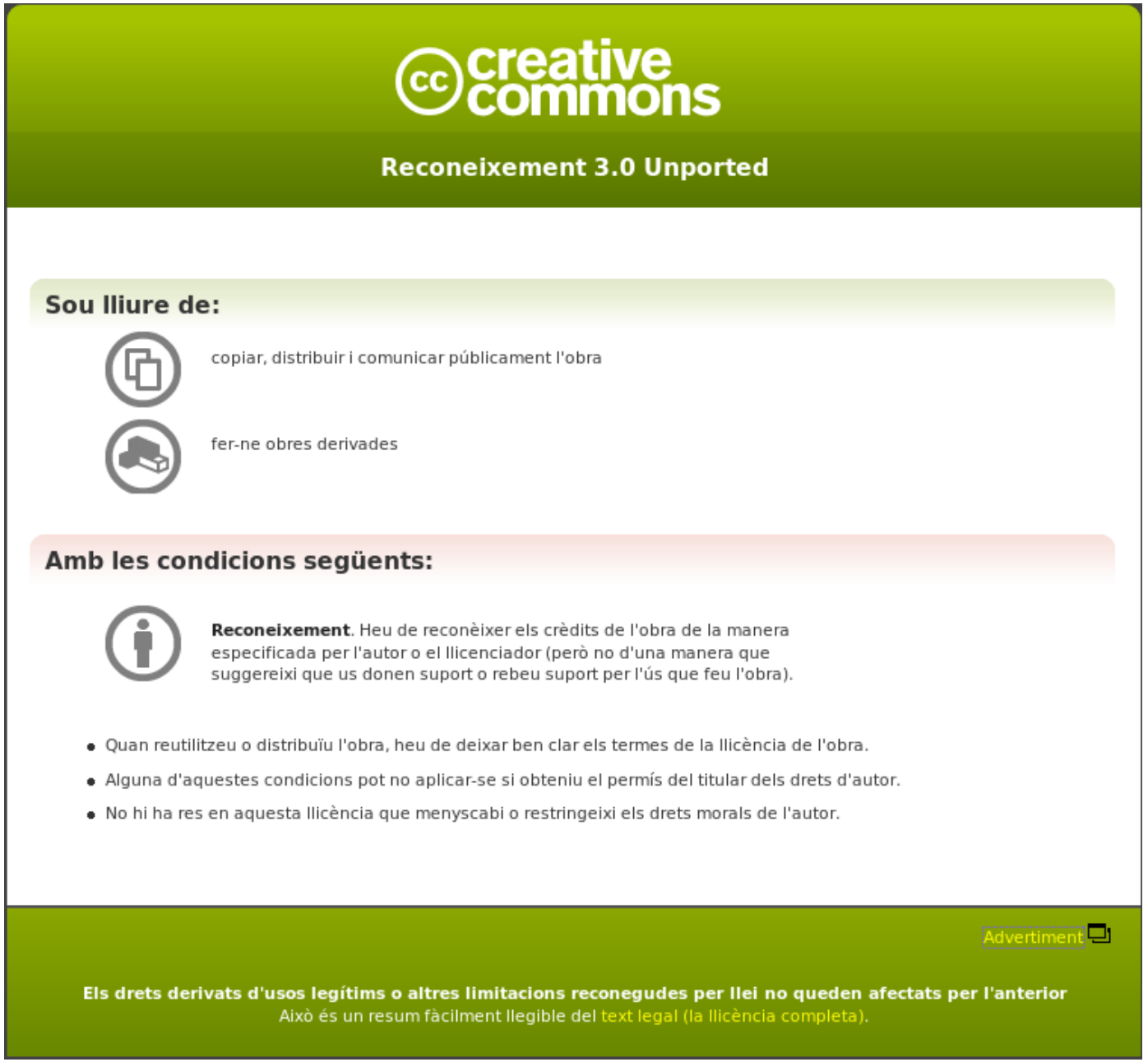
```
START TRANSACTION;
UPDATE nomina SET salari=salari + (salari*1/100);
INSERT INTO impostos (nom,salari) SELECT nom,salari FROM
nomina WHERE salari > 1500;
COMMIT;
```

```
AUTOCOMMIT=0;
UPDATE nomina SET salari=salari + (salari*1/100);
ROLLBACK;
INSERT INTO impostos (nom,salari) SELECT nom,salari FROM
nomina WHERE salari > 1500;
COMMIT;
```



**Autor**: Juan José Lopez

Llicència Creative Commons




http://creativecommons.org/licenses/by/3.0/deed.ca





**Autor:** Juan José López