Architettura MySQL

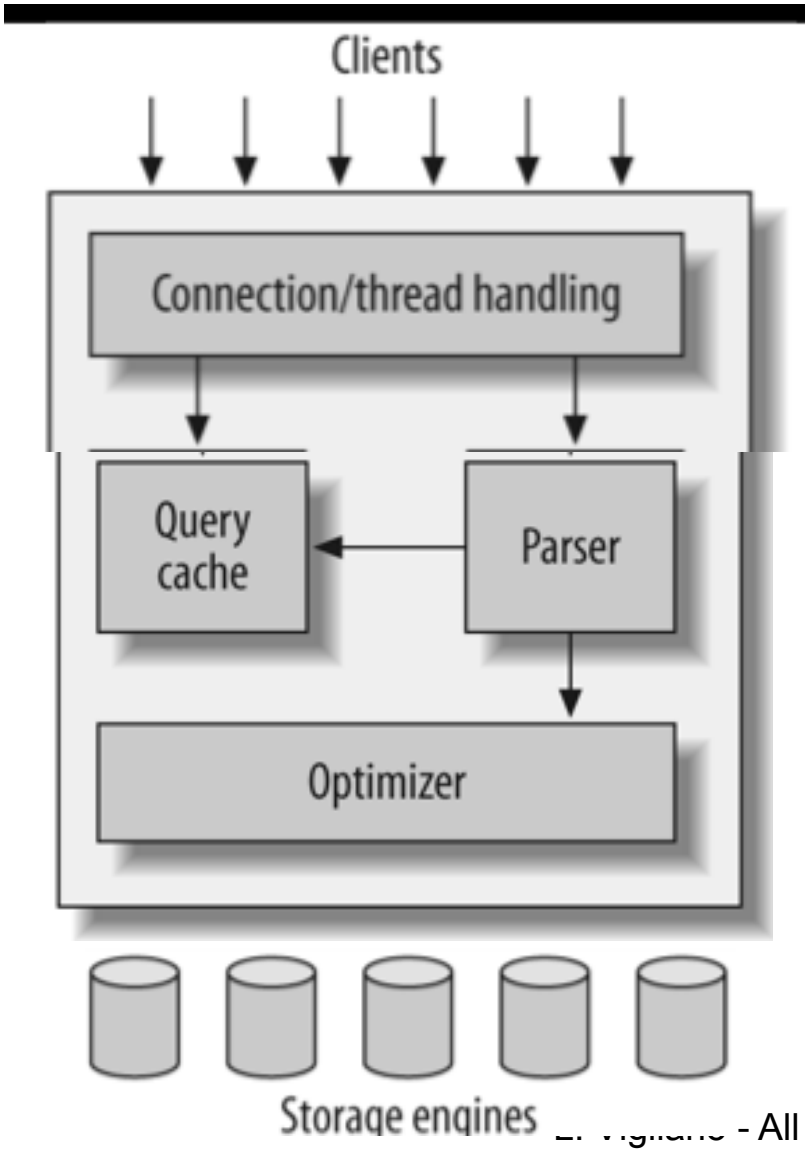
E Motori MySQL

Architettura MySQL
Caratteristiche

- Differente dagli altri DBMS
- Non perfetta
- Ma flessibile
- Gestisce DataWarehouses, OLTP, ecc.

- Architettura delle Storage Engine
  - Separa il query processing dai task di memorizzazione e ricerca dati

Architettura MySQL



- 1° liv. : servizi di rete, connessione, ecc. Simile ad altri DBMS.
- 2° liv. : cuore di MySql. Codice per query parsing, analisi, ottimizzazione, caching, ecc.
- 3° liv.: Motori di storage. Memorizzazione e ripescaggio di tutti i dati.

Architettura MySQL
Cosa avviene

- Connessione client
- Il server autentica e verifica privilegi client
- Controllo query cache
- MySQL analizza le query per creare il "parse tree"
- Ottimizzatore
- La Storage Engine scelta influisce sull' ottimizzazione

Architettura MySQL
Controllo concorrenza

- MySQL lo deve effettuare su due livelli :
  - Livello server
  - Livello storage engine
- Lock di due tipi :
  - shared lock
  - exclusive lock
- Granularità dei lock
  - Fissarne il giusto grado (table lock, row lock)
- La gestione dei lock è all'interno delle storage engine

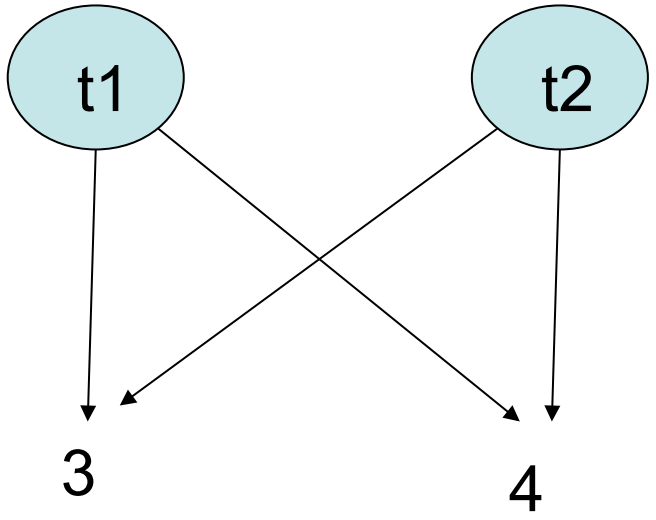
Architettura MySQL
Transazioni

- Decido, tramite le storage engine, se il DB è transazionale
- Ogni motore implementa 4 livelli di isolamento (in maniera però differente) :
  - Read uncommitted
  - Read committed
  - Repeatable read (default)
  - Serializable
- Alcune storage engine utilizzano il Transaction logging

Architettura MySQL
Deadlock

T1 :
aggiorna 3
aggiorna 4
commit

T2 :
aggiorna 4
aggiorna 3
commit



I DBMS adottano, per evitare il deadlock, varie forme di riconoscimento dello stallo e di timeout

InnoDB riconosce le dipendenze circolari.

Architettura MySQL
Transazioni

- MySQL ha tre storage engine transazionali :
  - InnoDB, NDBCluster e Falcon
  - MyIsam non è transazionale

- MySQL opera in “autocommit mode”, ma si può variare:
  - Select @@autocommit;
  - set autocommit = 0 ;

Architettura MySQL
Transazioni

- MySQL setta livello di isolamento sull'intero server o sulla singola sessione di lavoro :
  - Set session transaction isolation level read committed;

- MySQL non permette di avere, in una singola transazione, un mix di storage engine….. o meglio…..

Architettura MySQL
MVCC

- InnoDB, Falcon e PBXT non usano un semplice row-locking ma l'MVCC.
- La tecnica MVCC (MultiVersion Concurrency Control) dà a tutti un'istantanea (snapshot) consistente dei dati.

- MVCC (di InnoDB) assegna ad ogni riga di tabella due valori (versione della transazione) :
  - Un valore quando la riga viene creata;
  - Un valore quando la riga viene cancellata;
- Ogni transazione mantiene la riga con la sua versione.

Architettura MySQL
MVCC

SELECT

InnoDB must examine each row to ensure that it meets two criteria:

1. InnoDB must find a version of the row that is at least as old as the transaction (i.e., its version must be less than or equal to the transaction's version). This ensures that either the row existed before the transaction began, or the transaction created or altered the row.

2. The row's deletion version must be undefined or greater than the transaction's version. This ensures that the row wasn't deleted before the transaction began

Rows that pass both tests may be returned as the query's result.

INSERT

InnoDB records the current system version number with the new row.

DELETE

InnoDB records the current system version number as the row's deletion ID.

UPDATE

InnoDB writes a new copy of the row, using the system version number for the new row's version. It also writes the system version number as the old row's deletion version.

Architettura MySQL
MVCC

- InnoDB, in pratica, deve trovare una versione della riga che è vecchia almeno come quella della transazione.
- La versione dello storage engine deve essere quindi <= alla versione della transazione.

- Questo assicura che :
  - o la riga esisteva prima che iniziasse la transazione
  - o che la transazione stessa ha creato o cambiato la riga.

Architettura MySQL Sommario Locking e Concorrenza

able 1-2 summarizes the various locking models and concurrency levels in MySQL.

Table 1-2. Locking models and concurrency in MySQL using the default isolation leve

Locking strategy	Concurrency	Overhead	Engines
Table level	Lowest	Lowest	MyISAM, Merge, Memory
Row level	High	High	NDB Cluster
Row level with MVCC	Highest	Highest	InnoDB, Falcon, PBXT, solidDB

Architettura MySQL
Storage Engine

- Le Storage Engines, in generale, sono moduli software che si occupano della memorizzazione e del recupero delle informazioni.
- MySQL memorizza ciascun database come sottodirectory della sua directory di dati (file system).
- Ogni storage engine memorizza dati e indici della tabella differentemente, ma è il server che gestisce la table definition ;
  - show engines;
  - show table status like ‘nometabella’ \G

Architettura MySQL
MyIsam Engine

- Era il default di MySQL.
- Buon compromesso tra efficienza ed utilizzo.
- Non supporta transazioni e lock a livello riga.
- Ogni tabella in due file (portabili) :
  - uno di dati (.MYD) e uno di indice (.MYI).
- Tabelle di 256 Tb, indici anche su 500 char.
- Comprimere (‘pack’) tabelle : myisampack;
- Controllo e recupero automatico :
  - check table ‘nometab’ ;
  - repair table ‘nometab’ ;
  - myisamchk;

Architettura MySQL
MyIsam Merge Engine

- Combinazione di tabelle MyIsam identiche in una tabella virtuale.

- Utile per logging e datawarehouses

Architettura MySQL
InnoDB Engine

- Il più popolare per storage transazionale… ma anche per efficienza e crash-recovery automatico.
- Dati memorizzati in una serie di file detti ‘tablespace’
- Usa MVCC e i 4 livelli di isolamento
  - Repateable read default
  - Strategia ‘next-key locking’

Architettura MySQL
InnoDB Engine (2)

- Tabelle costruite su “clustered index”.
  - Primary key molto veloci.
  - Indici secondari meno veloci (no sort).
- Non comprime gli indici
- Supporta il constraint sulle foreign key
- Ottimizzazioni interne

Architettura MySQL
Memory Engine

- Tabelle heap (con dati in memoria);
- Accesso molto veloce a tabelle che non cambiano e che non recuperi.
- A un restart del server le tabelle sopravvivono, ma i dati no
- Utili per tenere il risultato periodico di dati aggregati, risultati intermedi…. e per le query di MySQL

Architettura MySQL
Archive Engine

- Solo insert e select;
- Non ha indici;
- Supporta il row-level locking ed emula l'MVCC
- Utili per logging e acquisizione dati

Architettura MySQL
CSV (Comma Separeted Value) Engine

- Tratta come tabelle, i file con valori separati da virgole
- Niente indici
- Utile per lo scambio di dati

Architettura MySQL
Federate Engine

- Tabelle Federated fanno riferimento a tabelle su un MySQL server remoto (nessun dato locale)
- Adatto per singole richieste sporadiche

Architettura MySQL
Black-hole Engine

- Nessun meccanismo di memoria
- Utile per simulare la replication o verificare il logging

Architettura MySQL
NDB Cluster Engine

- Alta velocità (Sony- Ericsson 2003)
- Loggato al disco, ma mantiene tutti i suoi dati in memoria
- Architettura complessa basata sul ‘non condivido nulla’
- Il database NDB consiste in nodi di dati, nodi di gestione, nodi SQL
- Ciascun nodo di dati (server) mantiene un “fragment” dei dati. Frammenti duplicati.

Architettura MySQL
Falcon Engine

- Jim Starkey, inventore MVCC
- Progettato per processori multipli a 64 bit e tanta memoria
- Usa MVCC e cerca di mantenere le transazioni tutte in memoria
- Non finito.

Architettura MySQL
Solid DB Engine

- Simile all'InnoDB
- Transazionale, usa MVCC e supporta foreign key

Architettura MySQL
PBXT (PrimeBaseXT) Engine

- Innovativo
- Transazionale, usa MVCC, supporta foreign key, riduce commit

Architettura MySQL
Maria Storage Engine

- Vuole sostituire MyIsam
- Ultimamente rinnovato e con nuove funzionalità

- Adatto per tabelle di privilegi e tabelle temporanee.

Storage engine	MySQL version	Transactions	Lock granularity	Key applications	Counter-indications
MyISAM	All	No	Table with concurrent inserts	`SELECT`, `INSERT`, bulk loading	Mixed read/write workload
MyISAM Merge	All	No	Table with concurrent inserts	Segmented archiving, data warehousing	Many global lookups
Memory (`HEAP`)	All	No	Table	Intermediate calculations, static lookup data	Large datasets, persistent storage
InnoDB	All	Yes	Row-level with MVCC	Transactional processing	None
Falcon	6.0	Yes	Row-level with MVCC	Transactional processing	None
Archive	4.1	No	Row-level	Logging, aggregate analysis	Random access needs, updates, deletes
CSV	4.1	No	Table	Logging, bulk loading of external data	Random access needs, indexing
Blackhole	4.1	N/A	N/A	Logged or replicated archiving	Any but the intended use
Federated	5.0	N/A	N/A	Distributed data sources	Any but the intended use
NDB Cluster	5.0	Yes	Row-level	High availability	Most typical uses
PBXT	5.0	Yes	Row-level with MVCC	Transactional processing, logging	Need for clustered indexes
solidDB	5.0	Yes	Row-level with MVCC	Transactional processing	None
Maria (planned)	6.x	Yes	Row-level with MVCC	MyISAM replacement	None

Architettura MySQL
Scegliere le Storage Engine

- Sceglierle al momento del design del DB
- Verificare se ho bisogno di:
  - Transazioni
  - Concorrenza
  - Backup
  - Crash-recovery
  - Altre caratteristiche particolari

Architettura MySQL
Esempi

- Logging di ogni chiamata telefonica :
  - MyIsam, Archive, PBXT
  - Non sottovalutare i bugs di MyIsam di fronte a un crash
- Operazioni bancarie, prenotazioni voli, ecc.:
  - InnoDB o simili

Architettura MySQL trasformare Storage Engine di tabelle

- Ci sono tre modi di trasformare tabelle da un tipo di storage engine a un altro:
  - Alter table mytable Engine= Falcon;
  - Usare mysqldump per usare i dati in una nuova create table;
  - Misto dei precedenti :
    - Create table innodb_table like myisam_table;
    - Alter table innodb_table engine=InnoDB;
    - Insert into innodb_table as select * from myisam_table;