Atitit 数据库存储引擎

目录

[1.1. BLACKHOLE 黑洞引擎 1](#_Toc14623)

[1.2. Myisam innodb 1](#_Toc30188)

[1.3. Archive 档案类 1](#_Toc28988)

[1.4. Fed 连接引擎 2](#_Toc13593)

[1.5. Memory 4](#_Toc10054)

[1.6. Merge 合并表引擎 4](#_Toc14814)

[1.7. Cluster 4](#_Toc8044)

[1.8. Bdb 4](#_Toc19635)

[1.9. Cluster/NDB 4](#_Toc18745)

[1.10. Custom 5](#_Toc11270)

[1.11. csv 5](#_Toc2607)

[1.12. MMAP内存映射存储引擎, 5](#_Toc27895)

[1.13. PERFORMANCE\_SCHEMA 5](#_Toc23049)

[2. 2.常用两种引擎的选择 6](#_Toc27392)

## BLACKHOLE 黑洞引擎

  BLACKHOLE存储引擎是一个非常有意思的存储引擎,功能恰如其名,就是一个"黑洞".就像Unix系统下面的"/dev/null"设备一样,不管我们写入任何信息,都是有去无回,那么BLACKHOLE存储引擎有什么用呢?在我最初接触MySQL的时候也有过同样的疑问,不知道MySQL提供这样一个存储引擎给我们用意何在?但是在后来的一次数据的迁移过程中,正是BLACKHOLE给我带来了非常大的帮助,在那次数据迁移过程中,数据须要经过一个中转的MySQL服务器做一些相关的转换操作,然后再通过复制移植到新的服务器上面,可当时我没有足够的空间来支持这个中转服务器的动作,这时候就显示出BLACKHOLE的功效了,他不会记录下任何数据,但是会在binlog中记录下所有的Query.而这些Query最终都会被复制利用,并实施到最终的slave端.  
    MySQL的用户手册上面介绍了BLACKHOLE存储引擎其他几项用途:  
a.Query语法的验证;  
b.测试二进制日志记录的性能开销,如通过比较允许二进制日志功能的BLACKHOLE与禁止二进制日志功能的BLACKHOLE的性能来实现.  
c.查找与存储引擎自身不相关的性能瓶颈,因为BLACKHOLE本质上是一个"no-op"的存储引擎.

## Myisam innodb

## Archive 档案类

拥有很好的压缩机制，它使用zlib压缩库，在记录被请求时会实时压缩。

支持最基本的插入和查询两种功能。在MySQL 5.5开始支持索引。

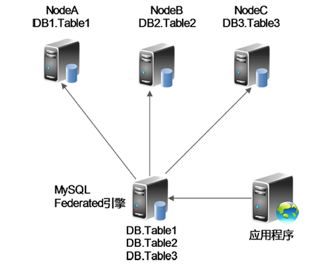
不支持事务。支持行级锁和专用的缓存区，所以可以实现高并发的插入。

适合存储大量日志、历史数据。

## Fed 连接引擎

该存储引擎可以不同的Mysql服务器联合起来，逻辑上组成一个完整的数据库。  
这种存储引擎非常适合数据库分布式应用。  
Federated存储引擎可以使你在本地数据库中访问远程数据库中的数据，针对federated存储引擎表的查询会被发送到远程数据库的表上执行，本地是不存储任何数据的。

场景： dblink。



缺点：

1.对本地虚拟表的结构修改，并不会修改远程表的结构

2.truncate 命令，会清除远程表数据

3. drop命令只会删除虚拟表，并不会删除远程表

4.不支持 alter table 命令

5. select count(), select  from limit M, N 等语句执行效率非常低，数据量较大时存在很严重的问题，但是按主键或索引列查询，则很快，如以下查询就非常慢（假设 id 为主索引）

select id from db.tablea where id >100 limit 10 ;

而以下查询就很快：

select id from db.tablea where id >100 and id<150

6.  如果虚拟虚拟表中字段未建立索引，而实体表中为此字段建立了索引，此种情况下，性能也相当差。但是当给虚拟表建立索引后，性能恢复正常。

7. 类似 where name like "str%" limit 1 的查询，即使在 name 列上创建了索引，也会导致查询过慢，是因为federated引擎会将所有满足条件的记录读取到本，再进行 limit 处理。

## Memory

## Merge 合并表引擎

Merge

Merge允许将一组使用MyISAM存储引擎的并且表结构相同（即每张表的字段顺序、字段名称、字段类型、索引定义的顺序及其定义的方式必须相同）的数据表合并为一个表，方便了数据的查询。

场景：MySQL中没有物化视图，视图的效率极低，故数据仓库中数据量较大的每天、每周或者每个月都创建一个单一的表的历史数据的集合可以通过Merge存储引擎合并为一张表。

## Cluster

## Bdb

## Cluster/NDB

该存储引擎用于多台数据机器联合提供服务以提高整体性能和安全性。适合数据量大、安全和性能要求高的场景。

CAP理论。CAP理论(Brewer’s CAP Theorem) ，是说Consistency(一致性), Availability(可用性), Partition tolerance(分布) 三部分在系统实现只可同时满足二点，没法三者兼顾。如果对"一致性"要求高，且必需要做到"分区"，那么就要牺牲可用性;而对大型网站，可用性与分区容忍性优先级要高于数据一致性，一般会尽量朝着 A、P 的方向设计，然后通过其它手段保证对于一致性的商务需求。

## Custom

## csv

每个表会生成一个.CSV文件，将CSV类型的文件当做表进行处理。

把数据以逗号分隔的格式存储在文本文件中，这种文件是一种普通文本文件，每个数据行占用一个文本行。

不支持索引，即使用该种类型的表没有主键列，也不允许表中的字段为null。

## MMAP内存映射存储引擎,

 MongoDB使用的是内存映射存储引擎,即Memory Mapped Storage Engine，简称MMAP。

## PERFORMANCE\_SCHEMA

PERFORMANCE\_SCHEMA：

该引擎主要用于收集数据库服务器性能参数。这种引擎提供以下功能：提供进程等待的详细信息，包括锁、互斥变量、文件信息；保存历史的事件汇总信息，为提供MySQL服务器性能做出详细的判断；对于新增和删除监控事件点都非常容易，并可以随意改变mysql服务器的监控周期，例如（CYCLE、MICROSECOND）。 MySQL用户是不能创建存储引擎为PERFORMANCE\_SCHEMA的表。

场景： DBA能够较明细得了解性能降低可能是由于哪些瓶颈

# 2.常用两种引擎的选择

#### MyISAM与InnoDB

InnoDB和MyISAM是许多人在使用MySQL时最常用的两个表类型，这两个表类型各有优劣，视具体应用而定。

基本的差别为： MyISAM类型不支持事务处理等高级处理，而InnoDB类型支持。MyISAM类型的表强调的是性能，其执行数度比InnoDB类型更快，但是不提供事务支持，而InnoDB提供事务支持以及外部键等高级数据库功能。

所以从宏观来讲，事务数据库关注细节，而数据仓库关注高层次的聚集，所以，InnoDB更适合作为线上的事务处理，而MyISAM更适合作为ROLAP型数据仓库。

从接触的应用逻辑来说，select count(\*) 和order by 是最频繁的，大概能占了整个sql总语句的60%以上的操作，而这种操作Innodb其实也是会锁表的，很多人以为Innodb是行级锁，那个只是where对它主键是有效，非主键的都会锁全表的。但MYISAM对于count操作只需要在元数据中读取，不用扫表。

6.如果是用MyISAM的话，merge引擎可以大大加快数据仓库开发速度，非常适合大项目总量约几亿的rows某一类型(如日志，调查统计)的业务表。

Atitti 存储引擎支持的国内点与特性attilax总结

MySQL各种存储引擎介绍与适用场景 - - SegmentFault 思否.html