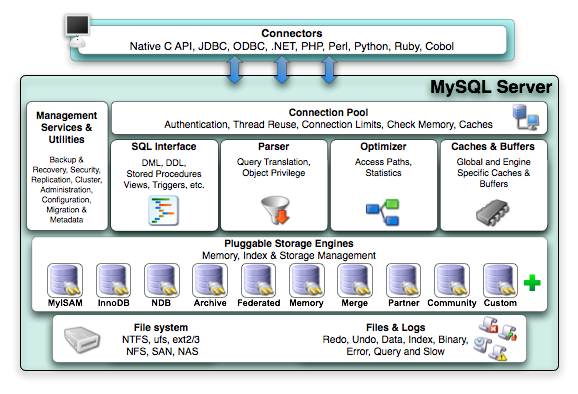
**MySQL数据库日常运维**



# MySQL架构

## 缺点

对比性能的话，单表还好点，三个表及其以上的关联查询，MySQL在这种环境下的性能和Oracle和SQLServer比较根本不是一个层次的，并不适合类似企业管理方面的复杂查询和关联

在可编程支持方面，MySQL和其他数据库相比还有一定的差距， 虽然最新版的MySQL已经开始提供一些简单的可编程支持，如开始支持 Procedure， Function， Trigger等， 但是所支持的功能还比较有限，和其他几大商用数据库管理系统相比，还存在较大的不足。

## 优点

免费

已于扩展

集群方案采用无共享架构，可以做大幅度的横向扩展

MySQL 最擅长的是查询性能，而不是事务处理

MySQL从诞生之初，采取的策略就是迎合互联网应用。所以MySQL主要应用在互联网

## 另外一个考虑

多用收费的软件出问题可以甩锅，这样就能保住自己工作不被解雇。

如果公司业务出问题，技术上都是你选的开源软件，别人往软件上甩锅就都甩到你那去

## MySQL的KV化

对于亿级规模的数据存储，尤其是涉及到水平拆分跨机分库分表的情况下，线上对数据库的访问只能做的越简单越好，group by/order by/分页/通用join/事务等等的支持 在这个量级下的MySQL系统都是不合适的。

## MySQL+Redius

业务层通过双写同时写MySQL及Redis。读通常在Redis，若读取不到，则从MySQL读取，然后将数据同步到Redis，Redis通常设置expire或者默认LRU进行数据淘汰。

这种使用方式会有如下问题：

(1) MySQL及Redis存在数据不一致风险，尤其是长时间运行的系统

(2) 业务层需要处理MySQL sql schema与Redis kv数据结构上的逻辑差异

(3) 无统一运维

(4) 无法方便扩容/缩容

mysql是传统关系数据库，redis是nosql的内存数据库，后者的出现主要是为了解决前者IO效率的问题。单独作为数据库解决方案，二者都能独当一面（redis也能做数据持久化，只是数据总容量受内存限值），由于mysql+redis是目前用的比较多的一个方案，这里搜集了下网友在用该方案的心得：

mysql作为主要数据库存储所有业务数据，对于访问量很大的特殊业务，将mysql数据读到redis中，客户端在这些特殊业务中直接读redis，其它业务还是与mysql交互。当这些特殊业务的数据有增删改时，同步修改mysql和redis即可；

数据使用情形同情况1，但当特殊业务的数据有增删改时，只操作mysql，对redis的相应数据做失效处理（删除），redis中缺的数据在需要时找mysql再同步过来；

根据实际业务场景，redis只一次性同步mysql中基本不需要增删改的某类数据，即使用中不用考虑二者数据同步的问题，保证高一致性。

以上三种情况中，mysql与redis的数据一致性是递增的。情况1中同步mysql和redis的过程有可能失败，导致二者数据不一致；情况2中失效处理本身需要保证二者的同步性，如果处理不好也会出现不一致的可能；情况3因为几乎没有一致性的考虑所以最能保证，但业务约束太大。

## MySQL做集群

(1 )组建MySQL集群的几种方案

LVS+Keepalived+MySQL（有脑裂问题？但似乎很多人推荐这个）

DRBD+Heartbeat+MySQL（有一台机器空余？Heartbeat切换时间较长？有脑裂问题？）

MySQL Proxy（不够成熟与稳定？使用了Lua？是不是用了他做分表则可以不用更改客户端逻辑？） -- 不错的项目，可惜官方半途夭折了，不建议用，无法高可用，是一个写分离

MySQL Cluster （社区版不支持INNODB引擎？商用案例不足？）

MySQL + MHA （如果配上异步复制，似乎是不错的选择，又和问题？）

MySQL + MMM （似乎反映有很多问题，未实践过，谁能给个说法）

## MySQL与缓存

# 安装数据库

下载并解压 tar -xvf mysql-5.7.21-1.el6.x86\_64.rpm-bundle.tar

mysql-community-libs-5.7.21-1.el6.x86\_64.rpm

mysql-community-devel-5.7.21-1.el6.x86\_64.rpm

mysql-community-server-5.7.21-1.el6.x86\_64.rpm

mysql-community-test-5.7.21-1.el6.x86\_64.rpm

mysql-community-embedded-5.7.21-1.el6.x86\_64.rpm

mysql-community-client-5.7.21-1.el6.x86\_64.rpm

mysql-community-libs-compat-5.7.21-1.el6.x86\_64.rpm

mysql-community-common-5.7.21-1.el6.x86\_64.rpm

mysql-community-embedded-devel-5.7.21-1.el6.x86\_64.rpm

rpm -ivh mysql-community-common-5.7.21-1.el6.x86\_64.rpm

rpm -ivh mysql-community-libs-5.7.21-1.el6.x86\_64.rpm

rpm -ivh mysql-community-client-5.7.21-1.el6.x86\_64.rpm

rpm -ivh mysql-community-server-5.7.21-1.el6.x86\_64.rpm

rpm -ivh mysql-community-devel-5.7.21-1.el6.x86\_64.rpm

启动mysql服务：

service mysqld start

查看初始密码 ：

grep 'temporary password' /var/log/mysqld.log

修改密码：

ALTER USER 'root'@'localhost' IDENTIFIED BY 'Workhard\_1234';

# 物理结构

## 日志文件

1. 错误日志：Error Log
2. 二进制日志：Binary Log
3. 更新日志：Update Log
4. 查询日志：Query Log
5. 慢日志：Slow Query Log
6. Innodb的在线redo日志

## 数据文件

## Replication相关文件

# 用户权限管理

## mysql权限系统的工作原理

MySQL权限系统通过下面两个阶段进行认证：

  （1）对连接的用户进行身份认证，合法的用户通过认证、不合法的用户拒绝连接。

  （2）对通过认证的合法用户赋予相应的权限，用户可以在这些权限范围内对数据库做相应的操作。

  对于身份，MySQL是通过IP地址和用户名联合进行确认的，例如MySQL安装默认创建的用户root@localhost表示用户root只能从本地（localhost）进行连接才可以通过认证，此用户从其他任何主机对数据库进行的连接都将被拒绝。也就是说，**同样的一个用户名，如果来自不同的IP地址，则MySQL将其视为不同的用户**。

  MySQL的权限表在数据库启动地时候就载入内存，当用户通过身份认证后，就在内存中进行相应权限的存取，这样，此用户就可以在数据库中做权限范围内的各种操作了。

## 重置mysql的密码

-bash-4.2$ mysql

ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)

在配置文件里设置跳过密码验证：

#vim /etc/my.cnf(注：windows下修改的是my.ini)

skip-grant-tables

explicit\_defaults\_for\_timestamp=true

重启mysql

/etc/init.d/mysql restart

进入mysql后重置密码：

|  |
| --- |
| mysql> use mysql; mysql> update user set password=password("zabbix") where user="root"; mysql> flush privileges; mysql> quit |

## 进入了mysql进行密码重置

|  |
| --- |
| mysql> UPDATE mysql.user SET Password=PASSWORD('新密码') where USER='root';  mysql> flush privileges;  mysql> exit |

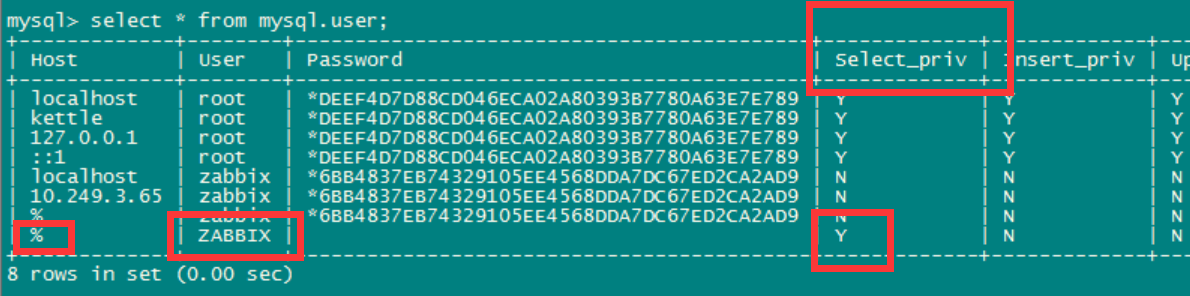
## 建立用户

|  |
| --- |
| mysql> CREATE USER username IDENTIFIED BY 'password'; |

## 给用户赋权

给ZABBIX@%用户访问任何数据库的任何对象的SELECT权限

|  |
| --- |
| mysql> GRANT SELECT ON \*.\* TO ‘ZABBIX’ @'%'; |



给ZABBIX@%用户访问zabbix数据库的任何对象的SELECT权限

|  |
| --- |
| mysql> GRANT SELECT ON ZABBIX.\* TO ‘ZABBIX’ @'%'; |

给ZABBIX@%用户访问zabbix数据库的任何对象的所有权限

|  |
| --- |
| mysql> GRANT ALL PRIVILEGES ON ZABBIX.\* TO ‘ZABBIX’ @'%'; |

给root用户赋予通过某台主机的SELECT权限

|  |
| --- |
| mysql> GRANT SELECT ON \*.\* TO 'ROOT'@'10.249.6.85'; |

回收root用户通过某台主机的SELECT权限

|  |
| --- |
| mysql> REVOKE SELECT ON \*.\* TO 'ROOT'@'10.249.6.85'; |

刷新缓存里的权限

|  |
| --- |
| mysql> FLUSH PRIVILEGES; |

## mysql现有的权限定义

| **Privilege** | **Column** | **Context** |
| --- | --- | --- |
| [ALL [PRIVILEGES]](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_all) | Synonym for “all privileges” | Server administration |
| [ALTER](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_alter) | Alter\_priv | Tables |
| [ALTER ROUTINE](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_alter-routine) | Alter\_routine\_priv | Stored routines |
| [CREATE](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_create) | Create\_priv | Databases, tables, or indexes |
| [CREATE ROUTINE](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_create-routine) | Create\_routine\_priv | Stored routines |
| [CREATE TABLESPACE](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_create-tablespace) | Create\_tablespace\_priv | Server administration |
| [CREATE TEMPORARY TABLES](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_create-temporary-tables) | Create\_tmp\_table\_priv | Tables |
| [CREATE USER](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_create-user) | Create\_user\_priv | Server administration |
| [CREATE VIEW](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_create-view) | Create\_view\_priv | Views |
| [DELETE](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_delete) | Delete\_priv | Tables |
| [DROP](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_drop) | Drop\_priv | Databases, tables, or views |
| [EVENT](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_event) | Event\_priv | Databases |
| [EXECUTE](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_execute) | Execute\_priv | Stored routines |
| [FILE](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_file) | File\_priv | File access on server host |
| [GRANT OPTION](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_grant-option) | Grant\_priv | Databases, tables, or stored routines |
| [INDEX](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_index) | Index\_priv | Tables |
| [INSERT](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_insert) | Insert\_priv | Tables or columns |
| [LOCK TABLES](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_lock-tables) | Lock\_tables\_priv | Databases |
| [PROCESS](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_process) | Process\_priv | Server administration |
| [PROXY](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_proxy) | See proxies\_priv table | Server administration |
| [REFERENCES](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_references) | References\_priv | Databases or tables |
| [RELOAD](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_reload) | Reload\_priv | Server administration |
| [REPLICATION CLIENT](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_replication-client) | Repl\_client\_priv | Server administration |
| [REPLICATION SLAVE](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_replication-slave) | Repl\_slave\_priv | Server administration |
| [SELECT](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_select) | Select\_priv | Tables or columns |
| [SHOW DATABASES](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_show-databases) | Show\_db\_priv | Server administration |
| [SHOW VIEW](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_show-view) | Show\_view\_priv | Views |
| [SHUTDOWN](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_shutdown) | Shutdown\_priv | Server administration |
| [SUPER](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_super) | Super\_priv | Server administration |
| [TRIGGER](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_trigger) | Trigger\_priv | Tables |
| [UPDATE](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_update) | Update\_priv | Tables or columns |
| [USAGE](https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html#priv_usage) | Synonym for “no privileges” | Server administration |

# 数据库

## 创建数据库

|  |
| --- |
| mysql > CREATE DATABASE IF NOT EXISTS DGLJW\_DB DEFAULT CHARSET UTF8 COLLATE UTF8\_GENERAL\_CI; |

# 配置管理

## 查看数据库软件的安装目录

|  |
| --- |
| mysql> select @@basedir; |

## 查看数据库的数据目录

|  |
| --- |
| select @@datadir |

## 查看数据库的参数文件

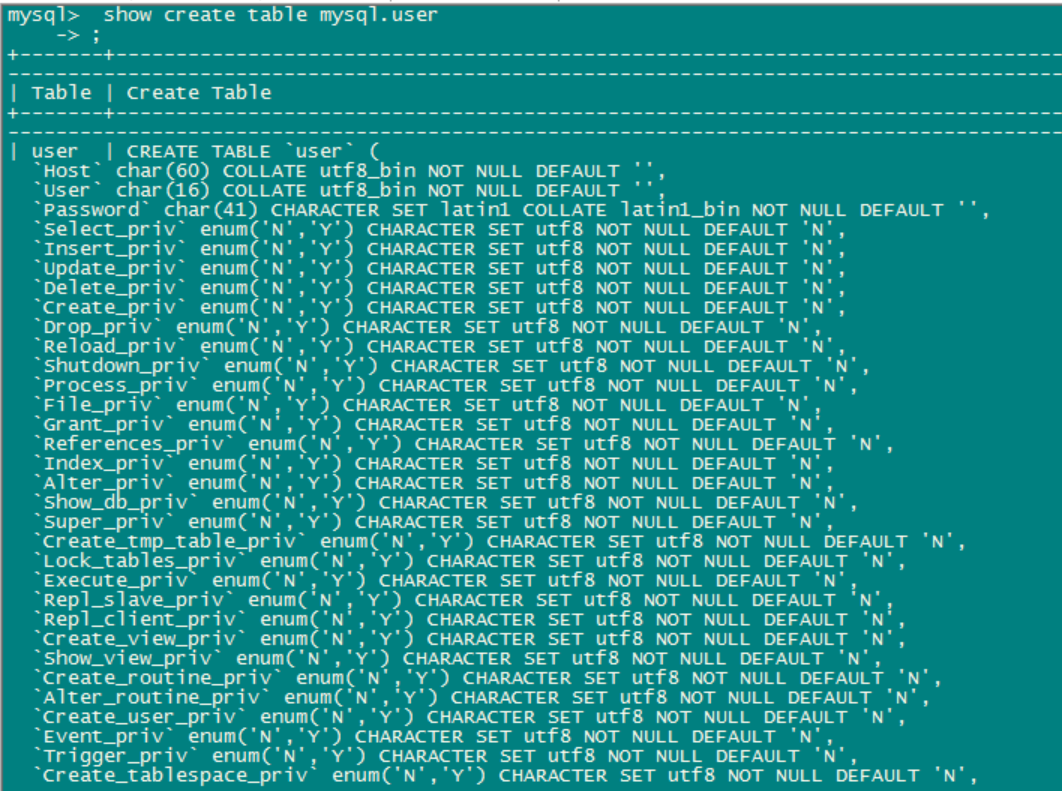
|  |
| --- |
| cat /etc/my.cnf |

# 表

## 建表指定存储引擎

## 查看建表语句

|  |
| --- |
| mysql> show create table mysql.user |



## 修改表的存储引擎

|  |
| --- |
| mysql> alter table mysql.user engine = InnoDB |

## 导出数据字典

|  |
| --- |
| select TABLE\_SCHEMA,TABLE\_NAME,COLUMN\_NAME, COLUMN\_TYPE,COLUMN\_COMMENT from information\_schema.columns where TABLE\_SCHEMA='zabbix' or TABLE\_SCHEMA='zabbix' |

## 统计各个表的大小

|  |
| --- |
| SELECT  table\_name AS "Tables",  round(((data\_length + index\_length) / 1024 / 1024), 2) AS "Size in MB"  FROM  information\_schema.TABLES  WHERE  table\_schema = 'zabbix'  ORDER BY  (data\_length + index\_length) DESC; |

# 并发机制

## 查看是否事务自动提交

show variables like ‘autocommit’

## 起停事务

start transaction;

xxx sql

commit;

# 性能

## 慢日志

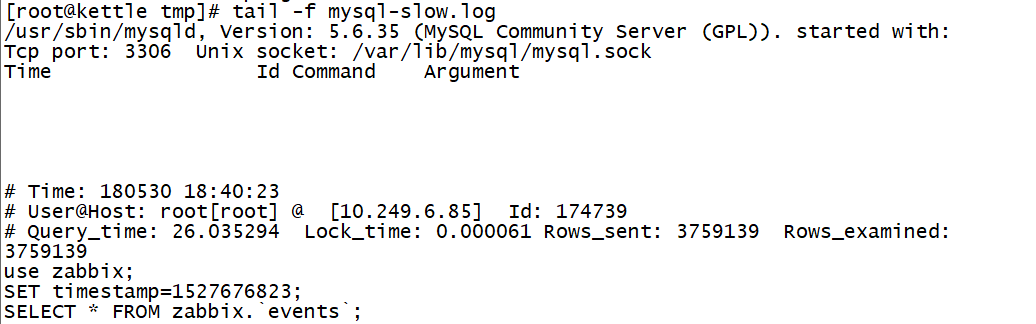
临时开启慢日志

|  |
| --- |
| mysql> set global slow\_query\_log=ON; mysql> set global slow\_query\_log\_file='/tmp/mysql-slow.log'; |

永久开启慢日志

|  |
| --- |
| 修改/etc/my.cnf  [mysqld]  slow\_query\_log = 1  slow\_query\_log\_file = /tmp/mysql-slow.log  long\_query\_time = 1  然后重启mysqld服务 |

慢日志的格式



慢日志文件分析

|  |
| --- |
| mysqldumpslow |

## MySQL数据库性能提升思路

1. 如果使用到InnoDB，将innodb\_buffer\_pool\_size设置得越大越好，甚至到内存的80%。
2. 使用行级别锁，避免表级别或页级别锁

尽量使用支持行级别锁的存储引擎，如InnoDB;只在读操作显著多于写作的场景中(如数据仓库类的应用)使用表级别锁的存储引擎，如MyISAM;。

1. 降低热巨锁(hot gaint lock)出现的可能性以尽可能避免全局互斥量

临界区(仅允许单一线程访问的资源)会严重降低MySQL系统并发性;InnoDB缓冲池(buffer pool)、数据字典等都是常见的临界区;幸运的是，新版本的InnoDB已经能够较好的运行于多核处理器，支持使用 innodb\_buffer\_pool\_instances服务器变量建立多个缓冲池实例，每个缓冲池实例分别自我管理空闲列表、列表刷写、LRU以及其它跟缓冲池相关的数据结构，并通过各自的互斥锁进行保护。

1. 并行运行多个I/O线程

通过innodb\_io\_capacity服务器变量等增加磁盘I/O线程的数量可以提高前端操作(如SELECT)的性能，不过，磁盘I/O线程的数量不应该超过磁盘的IOPS(7200RPM的单块硬件的IOPS数量一般为100个左右)。

异步I/O也可以在一定程度上提高系统的并发能力，在Linux系统上，可以通过将MySQL的服务器变量innodb\_use\_native\_aio的值设定为ON设定InnoDB可以使用Linux的异步I/O子系统。

1. 并行后端任务

默认情况下，MySQL的清写(purge)操作(用于移除带删除标记的记录)由InnoDB的主线程完成，这可以降低内部资源竞争发生的概率，进而增强MySQL服务伸缩能力。不过，随着InnoDB内部各式各样的竞争越来越多，这种设置带来的性能优势已几乎不值一提，因此，生产环境中应该通过为innodb\_purge\_threads服务器变量设定为ON将主线程与清写线程分开运行。

1. 单线程复制模型中的SQL线程是一个热区

在从服务器上并行运行多个SQL线程可有效提高MySQL从服务器性能，MySQL 5.6支持多线程复制(每库一个复制线程);

# 存储引擎

## mysql的存储引擎

InnoDB引擎：

1.将数据存储在表空间中，表空间由一系列的数据文件组成，由InnoDB管理；

2.支持每个表的数据和索引存放在单独文件中(innodb\_file\_per\_table)；

3.支持事务，采用MVCC来控制并发，并实现标准的4个事务隔离级别，支持外键；

4.索引基于聚簇索引建立，对于主键查询有较高性能；

5.数据文件的平台无关性，支持数据在不同的架构平台移植；

6.能够通过一些工具支持真正的热备。如XtraBackup等；

7.内部进行自身优化如采取可预测性预读，能够自动在内存中创建hash索引等。

MyISAM引擎：

1.MySQL5.1中默认，不支持事务和行级锁；

2.提供大量特性如全文索引、空间函数、压缩、延迟更新等；

3.数据库故障后，安全恢复性差；

4.对于只读数据可以忍受故障恢复，MyISAM依然非常适用；

5.日志服务器的场景也比较适用，只需插入和数据读取操作；

6.不支持单表一个文件，会将所有的数据和索引内容分别存在两个文件中；

7.MyISAM对整张表加锁而不是对行，所以不适用写操作比较多的场景；

8.支持索引缓存不支持数据缓存。

Archive引擎：

1.只支持insert和select操作；

2.缓存所有的写数据并进行压缩存储，支持行级锁但不支持事务；

3.适合高速插入和数据压缩，减少IO操作，适用于日志记录和归档服务器。

Blackhole引擎：

1.没有实现任何存储机制，会将插入的数据进行丢弃，但会存储二进制日志；

2.会在一些特殊需要的复制架构的环境中使用。

CSV引擎：

1.可以打开CSV文件存储的数据，可以将存储的数据导出，并利用excel打开；

2.可以作为一种数据交换的机制，同样经常使用。

Memory引擎：

1.将数据在内存中缓存，不消耗IO；

2.存储数据速度较快但不会被保留，一般作为临时表的存储被使用。

Federated引擎：

能够访问远程服务器上的数据的存储引擎。能够建立一个连接连到远程服务器。

Mrg\_MyISAM引擎：

将多个MYISAM表合并为一个。本身并不存储数据，数据存在MyISAM表中间。

NDB集群引擎：

MySQL Cluster专用。

以下这些是第三方存储引擎

XtraDB：

InnoDB的改进版本。

PBXT：

类似InnoDB，但提供引擎级别的复制和外键约束，适当支持SSD存储。

TokuDB(开源)：

支持分形树索引结构，支持海量数据的分析。

Infobright:

支持数十TB的数据量，为数据分析和数据仓库设计的。数据高度压缩。

InfiniDB:

可以在一组集群间做分布式查询，有商业版但没有典型应用案例。

Aria：

解决MyISAM崩溃安全恢复问题，并能够进行数据缓存。

Groona:

全文索引引擎。

QQGraph:

由Open query研发支持图操作，比如查找两点间最短距离。

SphinxSE:

该引擎为Sphinx全文索引搜索服务器提供SQL接口。

Spider:

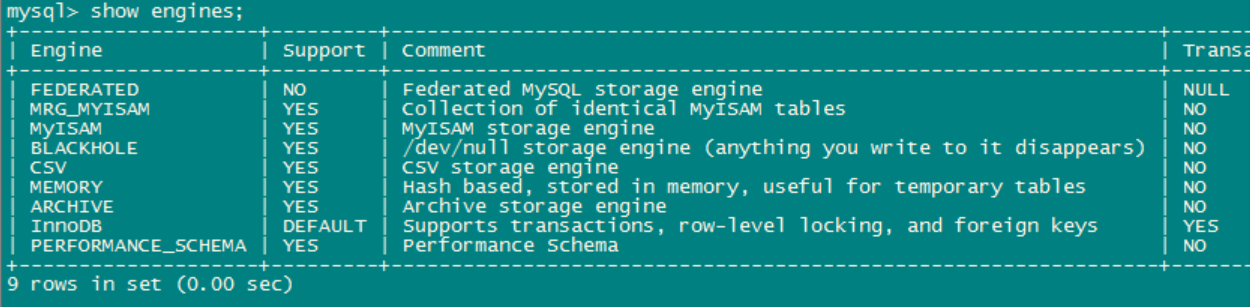
支持sharding并能够基于分片实现并列查询。

VPForMySQL:

支持垂直分区。

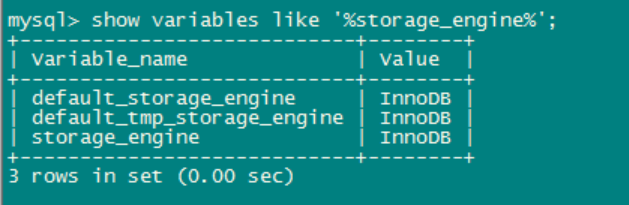
## 当前数据库支持什么存储引擎

|  |
| --- |
| mysql> show engines; |



## 当前数据库默认使用的存储引擎

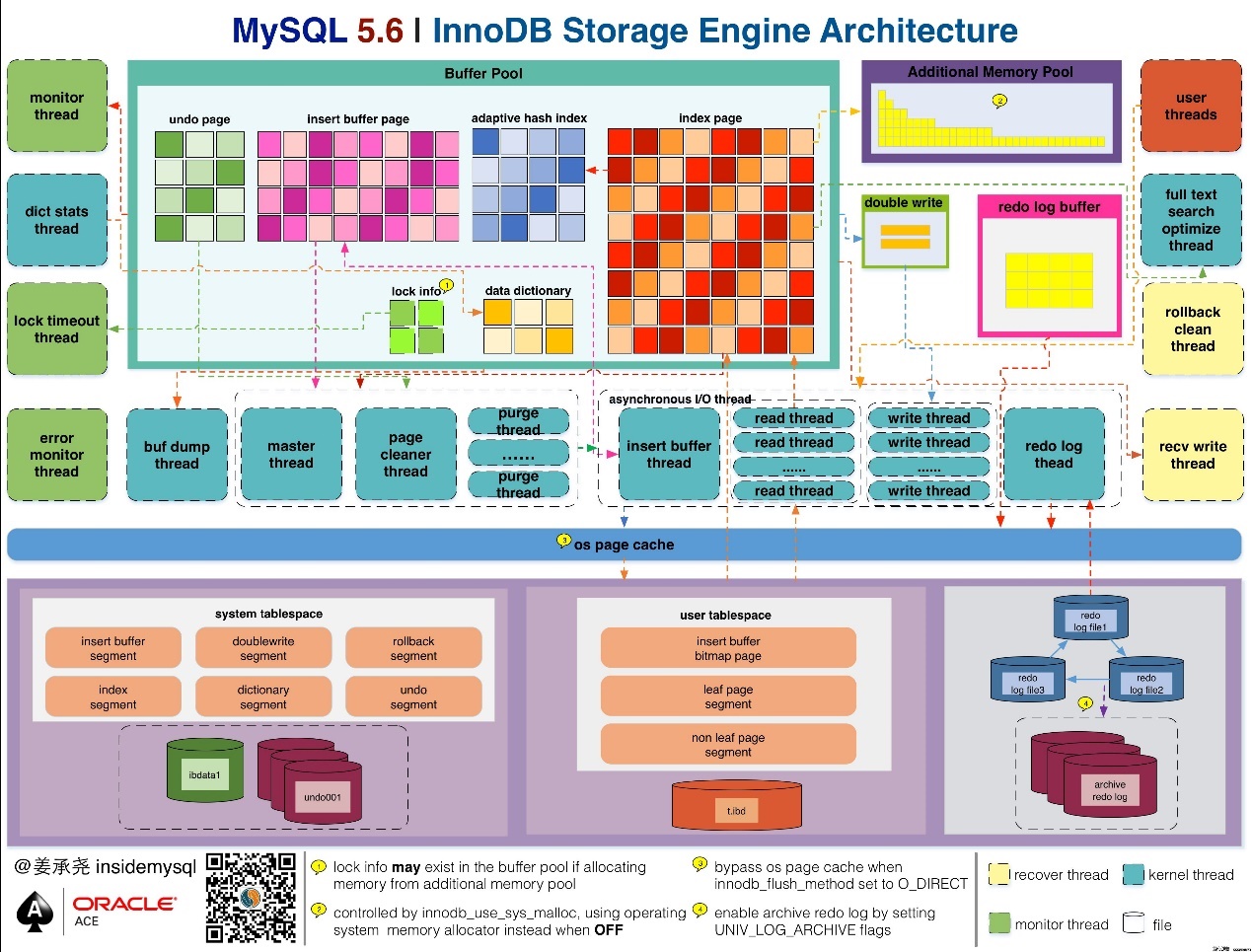
|  |
| --- |
| mysql> show variables like '%storage\_engine%'; |



## 表引擎转换方法

alter table as engine = InnoDB;

## Innodb



# 备份与恢复

## 原生mysql命令执行全库备份

mysqldump -uroot -pzabbix zabbix>/zabbix.sql

## 原生mysql命令备份表结构

mysqldump --no-data --databases zabbix sonar > test.dump

## 数据库恢复

source xxxx.sql

# 数据迁移

# 编码问题