**[四种mysql存储引擎](http://www.cnblogs.com/wcwen1990/p/6655416.html)**

**前言**

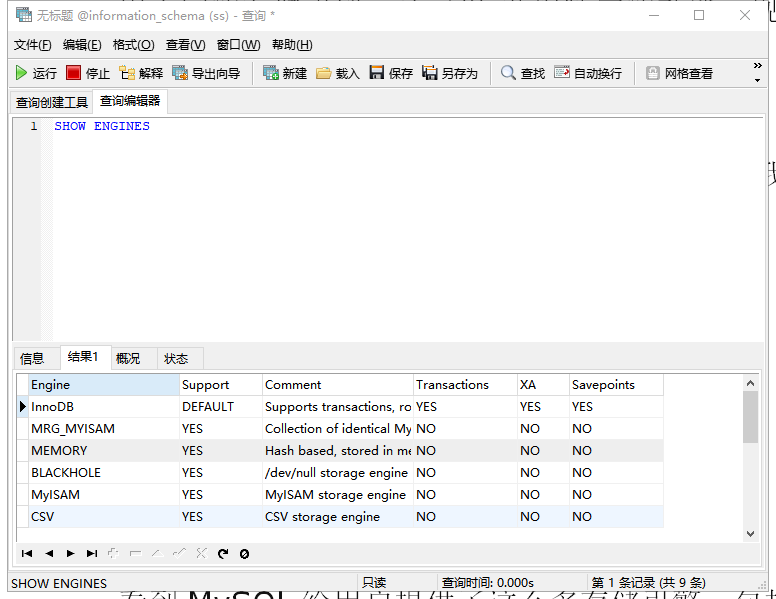
[**数据库**](http://lib.csdn.net/base/mysql)存储引擎是数据库底层软件组织，数据库管理系统（DBMS）使用数据引擎**进行创建、查询、更新和删除数据**。不同的存储引擎提供不同的存储机制、索引技巧、锁定水平等功能，使用不同的存储引擎，还可以 获得特定的功能。现在许多不同的数据库管理系统都支持多种不同的数据引擎。[**MySQL**](http://lib.csdn.net/base/mysql)**的核心就是存储引擎**。

**存储引擎查看**

MySQL给开发者提供了查询存储引擎的功能，我这里使用的是MySQL5.1，可以使用：

SHOW ENGINES

命令来查看MySQL使用的引擎，命令的输出为（我用的Navicat Premium）：



看到MySQL给用户提供了这么多存储引擎，包括处理**事务安全表的引擎和出来了非事物安全表的引擎。**

如果要想查看数据库默认使用哪个引擎，可以通过使用命令：

SHOW VARIABLES LIKE 'storage\_engine';

来查看，查询结果为：

[wps165E.tmp](http://images2015.cnblogs.com/blog/669905/201704/669905-20170401120009602-887472830.jpg)

在MySQL中，不需要在整个服务器中使用同一种存储引擎，针对具体的要求，可以对每一个表使用不同的存储引擎**。Support列的值表示某种引擎是否能使用：YES表示可以使用、NO表示不能使用、DEFAULT表示该引擎为当前默认的存储引擎 。**下面来看一下其中几种常用的引擎。

**InnoDB存储引擎(默认)**

**InnoDB是事务型数据库的首选引擎，支持事务安全表（ACID），支持行锁定和外键，**上图也看到了，InnoDB是默认的MySQL引擎。InnoDB主要特性有：

1、InnoDB给MySQL提供了具有**提交、回滚和崩溃恢复能力**的**事务**安全（ACID兼容）存储引擎。InnoDB锁定在行级并且也在**SELECT语句中提供一个类似[Oracle](http://lib.csdn.net/base/oracle)的非锁定读**。这些功能增加了多用户部署和性能。在SQL查询中，可以自由地将InnoDB类型的表和其他MySQL的表类型混合起来，甚至在同一个查询中也可以混合

2、InnoDB是为**处理巨**[**大数据**](http://lib.csdn.net/base/hadoop)**量的最大性能设计**。它的CPU效率可能是任何其他基于磁盘的关系型数据库引擎锁不能匹敌的

3、InnoDB存储引擎完全与MySQL服务器整合，InnoDB存储引擎为在主内存中缓存数据和索引而维持它自己的缓冲池。InnoDB将它的**表和索引在一个逻辑表空间中（就是索引的key和data放在一起，聚集索引），表空间可以包含数个文件（或原始磁盘文件）。**这与MyISAM表不同，比如在MyISAM表中**每个表被存放在分离的文件中（索引的key和data不放在一起，而data存的是地址的指针，非聚集索引）**。InnoDB表可以是任何尺寸，即使在文件尺寸被限制为2GB的**[操作系统](http://lib.csdn.net/base/operatingsystem)**上

4、InnoDB**支持外键完整性约束**，存储表中的数据时，每张表的存储都按主键顺序存放，如果没有显示在表定义时指定主键，InnoDB会为每一行生成一个6字节的ROWID，并以此作为主键

5、InnoDB被用在众多需要高性能的大型数据库站点上

InnoDB不创建目录，使用InnoDB时，MySQL将在MySQL数据目录下创建一个名为ibdata1的10MB大小的自动扩展数据文件，以及两个名为ib\_logfile0和ib\_logfile1的5MB大小的日志文件

**MyISAM存储引擎**

MyISAM基于ISAM存储引擎，并对其进行扩展。它是在Web、数据仓储和其他应用环境下**最常使用的存储引擎之一**。MyISAM拥有**较高的插入、查询速度**，但**不支持事务**。MyISAM主要特性有：

1、大文件（达到63位文件长度）在支持大文件的文件系统和操作系统上被支持

2、当把删除和更新及插入操作混合使用的时候，动态尺寸的行产生更少碎片。这要通过合并相邻被删除的块，以及若下一个块被删除，就扩展到下一块自动完成

3、**每个MyISAM表最大索引数是64**，这可以通过重新编译来改变。每个索引最大的列数是16

4、最大的键长度是1000字节，这也可以通过编译来改变，对于键长度超过250字节的情况，一个超过1024字节的键将被用上

5、BLOB和TEXT列可以被索引

6、NULL被允许在索引的列中，这个值占每个键的0~1个字节

7、所有数字键值以高字节优先被存储以允许一个更高的索引压缩

8、每个MyISAM类型的表都有一个AUTO\_INCREMENT的内部列，当INSERT和UPDATE操作的时候该列被更新，同时AUTO\_INCREMENT列将被刷新。所以说，MyISAM类型表的AUTO\_INCREMENT列更新比InnoDB类型的AUTO\_INCREMENT更快

9、**可以把数据文件和索引文件放在不同目录**

10、每个字符列可以有不同的字符集

11、有VARCHAR的表可以固定或动态记录长度

12、VARCHAR和CHAR列可以多达64KB

使用MyISAM引擎创建数据库，将产生3个文件。文件的名字以表名字开始，扩展名之处文件类型：frm文件存储表定义、数据文件的扩展名为.MYD（MYData）、索引文件的扩展名时.MYI（MYIndex）

**MEMORY存储引擎**

MEMORY存储引擎将表中的数据存储到内存中，未查询和引用其他表数据提供快速访问。MEMORY主要特性有：

1、MEMORY表的每个表可以有多达32个索引，每个索引16列，以及500字节的最大键长度

2、MEMORY存储引擎执行HASH和BTREE缩影

3、可以在一个MEMORY表中有非唯一键值

4、MEMORY表使用一个固定的记录长度格式

5、MEMORY不支持BLOB或TEXT列

6、MEMORY支持AUTO\_INCREMENT列和对可包含NULL值的列的索引

7、MEMORY表在所由客户端之间共享（就像其他任何非TEMPORARY表）

8、MEMORY表内存被存储在内存中，内存是MEMORY表和服务器在查询处理时的空闲中，创建的内部表共享

9、当不再需要MEMORY表的内容时，要释放被MEMORY表使用的内存，应该执行DELETE FROM或TRUNCATE TABLE，或者删除整个表（使用DROP TABLE）

**存储引擎的选择**

不同的存储引擎都有各自的特点，以适应不同的需求，如下表所示：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **特点** | **InnoDB** | **Myisam** | **Memory** | **Archive** | **BDB** |
| 存储限制 | 64TB | 没有 | 有 | 没有 | 没有 |
| 事务安全 | 支持 |  |  |  | 支持 |
| 锁机制 | 行锁 | 表锁 | 表锁 | 行锁 | 页锁 |
| B树索引 | 支持 | 支持 | 支持 |  | 支持 |
| 哈希索引 | 支持 |  | 支持 |  |  |
| 全文索引 |  | 支持 |  |  |  |
| 集群索引 | 支持 |  |  |  |  |
| 数据缓存 | 支持 |  | 支持 |  |  |
| 索引缓存 | 支持 | 支持 | 支持 |  |  |
| 数据可压缩 |  | 支持 |  | 支持 |  |
| 空间使用 | 高 | 低 | N/A | 非常低 | 低 |
| 内存使用 | 高 | 低 | 中等 | 低 | 低 |
| 批量插入的速度 | 低 | 高 | 高 | 非常高 | 高 |
| 支持外键 | 支持 |  |  |  |  |

**如果要提供提交、回滚、崩溃恢复能力的事物安全（ACID兼容）能力，并要求实现并发控制，InnoDB是一个好的选择**

如果**数据表**主要**用来插入和查询记录，则MyISAM引擎能提供较高的处理效率**

如果**只是临时存放数据，数据量不大，并且不需要较高的数据安全性，可以选择将数据保存在内存中的Memory引擎，MySQL中使用该引擎作为临时表，存放查询的中间结果**

如果**只有INSERT和SELECT操作，可以选择Archive，Archive支持高并发的插入操作**，但是本身不是事务安全的。Archive非常适合存储归档数据，如**记录日志信息可以使用Archive**

使用哪一种引擎需要灵活选择，**一个数据库中多个表可以使用不同引擎以满足各种性能和实际需求，**使用合适的存储引擎，将会提高整个数据库的性能

两种索引的功能和结构都是不同的  
**普通索引的结构主要以B+树和哈希索引为主，用于实现对字段中数据的精确查找，比如查找某个字段值等于给定值的记录，A=10这种查询，因此适合数值型字段和短文本字段  
全文索引是用于检索字段中是否包含或不包含指定的关键字**，有点像搜索引擎的功能，其**内部的索引结构采用的是与搜索引擎相同的**[**倒排索引**](https://www.baidu.com/s?wd=%E5%80%92%E6%8E%92%E7%B4%A2%E5%BC%95&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3P1FbPH7-ujb4ryFbPAF-0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnH6krjDzrjmsrj03rjDdrjR3r0)**结构**，其原理是对字段中的文本进行分词，**然后为每一个出现的单词记录一个索引项**，这个索引项中保存了所有出现过该单词的记录的信息，也就是说在索引中找到这个单词后，就知道哪些记录的字段中包含这个单词了。因此适合用大文本字段的查找。  
  
大字段之所以不适合做普通索引，最主要的原因是普通索引对检索条件只能进行精确匹配，而大字段中的文本内容很多，通常也不会在这种字段上执行精确的文本匹配查询，而更多的是基于关键字的[全文检索](https://www.baidu.com/s?wd=%E5%85%A8%E6%96%87%E6%A3%80%E7%B4%A2&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3P1FbPH7-ujb4ryFbPAF-0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnH6krjDzrjmsrj03rjDdrjR3r0)查询，例如你查一篇文章信息，你会只输入一些关键字，而不是把整篇文章输入查询（如果有整篇文章也就不用查询了）。而全文索引正是适合这种查询需求。