MySQL 表分区

MySQL 表分区和分库分表一样，都是为了提高数据库的吞吐量。分区类似与分表，分表是逻辑上将一个大数据量的表分成多个，可以是水平分也可以是垂直分。而分区是将表的一个数据文件拆分成多个。不同的数据拆分到不同的文件中。这样对于一个数据量非常大的表，有多个数据文件来进行存储，这样就提高了数据库的 io 性能。

既然是针对的数据表的文件进行操作，那么我们就需要先来了解 MySQL 表的存储。我们知道，MySQL 有多种存储引擎，不同的存储引擎所存储的文件格式不同。这里主要以 InnoDB 和 MyISAM 这两种存储引擎来说明。

InnoDB

.frm 文件 数据表的结构

.idb 文件 表的数据文件，独享表空间，每个表有一个.idb 文件

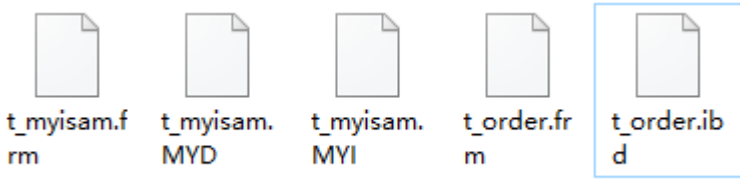
.ibdata 文件 表的数据文件，共享表空间，所有的表使用这一个数据文件

MyISAM

.frm 文件 数据表的结构

.myd 文件 数据文件

.myi 文件 索引文件



首先要查看一下我们当前的数据库版本是否支持分区

```
show variables like '%partition%';
```

如何进行分区呢？在进行数据库水平切分的时候我们知道，水平切分可以根据指定字段取模的方式来分到不同的表中，也可以根据日期来进行切分，或者根据 id 来分段，1-100 万在第一张表中，100 万零 1 到 200 万在第二张表中以此类推等等。总之我们在进行切分的过程中有很多的途径。那么在表分区上数据库也给我们提供了多种方案可供我们选择。

MySQL 表分区策略

RANGE 分区 基于属于一个给定连续区间的列值，把多行分配给分区。

```
DROP TABLE IF EXISTS `p_range`;
CREATE TABLE `p_range` (
`id` int(10) NOT NULL AUTO_INCREMENT,
`name` char(20) NOT NULL,
```

```
PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=9 DEFAULT CHARSET=utf8
/*!50100 PARTITION BY RANGE (id)
(PARTITION p0 VALUES LESS THAN (8) ENGINE = MyISAM) */;
```

最大值

```
PARTITION BY RANGE (id)
(
PARTITION p0 VALUES LESS THAN (8),
PARTITION p1 VALUES LESS THAN MAXVALUE)
```

适用场景：

这样就表示，所有 id 大于 7 的数据记录存在在 p1 分区里。

RANGE 分区在如下场合特别有用：

- 当需要删除“旧的”数据时。如果你使用上面最近的那个例子给出的分区方案，你只需简单地使用 “ALTER TABLE employees DROP PARTITION p0；”来删除所有在 1991 年前就已经停止工作的雇员相对应的所有行。对于有大量行的表，这比运行一个如“DELETE FROM employees WHERE YEAR(separated) <= 1990；”这样的一个 DELETE 查询要有效得多。
- 想要使用一个包含有日期或时间值，或包含有从一些其他级数开始增长的值的列。
- 经常运行直接依赖于用于分割表的列的查询。例如，当执行一个如“SELECT COUNT(*) FROM employees WHERE YEAR(separated) = 2000 GROUP BY store_id；”这样的查询时，MySQL 可以很迅速地确定只有分区 p2 需要扫描，这是因为余下的分区不可能包含有符合该 WHERE 子句的任何记录

LIST 分区

类似于按 RANGE 分区，区别在于 LIST 分区是基于列值匹配一个离散值集合中的某个值来进行选择。

```
DROP TABLE IF EXISTS `p_list`;
CREATE TABLE `p_list` (
`id` int(10) NOT NULL AUTO_INCREMENT,
`typeid` mediumint(10) NOT NULL DEFAULT '0',
`typename` char(20) DEFAULT NULL,
PRIMARY KEY (`id`,`typeid`)
) ENGINE=MyISAM AUTO_INCREMENT=9 DEFAULT CHARSET=utf8
/*!50100 PARTITION BY LIST (typeid)
(PARTITION p0 VALUES IN (1,2,3,4) ENGINE = MyISAM,
PARTITION p1 VALUES IN (5,6,7,8) ENGINE = MyISAM) */;
```

HASH 分区

基于用户定义的表达式的返回值来进行选择的分区，该表达式使用将要插入到表中的这些行的列值进行计算。这个函数可以包含 MySQL 中有效的、产生非负整数值的任何表达式。 HASH 分区主要用来确保数据在预先确定数目的分区中平均分布。在 RANGE 和 LIST 分区中，必须明确指定一个给定的列值或列值集合应该保存在哪个分区中；而在 HASH 分区中，MySQL 自动完成这些工作，你所要做的只是基于将要被哈希的列值指定一个列值或表达式，以及指定被分区的表将要被分割成的分区数量

```
DROP TABLE IF EXISTS `p_hash`;
CREATE TABLE `p_hash` (
`id` int(10) NOT NULL AUTO_INCREMENT,
`storeid` mediumint(10) NOT NULL DEFAULT '0',
`storename` char(255) DEFAULT NULL,
PRIMARY KEY (`id`,`storeid`)
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=utf8
/*!50100 PARTITION BY HASH (storeid)
PARTITIONS 4 */;
```

简单点说就是数据的存入可以按 partition by hash(expr)；这里的 expr 可以是键名也可以是表达式比如 YEAR(time)，如果是表达式的情况下
“但是应当记住，每当插入或更新（或者可能删除）一行，这个表达式都要计算一次；这意味着非常复杂的表达式可能会引起性能问题，尤其是在执行同时影响大量行的运算（例如批量插入）的时候。 ”
在执行删除、写入、更新时这个表达式都会计算一次。
数据的分布采用基于用户函数结果的模数来确定使用哪个编号的分区。换句话，对于一个表达式“expr”，将要保存记录的分区编号为 N ，其中“N = MOD(expr, num)”。
比如上面的 storeid 为 10；那么 N=MOD(10,4) ;N 是等于 2 的，那么这条记录就存储在 p2 的分区里面。
如果插入一个表达式列值为'2005-09-15′ 的记录到表中，那么保存该条记录的分区确定如下：MOD(YEAR('2005-09-01′),4) = MOD(2005,4) = 1 ；就存储在 p1 分区里面了。

分区注意点
1、重新分区时，如果原分区里面存在 maxvalue 则新的分区里面也必须包含 maxvalue 否则就错误。

```
alter table p_range2x
reorganize partition p1,p2
into (partition p0 values less than (5), partition p1 values less
than maxvalue);
[Err] 1520 - Reorganize of range partitions cannot change total
ranges except for last partition where it can extend the range
```

2、分区删除时，数据也同样会被删除
alter table p_range drop partition p0;
3、如果 range 分区列表里面没有 maxvalue 则如有新数据大于现在分区 range 数据值那么这个数据是无法写入到数据库表的。
4、修改表名不需要 删除分区后在进行更改，修改表名后分区存储 myd myi 对应也会自动更改。
如果希望从所有分区删除所有的数据，但是又保留表的定义和表的分区模式，使用 TRUNCATE TABLE 命令。（请参见 13.2.9 节，“TRUNCATE 语法”）。
如果希望改变表的分区而又不丢失数据，使用“ALTER TABLE … REORGANIZE PARTITION”语句。参见下面的内容，或者在 13.1.2 节，“ALTER TABLE 语法” 中参考关于 REORGANIZE PARTITION 的信息。
5、对表进行分区时，不论采用哪种分区方式如果表中存在主键那么主键必须在分区列中。表分区的局限性。
6、list 方式分区没有类似于 range 那种 less than maxvalue 的写法，也就是说 list 分区表的所有数据都必须在分区字段的值列表集合中。
7、在 MySQL 5.1 版中，同一个分区表的所有分区必须使用同一个存储引擎；例如，不能对一个分区使用 MyISAM，而对另一个使用 InnoDB。
8、分区的名字是不区分大小写的，myp1 与 MYp1 是相同的。