14 你应该知道的系统数据库及常用系统表

更新时间：2020-04-01 09:51:37

> 时间像海绵里的水，只要你愿意挤，总还是有的。——鲁迅

我们刚刚安装完 MySQL 之后，你就可以发现，此时已经有了四个库：information_schema、mysql、performance_schema 和 sys。它们被称之为系统库，用于记录或收集 MySQL 服务器的相关信息。你不需要把这些库中的每一张表都搞清楚，这很难，毕竟有200多张表（还包括视图、函数、存储过程）。虽然，绝大多数情况下，这些表是为 DBA 准备的，但是，理解它们并且能做到简单的应用，一定让你能够更懂 MySQL。

1 MySQL 的数据字典 - information_schema

information_schema 库提供了对数据库元数据（服务器中的所有库名、表名、字段数据类型、访问权限等等）、统计信息、以及有关 MySQL 服务器的信息访问。所以，这个库也被称为 MySQL 的数据字典。

1.1 认识 information_schema

在每个 MySQL 实例中都会存在 information_schema 库，用于存储 MySQL 实例中所有其他数据库的信息。这个库中包含多个只读表（非持久化表，不能执行 INSERT、UPDATE、DELETE 等数据变更操作），所以，我们不能在磁盘中找到关于它的数据文件，同时，也不能对这些表设置触发器。

我们想要知道当前 MySQL 实例中定义了哪些库，可以使用 SHOW DATABASES 语句；想要知道某一个库中定义了哪些数据表，可以使用 SHOW TABLES 语句。其实，这些数据都是保存在 information_schema 库中的，也就当然可以通过查表的方式获取。另外，对于查询 information_schema 中的表去获取元数据，还有以下优点：

- 因为经常用，所以会让你更加熟悉 information_schema
- 获取数据通过访问表的方式，符合 “Codd 法则”（科德十二定律）
- 使用 SELECT 的语法，可以定义个性化条件，例如：WHERE、ORDER BY

当然，以上讨论的优点并不是证明 SHOW 语句不好，只是说还有另一种方式可以替代 SHOW。最后，虽然所有的用户都可以去访问 information_schema 中的表，但是只能看到这些表中用户具有访问权限的行。所以，当查询出错或者返回 NULL，需要确定下是否有相应的权限。

1.2 information_schema 的组成

我们先来看一看 information_schema 库中有多少张表（需要注意，不同版本的 MySQL，库中的表不一定一样），有意思的是，我们查询的正是 information_schema 库中的表。如下所示：

```
mysql> SELECT COUNT(*) TABLES, table_schema FROM information_schema.TABLES WHERE table_schema = 'information_schema' GROUP BY
table_schema;
+--------+--------------------+
| TABLES | table_schema       |
+--------+--------------------+
|     61 | information_schema |
+--------+--------------------+
```

可以看到，information_schema 库中定义了61张表，确实是不少。这其中，有10张 InnoDB 存储引擎临时表（数据字典表），51张 Memory 引擎临时表。且由于它们是临时表，在数据库重启之后数据就会丢失，这个库也是唯一一个在文件系统上没有对应库表的目录和文件的系统库 。

下面，我们去大致了解下库中都有哪些表，这些表又会有怎样的用途（由于表很多，会根据用途的相似度进行分类讲解）。

MySQL 实例统计信息字典表

- COLUMNS：InnoDB 存储引擎临时表，记录表中的列信息
- KEY_COLUMN_USAGE：Memory 引擎临时表，记录索引列存在的约束条件
- REFERENTIAL_CONSTRAINTS：Memory 引擎临时表，记录外键约束
- STATISTICS：Memory 引擎临时表，记录索引信息
- TABLE_CONSTRAINTS：Memory 引擎临时表，记录表约束信息
- FILES：Memory 引擎临时表，记录 MySQL 数据表空间文件相关的信息
- ENGINES：Memory 引擎临时表，记录 MySQL 服务器支持的引擎的相关信息
- TABLESPACES：Memory 引擎临时表，记录 NDB 存储引擎表空间相关的信息
- SCHEMATA：Memory 引擎临时表，记录数据库信息

MySQL 实例表级别对象字典表

- VIEWS：InnoDB 存储引擎临时表，记录视图信息
- TRIGGERS：InnoDB 存储引擎临时表，记录某个数据库下的触发器信息
- TABLES：Memory 引擎临时表，记录表信息
- ROUTINES：InnoDB 存储引擎临时表，记录存储过程和函数的信息
- PARTITIONS：InnoDB 存储引擎临时表，记录分区表的信息
- EVENTS：InnoDB 存储引擎临时表，记录计划任务事件信息
- PARAMETERS：InnoDB 存储引擎临时表，记录存储过程和函数的参数信息，以及有关函数的返回值信息

MySQL 实例混合信息字典表

- GLOBAL_STATUS、GLOBAL_VARIABLES、SESSION_STATUS、SESSION_VARIABLES：Memory 引擎临时表，记录全局、会话级别的状态变量与系统变量信息
- OPTIMIZER_TRACE：InnoDB 存储引擎临时表，记录优化程序跟踪功能产生的信息
- PLUGINS：InnoDB 存储引擎临时表，记录 MySQL 支持哪些插件
- PROCESSLIST：InnoDB 存储引擎临时表，记录线程运行过程中的状态信息
- PROFILING：Memory 引擎临时表，记录查询关于语句性能分析的信息
- CHARACTER_SETS：Memory 引擎临时表，记录 MySQL 支持的可用字符集
- COLLATIONS：Memory 引擎临时表，记录 MySQL 支持的可用校对规则
- COLLATION_CHARACTER_SET_APPLICABILITY：Memory 引擎临时表，记录 MySQL 中哪种字符集适用于什么校对规则
- COLUMN_PRIVILEGES：Memory 引擎临时表，记录列的权限信息
- SCHEMA_PRIVILEGES：Memory 引擎临时表，记录库的权限信息
- TABLE_PRIVILEGES：Memory 引擎临时表，记录表的权限信息
- USER_PRIVILEGES：Memory 引擎临时表，记录全局权限信息

InnoDB 系统字典表

- INNODB_SYS_DATAFILES：Memory 引擎临时表，记录 InnoDB 所有表空间类型的元数据
- INNODB_SYS_VIRTUAL：Memory 引擎临时表，记录有关 InnoDB 虚拟生成列和与之关联的列的元数据信息
- INNODB_SYS_INDEXES：Memory 引擎临时表，记录有关 InnoDB 索引的元数据信息
- INNODB_SYS_TABLES：Memory 引擎临时表，记录有关 InnoDB 表的元数据
- INNODB_SYS_FIELDS：Memory 引擎临时表，记录有关 InnoDB 索引列的元数据信息
- INNODB_SYS_TABLESPACES：Memory 引擎临时表，记录有关 InnoDB 独立表空间和普通表空间的元数据信息
- INNODB_SYS_FOREIGN_COLS：Memory 引擎临时表，记录有关 InnoDB 外键列的状态信息
- INNODB_SYS_COLUMNS：Memory 引擎临时表，记录有关 InnoDB 表列的元数据信息
- INNODB_SYS_FOREIGN：Memory 引擎临时表，记录有关 InnoDB 外键的元数据信息
- INNODB_SYS_TABLESTATS：Memory 引擎临时表，记录有关 InnoDB 表的较低级别的状态信息视图

InnoDB 锁、事务、统计信息字典表

- INNODB_LOCKS：Memory 引擎临时表，记录 InnoDB 引擎事务中正在请求的且并未获得的且同时阻塞了其他事务的锁信息
- INNODB_TRX：Memory 引擎临时表，记录 InnoDB 引擎中执行的事务的信息
- INNODB_BUFFER_PAGE_LRU：Memory 引擎临时表，记录缓冲池中的页面信息
- INNODB_LOCK_WAITS：Memory 引擎临时表，记录关于每个被阻塞的 InnoDB 事务的锁等待记录
- INNODB_TEMP_TABLE_INFO：Memory 引擎临时表，记录在 InnoDB 实例中处于活动状态的用户创建的 InnoDB 临时表信息
- INNODB_BUFFER_PAGE：Memory 引擎临时表，记录关于 buffer pool 中的页相关信息
- INNODB_METRICS：Memory 引擎临时表，记录 InnoDB 更为细致的性能信息
- INNODB_BUFFER_POOL_STATS：Memory 引擎临时表，记录 InnoDB buffer pool 中的状态信息

InnoDB 全文索引字典表

- INNODB_FT_CONFIG：Memory 引擎临时表，记录 InnoDB 表的 FULLTEXT 索引和关联的元数据信息
- INNODB_FT_BEING_DELETED：Memory 引擎临时表，它仅在 OPTIMIZE TABLE 语句执行维护操作期间作为 INNODB_FT_DELETED 表的快照数据存放使用
- INNODB_FT_DELETED：Memory 引擎临时表，记录 InnoDB 表的 FULLTEXT 索引中删除的行信息
- INNODB_FT_DEFAULT_STOPWORD：Memory 引擎临时表，默认的全文索引停用词表，记录停用词列表值
- INNODB_FT_INDEX_TABLE：Memory 引擎临时表，记录 InnoDB 表全文索引中用于反向文本查找的倒排索引的分词信息
- INNODB_FT_INDEX_CACHE：Memory 引擎临时表，记录包含 FULLTEXT 索引的 InnoDB 存储引擎表中新插入行的全文索引标记信息

InnoDB 压缩相关字典表

- INNODB_CMP、INNODB_CMP_RESET：Memory 引擎临时表，包含了与压缩的 InnoDB 表页有关的操作状态信息
- INNODB_CMP_PER_INDEX、INNODB_CMP_PER_INDEX_RESET：Memory 引擎临时表，记录 InnoDB 压缩表数据和索引相关的操作状态信息
- INNODB_CMPMEM、INNODB_CMPMEM_RESET：Memory 引擎临时表，记录 InnoDB 缓冲池中压缩页上的状态信息

1.3 information_schema 中的常用表

虽然 information_schema 中定义了61张表，但是，我们日常工作中能够使用到的并不多。下面，我以 SQL 查询的方式讲解库中的常用表及其使用方法。

```
-- 查询表相关的信息
SELECT * FROM information_schema.TABLES WHERE TABLE_NAME = ...

-- 查询列相关的信息
SELECT * FROM information_schema.COLUMNS

-- 查询表主键
SELECT * FROM information_schema.KEY_COLUMN_USAGE WHERE table_schema = ... AND table_name = ...

-- 查询表的约束
SELECT * FROM information_schema.TABLE_CONSTRAINTS

-- 查询表的索引
SELECT * FROM information_schema.STATISTICS
```

2 控制和管理信息库 - mysql

这个库的名字比较特殊，与 MySQL 同名，想必一定是非常重要的（事实也确实如此）。如果你用过其他数据库的话，可能就不会陌生了，例如：SQL Server 中有 master 库；Oracle 中也有 system，它们都与 mysql 的功能类似。那么，mysql 在数据库中起到什么样的作用呢？一起来看看吧。

2.1 认识 mysql

mysql 是 MySQL 服务器的核心数据库，主要负责存储数据库的用户、权限、关键字等等 MySQL（注意，要时刻区分开 mysql 和 MySQL）自己需要使用的控制和管理信息。

相对来说，理解 mysql 库是比较简单的（即使它非常的重要），它也是四个系统库中定义表数量最少的一个。同样，类似于 information_schema，我们也可以按照功能将这些表分成很多类，例如：权限授予系统表、日志系统表、时区系统表等等。

2.2 mysql 的组成

mysql 库中一共定义了31张表（可以自行查询下 information_schema），按照各个表负责的功能划分，一共可以分为8类。接下来，我们就去看一看这些表各自都会有怎样的功能。

权限授予系统表（包含用户账户以及授权信息）

- user：记录用户账户以及全局权限
- db：记录数据库级别的权限
- tables_priv：记录表级别的权限
- columns_priv：记录列级别的权限
- procs_priv：记录存储过程、函数的权限
- proxies_priv：记录 proxy-user 的权限

Object 信息系统表（包含存储过程、用户自定义函数、插件的相关信息）

- event：记录 event 调度程序信息
- func：记录用户自定义函数（也就是常常看到的 UDF）的信息
- plugin：记录插件的信息
- proc：记录存储过程和函数的信息

日志系统表（MySQL 使用这些表记录日志，且它们使用 CSV 存储引擎）

- general_log：记录查询语句
- slow_log：记录慢查询语句

帮助系统表（MySQL 的帮助信息）

- help_category：记录帮助类别的信息
- help_keyword：记录帮助主题相关的关键字
- help_relation：记录关键字和帮助主题之间的映射关系
- help_topic：记录帮助主题（帮助信息）

时区系统表（包含时区相关的信息）

- time_zone：时区 id，以及它们是否使用闰秒
- time_zone_leap_second：时区的闰秒信息
- time_zone_name：时区 id 和名称之间的映射
- time_zone_transition、time_zone_transition_type：时区的描述信息

复制系统表（MySQL 使用这些表来支持复制，与 Binlog 相关）

- gtid_executed：用于记录 GTID
- ndb_binlog_index：NDB Cluster 复制的 Binlog 信息
- slave_master_info、slave_relay_log_info、slave_worker_info：用于在从服务器上存储复制信息

优化程序系统表（提供给 SQL 优化器使用）

- innodb_index_stats、innodb_table_stats：用于 InnoDB 持久优化器统计
- server_cost：包含 MySQL 操作优化程序成本估算
- engine_cost：包含特定存储引擎特定操作的估计值

其他系统表

- servers：由 FEDERATED 存储引擎使用

经过对 information_schema 和 mysql 这两个系统库的介绍，可以知道，它们其实更多的是在为 MySQL 系统服务。接下来，我们要去介绍的两个库（performance_schema、sys）则是为用户而服务的，走着，一起来看看吧。

3 服务器性能指标 - performance_schema

MySQL 5.7 做了相当多的优化，其中就包括对 performance_schema 的改进，包括引入大量新增加的监控项、降低表占用的存储空间和负载、以及通过 sys 库机制显著提升易用性。

3.1 认识 performance_schema

performance_schema 在 MySQL 运行的过程中，收集服务器性能参数，并经过计算，存储为数据库运行的统计信息。但是，这个数据的信息相当复杂，即使你对 MySQL 了解很多、很有使用经验，也很难读懂这些数据。所以，Oracle 公司官方把 performance_schema 库简化为 sys 库（所以，你也知道了，sys 其实是依赖于 performance_schema 的）。

performance_schema 库中的表存储引擎均为 PERFORMANCE_SCHEMA，而用户是不能创建存储引擎为 PERFORMANCE_SCHEMA 的表的。库中一共定义了87张表（由于各个表都很复杂，且常用的不多，这里不再一一讲解），可以分为5类：系统配置表、当前事件表、事件历史记录表、事件统计表、杂项表。

另外，还需要知道，虽然 performance_schema 提供了数据库运行过程中的各种统计信息，但是，它对数据库的性能影响很小。它的设计实现遵守了两个目标：

- 不影响数据库的行为、不影响线程调用的顺序、不影响 SQL 的执行计划
- 没有新增任何关键字或语句，所以，不会影响 SQL 语法分析器

3.2 performance_schema 承载的功能

刚刚已经简单介绍了 performance_schema 中都有哪些类型的表，这里，我们细化来叙述下 performance_schema 表所承载的功能。

- 进度跟踪：可以通过 events_stages_current 表来查看当前事件的进度信息。通常用于跟踪长时间执行的任务（例如：批量插入、删除等等），performance_schema 会自动的提供执行语句的进度信息
- 内存使用情况：performance_schema 分别从账户、线程、用户以及事件的角度统计了内存使用过程，了解这些，有助于我们调整服务器的内存消耗
- 元数据锁：metadata_locks 表记录了元数据锁的相关信息。你可以知道：哪些会话占据元数据锁、哪些会话正在等待元数据锁等等
- 检测器：可以用于存储过程、函数、事件调度器和触发器的检测
- 事务：配置 setup_consumers、setup_instruments 表打开事务监控，就可以通过 events_transactions_current

表查询当前事务的状态

performance_schema 的功能非常强大，这里仅仅是列出了它常用的方面，如果想要知道的更多，可以查看官方文档（但是内容非常多，且不易理解，需要花点功夫）。当然，有兴趣的话，也可以在 MySQL 运行的过程中，看看表数据的变化。

3.3 performance_schema 中的常用表

同样，类似于 information_schema，这里，我也以 SQL（部分语句太长，做了格式化处理）查询的方式讲解库中的常用表及其使用方法。

```
-- 执行最多的10条 SQL 语句
SELECT * FROM performance_schema.events_statements_summary_by_digest ORDER BY COUNT_STAR DESC LIMIT 10;

-- 平均耗时最多的10条语句
SELECT * FROM performance_schema.events_statements_summary_by_digest ORDER BY AVG_TIMER_WAIT DESC LIMIT 10;

-- 哪个索引使用的最多
SELECT
  *
 FROM
  performance_schema.table_io_waits_summary_by_index_usage
 ORDER BY
  SUM_TIMER_WAIT DESC LIMIT 1

-- 哪个索引没有使用过
SELECT
  *
 FROM
  performance_schema.table_io_waits_summary_by_index_usage
 WHERE
  INDEX_NAME IS NOT NULL
  AND COUNT_STAR = 0
  AND OBJECT_SCHEMA <> 'mysql'
 ORDER BY
  OBJECT_SCHEMA,
  OBJECT_NAME

-- 哪个事件消耗的时间最多
SELECT
  *
 FROM
  performance_schema.events_waits_summary_global_by_event_name
 WHERE
  EVENT_NAME != 'idle'
 ORDER BY
  SUM_TIMER_WAIT DESC LIMIT 1
```

4 DBA 的好帮手 - sys

之所以用好帮手这个词来形容 sys，就是因为这个库用于排查问题、优化系统，且简单好用。但是，这不能说明库中定义的表就很少，实际上：非常的多。不过，莫慌，你平时能用到的也不会有很多。

4.1 认识 sys

sys 是 MySQL 5.7.7 中引入的一个系统库，所以，如果你的数据库版本比它低，你需要先做升级。这个库专注于 MySQL 的易用性，包含了一系列视图、函数和存储过程。在 MySQL 中，需要根据表去创建视图，对于 sys 来说，它的视图数据来自于 information_schema 和 performance_schema。那么，为什么需要这样做呢 ？MySQL Server Blog 中对此作了说明：

> For Linux users I like to compare performance_schema to /proc, and SYS to vmstat.

也就是说，虽然 information_schema 和 performance_schema 统计了这些数据，但是并没有将数据好好的组织起来，以至于不太好用。而 sys 则通过视图的方式解决了这个问题。通过 sys，我们可以快速的查询出：哪些语句使用了临时表、哪个线程占用了最多的内存、哪些索引是冗余的等等。这听起来就很有意思，所以，接着往下看吧。

4.2 sys 中的视图、函数和存储过程

首先，你可以使用如下 SQL 语句查看下 sys 库中都定义了什么：

```
-- sys 中定义的所有表
SHOW FULL TABLES FROM sys

-- sys 中定义的函数
SHOW FUNCTION STATUS WHERE DB = 'sys'

-- sys 中定义的存储过程
SHOW PROCEDURE STATUS WHERE DB = 'sys'
```

sys 中定义了很多视图，关于每一个视图，都有两种形式，执行如下 SQL 语句：

```
mysql> SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'sys' AND TABLE_NAME LIKE '%memory%';
+---------------------------------------+
| TABLE_NAME                            |
+---------------------------------------+
| memory_by_host_by_current_bytes       |
| memory_by_thread_by_current_bytes     |
| memory_by_user_by_current_bytes       |
| memory_global_by_current_bytes        |
| memory_global_total                   |
| x$memory_by_host_by_current_bytes     |
| x$memory_by_thread_by_current_bytes   |
| x$memory_by_user_by_current_bytes     |
| x$memory_global_by_current_bytes      |
| x$memory_global_total                 |
+---------------------------------------+
```

以 “x$” 开头的视图，是便于代码做处理的。而不带的，是给我们去阅读的，它们格式化了时间单位和字节单位。我们可以分别去查询下两种不同形式的视图，从数据中你就可以看到它们的区别了。如下所示：

```
-- 格式化了时间单位，易于阅读
mysql> SELECT * FROM host_summary_by_file_io;
+------------+-------+------------+
| host       | ios   | io_latency |
+------------+-------+------------+
| localhost  | 32933 | 1.96 m     |
| background |  6535 | 2.66 s     |
+------------+-------+------------+

-- 未格式化时间单位，易于代码处理
mysql> SELECT * FROM x$host_summary_by_file_io;
+------------+-------+-----------------+
| host       | ios   | io_latency      |
+------------+-------+-----------------+
| localhost  | 32933 | 117876751169471 |
| background |  6538 |  2661720109685 |
+------------+-------+-----------------+
```

有兴趣的话，你可以去看看 host_summary_by_file_io 视图的定义，你会惊讶的发现：这也太复杂了。所以，简单的查询背后，MySQL 实际是做了很多工作的。另外，sys 中还定义了很多函数和存储过程，有兴趣的话，可以去尝试阅读下它们的代码。这里，我给出查询这些函数和存储过程的 SQL 语句。如下所示：

```
SELECT ROUTINE_SCHEMA, ROUTINE_NAME, ROUTINE_TYPE FROM information_schema.ROUTINES WHERE ROUTINE_TYPE = 'FUNCTION';
SELECT ROUTINE_SCHEMA, ROUTINE_NAME, ROUTINE_TYPE FROM information_schema.ROUTINES WHERE ROUTINE_TYPE = 'PROCEDUR
E';
```

4.3 sys 中的常用表

sys 非常的好用，优化的视图将查询过程简单化，好好的利用这些视图一定能够对你的系统做一些优化。同样，我还是以 SQL 查询的方式讲解库中的常用表及其使用方法。

```
-- 查看当前连接情况
SELECT host, current_connections, statements FROM sys.host_summary

-- 基于主机地址查看谁使用了最多的资源
SELECT * FROM sys.host_summary LIMIT 10

-- 基于用户查看谁使用了最多的资源
SELECT * FROM sys.user_summary LIMIT 10

-- 查看当前正在执行的 SQL
SELECT conn_id, user, current_statement, last_statement FROM sys.session

-- 查看执行最多的 SQL 语句（前10条）
SELECT * FROM sys.statement_analysis ORDER BY exec_count DESC LIMIT 10

-- 查看哪个表的 IO 最多（前10条）
SELECT * FROM sys.io_global_by_file_by_bytes LIMIT 10

-- 查看冗余的索引
SELECT * FROM sys.schema_redundant_indexes

-- 查看未使用的索引
SELECT * FROM sys.schema_unused_indexes

-- 查看访问最多的表（前10条）
SELECT * FROM sys.statement_analysis ORDER BY exec_count DESC LIMIT 10

-- 查看延迟最高的 SQL 语句（前10条）
SELECT * FROM sys.statement_analysis ORDER BY avg_latency DESC LIMIT 10

-- 查看每个用户各自占据多少连接
SELECT user, COUNT(*) FROM sys.processlist GROUP BY user

-- 查看扫描了全表的 SQL 语句（前10条）
SELECT * FROM sys.statements_with_full_table_scans LIMIT 10
```

这些语句虽然看起来都很简单，但是功能却很强大，这也得益于视图对表的“封装”。这里，我强烈建议你去执行下这些 SQL 语句，看看打印的结果。并尝试分析当前数据库存在的问题，最后，努力解决问题（这也就是水平在不知不觉中不断提高的过程）。

5 总结

想要完全掌握 MySQL 的四个系统库并非易事，但是，这样做实则也是没有必要的。学习知识，重要的在于理解，而不在于具体实现。思想理解了，用的时候再去查就好了。这一节的内容更像是工具书一样，你不需要想方设法的把它们都记下来，平时多去用一用，理解它们的内涵，在需要对数据库性能做优化的时候，再去对照着查找并运用。

6 问题

执行一遍我给出的各个库的常用 SQL 语句，分析结果输出，你能得到怎样的结论？

查询这些系统库，你能对你的数据库做出哪些优化呢？是怎么做的呢？

7 参考资料

《高性能 MySQL（第三版）》

MySQL Server Blog：Using SYS.SESSION as an alternative to SHOW PROCESSLIST

MySQL 官方文档：INFORMATION_SCHEMA Tables

MySQL 官方文档：MySQL Performance Schema

MySQL 官方文档：MySQL sys Schema

MySQL 官方文档：The mysql System Database

MySQL 官方文档：InnoDB In-Memory Structures

MySQL 官方文档：The CSV Storage Engine

MySQL 官方文档：The MEMORY Storage Engine

}

13 学会对MySQL做基准测试，掌握数据库性能

15 认识日志系统，掌握系统运行过程