

数据库系统表

1.如何利用数据库的功能读写文件，需要满足什么样的条件才可以读写？

首先本次使用的是mysql数据库。

通过load_file("参数是文件名的完整路径")函数将文件内容获取

前提条件：

- 1.当前权限对该文件**可读**
- 2.文件**在**该服务器上
- 3.路径完整
- 4.文件大小小于max_allowed_packet
- 5.当前数据库用户有对文件操作的权限
- 6.secure_file_priv的值不为null，其他的都可以，如果是某个指定路径，那就只能对在该路径下的文件进行读写，如果secure_file_priv的值没有设置，那么可以在任意位置进行读写。

通过SELECT...INTO OUTFILE写文件

前提条件：

和上述条件一样，唯一的区别在于，在服务器上不能有你要写的这个文件，这样会报错。即：

- 1.当前权限对该文件**可写**
- 2.文件**不在**该服务器上（文件名不可以重复）
- 3.路径完整
- 4.文件大小小于max_allowed_packet
- 5.当前数据库用户有对文件操作的权限
- 6.secure_file_priv的值不为null，其他的都可以，如果是某个指定路径，那就只能对在该路径下的文件进行读写，如果secure_file_priv的值没有设置，那么可以在任意位置进行读写。

下面是实例操作以及结果：

```
mysql> insert into Student (id,name,address,email) values (1,"tom","china","tom@.com");
Query OK, 1 row affected (0.00 sec)

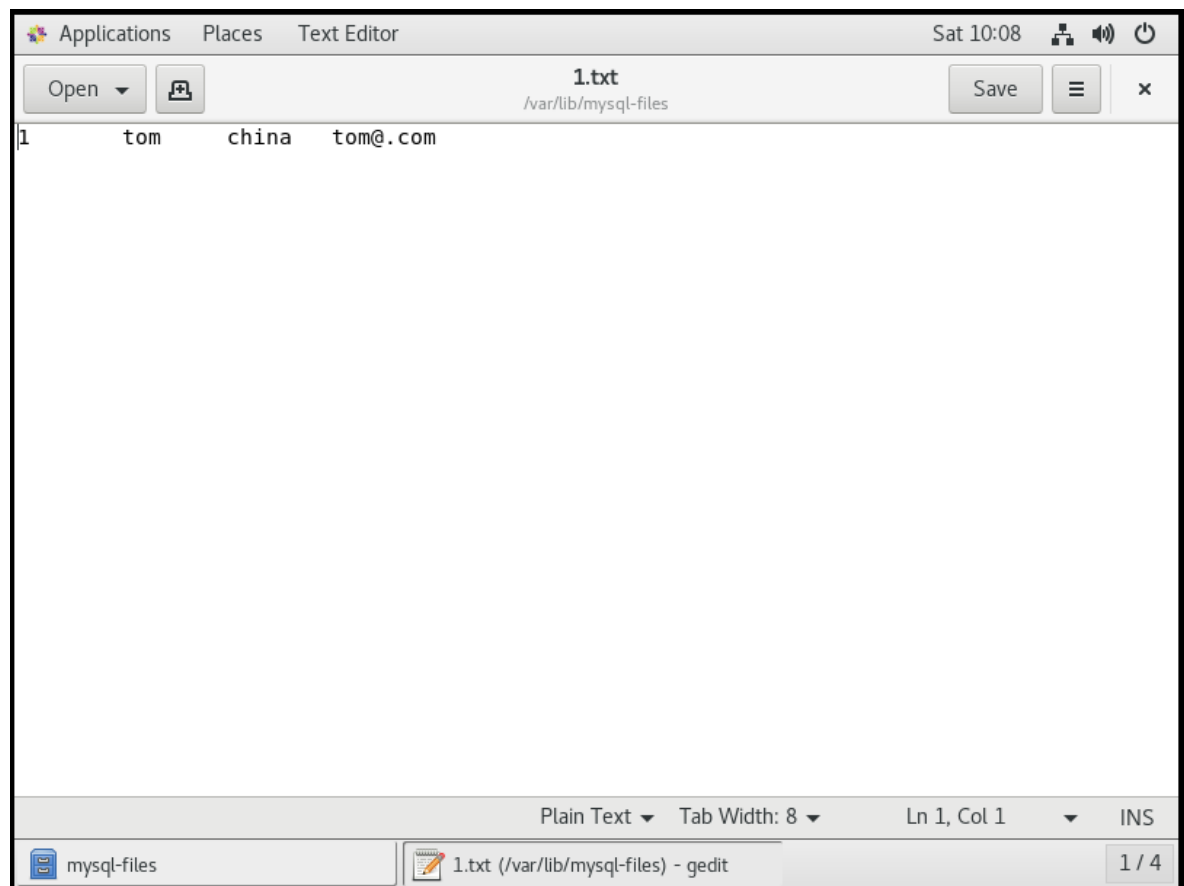
mysql> select * from Student;
+----+-----+-----+-----+
| id | name | address | email |
+----+-----+-----+-----+
| 1  | tom  | china   | tom@.com |
+----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> show variables like "secure_file_priv";
+-----+-----+
| Variable_name | Value |
+-----+-----+
| secure_file_priv | /var/lib/mysql-files/ |
+-----+-----+
1 row in set (0.01 sec)

mysql> select id,name,address,email from Student where id=1 order by id id into outfile "/var/lib/mysql-files/1.txt"
-> ;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'id into outfile "/var/lib/mysql-files/1.txt"' at line 1
mysql> select id,name,address,email from Student where id=1 order by id into outfile "/var/lib/mysql-files/1.txt";
Query OK, 1 row affected (0.00 sec)

mysql> _
```

上图可以看出secure_file_priv的值不为null，而是指定的文件路径，这说明我们只可以在该文件目录下进行文件的读写操作，于是我先创建了一个表，将表中的数据写进了var/lib/musql-files/1.txt文件中，最后执行成功了，下图是写文件的内容。



可以看到在secure_file_priv为null值的时候，文件写操作执行失败。

```
mysql> select * from course into outfile "C:\\User\\1\\Desktop\\1.txt";
ERROR 1290 (HY000): The MySQL server is running with the --secure-file-priv option so it cannot execute this statement
mysql> show global variables like "%secure%";
+-----+-----+
| Variable_name | Value |
+-----+-----+
| require_secure_transport | OFF |
| secure_auth | ON |
| secure_file_priv | NULL |
+-----+-----+
3 rows in set, 1 warning (0.01 sec)

mysql>
```

value为null，表示不可以对其进行任何操作

而另一方面在secure_file_priv值为某路径的语句读写操作都正常。

```
mysql> select load_file('/var/lib/mysql-files/1.txt') as result;
+-----+-----+
| result |
+-----+-----+
| 1      tom      china  tom@.com |
+-----+-----+
1 row in set (0.00 sec)

mysql>
```

未解决的问题：

文件的读写，文件这个词个人理解包括的范围很广，例如图片是文件，视频是文件，pdf也是文件，还有其他除了不是txt的各种文件，这里说的文件读写，是任何文件都可以读写吗？如果可以，怎么做？而且好像txt文件也必须有一定规律才可以读写进数据库，不知自己的理解是否准确，自己比较菜，如果有朋友知道这方面知识，不吝赐教。

2.数据库系统表的功能

首先我linux虚拟机中装的mysql版本号为5.7.27

```
[root@localhost ~]# mysql --version
mysql Ver 14.14 Distrib 5.7.27, for Linux (x86_64) using EditLine wrapper
[root@localhost ~]#
```

MySQL5.7 默认的模式有：

information_schema，具有 61个表；

mysql，具有31个表；

performance_schema，具有87个表；

sys，具有1个表，100个视图。

Information_schema数据库是MySQL自带的，它提供了访问数据库元数据的方式。什么是元数据呢？元数据是关于数据的数据，如数据库名或表名，列的数据类型，或访问权限等。有些时候用于表述该信息的其他术语包括“数据词典”和“系统目录”。在MySQL中，把 information_schema 看作是一个数据库，确切说是信息数据库。其中保存着关于MySQL服务器所维护的所有其他数据库的信息。如数据库名，数据库的表，表栏的数据类型与访问权限等。在INFORMATION_SCHEMA中，有数个只读表。它们实际上是视图，而不是基本表，因此，你将无法看到与之相关的任何文件。

1.使用sql语句查询数据库名。

```
show databases;
```

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| demo |
| mysql |
| performance_schema |
| sys |
+-----+
5 rows in set (0.00 sec)

mysql> █
```

2.使用sql语句查询表名。

```
use mysql;
```

```
show tables;
```

```
mysql> use mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_mysql |
+-----+
| columns_priv |
| db |
| engine_cost |
| event |
| func |
| general_log |
| gtid_executed |
| help_category |
| help_keyword |
| help_relation |
| help_topic |
| innodb_index_stats |
| innodb_table_stats |
| ndb_binlog_index |
| plugin |
| proc |
| procs_priv |
+-----+
```

首先选定你要查找的数据库名称，然后再查看这个数据库中有什么表。

3.使用sql语句查询字段名。

```
show columns from user;
```

这个命令是查询表的字段名

```
mysql> show columns from user;
```

Field	Extra	Type	Null	Key	Default
Host		char(60)	NO	PRI	
User		char(32)	NO	PRI	
Select_priv		enum('N','Y')	NO		N
Insert_priv		enum('N','Y')	NO		N
Update_priv		enum('N','Y')	NO		N
Delete_priv		enum('N','Y')	NO		N
Create_priv		enum('N','Y')	NO		N
Drop_priv		enum('N','Y')	NO		N
Reload_priv		enum('N','Y')	NO		N

4.使用sql语句查询内容。

假设我们要查找user表中的host内容，则可以这么写

```
select host from user;
```

```
mysql> select host from user;
+-----+
| host      |
+-----+
| localhost |
| localhost |
| localhost |
+-----+
3 rows in set (0.02 sec)

mysql>
```

root@localhost:~/data/navicat/navic...

5.使用sql语句查询当前用户的基本信息。

```
select user();
```

使用该命令可以看到当前用户是谁。

```
mysql> select user();
+-----+
| user()      |
+-----+
| root@localhost |
+-----+
1 row in set (0.00 sec)

mysql>
```

那么查看档期啊用户的用户名和密码呢？

用户名密码存储在mysql数据库下的user表内

字段名分别是：User和authentication_string

那么可以这么写sql命令：

```
select User,authentication_string from user;
```


database changed

```
mysql> select User,authentication_string from user;
```

User	authentication_string
root	*6D63738A28F7C1F82626986F1B18D74EC1CD26C9
mysql.session	*THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE
mysql.sys	*THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE

3 rows in set (0.00 sec)

```
mysql>
```



这样就获取到了数据库中的用户名和密码，不过这个密码是加密的密码。

通过网上查找资料可以了解到:MySQL 4.1版本之前是MySQL323加密，MySQL 4.1和之后的版本都是MySQLSHA1加密，其中*不加入实际的密码运算中，因此MySQLSHA1加密的实际位数是40位。

3.数据库系统表各个字段的功能：学习的网址：<https://blog.csdn.net/xlxxcc/article/details/51754524>

1. MYSQL SHOW 命令

```
desc 表名;           // 表信息
show columns from 表名;       // 表字段
describe 表名;         // 表信息
show create table 表名;      // 表创建语句
show create database 数据库名; // 显示数据库 信息
show table status from 数据库名; // 数据库状态
show tables或show tables from database_name; // 显示当前数据库中所有表的名称
show databases;          // 显示mysql中所有数据库的名称
show processlist;        // 显示系统中正在运行的所有进程，也就是当前正在执行的查询。大多数用户可以查看他们自己的进程，但是如果他们拥有process权限，就可以查看所有人的进程，包括密码。
show table status;       // 显示当前使用或者指定的database中的每个表的信息。信息包括表类型和表的最新更新时间
show columns from table_name from database_name; // 显示表中列名称
show columns from database_name.table_name; // 显示表中列名称
show grants for user_name@localhost; // 显示一个用户的权限，显示结果类似于grant 命令
show index from table_name; // 显示表的索引
show status;解释：显示一些系统特定资源的信息，例如，正在运行的线程数量
show variables; // 显示系统变量的名称和值
show privileges;解释：显示服务器所支持的不同权限
show create database database_name ; // 显示create database 语句是否能够创建指定的数据库
show create table table_name; // 显示create database 语句是否能够创建指定的数据库
show engines; // 显示安装以后可用的存储引擎和默认引擎。
show innodb status ; // 显示innodb存储引擎的状态
show logs; // 显示BDB存储引擎的日志
show warnings; // 显示最后一个执行的语句所产生的错误、警告和通知
show errors; // 只显示最后一个执行语句所产生的错误
```

使用如下语句可以查看当前数据库的各IP连接数:

```
select SUBSTRING_INDEX(host,':',1) as ip , count(*) from  
information_schema.processlist group by ip;
```

```
mysql> select SUBSTRING_INDEX(host,':',1) as ip , count(*) from information_sche  
ma.processlist group by ip  
-> ;  
+-----+-----+  
| ip      | count(*) |  
+-----+-----+  
| localhost |          1 |  
+-----+-----+  
1 row in set (0.05 sec)  
  
mysql>
```

2.information_schema数据库部分表说明

TABLES表: 提供了关于数据库中的表的信息(包括视图)。详细表述了某个表属于哪个**schema**, 表类型, 表引擎, 创建时间等信息。是**show tables from schemaname**的结果取之此表。

COLUMNS表: 提供了表中的列信息。详细表述了某张表的所有列以及每个列的信息。是**show columns from schemaname.tablename**的结果取之此表。

STATISTICS表: 提供了关于表索引的信息。是**show index from schemaname.tablename**的结果取之此表。

USER_PRIVILEGES (用户权限) 表: 给出了关于全程权限的信息。该信息源自**mysql.user**授权表。是非标准表。

SCHEMA_PRIVILEGES (方案权限) 表: 给出了关于方案(数据库)权限的信息。该信息来自**mysql.db**授权表。是非标准表。

TABLE_PRIVILEGES (表权限) 表: 给出了关于表权限的信息。该信息源自**mysql.tables_priv**授权表。是非标准表。

COLUMN_PRIVILEGES (列权限) 表: 给出了关于列权限的信息。该信息源自**mysql.columns_priv**授权表。是非标准表。

CHARACTER_SETS (字符集) 表: 提供了**mysql**实例可用字符集的信息。是**SHOW CHARACTER SET**结果集取之此表。

COLLATIONS表: 提供了关于各字符集的对照信息。

COLLATION_CHARACTER_SET_APPLICABILITY表: 指明了可用于校对的字符集。这些列等效于**SHOW COLLATION**的前两个显示字段。

TABLE_CONSTRAINTS表: 描述了存在约束的表。以及表的约束类型。

KEY_COLUMN_USAGE表: 描述了具有约束的键列。

ROUTINES表: 提供了关于存储子程序(存储程序和函数)的信息。此时, **ROUTINES**表不包含自定义函数(UDF)。名为“**mysql.proc name**”的列指明了对应于**INFORMATION_SCHEMA.ROUTINES**表的**mysql.proc**表列。

VIEWS表: 给出了关于数据库中的视图的信息。需要有**show views**权限, 否则无法查看视图信息。

TRIGGERS表: 提供了关于触发程序的信息。必须有**super**权限才能查看该表。

3.performance_schema数据库部分表说明

setup_table : 设置表, 配置监控选项。
current_events_table : 记录当前那些thread 正在发生什么事情。
history_table : 发生的各种事件的历史记录表
summary_table : 对各种事件的统计表
setup_consumers\ Setup_instruments : 描述各种事件, 设置哪些事件能够被收集
setup_instruments : 描述这个数据库下的表名以及是否开启监控。
setup_timers : 描述 监控选项已经采样频率的时间间隔
Threads : 监控服务器所有连接
Performance_timers : 设置一些监控信息, 指定mysql服务可用的监控周期, CYCLE表示按每秒检测2603393034次, 目前 performance-schema 只支持 'wait' 时间的监控, 代码树上 wait/ 下的函数都可以监控到。

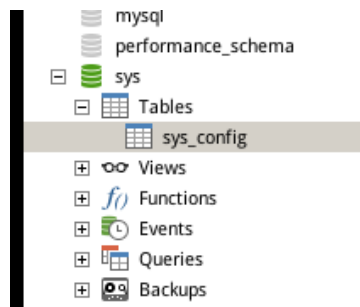
4.mysql数据库部分表说明

在mysql数据库中, 有mysql_install_db脚本初始化权限表, 存储权限的表有:

- 1、user表: 用户列、权限列、安全列、资源控制列
- 2、db表 : 用户列、权限列
- 3、host表
- 4、table_priv表
- 5、columns_priv表
- 6、proc_priv表

5.sys数据库表说明

sys_config : 这是在这个系统库上存在的唯一一个表。



variable	value	set_time	set_by
diagnostics.allow_i_s_tables	OFF	2019-08-03 11:20:35	(Null)
diagnostics.include_raw	OFF	2019-08-03 11:20:35	(Null)
ps_thread_trx_info.max_length	65535	2019-08-03 11:20:35	(Null)
statement_performance_analy	100	2019-08-03 11:20:35	(Null)
statement_performance_analy	(Null)	2019-08-03 11:20:35	(Null)
statement_truncate_len	64	2019-08-03 11:20:35	(Null)

variable : 配置选项名称

value : 配置选项值

set_time: 该行配置修改的时间

set_by : 该行配置信息修改者, 如果从被安装没有修改过, 那么这个数据应该为NULL

以上值的会话变量为@sys.+表中variable字段, 譬如: @sys.statement_truncate_len可以set @sys.statement_truncate_len = 32 临时改变值, 在会话中会一直使用这个值, 如果想要恢复使用表的默认值, 只需要将这个会话值设置为null; set @sys.statement_truncate_len = null;

- diagnostics.allow_i_s_tables : 默认为OFF , 如果开启表示允许diagnostics() 存储过程执行扫描information_schema.tables 表, 如果表很多, 那么可能会很耗性能,
- diagnostics.include_raw : 默认为OFF , 开启将会从metrics 视图输出未加工处理的数据 。
- statement_performance_analyzer.limit : 视图在没有加limit限制时, 返回的最大行数
- statement_performance_analyzer.view
- statement_truncate_len : 通过format_statement()函数返回值的最大长度

这个表非默认选项还有一个@sys.debug参数, 可以手动加入:

```

INSERT INTO sys_config (variable, value) VALUES('debug', 'ON');
UPDATE sys_config SET value = 'OFF' WHERE variable = 'debug';
SET @sys.debug = NULL;
  
```


更多关于mysql用户及权限

参考<https://zhuanlan.zhihu.com/p/55798418>