MySQL 正则表达式 MySQL ALTER命令

MySQL 事务

MySQL 事务主要用于处理操作量大，复杂度高的数据。比如说，在人员管理系统中，你删除一个人员，你即需要删除人员的基本资料，也要删除和该人员相关的信息，如信箱，文章等等，这样，这些数据库操作语句就构成一个事务！

- 在 MySQL 中只有使用了 Innodb 数据库引擎的数据库或表才支持事务。
- 事务处理可以用来维护数据库的完整性，保证成批的 SQL 语句要么全部执行，要么全部不执行。
- 事务用来管理 insert,update,delete 语句

一般来说，事务是必须满足4个条件（ACID）：：原子性（**A**tomicity，或称不可分割性）、一致性（**C**onsistency）、隔离性（**I**solation，又称独立性）、持久性（**D**urability）。

- **原子性：**一个事务（transaction）中的所有操作，要么全部完成，要么全部不完成，不会结束在中间某个环节。事务在执行过程中发生错误，会被回滚（Rollback）到事务开始前的状态，就像这个事务从来没有执行过一样。
- **一致性：**在事务开始之前和事务结束以后，数据库的完整性没有被破坏。这表示写入的资料必须完全符合所有的预设规则，这包含资料的精确度、串联性以及后续数据库可以自发性地完成预定的工作。
- **隔离性：**数据库允许多个并发事务同时对其数据进行读写和修改的能力，隔离性可以防止多个事务并发执行时由于交叉执行而导致数据的不一致。事务隔离分为不同级别，包括读未提交（Read uncommitted）、读提交（read committed）、可重复读（repeatable read）和串行化（Serializable）。
- **持久性：**事务处理结束后，对数据的修改就是永久的，即便系统故障也不会丢失。

> *在 MySQL 命令行的默认设置下，事务都是自动提交的，即执行 SQL 语句后就会马上执行 COMMIT 操作。因此要显式地开启一个事务务须使用命令 BEGIN 或 START TRANSACTION，或者执行命令 SET AUTOCOMMIT=0，用来禁止使用当前会话的自动提交。*

事务控制语句：

- BEGIN 或 START TRANSACTION 显式地开启一个事务；
- COMMIT 也可以使用 COMMIT WORK，不过二者是等价的。COMMIT 会提交事务，并使已对数据库进行的所有修改成为永久性的；
- ROLLBACK 也可以使用 ROLLBACK WORK，不过二者是等价的。回滚会结束用户的事务，并撤销正在进行的所有未提交的修改；
- SAVEPOINT identifier，SAVEPOINT 允许在事务中创建一个保存点，一个事务中可以有多个 SAVEPOINT；
- RELEASE SAVEPOINT identifier 删除一个事务的保存点，当没有指定的保存点时，执行该语句会抛出一个异常；

- ROLLBACK TO identifier 把事务回滚到标记点；
- SET TRANSACTION 用来设置事务的隔离级别。InnoDB 存储引擎提供事务的隔离级别有READ U NCOMMITTED、READ COMMITTED、REPEATABLE READ 和 SERIALIZABLE。

MYSQL 事务处理主要有两种方法：

1、用 BEGIN, ROLLBACK, COMMIT来实现

- **BEGIN** 开始一个事务
- **ROLLBACK** 事务回滚
- **COMMIT** 事务确认

2、直接用 SET 来改变 MySQL 的自动提交模式:

- **SET AUTOCOMMIT=0** 禁止自动提交
- **SET AUTOCOMMIT=1** 开启自动提交

事务测试

```
mysql> use RUNOOB;
Database changed
mysql> CREATE TABLE runoob_transaction_test( id int(5)) engine=innodb; # 创建数据表
Query OK, 0 rows affected (0.04 sec)
mysql> select * from runoob_transaction_test;
Empty set (0.01 sec)
mysql> begin; # 开始事务
Query OK, 0 rows affected (0.00 sec)
mysql> insert into runoob_transaction_test value(5);
Query OK, 1 rows affected (0.01 sec)
mysql> insert into runoob_transaction_test value(6);
Query OK, 1 rows affected (0.00 sec)
mysql> commit; # 提交事务
Query OK, 0 rows affected (0.01 sec)
mysql> select * from runoob_transaction_test;
+------+
| id |
+------+
| 5 |
| 6 |
+------+
2 rows in set (0.01 sec)
mysql> begin; # 开始事务
Query OK, 0 rows affected (0.00 sec)
mysql> insert into runoob_transaction_test values(7);
Query OK, 1 rows affected (0.00 sec)
mysql> rollback; # 回滚
Query OK, 0 rows affected (0.00 sec)
mysql> select * from runoob_transaction_test; # 因为回滚所以数据没有插入
+------+
| id |
```

```
+------+
| 5 |
| 6 |
+------+
2 rows in set (0.01 sec)
mysql>
```

PHP中使用事务实例

MySQL ORDER BY 测试：

```php
<?php
$dbhost = 'localhost:3306'; // mysql服务器主机地址
$dbuser = 'root'; // mysql用户名
$dbpass = '123456'; // mysql用户名密码
$conn = mysqli_connect($dbhost, $dbuser, $dbpass);
if(! $conn )
{
die('连接失败: ' . mysqli_error($conn));
}
// 设置编码，防止中文乱码
mysqli_query($conn, "set names utf8");
mysqli_select_db( $conn, 'RUNOOB' );
mysqli_query($conn, "SET AUTOCOMMIT=0"); // 设置为不自动提交，因为MYSQL默认立即执行
mysqli_begin_transaction($conn); // 开始事务定义
if(!mysqli_query($conn, "insert into runoob_transaction_test (id) values(8)"))
{
mysqli_query($conn, "ROLLBACK"); // 判断当执行失败时回滚
}
if(!mysqli_query($conn, "insert into runoob_transaction_test (id) values(9)"))
{
mysqli_query($conn, "ROLLBACK"); // 判断执行失败时回滚
}
mysqli_commit($conn); //执行事务
mysqli_close($conn);
?>
```

← MySQL 正则表达式 MySQL ALTER命令 →