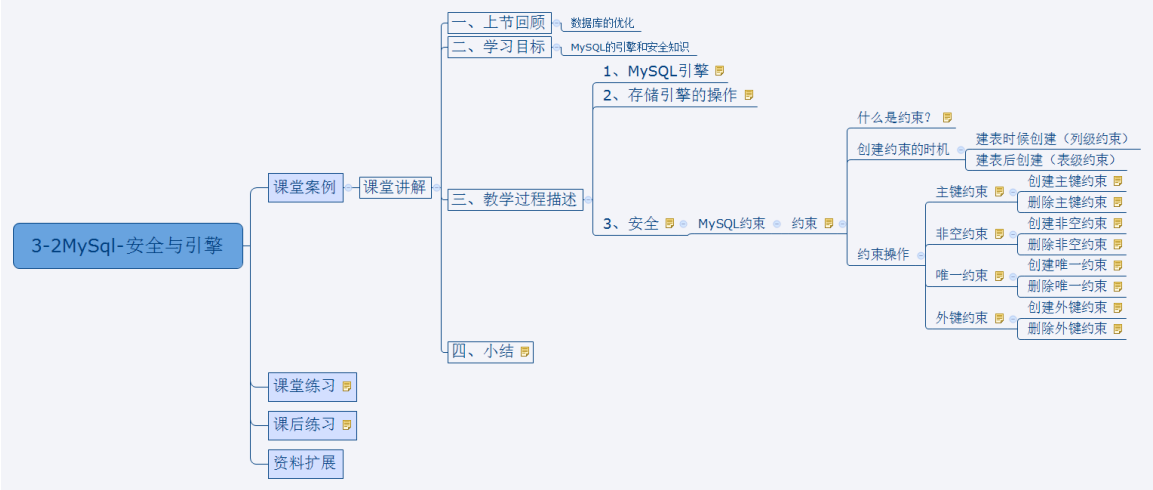
3-2MySql-安全与引擎



1. 课堂案例

课堂讲解

一、上节回顾

数据库的优化

二、学习目标

MySQL的引擎和安全知识

三、教学过程描述

1、MySQL引擎

那么什么是存储引擎呢？

存储引擎说白了就是如何存储数据、如何为存储的数据建立索引和如何更新、查询数据等技术的实现方法。因为在关系数据库中数据的存储是以表的形式存储的，所以存储引擎也可以称为表类型（即存储和操作此表的类型）。

在Oracle和SQLServer等数据库中只有一种存储引擎，所有数据存储管理机制都是一样的。而MySql数据库提供了多种存储引擎。用户可以根据不同的需求为数据表选择不同的存储引擎，用户也可以根据自己的需要编写自己的存储引擎。

1、MyISAM：这种引擎是mysql最早提供的。这种引擎又可以分为静态MyISAM、动态MyISAM和压缩MyISAM三种：

静态MyISAM：如果数据表中的各数据列的长度都是预先固定好的，服务器将自动选择这种表类型。因为数据表中每一条记录所占用的空间都是一样的，所以这种表存取和更新的效率非常高。当数据受损时，恢复工作也比较容易做。

动态MyISAM：如果数据表中出现varchar、xxxtext或xxxBLOB字段时，服务器将自动选择这种表类型。相对于静态MyISAM，这种表存储空间比较小，但由于每条记录的长度不一，所以多次修改数据后，数据表中的数据就可能离散的存储在内存中，进而导致执行效率下降。同时，内存中也可能会出现很多碎片。因此，这种类型的表要经常用optimize table 命令或优化工具来进行碎片整理。

压缩MyISAM：以上说到的两种类型的表都可以用myisamchk工具压缩。这种类型的表进一步减小了占用的存储，但是这种表压缩之后不能再被修改。另外，因为是压缩数据，所以这种表在读取的时候要先时行解压缩。

但是，不管是何种MyISAM表，目前它都不支持事务，行级锁和外键约束的功能。

2、Merge引擎：这种类型是MyISAM类型的一种变种。合并表是将几个相同的MyISAM表合并为一个虚表。常应用于日志和数据仓库。

3、InnoDB：InnoDB表类型可以看作是对MyISAM的进一步更新产品，它提供了事务、行级锁机制和外键约束的功能。

4、memory(heap)：这种类型的数据表只存在于内存中。它使用散列索引，所以数据的存取速度非常快。因为是存在于内存中，所以这种类型常应用于临时表中。

5、archive：这种类型只支持select 和 insert语句，而且不支持索引。常应用于日志记录和聚合分析方面。

2、存储引擎的操作

1、查看数据库可以支持的存储引擎

用show engines; 命令可以显示当前数据库支持的存储引擎情况，如图1所示：

```
mysql> show engines
    -> ;
+--------------------+---------+----------------------------------------------------------------+--------------+------+------------+
| Engine             | Support | Comment                                                        | Transactions | XA   | Savepoints |
+--------------------+---------+----------------------------------------------------------------+--------------+------+------------+
| InnoDB             | YES     | Supports transactions, row-level locking, and foreign keys     | YES          | YES  | YES        |
| MRG_MYISAM         | YES     | Collection of identical MyISAM tables                          | NO           | NO   | NO         |
| MEMORY             | YES     | Hash based, stored in memory, useful for temporary tables      | NO           | NO   | NO         |
| BLACKHOLE          | YES     | /dev/null storage engine (anything you write to it disappears) | NO           | NO   | NO         |
| MyISAM             | DEFAULT | MyISAM storage engine                                          | NO           | NO   | NO         |
| CSV                | YES     | CSV storage engine                                             | NO           | NO   | NO         |
| ARCHIVE            | YES     | Archive storage engine                                         | NO           | NO   | NO         |
| PERFORMANCE_SCHEMA | YES     | Performance Schema                                             | NO           | NO   | NO         |
| FEDERATED          | NO      | Federated MySQL storage engine                                 | NULL         | NULL | NULL       |
+--------------------+---------+----------------------------------------------------------------+--------------+------+------------+
```

由上图可见当前系统的默认数据表类型是MyISAM。当然，我们可以通过修改数据库配置文件中的选项，设定默认表类型。

2、查看表的数据引擎

Show create table tablename;

```
| mvc_news | CREATE TABLE `mvc_news` (
  `id` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `category_id` int(11) DEFAULT NULL COMMENT '分类id',
  `title` varchar(100) NOT NULL COMMENT '新闻标题',
  `excerpt` varchar(255) DEFAULT NULL COMMENT '摘要',
  `image_url` varchar(255) DEFAULT NULL COMMENT '图片地址',
  `content` text COMMENT '新闻内容',
  `time` int(11) DEFAULT NULL COMMENT '新闻添加时间',
  `update_time` int(11) DEFAULT NULL COMMENT '修改时间',
  `type` tinyint(1) DEFAULT '1' COMMENT '1不删除0删除',
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=307 DEFAULT CHARSET=utf8 COMMENT='新闻表' |
```

SHOW TABLE STATUS LIKE 'mvc_news'\G;

```
mysql> SHOW TABLE STATUS LIKE 'mvc_news'\G;
*************************** 1. row ***************************
           Name: mvc_news
         Engine: MyISAM
        Version: 10
     Row_format: Dynamic
           Rows: 303
 Avg_row_length: 622
    Data_length: 188520
Max_data_length: 281474976710655
   Index_length: 6144
      Data_free: 0
 Auto_increment: 307
    Create_time: 2017-06-01 17:07:09
    Update_time: 2017-06-16 20:34:28
     Check_time: NULL
      Collation: utf8_general_ci
       Checksum: NULL
 Create_options:
        Comment: 新闻表
1 row in set (0.00 sec)
```

3、设置或修改表的存储引擎

3.1创建数据库表时设置存储存储引擎的基本语法是:

Create table tableName(

columnName(列名1) type(数据类型) attri(属性设置),

columnName(列名2) type(数据类型) attri(属性设置),

……..) engine = engineName

3.2修改存储引擎，可以用命令Alter table tableName engine =engineName

3、安全

1、避免从互联网访问MySQL数据库，确保特定主机才拥有访问特权

直接通过本地网络之外的计算机改变生产环境中的数据库是异常危险的。

2、定期备份数据库

3、禁用或限制远程访问

4、设置root用户的口令并改变其登录名

5、移除测试(test)数据库

6、禁用LOCAL INFILE

7、移除匿名账户和废弃的账户：使用命令检查select * from mysql.user where user="";

8、降低系统特权

9、降低用户的数据库特权

10、移除和禁用.mysql_history文件

11、安全补丁

12、启用日志

13、改变root目录

14、禁用LOCAL、INFILE命令。LOAD、DATA、LOCAL、INFILE可以从文件系统中读取文件，并显示在屏幕中或保存在数据库中。如果攻击者能够从应用程序找到SQL注入漏洞，这个命令就相当危险了

面的命令可以从MySQL控制台进行操作：SELECT LOAD_FILE("/etc/passwd");

该命令列示了所有的用户。解决此问题的最佳方法是在MySQL配置中禁用它，在CentOS中找到/etc/my.cnf或在Ubuntu中找到/etc/mysql/my.cnf，在[mysqld]部分增加下面一行：set-variable=local-infile=0。搞定。

MySQL约束

约束

约束

约束是一种限制，它通过对表的行或列的数据做出限制，来确保表的数据的完整性、唯一性。

MYSQL中，常用的几种约束：

约束类型：	主键	默认值	唯一	外键	非空
关键字：	PRIMARY KEY	DEFAULT	UNIQUE	FOREIGN KEY	NOT NULL

什么是约束？

1.什么是约束？

1）约束是在表上强制执行的数据校验规则。

2）约束主要用于保证数据库的完整性。

3）当表中数据有相互依赖性时，可以保护相关的数据不被删除。

约束

生活中的约束有哪些

每个人的指纹信息必须唯一

每个人的身份证要求唯一

网上购物需要先登录才能下单

唯一约束

对一张表的某个字段或者某几个字段设置唯一键约束，保证在这个表里对应的数据必须唯一，如:用户ID、手机号、身份证等。

创建约束的时机

建表时候创建（列级约束）

建表后创建（表级约束）

约束操作

主键约束

1.主键从功能上看相当于“非空约束+唯一约束”，用来确定表中的一行数据；

2.一个表中只允许有一个主键；

3.主键字段可以是单字段或者是多字段的组合；

创建主键约束

```
-- 定义列级主键约束
CREATE TABLE STUDENT_TABLE
(
   STUDENT_ID INT PRIMARY KEY AUTO_INCREMENT,
   STUDENT_NAME VARCHAR(255)
);

--定义表级主键约束
```

```
ALTER TABLE STUDENT_TABLE  ADD PRIMARY KEY (STUDENT_ID);


--定义表级主键约束以及自增长
 ALTER TABLE STUDENT_TABLE CHANGE id id INT PRIMARY KEY AUTO_INCREMENT;
```

删除主键约束

```
-- 删除主键约束
ALTER TABLE STUDENT_TABLE  DROP PRIMARY KEY;
```

非空约束

1.非空约束属于列级约束，只能使用列级约束语法定义；

2.确保字段值不允许为空；

注意：

1）所有数据类型的值都可以是NULL。

2）空字符串不等于NULL。

3）0也不等于NULL。

创建非空约束

```
-- 定义列级非空约束
CREATE TABLE STUDENT_TABLE
(
   STUDENT_ID INT PRIMARY KEY AUTO_INCREMENT,
   STUDENT_NAME VARCHAR(255) NOT NULL
);


--定义表级非空约束
ALTER TABLE STUDENT_TABLE
MODIFY STUDENT_NAME VARCHAR(255) NOT NULL;
```

删除非空约束

```
-- 删除非空约束
ALTER TABLE STUDENT_TABLE
MODIFY STUDENT_NAME VARCHAR(255) NULL;
```

唯一约束

1.唯一性约束条件保证字段或者字段组合不会出现重复值；

2.可以对多列建立唯一约束。

创建唯一约束

唯一约束是一种特殊的索引

唯一约束可以是一个或者多个字段

唯一约束可以在创建表的时候建好，也可以后面再补上

主键也是一种唯一约束

创建唯一约束

```
-- 建立列级唯一约束
CREATE TABLE STUDENT_TABLE
(
   STUDENT_ID INT PRIMARY KEY AUTO_INCREMENT,
   STUDENT_NAME VARCHAR(255) UNIQUE
);
```

```
-- 建立表级唯一约束
ALTER TABLE STUDENT_TABLE
ADD UNIQUE KEY STUDENT_NAME_UNIQUE(STUDENT_NAME);
```

STUDENT_NAME_UNIQUE是一个自定义的唯一约束名称，方便管理这个约束。

```
-- 对多列添加唯一索引
ALTER TABLE STUDENT_TABLE
ADD UNIQUE(FIRST_NAME, LAST_NAME);
```

删除唯一约束

```
-- 删除唯一约束
ALTER TABLE STUDENT_TABLE
DROP INDEX STUDENT_NAME_UNIQUE;
```

外键约束

1.外键是构建于一个表的两个字段或者两个表的两个字段之间的关系；

2.外键确保了相关的两个字段的两个关系：

1）子(从)表外键列的值必须在主表参照列值的范围内，或者为空（也可以加非空约束，强制不允许为空）；

2）当主表的记录被子表参照时，主表记录不允许被删除；

3.外键参照的只能是主表主键或者唯一键，保证子表记录可以准确定位到被参照的记录；

添加外键约束的语法：

```
CREATE TABLE t_dept(
   dept_id INT PRIMARY KEY,
   NAME VARCHAR(18),
   description VARCHAR(255)
   );

CREATE TABLE t_employee(
   employee_id INT PRIMARY KEY,
   NAME VARCHAR(18),
   gender VARCHAR(10),
   dept_id INT REFERENCES t_dept(dept_id),
   address VARCHAR(255)
);
```

或者：

```
CREATE TABLE t_employee(
   employee_id INT PRIMARY KEY,
   NAME VARCHAR(18),
   gender VARCHAR(10),
   dept_id INT,
   address VARCHAR(255),
   CONSTRAINT FOREIGN KEY t_employee (dept_id) REFERENCES t_dept(dept_id)
);
```

创建外键约束

```
CREATE TABLE SCHOOL_TABLE
(
   SCHOOL_ID INT PRIMARY KEY AUTO_INCREMENT,
   SCHOOL_NAME VARCHAR(255)
);


-- 定义列级外键约束
CREATE TABLE STUDENT_TABLE
(
   STUDENT_ID INT PRIMARY KEY AUTO_INCREMENT,
   STUDENT_NAME VARCHAR(255),
   SCHOOL_ID INT,
   FOREIGN KEY (SCHOOL_ID) REFERENCES SCHOOL_TABLE(SCHOOL_ID) ON DELETE CASCADE
);


-- 定义表级外键约束
ALTER TABLE STUDENT_TABLE
ADD CONSTRAINT FK_STUDENT_SCHOOL_ID FOREIGN KEY(SCHOOL_ID) REFERENCES SCHOOL_TABLE(SCHOOL_ID);
```

注意:

ON DELETE CASCADE:当主表记录被删除,从表中的对应记录也会被删除。

ON UPDATE CANSCADE:当主表记录被更新,从表中的对应记录也会被更新。

ON DELETE SET NULL:当主表记录被删除,从表中的对应记录的外键列会被设置为NULL。

ON UPDATE SET NULL:当主表记录被更新,从表中的对应记录也会被会被设置为NULL。

删除外键约束

```
-- 删除外键
ALTER TABLE STUDENT_TABLE
DROP FOREIGN KEY FK_STUDENT_SCHOOL_ID;
```

四、小结

1、MyISAM与InnoDB的区别。MyISAM类型不支持事务处理等高级处理,而InnoDB类型支持。MyISAM类型的表强调的是性能,其执行数度比InnoDB类型更快,但是不提供事务支持,而InnoDB提供事务支持已经外部键等高级数据库功能。

2、mysql的安全注意

2. 课堂练习

1、区分各种引擎的作用

2、如何注意安全问题

3. 课后练习

1、检查以前的项目数据库是什么 引擎

2、有什么安全问题

4. 资料扩展