

第 3 章

◀ 数据库的基本操作 ▶



学习目标 | Objective

MySQL 安装好以后，首先需要创建数据库，这是使用 MySQL 各种功能的前提。本章将详细介绍数据的基本操作，主要内容包括：创建数据库、删除数据库、不同类型的数据存储引擎和存储引擎的选择。



内容导航 | Navigation

- 掌握如何创建数据库
- 熟悉数据库的删除操作
- 了解数据存储引擎的简介
- 熟悉常见的存储引擎工作原理
- 熟悉如何选择符合需求的存储引擎
- 掌握综合案例中数据库的创建和删除方法

3.1 创建数据库

MySQL 安装完成之后，将会在其 data 目录下自动创建几个必需的数据库，可以使用 SHOW DATABASES; 语句来查看当前所有存在的数据库，输入语句如下。

```
mysql> SHOW DATABASES;
```

```
+-----+
| Database |
+-----+
| information_schema |
| mysql          |
| performance_schema |
| sakila         |
| test           |
| world          |
```

```
+-----+
6 rows in set (0.04 sec)
```

可以看到, 数据库列表中包含了 6 个数据库, MySQL 是必需的, 它描述用户访问权限, 用户经常利用 test 数据库做测试的工作, 其他数据库将在后面的章节中介绍。

创建数据库是在系统磁盘上划分一块区域用于数据的存储和管理, 如果管理员在设置权限的时候为用户创建了数据库, 则可以直接使用, 否则, 需要自己创建数据库。MySQL 中创建数据库的基本 SQL 语法格式为:

```
CREATE DATABASE database_name;
```

“database_name”为要创建的数据库的名称, 该名称不能与已经存在的数据库重名。

【例 3.1】创建测试数据库 test_db, 输入语句如下。

```
CREATE DATABASE test_db;
```

数据库创建好之后, 可以使用 SHOW CREATE DATABASE 声明查看数据库的定义。

【例 3.2】查看创建好的数据库 test_db 的定义, 输入语句如下。

```
mysql> SHOW CREATE DATABASE test_db\G
*** 1. row ***
      Database: test_db
Create Database: CREATE DATABASE `test_db` /*!40100 DEFAULT CHARACTER SET utf8 */
1 row in set (0.00 sec)
```

可以看到, 如果数据库创建成功, 将显示数据库的创建信息。

再次使用 SHOW DATABASES; 语句来查看当前所有存在的数据库, 输入语句如下。

```
mysql> SHOW databases;
+-----+
| Database          |
+-----+
| information_schema |
| mysql              |
| performance_schema |
| sakila             |
| test               |
| test_db            |
| world              |
+-----+
7 rows in set (0.05 sec)
```

可以看到, 数据库列表中包含了刚刚创建的数据库 test_db 和其他已经存在的数据库的名称。

3.2 删除数据库

删除数据库是将已经存在的数据库从磁盘空间上清除，清除之后，数据库中的所有数据也将一同被删除。删除数据库语句和创建数据库的命令相似，MySQL 中删除数据库的基本语法格式为：

```
DROP DATABASE database_name;
```

“database_name”为要删除的数据库的名称，如果指定的数据库不存在，则删除出错。

【例 3.3】删除测试数据库 test_db，输入语句如下。

```
DROP DATABASE test_db;
```

语句执行完毕之后，数据库 test_db 将被删除，再次使用 SHOW CREATE DATABASE 声明查看数据库的定义，结果如下。

```
mysql> SHOW CREATE DATABASE test_db\G
ERROR 1049 (42000): Unknown database 'test_db'
ERROR:
No query specified
```

执行结果给出一条错误信息：“ERROR 1049 <42000>: Unknown database 'test_db'”，即数据库 test_db 已不存在，删除成功。

提示

使用 DROP DATABASE 命令时要非常谨慎，在执行该命令时，MySQL 不会给出任何提醒确认信息，DROP DATABASE 声明删除数据库后，数据库中存储的所有数据表和数据也将一同被删除，而且不能恢复。

3.3 数据库存储引擎

数据库存储引擎是数据库底层软件组件，数据库管理系统（DBMS）使用数据引擎进行创建、查询、更新和删除数据操作。不同的存储引擎提供不同的存储机制、索引技巧、锁定水平等功能，使用不同的存储引擎，还可以获得特定的功能。现在许多不同的数据库管理系统都支持多种不同的数据引擎。MySQL 的核心就是存储引擎。

3.3.1 MySQL 存储引擎简介

MySQL 提供了多个不同的存储引擎，包括处理事务安全表的引擎和处理非事务安全表的引擎。在 MySQL 中，不需要在整个服务器中使用同一种存储引擎，针对具体的要求，可以对

每一个表使用不同的存储引擎。MySQL 5.7 支持的存储引擎有: InnoDB, MyISAM, Memory, Merge, Archive, Federated, CSV, BLACKHOLE 等。可以使用 SHOW ENGINES 语句查看系统所支持的引擎类型, 结果如下。

```
mysql> SHOW ENGINES \G
*** 1. row ***
      Engine: FEDERATED
      Support: NO
      Comment: Federated MySQL storage engine
Transactions: NULL
      XA: NULL
Savepoints: NULL
*** 2. row ***
      Engine: MRG_MYISAM
      Support: YES
      Comment: Collection of identical MyISAM tables
Transactions: NO
      XA: NO
Savepoints: NO
*** 3. row ***
      Engine: MyISAM
      Support: YES
      Comment: MyISAM storage engine
Transactions: NO
      XA: NO
Savepoints: NO
*** 4. row ***
      Engine: BLACKHOLE
      Support: YES
      Comment: /dev/null storage engine (anything you write to it disappears)
Transactions: NO
      XA: NO
Savepoints: NO
*** 5. row ***
      Engine: CSV
      Support: YES
      Comment: CSV storage engine
Transactions: NO
      XA: NO
Savepoints: NO
*** 6. row ***
      Engine: MEMORY
      Support: YES
      Comment: Hash based, stored in memory, useful for temporary tables
Transactions: NO
```

```

XA: NO
Savepoints: NO
*** 7. row ***
Engine: ARCHIVE
Support: YES
Comment: Archive storage engine
Transactions: NO
XA: NO
Savepoints: NO
*** 8. row ***
Engine: InnoDB
Support: DEFAULT
Comment: Supports transactions, row-level locking, and foreign keys
Transactions: YES
XA: YES
Savepoints: YES
*** 9. row ***
Engine: PERFORMANCE_SCHEMA
Support: YES
Comment: Performance Schema
Transactions: NO
XA: NO
Savepoints: NO
9 rows in set (0.00 sec)

```

Support 列的值表示某种引擎是否能使用：YES 表示可以使用，NO 表示不能使用，DEFAULT 表示该引擎为当前默认存储引擎。

3.3.2 InnoDB 存储引擎

InnoDB 是事务型数据库的首选引擎，支持事务安全表（ACID），支持行锁定和外键。MySQL 5.5.5 之后，InnoDB 作为默认存储引擎，InnoDB 主要特性有：

(1) InnoDB 给 MySQL 提供了具有提交、回滚和崩溃恢复能力的事务安全（ACID 兼容）存储引擎。InnoDB 锁定在行级并且也在 SELECT 语句中提供一个类似 Oracle 的非锁定读。这些功能增加了多用户部署和性能。在 SQL 查询中，可以自由地将 InnoDB 类型的表与其他 MySQL 的表的类型混合起来，甚至在同一个查询中也可以混合。

(2) InnoDB 是为处理巨大数据量的最大性能设计。它的 CPU 效率可能是任何其他基于磁盘的关系数据库引擎所不能匹敌的。

(3) InnoDB 存储引擎完全与 MySQL 服务器整合，InnoDB 存储引擎为在主内存中缓存数据和索引而维持它自己的缓冲池。InnoDB 将它的表和索引存在一个逻辑表空间中，表空间

可以包含数个文件（或原始磁盘分区）。这与 MyISAM 表不同，比如在 MyISAM 表中每个表被存在分离的文件中。InnoDB 表可以是任何尺寸，即使在文件尺寸被限制为 2GB 的操作系统上。

(4) InnoDB 支持外键完整性约束 (FOREIGN KEY)。

存储表中的数据时，每张表的存储都按主键顺序存放，如果没有显示在表定义时指定主键，InnoDB 会为每一行生成一个 6B 的 ROWID，并以此作为主键。

(5) InnoDB 被用在众多需要高性能的大型数据库站点上。

InnoDB 不创建目录，使用 InnoDB 时，MySQL 将在 MySQL 数据目录下创建一个名为 ibdata1 的 10MB 大小的自动扩展数据文件，以及两个名为 ib_logfile0 和 ib_logfile1 的 5MB 大小的日志文件。

3.3.3 MyISAM 存储引擎

MyISAM 基于 ISAM 的存储引擎，并对其进行扩展。它是在 Web、数据存储和其他应用环境下最常使用的存储引擎之一。MyISAM 拥有较高的插入、查询速度，但不支持事务。在 MySQL 5.5.5 之前的版本中，MyISAM 是默认存储引擎。MyISAM 主要特性有：

(1) 大文件（达 63 位文件长度）在支持大文件的文件系统和操作系统上被支持。

(2) 当把删除、更新及插入操作混合使用的时候，动态尺寸的行产生更少碎片。这要通过合并相邻被删除的块，以及若下一个块被删除，就扩展到下一块来自动完成。

(3) 每个 MyISAM 表最大索引数是 64，这可以通过重新编译来改变。每个索引最大的列数是 16 个。

(4) 最大的键长度是 1000B，这也可以通过编译来改变。对于键长度超过 250B 的情况，一个超过 1024B 的键将被用上。

(5) BLOB 和 TEXT 列可以被索引。

(6) NULL 值被允许在索引的列中。这个值占每个键的 0~1 个字节。

(7) 所有数字键值以高字节优先被存储以允许一个更高的索引压缩。

(8) 每表一个 AUTO_INCREMENT 列的内部处理。MyISAM 为 INSERT 和 UPDATE 操作自动更新这一列。这使得 AUTO_INCREMENT 列更快（至少 10%）。在序列顶的值被删除之后就不能再利用。

(9) 可以把数据文件和索引文件放在不同目录。

(10) 每个字符列可以有不同的字符集。

(11) 有 VARCHAR 的表可以固定或动态记录长度。

(12) VARCHAR 和 CHAR 列可以多达 64KB。

使用 MyISAM 引擎创建数据库，将生产 3 个文件。文件的名称以表的名称开始，扩展名指出文件类型：frm 文件存储表定义，数据文件的扩展名为 .MYD (MYData)，索引文件的扩

展名是.MYI (MYIndex)。

3.3.4 MEMORY 存储引擎

MEMORY 存储引擎将表中的数据存储在内存中, 为查询和引用其他表数据提供快速访问。MEMORY 主要特性有:

- (1) MEMORY 表的每个表可以有多达 32 个索引, 每个索引 16 列, 以及 500B 的最大键长度。
- (2) MEMORY 存储引擎执行 HASH 和 BTREE 索引。
- (3) 可以在一个 MEMORY 表中有非唯一键。
- (4) MEMORY 表使用一个固定的记录长度格式。
- (5) MEMORY 不支持 BLOB 或 TEXT 列。
- (6) MEMORY 支持 AUTO_INCREMENT 列和对可包含 NULL 值的列的索引。
- (7) MEMORY 表在所有客户端之间共享 (就像其他任何非 TEMPORARY 表)。
- (8) MEMORY 表内容被存在内存中, 内存是 MEMORY 表和服务器在查询处理时的空闲中创建的内部表共享。
- (9) 当不再需要 MEMORY 表的内容时, 要释放被 MEMORY 表使用的内存, 应该执行 DELETE FROM 或 TRUNCATE TABLE, 或者删除整个表 (使用 DROP TABLE)。

3.3.5 存储引擎的选择

不同存储引擎都有各自的特点, 以适应不同的需求, 如表 3.1 所示。为了做出选择, 首先需要考虑每一个存储引擎提供了哪些不同的功能。

表 3.1 存储引擎比较

功能	MyISAM	Memory	InnoDB	Archive
存储限制	256TB	RAM	64TB	None
支持事务	No	No	Yes	No
支持全文索引	Yes	No	No	No
支持数索引	Yes	Yes	Yes	No
支持哈希索引	No	Yes	No	No
支持数据缓存	No	N/A	Yes	No
支持外键	No	No	Yes	No

如果要提供提交、回滚和崩溃恢复能力的事务安全 (ACID 兼容) 能力, 并要求实现并发控制, InnoDB 是个很好的选择。如果数据表主要用来插入和查询记录, 则 MyISAM 引擎能提供较高的处理效率; 如果只是临时存放数据, 数据量不大, 并且不需要较高的数据安全性, 可以选择将数据保存在内存中的 Memory 引擎, MySQL 中使用该引擎作为临时表, 存放查询的中间结果。如果只有 INSERT 和 SELECT 操作, 可以选择 Archive 引擎, Archive 存储引擎支

持高并发的插入操作，但是本身并不是事务安全的。Archive 存储引擎非常适合存储归档数据，如记录日志信息可以使用 Archive 引擎。

使用哪一种引擎要根据需要灵活选择，一个数据库中多个表可以使用不同引擎以满足各种性能和实际需求。使用合适的存储引擎，将会提高整个数据库的性能。

3.4 综合案例——数据库的创建和删除

本章分别介绍了数据库的基本操作，包括数据库的创建、查看当前数据库和删除数据库。最后介绍了 MySQL 中各种存储引擎。在这里，通过一个案例，让读者全面回顾数据库的基本操作。

1. 案例目的

登录 MySQL，使用数据库操作语句创建、查看和删除数据库，步骤如下：

- 步骤 01 登录数据库。
- 步骤 02 创建数据库 zoo。
- 步骤 03 选择当前数据库为 zoo，并查看 zoo 数据库的信息。
- 步骤 04 删除数据库 zoo。

2. 案例操作过程

- 步骤 01 登录数据库。

打开 Windows 命令行，输入登录用户名和密码。

```
C:\>mysql -h localhost -u root -p
Enter password: **
```

或者打开 MySQL 5.7 Command Line Client，只输入用户密码也可以登录。登录成功后显示如下信息：

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.10 MySQL Community Server (GPL)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
mysql>
```

出现 MySQL 命令输入提示符时表示登录成功，可以输入 SQL 语句进行操作。

步骤 02 创建数据库 zoo，执行过程如下。

```
mysql> CREATE DATABASE zoo;
Query OK, 1 row affected (0.01 sec)
```

提示信息表明语句成功执行。

查看当前系统中所有的数据库，执行过程如下。

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| test |
| sakila |
| test |
| world |
| zoo |
+-----+
```

可以看到，数据库列表中已经有了名称为 zoo 的数据库，数据库创建成功。

步骤 03 选择当前数据库为 zoo，查看数据库 zoo 的信息，执行过程如下。

```
mysql> USE zoo;
Database changed
```

提示信息 Database changed 表明选择成功。

查看数据库信息，如下所示：

```
mysql> SHOW CREATE DATABASE zoo \G
*** 1. row ***
      Database: zoo
Create Database: CREATE DATABASE `zoo` /*!40100 DEFAULT CHARACTER SET utf8 */
```

Database 值表明当前数据库名称；Create Database 值表示创建数据库 zoo 的语句，后面为注释信息。

步骤 04 删除数据库 zoo，执行过程如下。

```
mysql> DROP DATABASE zoo;
```

Query OK, 0 rows affected (0.00 sec)

语句执行完毕，将数据库 zoo 从系统中删除。

```
mysql> SHOW DATABASES;
+-----+
| Database          |
+-----+
| information_schema |
| mysql              |
| performance_schema |
| test               |
| sakila              |
| test               |
| world              |
+-----+
```

可以看到，数据库列表中已经没有名称为 zoo 的数据库了。

3.5 专家解惑

疑问：如何查看默认存储引擎？

在前面介绍了使用 SHOW ENGINES 语句查看系统中所有的存储引擎，其中包括默认的存储引擎，还可以使用一种直接的方法查看默认存储引擎，输入语句如下：

```
mysql> SHOW VARIABLES LIKE 'storage_engine';
+-----+
| Variable_name | Value |
+-----+
| storage_engine | InnoDB |
+-----+
```

执行结果直接显示了当前默认的存储引擎为 InnoDB。

3.6 经典习题

- (1) 查看当前系统中的数据库。
- (2) 创建数据库 Book，使用 SHOW CREATE DATABASE 语句查看数据库定义信息。
- (3) 删除数据库 Book。