



下载APP



21 | 数据备份：异常情况下，如何确保数据安全？

2021-04-27 朱晓峰

MySQL 必知必会

[进入课程 >](#)**讲述：朱晓峰**

时长 09:36 大小 8.80M



你好，我是朱晓峰。今天，我来和你聊一聊数据备份。

数据备份，对咱们技术人员来说十分重要。当成千上万的用户，每天使用我们开发的应用做着他们的日常工作的时候，数据的安全性就不光是你一个人的事了。要是有一天，突然发生了某种意想不到的情况，导致数据库服务器上的数据全部丢失，所有使用这个应用的人都会受到严重影响。

所以，我们必须“未雨绸缪”，及时把数据备份到安全的地方。这样，当突发的异常来临时，我们就能把数据及时恢复回来，就不会造成太大损失。



MySQL 的数据备份有 2 种，一种是物理备份，通过把数据文件复制出来，达到备份的目的；另外一种逻辑备份，通过把描述数据库结构和内容的信息保存起来，达到备份的目的。

的。逻辑备份这种方式是免费的，广泛得到使用；而物理备份的方式需要收费，用得比较少。所以，这节课我重点和你聊聊逻辑备份。

我还会给你介绍一下 MySQL 中的数据备份工具 `mysqldump`、数据恢复的命令行客户端工具 `mysql`，以及数据表中数据导出到文件和从文件导入的 SQL 语句，帮助你提高你所开发的应用中的数据安全性。

如何进行数据备份？

首先，我们来学习下用于数据备份的工具 `mysqldump`。它总共有三种模式：

1. 备份数据库中的表；
2. 备份整个数据库；
3. 备份整个数据库服务器。

接下来，我就来介绍下这 3 种备份的具体方法。

如何备份数据库中的表？

`mysqldump` 备份数据库中的表的语法结构是：

```
1 mysqldump -h 服务器 -u 用户 -p 密码 数据库名称 [表名称 ... ] > 备份文件名称
```

[复制代码](#)

我简单解释一下这里的核心内容。

“-h” 后面跟的服务器名称，如果省略，默认是本机 “localhost”。

“-u” 后面跟的是用户名。

“-p” 后面跟的是密码，如果省略，执行的时候系统会提示录入密码。

我举个小例子，带你看看怎么使用这个工具。

假设数据库 demo 中有 2 个表，分别是商品信息表 (demo.goodsmaster) 和会员表 (demo.membermaster) 。


商品信息表：

itemnumber (商品编号)	barcode (条码)	goodsname (商品名称)	specification (规格)	unit (单位)	salesprice (售价)
1	0001	书	16开	本	89
2	0002	笔	10只装	包	5
3	0003	橡皮	NULL	个	3

会员表：

id (编号)	cardno (会员卡号)	membername (名称)	memberphone (电话)	memberpid (身份证号)	memberaddress (地址)	sex (性别)	birthday (生日)
1	10000001	张三	13812345678	110123200001017890	北京	男	2000-01-01
2	10000002	李四	13512345678	123123199001012356	上海	女	1990-01-01


现在，我需要把数据库 demo 备份到文件中，就可以用下面的代码实现：

 复制代码

```
1 H:\>mysqldump -u root -p demo goodsmaster membermaster > test.sql
2 Enter password: *****
```

这个指令的意思，就是备份本机数据库服务器上 demo 数据库中的商品信息表和会员信息表的所有信息。

备份文件是以文本格式保存的，我们可以用记事本打开，看一下备份的内容：

 复制代码

```
1 -- MySQL dump 10.13 Distrib 8.0.23, for Win64 (x86_64)
2 --
3 -- Host: localhost Database: demo -- 表示从本地进行备份，数据库是demo
4 -- -----
5 -- Server version 8.0.23
6
7 /*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
8 /*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
9 /*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
10 /*!50503 SET NAMES utf8mb4 */;
11 /*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
12 /*!40103 SET TIME_ZONE='+00:00' */;
13 /*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
14 /*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=
15 /*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
16 /*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
17
18 --
19 -- Table structure for table `goodsmaster` -- 商品信息表的结构
20 --
21
22 DROP TABLE IF EXISTS `goodsmaster`;
23 /*!40101 SET @saved_cs_client = @@character_set_client */;
24 /*!50503 SET character_set_client = utf8mb4 */;
25 CREATE TABLE `goodsmaster` (
26   `itemnumber` int NOT NULL,
27   `barcode` text,
28   `goodsname` text,
29   `specification` text,
30   `unit` text,
31   `salesprice` decimal(10,2) DEFAULT NULL,
32   PRIMARY KEY (`itemnumber`)
33 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
34 /*!40101 SET character_set_client = @saved_cs_client */;
35
36 --
37 -- Dumping data for table `goodsmaster` -- 商品信息表的内容
38 --
39
40 LOCK TABLES `goodsmaster` WRITE;
41 /*!40000 ALTER TABLE `goodsmaster` DISABLE KEYS */;
42 INSERT INTO `goodsmaster` VALUES (1, '0001', '书', '16开', '本', 89.00), (2, '0002', '笔
43 /*!40000 ALTER TABLE `goodsmaster` ENABLE KEYS */;
```

```
44 UNLOCK TABLES;
45
46 --
47 -- Table structure for table `membermaster` -- 会员表的结构
48 --
49
50 DROP TABLE IF EXISTS `membermaster`;
51 /*!40101 SET @saved_cs_client = @@character_set_client */;
52 /*!50503 SET character_set_client = utf8mb4 */;
53 CREATE TABLE `membermaster` (
54   `id` int NOT NULL AUTO_INCREMENT,
55   `cardno` char(8) NOT NULL,
56   `membername` text,
57   `memberphone` text,
58   `memberpid` text,
59   `memberaddress` text,
60   `sex` text,
61   `birthday` datetime DEFAULT NULL,
62   PRIMARY KEY (`id`)
63 ) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_
64 /*!40101 SET character_set_client = @saved_cs_client */;
65
66 --
67 -- Dumping data for table `membermaster` -- 会员表的内容
68 --
69
70 LOCK TABLES `membermaster` WRITE;
71 /*!40000 ALTER TABLE `membermaster` DISABLE KEYS */;
72 INSERT INTO `membermaster` VALUES ('10000001','张三','13812345678','11012320000
73 /*!40000 ALTER TABLE `membermaster` ENABLE KEYS */;
74 UNLOCK TABLES;
75 /*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;
76
77 /*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
78 /*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
79 /*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
80 /*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
81 /*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
82 /*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
83 /*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;
84
85 -- Dump completed on 2021-04-11 10:43:04
```

从这个文件中，我们可以看到，它相当于一个 SQL 执行脚本，里面包括了创建商品信息表和会员表的 SQL 语句，以及把表里的数据插入到这两个表中的 SQL 语句。这样一来，商品信息表和会员信息表的结构信息和全部数据信息就都备份出来了。

下面我来介绍一下备份整个数据库的方法。

如何备份数据库？

mysqldump 备份数据库的语法结构是：

```
1 mysqldump -h 服务器 -u 用户 -p 密码 --databases 数据库名称 ... > 备份文件名
```

[复制代码](#)

举个小例子，假设我现在需要对本机的数据库服务器中的 2 个数据库 demo 和 demo1 进行备份，就可以用下面的指令：

```
1 H:\>mysqldump -u root -p --databases demo demo1 > test1.sql
2 Enter password: *****
```

[复制代码](#)

现在，我们来查看一下备份文件的内容：

```
1 -- MySQL dump 10.13 Distrib 8.0.23, for Win64 (x86_64)
2 --
3 -- Host: localhost Database: demo
4 -- -----
5 -- Server version 8.0.23
6
7 /*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
8 /*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
9 /*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
10 /*!50503 SET NAMES utf8mb4 */;
11 /*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
12 /*!40103 SET TIME_ZONE='+00:00' */;
13 /*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
14 /*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=
15 /*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
16 /*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
17
18 --
19 -- Current Database: `demo` -- 备份数据库demo
20 --
21
22 CREATE DATABASE /*!32312 IF NOT EXISTS*/ `demo` /*!40100 DEFAULT CHARACTER SET
23
24 USE `demo`; -- 备份数据库中的表
25
26 --
```

[复制代码](#)


```

27 -- Table structure for table `dailystatistics`
28 --
29
30 DROP TABLE IF EXISTS `dailystatistics`;
31 /*!40101 SET @saved_cs_client = @@character_set_client */;
32 /*!50503 SET character_set_client = utf8mb4 */;
33 CREATE TABLE `dailystatistics` (
34   `id` int NOT NULL AUTO_INCREMENT,
35   `itemnumber` int DEFAULT NULL,
36   `quantity` decimal(10,3) DEFAULT NULL,
37   `actualvalue` decimal(10,2) DEFAULT NULL,
38   `cost` decimal(10,2) DEFAULT NULL,
39   `profit` decimal(10,2) DEFAULT NULL,
40   `profitratio` decimal(10,4) DEFAULT NULL,
41   `salesdate` datetime DEFAULT NULL,
42   PRIMARY KEY (`id`),
43   KEY `index_dailystatistic_salesdate` (`salesdate`),
44   KEY `index_dailystatistic_itemnumber` (`itemnumber`)
45 ) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900
46 /*!40101 SET character_set_client = @saved_cs_client */;
47
48 --
49 -- Dumping data for table `dailystatistics`
50 --
51
52 LOCK TABLES `dailystatistics` WRITE;
53 /*!40000 ALTER TABLE `dailystatistics` DISABLE KEYS */;
54 INSERT INTO `dailystatistics` VALUES (15,1,3.000,267.00,100.50,166.50,0.6236,'
55 /*!40000 ALTER TABLE `dailystatistics` ENABLE KEYS */;
56 UNLOCK TABLES;
57
58 -- 这里省略了其他表的备份语句
59 --
60 -- Current Database: `demo1` -- 备份数据库demo1
61 --
62
63 CREATE DATABASE /*!32312 IF NOT EXISTS*/ `demo1` /*!40100 DEFAULT CHARACTER SE
64
65 USE `demo1`;
66 /*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;
67
68 /*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
69 /*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
70 /*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
71 /*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
72 /*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
73 /*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
74 /*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;
75
76 -- Dump completed on 2021-04-11 11:02:09

```

可以看到，这个文件里面包含了创建数据库 demo 和 demo1 的 SQL 语句，以及创建数据库中所有表、插入所有表中原有数据的 SQL 语句。

如何备份整个数据库服务器？

mysqldump 备份整个数据库服务器的语法结构是：

```
1 mysqldump -h 服务器 -u 用户 -p 密码 --all-databases > 备份文件名
```

[复制代码](#)

举个小例子，假设我要把本机上整个 MySQL 服务器备份下来，可以用下面的代码：

```
1 H:\>mysqldump -u root -p --all-databases > test2.sql
2 Enter password: *****
```

[复制代码](#)

这个指令表示，备份本机上运行的 MySQL 数据库服务器的全部内容，包含系统数据库和用户创建的数据库中的库结构信息、表结构信息和表里的数据。这种备份方式会把系统数据库也全部备份出来，而且消耗的资源也比较多，一般来说没有必要，我就不展开细说了。

备份文件有了，如何用它进行数据恢复呢？下面我就来给你介绍下具体的方法。

如何进行数据恢复？


mysqldump 的备份文件包含了创建数据库、数据表，以及插入数据表里原有数据的 SQL 语句，我们可以直接运行这些 SQL 语句，来进行数据恢复。

数据恢复的方法主要有 2 种：

使用 “mysql” 命令行客户端工具进行数据恢复；

使用 “SOURCE” 语句进行数据恢复。

使用 “mysql” 命令行客户端工具，进行数据恢复的命令如下：

 复制代码

```
1 H:\>mysql -u root -p demo < test.sql
2 Enter password: *****
```

我来简单介绍下这个数据恢复命令。

mysql 是一个命令行客户端工具，可以与 MySQL 服务器之间进行连接，执行 SQL 语句。

“-u” 后面跟的是用户。

“-p” 后面跟的是密码。


在这个命令里面，我指定了数据库，因为备份文件 test.sql 里面只有数据表的备份信息，需要指定恢复到哪个数据库中。如果使用的备份文件备份的是数据库的信息（比如 test1.sql），或者是整个 MySQL 数据库服务器的信息（比如 test2.sql），则不需要指定数据库。

第二种数据恢复的方法是，使用 “SOURCE” 语句恢复数据，语法结构如下：

 复制代码

```
1 SOURCE 备份文件名
```

举个小例子，刚才我们对商品信息表和会员信息表进行了备份，现在想用备份的文件进行恢复，就可以用下面的语句：

 复制代码

```
1 mysql> USE demo;
2 Database changed
3 mysql> SOURCE H:\\test.sql
4 Query OK, 0 rows affected (0.00 sec)
```

注意，这里需要先用 “USE” 语句把当前的数据库变更为 demo，这样商品信息表和会员表才能恢复到正确的数据库里面。否则，可能会恢复错误。

除此之外，你还可以通过这种方式，用整个数据库的备份文件把数据库恢复回来，甚至是用整个数据库服务器的备份文件，恢复整个 MySQL 服务器。

到这里，我们就掌握了备份和恢复整个数据库服务器、数据库和数据库中的表的方法。不过，有的时候，我们只关心表里的数据本身，希望能够把表里的数据，按照一定的格式保存下来。这个时候，mysqldump 就不够用了。所以，接下来我再给你介绍下 MySQL 数据导出和导入的方法。

如何导出和导入表里的数据？

先来学习下怎么把一个表的数据按照一定的格式，导出成一个文件。

SELECT 语句导出数据

使用 “SELECT ... INTO OUTFILE” 语句导出数据表的语法结构是：

 复制代码

```
1 SELECT 字段列表 INTO OUTFILE 文件名称
2 FIELDS TERMINATED BY 字符
3 LINES TERMINATED BY 字符
4 FROM 表名;
```

我来解释下这段代码。

INTO OUTFILE 文件名称，表示查询的结果保存到文件名称指定的文件中；

FIELDS TERMINATED BY 字符，表示列之间的分隔符是 “字符” ；

LINES TERMINATED BY 字符，表示行之间的分隔符是 “字符” 。

举个小例子，假设我们要把商品信息表导出到文件 H:\goodsmaster.txt 中，该如何实现呢？按照我刚刚介绍的语法结构来尝试一下：

 复制代码

```
1 mysql> SELECT * INTO OUTFILE 'H:\goodsmaster.txt'
2 -> FIELDS TERMINATED BY ','
3 -> LINES TERMINATED BY '\n'
4 -> FROM demo.goodsmaster;
5 ERROR 1290 (HY000): The MySQL server is running with the --secure-file-priv op
```

结果，系统提示错误。其实，这是因为**服务器的“secure-file-priv”参数选项，不允许把文件写入到 H:\goodsmaster.txt 中**。那怎么解决这个问题呢？

这个时候，我们可以通过 MySQL 的配置文件 my.ini，来查看一下“secure-file-priv”参数的设定，并且按照这个参数设定的要求准备导入文件。

打开 C:\ProgramData\MySQL\MySQL Server 8.0\my.ini，找到“secure-file-priv”参数设定，如下所示：

[复制代码](#)

```
1 # Secure File Priv.
2 secure-file-priv="C:/ProgramData/MySQL/MySQL Server 8.0/Uploads"
```

这个意思是说，只能把数据导出到“C:/ProgramData/MySQL/MySQL Server 8.0/Uploads”这个文件夹中，所以，如果我们把数据导出到 H:\goodsmaster.txt 中，就违反了系统参数的设定，导致发生错误。

现在，我们来修改一下数据导出的 SQL 语句，把导出文件的路径改到系统要求的文件目录，看看结果如何：

[复制代码](#)

```
1 mysql> SELECT * INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/go
2 -> FIELDS TERMINATED BY ','
3 -> LINES TERMINATED BY '\n'
4 -> FROM demo.goodsmaster;
5 Query OK, 3 rows affected (0.00 sec)
```

结果显示，执行成功了。下面我们来看一下结果文件的内容：

[复制代码](#)

```
1 1,0001,书,16开,本,89.00
2 2,0002,笔,10支装,包,5.00
3 3,0003,橡皮,\N,个,3.00
```

很显然，这很符合我们希望的导出格式：行与行之间用回车 “\n” 分隔，列与列之间用逗号 “,” 分隔。

到这里，我们就知道怎么把数据表中的数据按照一定的格式导出到文件了。那在实际工作中，我们还经常需要把一定格式的数据从文件中导入到数据表中。

“LOAD DATA” 语句，就是 MySQL 提供的一种快速数据读入的方法，在实际工作中常用于大量数据的导入，效率极高。

使用“LOAD DATA”语句导入数据

“LOAD DATA” 是与 “SELECT ... INTO OUTFILE” 相对应的数据导入语句。语句结构是：

```
1 LOAD DATA INFILE 文件名
2 INTO TABLE 表名
3 FIELDS TERMINATED BY 字符
4 LINES TERMINATED BY 字符;
```

[复制代码](#)

我举个小例子来演示一下 “LOAD DATA” 语句是如何工作的。

还是以我们刚才导出的那个文件 goodsmaster.txt 为例，现在我们把这个文件内的数据导入到商品信息表（demo.goodsmaster）中去。

为了演示方便，我会先把 demo.goodsmaster 中的数据先删除，然后使用 “LOAD DATA” 语句，把刚才的导出文件 goodsmaster.txt 的内容导入进来，再与删除之前的数据进行对比，来验证 “LOAD DATA” 语句的执行效果。

首先，我们把商品信息表中的数据删除：

```
1 mysql> DELETE FROM demo.goodsmaster
2 -> WHERE itemnumber>0;
3 Query OK, 3 rows affected (0.03 sec)
```

[复制代码](#)

然后，我们尝试把文件 `goodsmaster.txt` 中的数据导入进来：

[复制代码](#)

```
1 mysql> LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/goodsmaster
2 -> INTO TABLE demo.goodsmaster
3 -> FIELDS TERMINATED BY ','
4 -> LINES TERMINATED BY '\n';
5 Query OK, 3 rows affected (0.02 sec)
6 Records: 3 Deleted: 0 Skipped: 0 Warnings: 0
```

结果显示，导入成功了。我们再查看一下数据表中的内容：

[复制代码](#)

```
1 mysql> SELECT * FROM demo.goodsmaster;
2 +-----+-----+-----+-----+-----+-----+
3 | itemnumber | barcode | goodsname | specification | unit | salesprice |
4 +-----+-----+-----+-----+-----+-----+
5 | 1 | 0001 | 书 | 16开 | 本 | 89.00 |
6 | 2 | 0002 | 笔 | 10支装 | 包 | 5.00 |
7 | 3 | 0003 | 橡皮 | NULL | 个 | 3.00 |
8 +-----+-----+-----+-----+-----+-----+
9 3 rows in set (0.00 sec)
```

结果显示，与我们删除之前的数据完全一致。这说明，“LOAD DATA”语句成功导入了数据文件 `goodsmaster.txt` 中的数据。

总结

今天，我们重点学习了数据备份，包括数据备份的工具 `mysqldump`，以及用命令行客户端工具“mysql”和 SQL 语句“SOURCE”进行数据恢复的方法。同时，我还给你介绍了用于导出数据表中数据的语句“SELECT ... INTO OUTFILE”和导入的语句“LOAD DATA”。这些都是你在备份数据时必不可少的，对确保数据的安全性至关重要。

最后提醒你一点，“LOAD DATA”是很好用的工具，因为它的导入速度是非常惊人的。一个 400 万条数据的文件，用“LOAD DATA”语句，只需要几分钟就可以完成，而其他的方法，比如使用 Workbench 来导入数据，就需要花费好几个小时。

思考题

这节课，我介绍了数据导出语句“SELECT ... INTO OUTFILE”，并且在第一次数据导出时遇到了一个系统参数“secure-file-priv”设定的目录与导出文件目录不一致，从而导致导出失败的问题。最后，我通过修改导出文件的文件夹，解决了这个问题。

我想请你思考一下，如果还是想把导出文件保存到 H:\ 目录下，有没有办法实现呢？

欢迎在留言区写下你的思考和答案，我们一起交流讨论。如果你觉得今天的内容对你有所帮助，也欢迎你分享给你的朋友或同事，我们下节课见。

提建议

更多课程推荐

深入浅出计算机组成原理

带你掌握计算机体系全貌

徐文浩

bothub 创始人



涨价倒计时 🕒

今日订阅 **¥89**，5月12日涨价至 **¥199**

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 20 | 日志（下）：系统故障，如何恢复数据？

精选留言 (3)

[写留言](#)**lesserror**

2021-04-27

敲黑板了，几百万数据使用 LOAD DATA 命令回复只需几分钟，可以说是属于黑科技了，长见识了。

课程中讲到的命令，公司运维经常使用。今天跟着敲了一遍，加深了印象。

...

[展开](#)**危大爷危**

2021-04-27

可以修改my.ini 文件的secure-file-priv参数,指定为H盘,之后重启数据库

[展开](#)**Harry**

2021-04-27

本节需要掌握两点：

1. 备份和恢复整个数据库服务器、数据库和数据库中的表的方法。
2. MySQL 数据导出和导入的方法。...

[展开](#)