大数据科学丛书

Spark SQL

大数据实例开发教程

深度剖析Spark SQL内核架构

案例全面覆盖Spark SQL**核心应用**

Spark SQL源码与性能调优**全解密**

王家林 段智华 编著

机械工业出版社
CHINA MACHINE PRESS

大数据科学丛书

Spark SQL 大数据实例开发教程

王家林　段智华　编著

机 械 工 业 出 版 社

Spark SQL 是 Spark 生态环境中最核心和最基础的组件，是掌握 Spark 的关键所在。本书完全从企业级开发的角度出发，结合多个企业级应用案例，深入剖析 Spark SQL。全书共分为 8 章，包括：认识 Spark SQL、DataFrame 原理与常用操作、Spark SQL 操作多种数据源、Parquet 列式存储、Spark SQL 内置函数与窗口函数、Spark SQL UDF 与 UDAF、Thrift Server、Spark SQL 综合应用案例。

本书可以使读者对 Spark SQL 有深入彻底的理解，本书适合于 Spark 学习爱好者，是学习 Spark SQL 的入门和提高教材，也是 Spark 开发工程师开发过程中查阅 Spark SQL 的案头手册。

图书在版编目（CIP）数据

Spark SQL 大数据实例开发教程/王家林等编著．—北京：机械工业出版社，2017.9
（大数据科学丛书）
ISBN 978-7-111-59197-9

Ⅰ.①S… Ⅱ.①王… Ⅲ.①数据处理软件－教材 Ⅳ.①TP274

中国版本图书馆 CIP 数据核字（2018）第 033108 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
策划编辑：王 斌 责任编辑：王 斌
责任校对：张艳霞 责任印制：孙 炜
北京中兴印刷有限公司印刷

2018 年 3 月第 1 版·第 1 次印刷
184mm×260mm·16.5 印张·398 千字
0001－3000 册
标准书号：ISBN 978-7-111-59197-9
定价：59.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务
服务咨询热线：010-88361066
读者购书热线：010-68326294
010-88379203

网络服务
机 工 官 网：www.cmpbook.com
机 工 官 博：weibo.com/cmp1952
金 书 网：www.golden-book.com
教育服务网：www.cmpedu.com

前　言

"Use of MapReduce engine for Big Data projects will decline, replaced by Apache Spark."
MapReduce 计算模型的使用会越来越少，最终将被 Apache Spark 所取代。

——Hadoop 之父 Doug Cutting

写作背景

Spark 是一个快速大规模数据处理的通用引擎。它给 Java、Scala、Python 和 R 等语言提供了高级 API，并基于统一抽象的 RDD（弹性分布式数据集），逐渐形成了一套自己的生态系统。这个生态系统主要包括负责 SQL 和结构化数据处理的 Spark SQL、负责实时流处理的 Spark Streaming、负责图计算的 Spark GraphX 以及机器学习子框架 Mlib。Spark 在处理各种场景时，提供给用户统一的编程体验，可极大地提高编程效率。

Hive 是运行在 Hadoop 上的 SQL on Hadoop 工具，它的推出是为了给熟悉 RDBMS 但又不理解 MapReduce 的技术人员提供快速上手的工具，但是 MapReduce 在计算过程中消耗大量 I/O 资源，降低了运行效率。为了提高 SQL on Hadoop 的效率，Shark 出现了，它使得 SQL on Hadoop 的性能比 Hive 有了 10 ~ 100 倍的提高。但 Shark 对于 Hive 的过度依赖（如采用 Hive 的语法解析器、查询优化器等），制约了 Spark 的发展，所以提出了 Spark SQL 项目，Spark SQL 抛弃 Shark 原有的弊端，又汲取了 Shark 的一些优点，如内存列存储（In - Memory Columnar Storage）、Hive 的兼容性等，由于摆脱了对 Hive 的依赖性，Spark SQL 在数据兼容、性能优化、组件扩展等方面的性能都得到了极大的提升。

Spark SQL 是 Spark 生态环境中最核心和最基础的组件，是掌握 Spark 的关键所在。由于目前市场上介绍 Spark 技术的书籍比较少，尤其是单独讲解 Spark SQL 的书更是凤毛麟角，我们特意编写了这本理论和实战相结合的 Spark SQL 书籍，在介绍 Spark SQL 核心技术的同时又配备了丰富的示例，同时还穿插了源代码的分析，使读者能从更深层次来把握 Spark SQL 的核心技术。

内容速览

本书完全从企业级开发的角度出发，结合多个企业级应用案例，深入剖析 Spark SQL。

全书一共分为 8 章，主要内容概括如下：

第 1 章认识 Spark SQL，引领读者了解 Spark SQL 的基础知识，接下来的第 2 章至第 7 章，结合实战案例，引导读者掌握 Spark SQL 的核心知识，这 6 章内容分别为：DataFrame 原理与常用操作、Spark SQL 操作多种数据源、Parquet 列式存储、Spark SQL 内置函数与窗口函数、Spark SQL UDF 与 UDAF、Thrift Server；本书的最后部分，第 8 章 Spark SQL 综合应用案例归纳并综合运用了全部 Spark SQL 知识点，是深入理解 Spark SQL 的经典案例。

本书可以使读者对 Spark SQL 有深入的理解，是 Spark 爱好者用来学习 Spark SQL 的理想教程，也是 Spark 开发工程师在开发过程中可随时查阅的案头手册。

本书作者

本书由王家林和段智华编写。

预备知识

在学习本书之前读者需要熟悉基本的 Linux 命令及 Java、Scala 语言，掌握基本的 Spark 知识架构，能够搭建 Spark 集群环境。

致谢

在本书编写的过程中，作者参考了很多网络上的书籍和博客，在此谢谢各位作者，正是你们的无私奉献，才推动了 Spark 技术的快速发展。

特别感谢“小小”同学为本书的编写提供的各种协调和热心帮助。

由于笔者能力有限，书中难免存在错误或表达不准确的内容，恳请大家批评指正，希望大家一起努力使 Spark 技术在大数据计算领域里推广开来。

作者

2017.9

目　录

第1章 认识Spark SQL

Section 1.1

Spark SQL概述

Spark SQL是Spark的计算模块之一，它和Spark的基础模块RDD不一样，是专门用于处理结构化数据的。Spark SQL为Spark提供了更加便利的处理和计算结构化数据的能力。具备以下知识基础，可以让我们更好地了解和使用Spark SQL，包括：SQL、Spark SQL DataFrame、Spark SQL Dataset（SparkSQL Dataset在Spark 2.0之后才被正式推出，在Spark 1.6.3版本的生产环境中还没有大规模应用）。

Spark SQL既可以使用标准的SQL语法，也可以使用HiveQL来执行SQL的查询和读写，Spark SQL还可以从已经存在的Hive数据仓库中读取数据（关于这方面的配置，我们将在1.3节中介绍）。当我们在Spark程序中使用Spark SQL时，其结果将返回DataFrame。当然，我们还可以通过交互式命令行或者JDBC/ODBC来调用Spark SQL的API。

1.1.1 Spark SQL与DataFrame

Spark SQL是在Spark生态系统中除了Spark Core之外最大且最受关注组件，如图1-1所示。

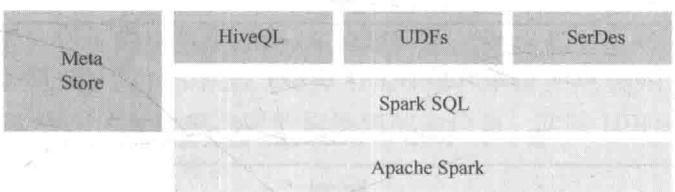

图1-1 Spark组件

Spark SQL具备以下特征：

1）处理一切存储介质和各种格式的数据，同时可以方便地扩展Spark SQL的功能来支持更多类型的数据，例如Kudu。

2）Spark SQL把数据仓库的计算能力推向了新的高度，不仅包括高效的计算速度（Spark SQL比Shark快了至少一个数量级，而Shark比Hive快了至少一个数量级，尤其是在Tungsten Project成熟以后Spark SQL会更加无可匹敌），更为重要的是Spark SQL大大提升了数据仓库的计算复杂度（Spark SQL推出的DataFrame可以让数据仓库直接使用机器学习、

图计算等复杂的算法库来对数据仓库进行复杂深度数据价值的挖掘)。

3) Spark SQL (DataFrame、DataSet) 不仅是数据仓库的引擎，而且也是数据挖掘的引擎，更为重要的是 Spark SQL 是数据科学计算和分析引擎。

4) Hive + Spark SQL + DataFrame 组成了目前在国内的大数据主流技术组合：

- Hive：负责低成本的数据仓库存储。
- Spark SQL：负责高速的计算。
- DataFrame：负责复杂的数据挖掘。

1.1.2 DataFrame 与 RDD 的差异

DataFrame 是一个分布式的面向列组成的数据集，在概念上类似于一张关系型数据库中的表，但是在 Spark 计算引擎下，有了更高效的优化。DataFrame 的组成来源很广泛，例如：结构化文件、Hive 中的 Table、外部数据库，或者由 RDD 转换而来。DataFrame 的 API 可以在 Scala、Java、Python、R 中使用 (Spark 支持以上 4 种语言的开发)。

在 R 和 Python 中都有 DataFrame，但 Spark SQL 中的 DataFrame 从形式上看最大的不同点在于，其天生是分布式的。从学习的角度讲，我们可以简单地把 Spark SQL 中的 DataFrame 抽象成是一个分布式的表，其形式如表 1-1 所示。

表 1-1 DataFrame 被抽象成分布式的表

姓名	年龄	电话
String	Int	Long
String	Int	Long
String	Int	Long
…	…	…
String	Int	Long
String	Int	Long

而在 RDD 中，DataFrame 的形式如表 1-2 所示。

表 1-2 RDD 中 Dataframe 的表现形式

Record
Record
Record
Record
Record
Record
Record

数据的表现在 RDD 和 DataFrame 之间有两点根本的差异：

1) RDD 是以 Record 为单位的，SparkSQL 在优化的时候无法了解 Record 内部的细节，

所以也就无法进行更深度的优化，这极大地限制了 Spark SQL 性能的提升。

2）DataFrame 是以列为单位的，包含每个 Record 的元数据信息，也就是说 DataFrame 在优化时基于列内部的优化，而不是像 RDD 一样，只能够基于行来进行优化。

1.1.3 Spark SQL 的发展历程

SparkSQL 发展历程图如图 1-2 所示。

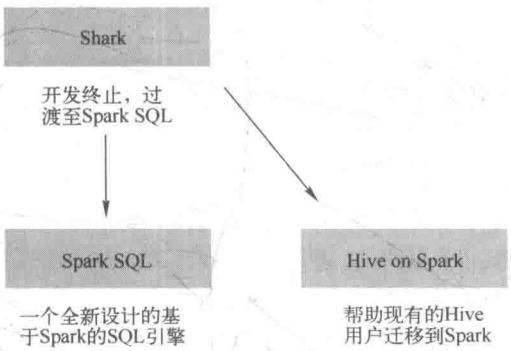

图 1-2 Spark SQL 发展历程图

SparkSQL 是从 Shark 发展而来的，2014 年 7 月 1 日的 Spark Summit 上，Databricks 宣布终止对 Shark 的开发，转向到 Spark SQL 上。Databricks 表示，Spark SQL 将涵盖 Shark 的所有特性，用户可以从 Shark 0.9 进行无缝的升级。同时 Databricks 推出 Spark SQL 和 Hiveon Spark。其中 Spark SQL 是为 Spark 设计的一代新的 SQL 引擎，作为 Spark 生态中的一员继续发展，而不再受限于 Hive，只是兼容 Hive；而 Hive on Spark 是将 Spark 作为一个替代执行引擎提供给 Hive 的，从而为已经存在的 Hive 用户提供了一个迁往 Spark 的途径。

下面简单地介绍一下 Shark 及 Shark 项目终止的原因。

Shark 发布时，Hive 可以说是 SQL on Hadoop 的唯一选择，负责将 SQL 编译成可扩展的 MapReduce 作业。鉴于 Hive 的性能及与 Spark 的兼容性，Shark 项目由此而生。

Shark 通过 Hive 的 HQL（Hibernate Query Language）解析，把 HQL 翻译成 Spark 上的 RDD 操作，然后通过 Hive 的元数据获取数据库里的表信息，实际 HDFS 上的数据和文件，会由 Shark 获取并放到 Spark 上运算。

Shark 的最大特性就是速度快以及与 Hive 的完全兼容，且可以在 Shell 模式下使用 API，把 HQL 得到的结果集继续在 Scala 环境下运算，支持自己编写简单的机器学习或简单分析处理函数，对 HQL 结果进一步分析计算。

但是 Shark 更多的是对 Hive 的改造，替换了 Hive 的物理执行引擎，因此会有一个很快的速度。然而，不容忽视的是，Shark 继承了大量的 Hive 代码，因此给进一步对其优化和维护带来了大量的麻烦。随着性能优化和先进分析整合的进一步加深，基于 MapReduce 设计的部分无疑成为整个项目的瓶颈。因此，为了更好地发展，给用户提供一个更好的体验，Databricks 宣布终止 Shark 项目，从而将更多的精力放到 Spark SQL 上。

Section 1.2

从零起步掌握 Hive

Hive 是基于 Hadoop 构建的一套数据仓库分析系统，提供了丰富的 SQL 查询方式来分析存储在 Hadoop 分布式文件系统中的数据，可以将结构化的数据文件映射为一张数据库表，并提供完整的 SQL 查询功能，可以将 SQL 语句转换为 MapReduce 任务进行运行，通过自己的 SQL 去查询分析需要的内容，这套 SQL 简称 HiveQL（Hive SQL）。

1.2.1 Hive 的本质是什么

Hive 是分布式数据仓库，同时又是查询引擎，所以 Spark SQL 取代的只是 Hive 查询引擎，在企业实际生产环境下，Hive + Spark SQL 是目前最为经典的数据分析组合，Hive 本身就是一个简单单机版本的软件，主要负责：

1）把 HQL 翻译成 Mapper - Reducer - Mapper 的代码，并且可能产生很多 MapReduce 的 Job。

2）把生产的 MapReduce 代码及资源打成 JAR 包，并自动发布到 Hadoop 集群中运行。

Hive 本身的架构如图 1-3 所示。

Hive 架构主要有下面 4 个部分组成：

1）驱动程序（Driver）：负责编译、优化、执行。

Hive 的入口是 Driver，执行的 SQL 语句首先提交到 Driver，然后调用编译器（Compiler）解释驱动，最终解释成 MapReduce 任务执行，最后将结果返回。Driver 调用编译器处理 HiveQL（Hive SQL），可能是一条 DDL、DML 或查询语句。编译器将字符串转化为策略（Plan）。策略仅由元数据操作和 HDFS 操作组成，元数据操作只包含 DDL 语句，HDFS 操作只包含 LOAD 语句。具体流程为：解析→语义分析→逻辑策略生成→优化→执行。

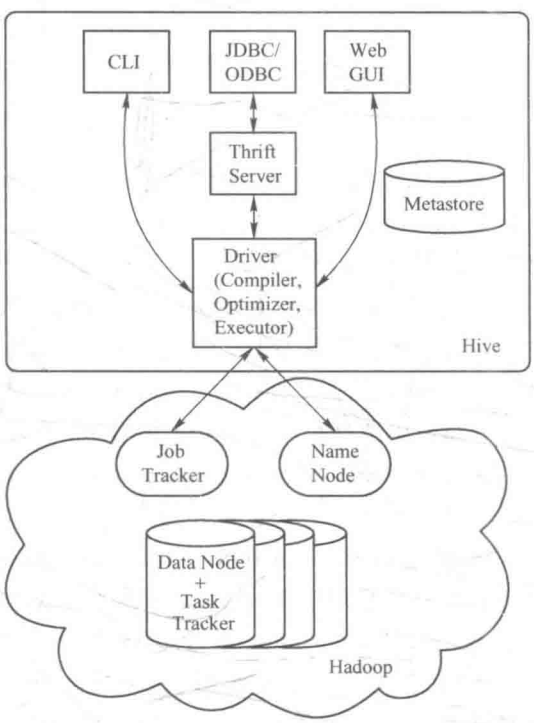

图 1-3 Hive 架构

2）3 种服务模式：Hive 驱动对外提供了 3 种服务模式，分别是：

- CLI：即 Hive 命令行模式，通过命令行终端来直接操作 Hive。
- Web GUI：即 Hive 的 Web 模式，通过浏览器来访问 Hive。
- Hive 的远程服务模式：通过 Thrift Server 的支持，远程 Client 程序可以通过 JDBC/ODBC 方式来访问连接 Hive，这也是程序员最需要的方式。

3）Metastore：元数据存储。

Hive 将元数据存储在 RDBMS 中，一般常用 MySQL 和 Derby。默认情况下，Hive 元数据保存在内嵌的 Derby 数据库中，只能允许一个会话连接，只适合简单的测试。实际生产环境中不适用，为了支持多用户会话，则需要一个独立的元数据库，使用 MySQL 作为元数据库，Hive 内部对 MySQL 提供了很好的支持。

4）Hadoop：Hadoop 是 Hive 的运行基石。

Hive 安装依赖 Hadoop 的集群，Hive 运行在 Hadoop 上，用 HDFS 进行存储，利用 MapReduce 进行计算。

1.2.2 Hive 安装和配置

要安装 Hive，首先要去 Hive 的官网（hive.apache.org）下载安装包（本书选择 Hive-1.2.1 版本）。在这里要说明一下，官网上的安装包不带基于 Web 的图形化的查询工具，如果需要的话，可以自行下载源码包，自己编译打包，生成基于 Web 的图形化查询工具，然后进行部署。这个时候就可以使用 Web 的管理工具来查询数据仓库中的数据了。

我们知道 Hive 是运行在 Hadoop 之上的，所以在安装 Hive 之前，我们要先安装好 Hadoop 环境，Hadoop 可以是单机环境，也可以是伪分布环境，还可以是集群环境，我们采用的是 Hadoop-2.6.0 版的集群环境。

下面介绍 Hive 的安装模式，Hive 有 3 种安装模式，分别是嵌入模式、本地模式、远程模式。

（1）嵌入模式安装

在这种模式下，Hive 的元数据信息被存储在 Hive 自带的 Derby 数据库中。Hive 的嵌入模式有很大的局限，在同一时间，Hive 只允许创建一个连接，这意味着，这个时候只能有一个人可以操作 Hive，这种模式一般只适用于做演示使用。

（2）本地模式安装

实际上本地模式和嵌入模式很相似，这个时候 Hive 的元数据存储在本地另外的数据库当中，通常我们使用 MySQL 作为 Hive 的元数据数据库。在这种模式下，允许多个用户同时连接，这种模式一般用在我们的开发和测试中。

（3）远程模式安装

一般生产环境采用的都是远程模式，在远程模式下，Hive 和元数据数据库 MySQL（一般是 MySQL，本书中同样采用的是 MySQL，如果不做特殊的说明，本章所有的元数据数据库都指的是 MySQL 数据库）运行在不同的机器上，且操作系统也可能不一样。在这种模式下，Hive 允许多用户同时连接。

下面分别介绍 Hive 的 3 种模式下的安装步骤。

安装环境说明：

- 操作系统：Ubuntu14.04 LTS。
- 软件版本：apache-hive-1.2.1-bin.tar.gz。
- Hadoop：hadoop-2.6.0 集群模式。

1. Hive 的嵌入模式安装步骤

1）解压安装包，并且将安装包复制到指定的目录，假设为：/opt/software。

```
tar -zxvf apache-hive-1.2.1-bin.tar.gz -C/opt/software
```

解压后进入到该目录中，可以看到下列子目录：

```
bin    examples  lib          NOTICE        RELEASE_NOTES.txt
conf   hcatalog  LICENSE      README.txt    scripts
```

简单介绍一下 Hive 的目录结构。

- bin 目录：存放的是一些可执行文件，比如 Hive 常用的一些指令等。
- conf 目录：存放的是 Hive 的配置文件，比如 Hive 的元数据存储信息等的配置，都在这个文件目录中。
- examples 目录：存放 Hive 官方提供的一些案例程序。
- lib 目录：存放 Hive 的一些 JAR 包，通过这些 JAR 包，我们就可以调用 Hive 的指令来执行操作了。

2）将 Hive 安装目录下的 lib 目录中的 jline-2.12.jar 文件复制到 Hadoop 安装目录下的 share/hadoop/yarn/lib 目录中，否则在启动 Hive 的时候会报错。

```
cp jline-2.12.jar /opt/software/hadoop-2.6.0/share/hadoop/yarn/lib
```

3）到 Hive 安装目录下的 bin 目录中，执行 ./hive 命令启动 Hive，在启动 Hive 的同时，Hive 会自动创建一个 Derby 数据库作为元数据存储介质，至此，Hive 的嵌入模式安装完成。如果见到下面的提示，说明 Hive 已经安装成功了。

```
hadoop@hadoop:/opt/software/hive/bin $ ./hive
Logging initialized using configuration in jar:file:/opt/software/hive/lib/hive-common-1.2.1.jar!/
    hive-log4j.properties
hive>
```

接下来可以试试 Hive 是否能够正常使用。如下所示，使用 SHOW DATABASES 查询数据库。

```
hive>SHOW DATABASES;
OK
Default
Time taken:1.58 seconds,Fetched:1 row(s)
hive>
```

现在也可以输入其他的命令，就像操作数据库一样，可以创建表、插入数据、删除数据等。

2. Hive 的本地模式和远程模式安装

由于 Hive 的本地模式和远程模式非常相似，所以这里就介绍 Hive 的远程模式安装。远程模式安装意味着 Hive 的元数据存储在远程机器上，远程机器可以是 Linux 系统，也可以是 Windows 系统。这个根据实际生产环境所决定。

在本章中，我们使用 MySQL 作为 Hive 的元数据数据库，远程机器操作系统是 Ubun-

tu14.04LTS 版。关于 MySQL 的安装，可以参考本章的第 1.3.2 小节。

下面将分为两个阶段进行安装工作。

第一阶段：进行与 MySQL 相关的准备工作。

1）在进行 Hive 的远程模式安装之前，先登录 MySQL 查看已有的数据库的信息。

```
hadoop@hadoop:/opt/software $ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with;or \g.

mysql > SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| hive               |
| mysql              |
| performance_schema |
+--------------------+
4 rows in set (0.00 sec)
```

结果显示，登录 MySQL 数据库查询到已有四个数据库，分别为：information_schema、hive、mysql、performance_schema。

2）创建元数据信息库。

在上面列表中，如果发现名称为 Hive 的数据库不存在，那么可以手动新建一个名为 hive 的数据库，用来存储 Hive 数据仓库中的元数据信息（当然也可以不手动创建它，因为 Hive 的相应配置可以支持自动创建元数据信息库）。

下面先手工新建一个名称为 hive 的数据库：

```
mysql > CREATE DATABASE hive;
Query OK,1 row affected (0.00 sec)
```

这个时候我们已经成功地创建了数据库，目前里面暂时是空的。

```
mysql > USE hive;
Database changed
mysql > SHOW TABLES;
Empty set (0.00 sec)
```

3）MySQL 驱动程序准备。

因为我们使用 MySQL 作为元数据数据库，所以还需要把 MySQL 的驱动放到 Hive 安装目录下的 lib 子目录中。MySQL 驱动程序可以去官网上下载，我们选择的是 mysql-connector-java-5.1.39-bin.jar 这个文件包。检查该包是否存在于 lib 目录中：

```
root@ master:/opt/software/apache - hive - 1. 2. 1-bin/lib# ll my *
- rw - r - - r - - 1 root root 855948 Nov  8 15:23 mysql - connector - java - 5. 1. 39 - bin. jar
```

如上所示，查询到 mysql - connector - java - 5. 1. 39 - bin. jar 已位于 lib 目录之下。

4）MySQL 的访问账号配置。

因为我们是使用 root 账号来做演示的，所以需要让 root 账号可以被远程连接，同时删除所有的匿名用户。下面是具体的操作步骤：

```
hadoop@ hadoop: ~ $ /usr/bin/mysql_secure_installation

NOTE:RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MySQL
     SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MySQL to secure it,we'll need the current
password for the root user.  If you've just installed MySQL,and
you haven't set the root password yet,the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK,successfully used password,moving on...

Setting the root password ensures that nobody can log into the MySQL
root user without the properauthorisation.

You already have a root password set,so you can safely answer 'n'.

Change the root password? [Y/n] n
```

当提示是否修改 root 密码时，由于已经设置好了，所以这里选择 n。接下来会显示下面这个提示界面：

```
By default,a MySQL installation has an anonymous user,allowing anyone
to log into MySQL without having to have a user account created for
them.  This is intended only for testing,and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
```

询问是否删除匿名用户，这里选择 Y 确认，之后的每一次询问都选择 Y。

```
By default,MySQL comes with a database named 'test'that anyone can
access.  This is also intended only for testing,and should be removed
before moving into a production environment.
Remove test database and access to it? [Y/n] y
 - Dropping test database...
```