第5章

数据仓库设计与开发

5.1 数据仓库设计概述

企业数据仓库的建设是以现有企业业务系统和业务数据的积累为基础。数据仓库不是静态的概念，只有把信息及时交给需要这些信息的使用者，供他们做出改善其业务经营的决策，信息才能发挥作用。而把信息加以整理归纳和重组，并及时提供给相应的管理决策人员，是数据仓库的根本任务。因此，数据仓库的设计目的是建立一个面向企业决策者的分析环境或系统，因此，它与传统的以业务为主的 OLTP 系统的设计有较大不同，不仅要设计一个数据库和用户接口，还必须设计数据装载策略、数据存取工具和不间断的维护方案。

扫一扫

视频讲解

5.1.1 数据仓库设计的特点

数据仓库设计是以业务和需求为中心，由数据驱动。其中，以业务和需求为中心是指围绕业务需求与问题，确定系统范围和总体框架，由数据驱动是指所有数据建立在已有数据源的基础上，从已存在于操作型环境中的数据出发进行数据仓库设计。数据仓库的设计具有以下几个特点。

1. “数据驱动”的设计

数据仓库的构建是在原有的事务数据库基础上进行的，即数据仓库的数据是事务数据库中数据的另一种形式。因此，数据仓库设计要从面向事务处理环境的数据出发，将

其转换为数据仓库面向主题的数据，并使其提高决策支持效果。

2. “决策驱动”的设计

“决策驱动”的设计方法不再面向事务处理，而是从决策分析需求出发进行设计。因此，数据仓库设计时要从已有的数据源出发，按照决策分析主题对数据源中的数据及其关系重新开始考查，重新组织形成数据仓库决策分析需要的主题数据。

3. “需求模糊”的设计

面向事务的数据库设计针对一系列明确的事务处理需求，而在数据仓库设计中，不存在事务处理中固定和确定的数据处理流程，数据的分析处理需求更加灵活。因此，数据仓库的分析需求在设计初期往往比较模糊或不明确。

4. “螺旋周期”的设计

数据仓库在设计过程中，由于其内容、结构、粒度及其他物理设计，需要根据用户的反馈和建议进行调整，同时，用户在使用数据仓库的过程中，又会不断增加新的决策主题。因此，数据仓库系统的开发是动态反馈的启发式过程。

5.1.2　数据仓库设计与业务系统模型设计的区别

面向 OLTP 的事务型系统模型设计和面向 OLAP 的分析型数据仓库设计，在设计目标和方法等方面都有很多不同，具体如表 5-1 所示。

表 5-1　数据仓库设计与业务系统模型设计的区别

不同点	业务系统 DB	分析系统 DW
处理类型	操作型数据环境，面向业务和应用	面向主题的分析型数据环境
面向需求	确定的应用和业务需求	需求不确定
设计目标	快速响应，大量用户并发数据操作	少量用户大数据量的查询和分析
数据来源	业务流程中产生的数据	生产系统中的历史数据及外部数据
设计方法	应用需求驱动	业务驱动＋数据驱动

扫一扫

视频讲解

5.1.3　数据仓库构建模式

数据仓库的设计模式一般有自顶向下、自底向上、混合方法和联邦方法。下面主要介绍先整体再局部的自顶向下构建模式和先局部再整体的自底向上构建模式。

1. 自顶向下构建模式

Ralph Kimball 和 Inmon 都是数据仓库的首创者，但在数据仓库的设计和机构上却很不相同。Inmon 支持“独立数据集市机构”，主张先创建数据仓库，即对分散于各业务数据库中的数据特征进行分析，在此基础上实施数据仓库的总体规划和设计，构建完整

的数据仓库并提供全局数据视图，然后从数据仓库中分离部门业务的数据集市，即逐步建立面向各主题的数据集市，以满足决策需求。自顶向下构建模式如图 5-1 所示，可以明显看出数据由数据仓库流向数据集市。

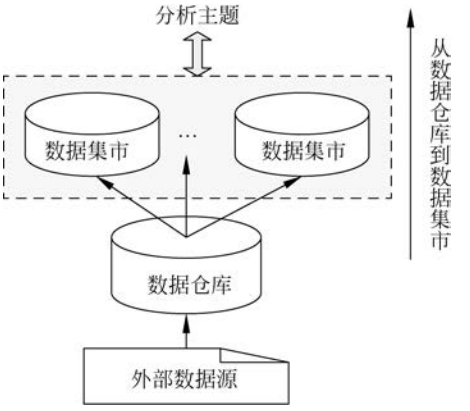

图 5-1　自顶向下构建模式

Inmon 的自顶向下构建模式数据规范化程度高，面向企业构建结构稳定和数据质量可靠的数据中心，可以快速、有效地分离面向部门的应用，从而降低数据冗余与不一致性程度，并且当前数据、历史数据与详细数据整合，有利于全局数据的分析挖掘。其缺点是建设周期长，风险程度大。

2. 自底向上构建模式

Ralph Kimball 提出先将企业内部各部门的需求作为分解后的决策子目标，以这些子目标构建各自的数据集市，然后对系统不断扩充逐步完善数据仓库，实现企业级决策支持。自底向上构建模式如图 5-2 所示，可以明显看出这种构建模式是一种从数据集市到数据仓库再到数据源（先有数据集市再有数据仓库）的敏捷开发方法。

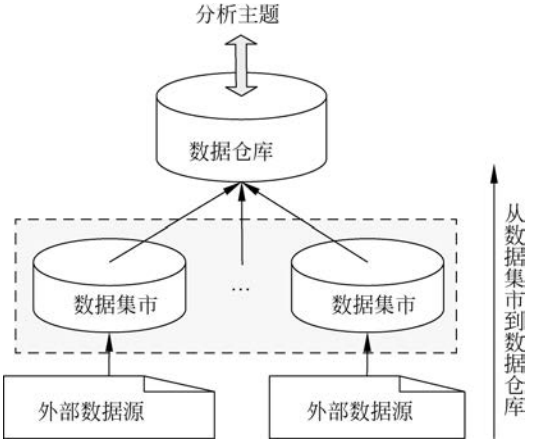

图 5-2　自底向上构建模式

Kimball 的自底向上构建模式往往意味着快速交付、敏捷迭代，不会对数据仓库架构做过多复杂的设计，但其缺点是数据需要逐步清洗，数据提取会有重复，因此信息需要进一步精炼。

5.1.4　数据仓库建立框架

数据仓库的设计从数据、技术和应用 3 方面展开，各方面工作完成之后进行数据仓库部署与运行维护，如图 5-3 所示。其中，技术路线的实施包括技术选择和产品选择，如何采用有效的技术和合适的开发工具是实现一个好的数据仓库的基本条件；数据路线的实施包括模型设计、物理设计和数据处理 3 个步骤，用以满足对数据的有效实施和管理；应用路线的实施分为应用设计和应用开发两部分。

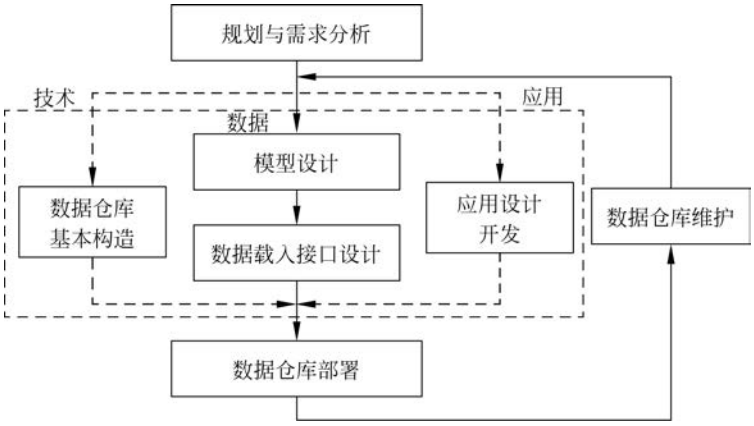

图 5-3　数据仓库建立的基本框架

5.2　数据仓库设计

数据仓库设计过程主要包括数据仓库规划与需求分析、数据仓库逻辑模型和物理模型设计以及数据仓库部署与运行维护。

5.2.1　数据仓库规划、需求分析及概念设计

扫一扫

视频讲解

1. 数据仓库规划

数据仓库规划主要产生建设数据仓库的策略，确定建立数据仓库的长期计划，并为每个建设阶段设定目标、范围和验证标准。

1）确定开发目标和实现范围

数据仓库是面向主题的用以决策支持的数据集合，因此，数据仓库的开发目标就是为服务对象提供决策支持。数据仓库的开发目标确定之后，就可以通过解决一系列与实现相关的问题逐步确定数据仓库的实现范围，即确定数据仓库在为用户提供决策支持时

需要哪些数据源以及需要什么工具来访问。

2）确定数据仓库系统的体系结构

数据仓库的设计中，一般有虚拟数据仓库、单纯数据仓库、单纯数据集市和数据仓库加数据集市4种体系。不同的体系结构实现的难易程度和实现代价也不尽相同。因此，需要根据数据仓库的目标和实现范围选择合适的体系结构。

3）确定数据仓库系统的应用结构

从用户视图看，一个数据仓库的功能分为前端显示和后端处理两部分，但从软件开发人员的角度看，系统功能应该分为数据管理、事务控制、应用服务和用户界面四大功能。这些系统功能可以在客户端和服务器端进行多种划分，且每种功能划分都称为一种应用结构。因此，在数据仓库设计时要确定具体的应用结构。

4）确定数据仓库的项目预算

在数据仓库规划时，要根据使用部门、使用人数、基础软硬件平台等粗略估计系统开发的成本。

2. 数据仓库需求分析

数据仓库按主题组织数据。因此，数据仓库应用系统的需求分析，必须紧紧围绕主题进行，主要包括主题分析、数据分析和环境要求分析。

1）主题分析

需求分析的中心工作是主题分析，主题是由用户提出的分析决策的目标和需求。在此阶段要通过大量沟通，对用户需求进行梳理，归纳出主题并分解为若干需求层次，构成从宏观到微观、从综合到细化的主题层次结构。对于每个主题，需要进行详细调研，确定要分析的指标和用户从哪些角度分析数据维度，并确定用户分析数据的细化或综合程度即粒度。

2）数据分析

数据仓库系统以数据为核心，在明确系统主题后就需要对业务系统的数据源进行分析。数据分析要对数据源状况、数据质量和数量进行分析。

3）环境要求分析

除了主题和数据分析，还需要对满足需求的系统平台与环境进行分析，如设备、网络、数据、接口和软件等。

3. 数据仓库概念设计

概念设计是数据仓库设计的一个重要阶段，其任务是将需求分析阶段确定的各主题转换为概念数据模型表示，并为这些主题的逻辑数据模型设计奠定基础。针对多维数据模型，数据仓库概念设计有以下3个步骤。

（1）设计每个主题的多维数据模型，包括事实和维度名称。

（2）设计每个维度的层次及其名称。

（3）设计每个主题的元数据，包括事实、维度等的类型和长度等。

扫一扫

视频讲解

5.2.2　数据仓库逻辑模型设计

逻辑模型是数据仓库实施中的重要一环，它能直接反映业务部门的需求，同时对系统的物理实施有重要指导作用。数据仓库逻辑模型设计主要是确定数据仓库中应该包含的数据类型及其相互关系，主要工作如下。

1. 确定主题

主题是在较高层次上将企业信息系统中的数据进行综合、归类和分析利用的一个抽象概念，每个主题对应一个宏观的分析领域。主题域是对某个主题进行分析后确定其边界，进一步理解业务关系。在确定系统包含的主题域后，要对每个主题域的内容进行详细描述，描述内容包括主题域的公共码键、主题域间的联系和代表主题的属性组。

2. 粒度设计

粒度指数据仓库的数据单位中保存数据的细化或综合程度的级别。细化程度越高，粒度就越细。粒度问题是设计数据仓库的一个重要方面，因为粒度问题影响存放在数据仓库中数据量的大小，同时影响数据仓库所能回答的查询类型。

3. 数据仓库建模

数据仓库建模是确定数据仓库的多维数据模型是星形模型、雪花模型还是事实星座模型，在此基础上设计相应的维表和事实表，从而得到数据仓库的逻辑模型。

对于第 4 章提到的电力公司售电数据，采用关系数据库时，其逻辑模型描述如下。

(1) 用户维表(Customer_id，用户名，市，区县，乡镇，地址)。

(2) 时间维表(Time_id，原始时间，年，季度，月，日)。

(3) 供电所维表(Station_id，名称，地址，类别)。

(4) 售电事实表(Time_id，Customer_id，Station_id，售电量)。

维表用于存放维信息，包括维属性和维成员，维表中的维一般包含层次关系，也称为概念分层，如在时间维上，按照“年份-季度-月份”形成一个层次。事实表是多维模型的核心，用来记录业务事实并统计相应指标，因此，在查询事实表时，通常还需要用到聚集函数。

扫一扫

视频讲解

5.2.3　数据仓库物理模型设计

数据仓库的物理模型是逻辑模型在数据仓库中的实现模式。构建数据仓库的物理模型与所选择的数据仓库开发工具密切相关。物理模型设计阶段需要确定数据的存储结构、索引策略和存储分配。

设计人员在设计数据仓库物理模型时，首先要全面了解所选用的数据仓库的开发工具，尤其是存储结构和存取方法，其次要了解数据环境、数据使用频度、数据规模和相应时间要求，以及外部存储设备的特性，如分块原则、块的大小的规定和设备的 I/O 特

性等。

1. 确定数据仓库的存储结构

一个数据仓库开发工具一般提供了多种存储结构，不同存储结构有不同的实现方式并具有各自适用的范围和优缺点。设计人员在选择合适的存储结构时需要权衡存取时间、存储空间利用率和维护代价 3 方面的效率。由于统一主题的数据并不要求存放在相同的介质上，因此，为了提高时间效率，在物理设计时，常常将数据按它们的重要程度、使用频率和响应时间进行分类并存储在不同的存储设备中，重要程度高、经常存取并对响应要求高的数据被存储在高速存取设备上，如硬盘；存取频率低或对存取响应时间要求低的数据则可以存储在低速存储设备上，如磁盘或磁带。此外，为了协调空间利用率和维护代价，还可以考虑归并表、引入冗余、分割表和生成派生数据等策略。

1）归并表

当几个表的记录分散在几个物理块时，多个表的存取和连接操作的代价会很大，这时可以将需要同时访问的表在物理上顺序存储，或者通过公共关键字将互相关联的记录放在一起。

2）引入冗余

一些表的某些属性可能在许多地方被用到，因此，通过修改关系模式把某些属性复制到多个不同的主题表中，从而减少一次分析过程需要访问的表的数量。

3）分割表

每个主题中各属性的存取频率是不同的，因此，可以将一张表按照各属性被存取的频率分成两个或多个表，把具有相似访问频率的数据组织在一起。

4）生成派生数据

在原始数据的基础上进行统计和计算，生成派生数据，可以在应用中直接使用这些派生数据，减少 I/O 次数，避免在分析过程中执行过多的统计和计算，以及不同用户进行重复统计可能产生的偏差，提高分析的性能。

2. 确定索引策略

数据仓库的数据量很大，但其中的数据是不常更新的，因此可以设计多种多样的索引以提高数据存取效率。数据仓库中常用的索引技术有树形索引、位图索引、连接索引、散列（哈希）表索引和倒排索引。

3. 确定存储分配

许多数据仓库开发工具提供了一些存储分配的参数供设计者进行物理优化处理，如块的大小、缓冲区的大小和个数等，这些问题都要在物理设计阶段进行确定。

5.2.4 数据仓库部署与维护

完成前面各项工作后，即进入数据仓库的部署和维护阶段。

1. 数据仓库部署

数据仓库的部署阶段主要包括用户认可、初始安装、桌面准备、初始培训、建立初始用户支持和分阶段部署，具体过程和内容如表 5-2 所示。

表 5-2　数据仓库的部署阶段和实施内容

阶　　段	实 施 内 容
用户认可	完成包括所有用户界面及系统性能方面的所有项目的最后测试
初始安装	加载维表、事实表，建立聚集表
桌面准备	安装有需要的桌面用户工具，测试每个用户的计算机
初始培训	培训用户学习数据仓库相关内容和数据访问工具
建立初始用户支持	建立对初始用户的基本使用支持，回答问题建立联系
分阶段部署	将部署分为用户同意的可管理阶段

2. 数据仓库维护

维护数据仓库的工作主要是管理日常数据装入的工作，包括利用接口定期从操作型环境向数据仓库追加数据、刷新数据仓库的当前详细数据、将过时数据转化为历史数据、清除不再使用的数据以及管理元数据等工作。

5.3　基于 Hive 的数据仓库实现

5.3.1　Hadoop/Hive 简介

1. Hadoop 简介

Hadoop 是 Apache 软件基金会旗下的一个开源分布式计算平台，以 Hadoop 分布式文件系统(Hadoop Distributed File System，HDFS)和分布式计算模型 MapReduce 为核心。Hadoop 为用户提供了系统底层细节透明的分布式基础架构，具有高容错性和高伸缩性等特点，允许用户将 Hadoop 部署在由廉价机器构成的集群中，形成分布式系统。MapReduce 计算模型允许用户轻松地组织计算机资源，从而搭建自己的分布式计算平台，充分利用集群的计算和存储能力，完成海量数据的处理。

1) Hadoop 软件框架

Hadoop 软件框架主要包括以下 4 个模块。

(1) Hadoop Common 模块，主要包含其他模块需要的库函数和实用函数。

(2) HDFS。HDFS 是在由普通服务器组成的集群中运行的分布式文件系统。HDFS 简化了文件的一致性模型，通过流式数据访问，提供高吞吐量应用程序数据访问功能，适合带有大型数据集的应用程序。

HDFS 支持文件形式的数据，采用主从(Master/Slave)结构模型，一个 HDFS 集群

由一个 NameNode 和若干个 DataNode 组成。NameNode 作为主服务器，管理文件系统命名空间和客户端对文件的访问操作，DataNode 管理存储的数据。

（3）运算资源调度系统 YARN。YARN（Yet Another Resource Negotiator）是一种新的 Hadoop 资源管理器，实现资源管理和任务调度。YARN 把集群的计算资源管理起来，为调度和执行用户程序提供资源支持。

（4）MapReduce 计算模型。MapReduce 是一种支持大数据处理的计算模型，用 Map 和 Reduce 两个函数编程实现基本的并行计算任务；同时，提供了抽象的操作和并行编程接口，便捷地完成大规模数据的编程和计算处理。

2）Hadoop 的生态结构

随着大数据和云计算时代的到来，越来越多的业务需要处理大数据。Hadoop 的核心 MapReduce 和 HDFS 为大数据处理提供了基本工具，但是在实际业务处理中还需要一些补充的工具。目前 Hadoop 已经发展成为一个包含多个子项目的集合，其中的 Hive、HBase 和 Mahout 等子项目提供了互补性的或更高层的服务。Hadoop 的生态结构如图 5-4 所示。

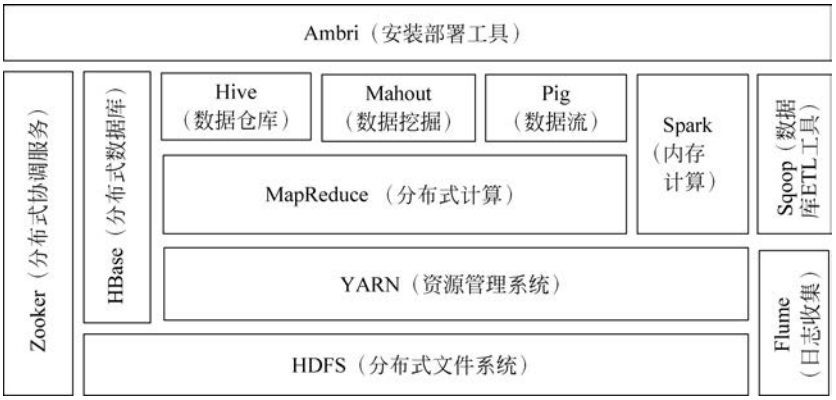

图 5-4　Hadoop 的生态结构

对于 Hadoop 生态中的组件，仅需先了解组件针对的问题，可在实际工作中用到时再具体学习研究。

2. Hive 基础

1）Hive 概述

Hive 是基于 Hadoop 的数据仓库工具，最初由 Facebook 开发，后来由 Apache 软件基金会开发。Hive 可对存储在 HDFS 上的文件中的数据集进行数据整理、特殊查询和分析处理，并将提供的 SQL 查询功能转换为 MapReduce 任务进行运行。对于数据存储，Hive 没有专门的数据存储格式，也没有为数据建立索引，用户可以非常自由地组织 Hive 中的表，只需要在创建表的时候告诉 Hive 数据中的列分隔符和行分隔符，Hive 就可以解析数据。也就是说，Hive 的本质是将 SQL 转换为 MapReduce 的任务进行运算，底层由 HDFS 提供数据的存储。

2）Hive 的特点

Hive 具有以下特点。

（1）Hive 存储架构在一个数据库中并处理数据到 HDFS。

Hive 所有数据都存储在 HDFS 中。Hive 在加载数据的过程中对数据不会进行任何修改，只是将数据移动或复制到 HDFS 中 Hive 设定的目录下。

（2）Hive 专为 OLAP 设计。

Hive 不适合那些需要低延迟的应用，如联机事务处理 OLTP，设计模式遵循联机分析处理 OLAP。

（3）Hive 提供了 SQL 类型语言查询（HiveQL 或 HQL）。

Hive 提供了一套类 SQL 的语言（HiveQL 或 HQL），用于执行查询，类 SQL 的查询方式将 SQL 查询转换为 MapReduce 的任务在 Hadoop 集群上执行。Hive 不提供实时查询和基于行级的数据更新操作。

（4）Hive 快速、简单并且可扩展。

Hive 基于 HDFS，并行查询的效率提高很多，与 HDFS 的扩展性一致，并且具有良好的容错性，节点出现问题时 SQL 仍可完成执行。

3）Hive 的结构

Hive 中主要包括用户接口/界面、元数据存储、HiveQL 处理引擎、执行引擎和 HDFS 或 HBase，结构如图 5-5 所示。

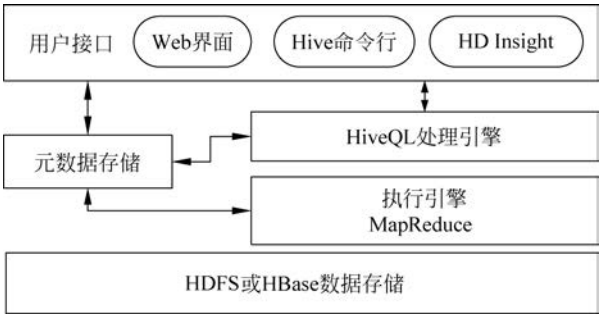

图 5-5　Hive 的结构

（1）用户接口/界面。Hive 是一个数据仓库基础工具软件，可以创建用户和 HDFS 之间的互动。Hive 支持的用户界面有 Web UI、Hive 命令行和 Hive HD Insight（在 Windows 服务器上）。

（2）元数据存储。Hive 将元数据存储在数据库中，如 MySQL、Derby。Hive 中的元数据包括表的名字、表的列和分区及其属性、表的属性（是否为外部表等）、表的数据所在目录等。Metastore 包括元数据服务和元数据存储两部分，因而 Hive 不仅是数据查询工具，还是一个管理海量数据的系统，将 HDFS 上的结构化数据通过元数据映射为一张张表，再使用 HQL 进行查询。

（3）HiveQL 处理引擎。HiveQL 类似于 SQL 用于查询 Metastore 上的模式信息。这是传统的方式进行 MapReduce 程序的替代品之一。相反，使用 Java 编写的

MapReduce 程序，可以编写为 MapReduce 工作，并处理它的查询。

（4）执行引擎。HiveQL 处理引擎和 MapReduce 的结合部分是 Hive 执行引擎。执行引擎处理查询并产生的结果和 MapReduce 的结果一样，它采用 MapReduce 方法。

（5）HDFS 或 HBase。HDFS 或 HBase 数据存储技术用于将数据存储到文件系统。Hive 处理的数据存储在 HDFS，分析数据底层的实现是 MapReduce，执行程序运行在 YARN 上。

4）Hive 与 Hadoop 之间的工作流程

Hive 接到命令之后，首先会去元数据库获取元数据，然后把元数据信息和作业计划发送至 Hadoop 集群执行任务，再将最终的结果返回。具体工作流程如图 5-6 所示。

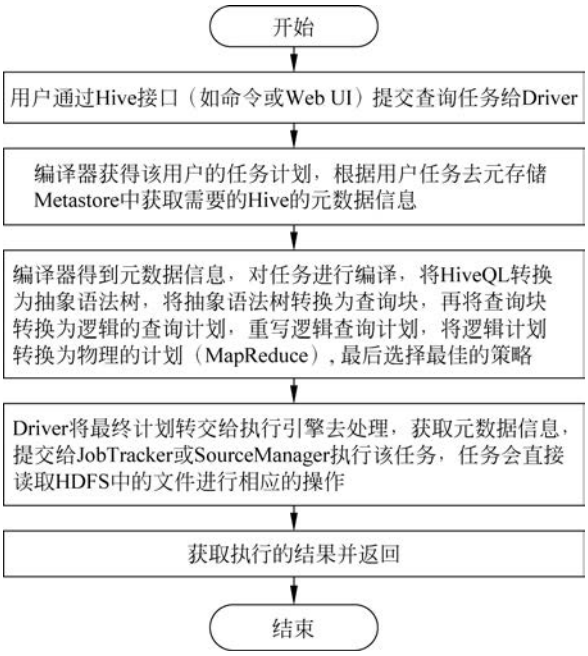

图 5-6　Hive 与 Hadoop 之间的工作流程

5）Hive 中的数据模型

Hive 支持原子和复杂的数据类型。原子数据类型包括数值型、布尔型、字符串和时间戳类型。复杂数据结构包括数组、映射和结构。这些数据是在 HiveQL 中使用的形式，而不是在表中序列化存储的格式。Hive 中主要包含内部表、外部表、分区和桶 4 种数据模型。

（1）内部表。Hive 中的内部表在概念上类似于数据库中的表，逻辑上由存储的数据和描述数据格式的相关元数据组成。内部表的数据存放在 HDFS 文件系统中，元数据存储在关系型数据库中。

（2）外部表。当多个用户对同一份数据进行分析时，创建的表分析完成后即可删除，但删除内部表的同时会把数据删除，因此需要使用外部表。与内部表不同，创建外部表仅记录数据所在的路径，删除外部表也只是删除元数据而不删除数据。这样，外部表相

对来说更加安全，数据组织也更加灵活，而且方便共享元数据。

(3) 分区。在 Hive 中查询会扫描整个表的内容，引入分区能使查询时只扫描表中所关心的部分数据，从而提高查询效率。分区就是将满足某些条件的记录打包，相当于按文件夹进行分类。在 Hive 存储中，一个表中可以有一个或多个分区，每个分区以文件夹的形式单独存放在指定文件夹的目录下，文件夹的名字即为所定义的分区字段。

(4) 桶。桶是相对分区粒度更小的划分，分桶可以获得更高的查询处理效率。桶将数据文件按一定的规律拆分成多个文件，每个桶就是表目录(或分区子目录)中的一个文件。数据分桶一般通过某列属性值的哈希值实现。

5.3.2　Hive 数据仓库和数据库比较

Hive 数据仓库与传统意义上的数据库是有区别的。一般来说，基于传统方式，可以用 Oracle 数据库或 MySQL 数据库搭建数据仓库，而 Hive 数据仓库是建立在 Hadoop 集群的 HDFS 之上，通过 ETL 的形式抽取、转换和加载数据。

Hive 提供了类似 SQL 的查询语言——HQL。因此，从结构上看，Hive 数据仓库和数据库除了拥有类似的查询语言之外，再无类似之处，两者的主要差异有以下几点。

1. 查询语言

Hive 提供了类 SQL 的 HQL，因此，熟悉 SQL 的开发者可以方便地使用 Hive。

2. 查询存储系统

Hive 数据仓库使用 Hadoop 集群的 HDFS 存储数据，而数据库的数据存储在块设备上或本地文件系统中。

3. 数据更新

数据库中的数据需要经常进行修改，因此可以使用 insert 和 update 语句。而 Hive 针对数据仓库设计，本身性质就是“读多写少”，因此在一些旧版本中并不支持 insert into 插入数据以及 update 更新数据。

4. 数据规模

Hive 数据仓库建立在 Hadoop 集群上，并利用 MapReduce 进行并行计算，支持很大规模的数据。相对而言，数据库支持的数据规模较小。

5. 计算引擎

Hive 实际上就是将用户输入的 HQL 语句转换为 MapReduce 程序来运行，但是数据库有自己的执行引擎。

6. 执行延迟

Hive 数据仓库没有对数据中的某些 key 建立索引，查询时需要扫描整个表中的所有数据，而且 MapReduce 本身具有较高的延迟，因此，相对于数据库来说，Hive 数据仓库执

行查询操作时延迟较高。但是当数据量很大，超出了数据库的处理能力范围时，Hive 的并行计算就具有很大的优势。

7. 可扩展性

Hive 数据仓库建立在 Hadoop 集群之上，因此其可扩展性和 Hadoop 集群的可扩展性一致，而数据库的扩展性非常有限。

8. 应用场景

Hive 数据仓库是为海量数据做数据分析而设计的，而数据库是为实时查询业务而设计的。Hive 数据仓库的实时性较差，因此导致 Hive 数据仓库的应用场景和数据库有很大不同。

5.3.3 Hive 常用数据操作

Hive 中针对数据库的常用操作如表 5-3 所示。

表 5-3 Hive 中常用的数据操作语句

操作类型	语　句	内　容
数据库操作	show databases;	查看数据库
	use 数据库;	进入某个数据库
	show tables;	展示所有表
	desc 表名;	显示表结构
	show partitions 表名;	显示表名分区
	show create table_name;	显示创建表的结构
表结构修改	create table xxx;	创建内部表
	create table xxx like tx;	创建和 tx 表结构相同的表
	create external table xx;	创建外部表
	create external table xxx (l int) partitioned by (d string);	分区表
	alter table table_name set TBLPROPROTIES ('EXTERNAL'='TRUE');	内部表转外部表
	alter table table_name set TBLPROPROTIES ('EXTERNAL'='FALSE');	外部表转内部表

5.3.4 利用 Hive 建立数据仓库

1. 创建数据库

```
hive> create database if not exists datamine;
hive> show databases;
hive> use datamine;
```

2. 创建用户维表并导入数据

```
hive > create external table users(user_id string, name string, address string, county
string, village string) row format delimited fields terminated by ',';

OK
Time taken: 0.527 seconds
hive > load data local inpath '/home/data/User.csv' into table users;
Loading data to table datamine.users
OK
Time taken: 1.596 seconds
hive > select * from users limit 5;
OK
1300157601   ***   甘肃省张掖市甘州区北大街北街街道北环路   甘州区   北街街道
1300157757   ***   甘肃省张掖市甘州区北大街北街街道北环路   甘州区   北街街道
1300157799   ***   甘肃省张掖市甘州区北大街北街街道北环路   甘州区   北街街道
1300157874   ***   甘肃省张掖市甘州区北大街北街街道北环路   甘州区   北街街道
1300157920   ***   甘肃省张掖市甘州区北大街北街街道北环路   甘州区   北街街道
Time taken: 0.19 seconds, Fetched: 5 row(s)
```

3. 创建时间维表并导入数据

```
hive > create external table time(time_id string, year_month string, year string, month
string) row format delimited fields terminated by ',';

OK
Time taken: 0.062 seconds
hive > load data local inpath '/home/data/time.csv' into table time;
Loading data to table datamine.time
OK
Time taken: 1.233 seconds
hive > select * from time limit 5;
OK
0       202009  2020    9
1       202010  2020    10
2       202011  2020    11
3       202012  2020    12
4       202101  2021    1
Time taken: 0.128 seconds, Fetched: 5 row(s)
```

4. 创建售电事实表并导入数据

```
hive > create external table electricity_sales(user_id string, time_id string, num_of_elec
string) row format delimited fields terminated by ',';
```

```
OK
Time taken: 0.042 seconds
hive> load data local inpath '/home/data/electricity_sales.csv' into table electricity_
sales;
Loading data to table datamine.electricity_sales
OK
Time taken: 1.044 seconds
hive> select * from electricity_sales limit 5;
OK
1300157601      0       101
1300157601      1       79
1300157601      2       97
1300157601      3       73
1300157601      4       316
Time taken: 0.2 seconds, Fetched: 5 row(s)
```

5. 查询售电量数据及具体时间

```
hive> select e.user_id, t.year_month, e.num_of_elec from electricity_sales e, time t where
e.time_id = t.time_id limit 5;
OK
1300157601      202009  101
1300157601      202010  79
1300157601      202011  97
1300157601      202012  73
1300157601      202101  316
Time taken: 29.081 seconds, Fetched: 5 row(s)
```

6. 查询售电量及对应的用户住址

```
hive> select e.user_id, u.address, e.num_of_elec from electricity_sales e, users u where e.
user_id = u.user_id limit 5;
OK
1300157601      甘肃省张掖市甘州区北大街北街街道北环路      101
1300157601      甘肃省张掖市甘州区北大街北街街道北环路      79
1300157601      甘肃省张掖市甘州区北大街北街街道北环路      97
1300157601      甘肃省张掖市甘州区北大街北街街道北环路      73
1300157601      甘肃省张掖市甘州区北大街北街街道北环路      316
Time taken: 29.384 seconds, Fetched: 5 row(s)
```

7. 查询并输出售电量最大的 10 个用户

```
hive > select e.user_id, u.address, t.time_id, t.year_month, e.num_of_elec from electricity
_sales e, users u, time t where e.user_id = u.user_id and e.time_id = t.time_id order by e.
num_of_elec desc limit 10;
OK
1303289202      甘肃省张掖市甘州区  312256  202004  99988
1303289202      甘肃省张掖市甘州区  312256  202004  99988
1301290103      甘肃省张掖市民乐县  186887  202101  9995
1301290103      甘肃省张掖市民乐县  186887  202101  9995
1307321485      甘肃省张掖市山丹县  888245  201812  9994
30026116        甘肃省张掖市甘州区  972024  201908  9994
30026116        甘肃省张掖市甘州区  972024  201908  9994
1307321485      甘肃省张掖市山丹县  888245  201812  9994
1301176151      甘肃省张掖市高台县  174698  201908  9992
1301176151      甘肃省张掖市高台县  174698  201908  9992
Time taken: 70.836 seconds, Fetched: 10 row(s)
```

5.4　小结

数据仓库的设计是以业务和需求为中心，由数据驱动。其中，以业务和需求为中心是指围绕业务需求与问题，确定系统范围和总体框架；由数据驱动是指所有数据建立在已有数据源的基础上，从已存在于操作型环境中的数据出发进行数据仓库设计。

数据仓库的设计模式一般有自顶向下、自底向上、混合方法和联邦方法。Inmon 支持“独立数据集市机构”，主张先创建数据仓库，即对分散于各业务数据库中的数据特征进行分析，在此基础上实施数据仓库的总体规划和设计，构建完整的数据仓库并提供全局数据视图，然后从数据仓库中分离部门业务的数据集市，即逐步建立面向各主题的数据集市，以满足决策需求。Ralph Kimball 提出先将企业内部各部门的需求作为分解后的决策子目标，以这些子目标构建各自的数据集市，然后对系统不断扩充，逐步完善数据仓库，实现企业级决策支持。

数据仓库设计过程主要包括数据仓库规划与需求分析、数据仓库逻辑模型和物理模型设计以及数据仓库部署与运行维护。数据仓库应用系统的需求分析主要包括主题分析、数据分析和环境要求分析。概念设计是数据仓库设计的一个重要阶段，其任务是将需求分析阶段确定的各主题转换为概念数据模型表示，并为这些主题的逻辑数据模型设计奠定基础。逻辑模型能直接反映业务部门的需求，同时对系统的物理实施有重要指导作用。数据仓库的物理模型是逻辑模型在数据仓库中的实现模式。构建数据仓库的物理模型与所选择的数据仓库开发工具密切相关。物理模型设计阶段需要确定数据的存储结构、索引策略和存储分配。数据仓库的部署阶段主要包括用户认可、初始装载、桌面准备、初始培训、建立初始用户支持和分阶段部署。

Hadoop是Apache软件基金会旗下的一个开源分布式计算平台，以HDFS和分布式计算模型MapReduce为核心。Hadoop为用户提供了系统底层细节透明的分布式基础架构，具有高容错性和高伸缩性等特点，允许用户将Hadoop部署在由廉价机器构成的集群中，形成分布式系统。Hadoop软件框架主要包括Hadoop Common模块、HDFS、运算资源调度系统YARN和MapReduce计算模型4个主要模块。

Hive是基于Hadoop的数据仓库工具，最初由Facebook开发，后来由Apache软件基金会开发。Hive可对存储在HDFS上的文件中的数据集进行数据整理、特殊查询和分析处理，并将提供的SQL查询功能转换为MapReduce任务进行运行。

习题5

1. 单项选择题

(1) 关于数据仓库设计，下列说法中正确的是(　　)。

A. 只能从各部门业务应用的方式设计数据模型

B. 不可能从用户的需求出发设计数据仓库

C. 在进行数据仓库概念模型设计时必须设计实体关系图

D. 在进行数据仓库主题数据模型设计时要强调数据的集成性

(2) 下列有关数据仓库粒度设计的叙述正确的是(　　)。

A. 粒度越细越好　　B. 粒度越粗越好

C. 粒度应该与数据仓库的主题对应　　D. 以上都不对

(3) 下列有关数据仓库与分割策略的叙述正确的是(　　)。

A. 分割越细越好

B. 分割策略与数据量大小和速度等因素有关

C. 分割越粗越好

D. 以上都不对

(4) 下列有关数据仓库建模的叙述正确的是(　　)。

A. 需求分析中已经考虑主题，建模时不再确定主题域

B. 需求分析中已经确定项目的所有功能，没必要再进行数据仓库建模工作

C. 数据仓库建模时设计概念模型，继而导出逻辑模型

D. 数据仓库建模指设计物理模型

(5) 下列有关数据仓库物理模型设计的叙述正确的是(　　)。

A. 存储结构中不能存在任何数据冗余

B. 尽可能多地建立索引

C. 要尽可能把在逻辑上关联的数据放在一个表中

D. 以上都不对

(6) Hive是建立在(　　)之上的一个数据仓库。

A. MySQL　　B. MapReduce　　C. Hadoop　　D. HBase

(7) Hive 的计算引擎是(　　)。

A. Spark　　B. MapReduce　　C. HDFS　　D. HQL

(8) Hive 所处理的数据存储在(　　)。

A. HBase　　B. MapReduce　　C. HDFS　　D. Hadoop

2. 简答题

(1) 简述数据仓库设计的步骤。

(2) 简述维表和事实表的含义。

(3) 简述数据仓库设计中自顶向下和自底向上模式的主要过程。

(4) 简述数据仓库设计的主要过程。

(5) 简述 Hadoop 系统的主要特点。

(6) 简述 Hive 的主要功能和特点。