Broadview®
www.broadview.com.cn

Spark SQL 内核剖析

朱锋 张韶全 黄明 著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Spark SQL 内核剖析

朱锋 张韶全 黄明 著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

Spark SQL 是 Spark 技术体系中较有影响力的应用（Killer application），也是 SQL-on-Hadoop 解决方案中举足轻重的产品。本书由 11 章构成，从源码层面深入介绍 Spark SQL 内部实现机制，以及在实际业务场景中的开发实践，其中包括 SQL 编译实现、逻辑计划的生成与优化、物理计划的生成与优化、Aggregation 算子和 Join 算子的实现与执行、Tungsten 优化技术、生产环境中的一些改造优化经验等。

本书不属于入门级教程，需要读者对基本概念有一定的了解。在企业中任职的系统架构师和软件开发人员，以及对大数据、分布式计算和数据库系统实现感兴趣的研究人员，均适合阅读本书。

图书在版编目（CIP）数据
Spark SQL 内核剖析 / 朱锋, 张韶全, 黄明著. —北京：电子工业出版社，2018.7
ISBN 978-7-121-34314-8
I. ①S… II. ①朱… ②张… ③黄… III. ①数据处理软件 IV. ①TP274
中国版本图书馆 CIP 数据核字 (2018) 第 111300 号

策划编辑：张春雨
责任编辑：牛　勇
印　　刷：三河市君旺印务有限公司
装　　订：三河市君旺印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编：100036
开　　本：787×980　1/16　印张：18　字数：390 千字
版　　次：2018 年 7 月第 1 版
印　　次：2018 年 9 月第 2 次印刷
定　　价：69.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：（010）51260888-819，faq@phei.com.cn。

推荐序 1

互联网技术经过几十年的发展已经渗透到人们生活的方方面面，从云计算、大数据到如今如火如荼的人工智能和区块链，相信无论是圈内人还是圈外人，对这些名词都耳熟能详了。仔细一算，“大数据”这个概念的出现已经有十多年了，背后催生的技术可以说是百花齐放、百家争鸣。

2009 年年初，腾讯从传统的数据仓库转向基于 Hadoop 架构的大数据平台，至今将近 10 年，历经了 3 代跨越式的发展：2009—2011 年是以 Hadoop 为基础的离线计算时代，2012—2014 年是以 Spark 和 Storm 为引擎的实时计算时代，2015 年至今是以腾讯自研的高性能机器学习平台 Angel 为核心的智能学习时代。从最简单的统计报表的计算，到万亿特征维度的算法训练，从结构化数据到图片、语音、文本等非结构化数据，腾讯一直用前沿技术来挖掘大数据背后的价值。

如今，腾讯大数据集群规模达到几万台服务器，存储数据量有几百 PB，每天有几十 PB 的计算量，支撑着腾讯包括微信、QQ、游戏、广告、支付、视频、音乐等关键业务，助力腾讯业务发展，服务着十亿级别的用户。正是历经了腾讯数以亿计的海量数据的锤炼，让腾讯大数据平台得到快速的发展，其技术在业内处于领先水平。

腾讯大数据起源于网络社区，并一直积极参与网络社区的建设。2014 年，腾讯大数据平台（TDW）的核心组件进行开源，我们在 Hadoop、Spark、Docker、Ceph、HBase、Kubernetes、Kafka、Storm、Flink、PostgreSQL 等众多社区项目上积极“反哺”社区。2017 年 6 月，我们在 GitHub 上把腾讯大数据第三代的高性能分布式机器学习平台 Angel 进行了开源，吸引了海内外众多知名企业用户，并于 2018 年 3 月贡献给 Linux 深度学习基金会（LF Deep Learning）。

除代码层面的开源外，近年来，腾讯也把大数据能力开放给传统企业，我们服务了政务民生、金融、交通、零售、教育、工业等各行各业的用户，旨在让没有大数据人才的企业也能具备使用大数据的能力。我们乐于把腾讯积累了十年的大数据技术和运维经验对外分享、对外输出，本书也可以看作是腾讯大数据技术开放的一部分。

本书的内容最初是腾讯内部为进行 Spark SQL 开发而整理的技术文档，最后剥离出通用的部分集结成册。从数据的维度来看，无论是单机还是分布式环境，SQL 对用户来说都是非常重要的。Spark SQL 作为腾讯大数据平台中最基础的部分，支撑全公司的数据分析业务。因此，书

中的内容并非是针对 Spark SQL 技术的空谈，而是立足于腾讯大数据平台的大量实践经验。

本书的几位作者正是工作在腾讯大数据一线的工程师和技术专家，在日均百万级别的 SQL 业务处理和优化中积累了丰富的经验。综观全书，条理非常清晰，读者既能在高度上知晓来龙去脉和他山之石，又能在深度上体会源码级别的技术点剖析。同时，书中结合实践展示了一些通用案例，避免读者陷入到代码的汪洋大海中。

于我个人而言，大学毕业后在传统的银行工作。后来，在数据爆发的时代，我有幸在国内数据最多的两家公司工作，我在阿里巴巴负责支付宝 BI 数据平台基础架构和应用架构，来到腾讯后一直负责腾讯的大数据业务。十多年的职业生涯，转换了公司，也转换了工作和生活的城市，但一直不变的是我的工作始终围绕着“数据”展开，无论是在传统 IT 行业，还是在互联网行业，“数据”始终是我工作的核心内容，而我自己最大的职业追求也离不开“数据”。

未来，在人们的生活中，数据将无时无刻无处不在，数据与商业的真正结合将爆发出强大的生命力和价值。作为服务于上层业务的基础支撑平台，最重要的地方在于技术的沉淀和积累，不断打磨优化。从技术研发人员的角度来讲，最重要的是修炼好自己的“内功”，不忘初心。最后，希望每一位读者都能够从本书中有所收获，练好数据的“内功”，与数据结缘。

蒋杰 博士
腾讯首席数据官、腾讯数据平台部总经理
CCF 大数据专家委员会委员
2018 年 7 月

推荐序 2

非常高兴收到了师弟朱锋、张博士和明哥关于 Spark SQL 的书稿，也非常荣幸被邀请为本书作友情序言。本人是朱锋在中科院软件所读博时的师兄，2014 年也曾经在明哥组内实习，目前在中科院从事大数据方面的科研工作，主要关注 Spark/Flink 等大数据处理系统及大数据分析算法，之前也在 GitHub 上写过 SparkInternals 等介绍技术原理的文章。

去年就听闻师弟他们在写一本关于 Spark SQL 的书，希望能够将生产环境中平台开发建设的一些经验总结出来。当时感觉有些惊讶，惊讶师弟从学术界到工业界能够快速转变，短时间内深入理解了整个系统，并能将经验总结成书。后来想了一下，这也是在情理之中的。朱锋的博士论文的内容就和 SQL-on-Hadoop 解决方案相关，在 Hadoop、Hive、HBase 等系统上也积累了多年的开发应用经验，在腾讯接触工业界实际案例后能够迅速应用所学知识，加上读博期间训练出来的抽象表达能力，与张博士一起，在明哥的全力支持下，足以写出一本有深度的技术图书。

虽然是熟人，但刚收到书时，我也有着顾虑。相信大多数读者也是如此，就是想知道这本书是否值得读，讲没讲干货，能否让读者快速理解 Spark SQL 的原理，能否对读者的实际工作有用。带着这些问题，我开始了阅读，发现越读干货越多，从 Spark SQL 历史到 Spark SQL 语法解析，从逻辑执行计划和物理执行计划生成，继续往下切入到 Tungsten 内存管理，全面讲述了 Spark SQL 技术的方方面面，不仅有原理介绍，还有实现细节的描述和总结，更有在海量数据和海量业务下的实践经验的总结。这些对想深入理解 Spark SQL，并对其进行优化改进，以及想更高效地使用 Spark SQL 的开发者、用户都有莫大的帮助。

抱着学习的态度读完本书，我在以下几个方面受益匪浅：逻辑执行计划、物理执行计划、优化方法等背后的技术原理，本书均透彻地讲清楚了。在读的过程中，我也在感慨，记得前几年带组里的师弟师妹们想在 Spark SQL 中添加关键词查询功能（类似 MySQL 中的 MATCH 关键词），发现需要在 Spark SQL 中添加新语法、改进语法解析、对翻译生成执行计划等一系列模块进行改进，由于当时缺少 Spark SQL 技术体系引导和深入解析的书籍，因此我们花了不少时间扎入代码，并自行总结技术体系和实现原理。本书的内容对于想对 Spark SQL 进行二次开发的读者非常有用，可以让二次开发事半功倍。

因为平时工作需要写论文，同时也会写技术报告和文档，所以深知写一本简洁易懂又包含复杂技术知识的图书有多困难，不仅需要花费大量精力阅读代码、分析细节，还需要在高层进行抽象总结、简洁表达。在阅读本书的过程中，可以从字里行间感受到作者们的用心和付出的努力。例如，作者在讲解 ANTLR 4 时，首先自己设计一个简单的例子，并进行相应的代码实现，在生成逻辑算子树时用到的访问者模式也会以实例进行说明。在技术展现方面，要画出图 3.8、图 7.10 等，不仅需要仔细阅读代码中的每个类的实现，而且需要考虑图形布局以达到直观的效果。

Spark 是一个易用性、通用性都很高的框架，除 Spark SQL 外，上层还有面向图计算的 GraphX 框架、机器学习 MLlib 库，流处理的 Spark Streaming 库，希望包括几位作者在内的业界专家能够为读者带来更多高水平的解析和总结。Spark SQL 本身也在演变中，希望本书有第二版、第三版等，能够不断加入更多的技术解析，不断完善。作为一名研究者，我自己也会努力去设计实现更多能够解决大数据实际问题的系统和方法，共勉之。

许利杰 (@JerryLead)
http://www.tcse.cn/~xulijie/
中科院软件所
2018 年 6 月 15 日

前言

极其迅速的信息传播将人们带入了大数据时代，也推动了大数据技术的发展。Spark 于 2009 年诞生于伯克利大学 AMP 实验室，至今已经形成完整的生态圈。除参与度高的开源社区外，各种相关的技术分享和论坛（如每年的 Spark Summit）也是如火如荼。得益于其灵活的 RDD 计算模型，Spark 系统高效地支持了各类应用，涉及 SQL 处理、图计算和机器学习等。

本书重点讲解 Spark SQL，该系统在企业中的应用非常广泛，也是 Spark 生态圈中较活跃的部分。从另一个视角来看，Spark SQL 是近年来 SQL-on-Hadoop 解决方案（包括 Hive、Presto 和 Impala 等）中的佼佼者，结合了数据库 SQL 处理和 Spark 分布式计算模型两个方面的技术，目标是取代传统的数据仓库。

在实际生产环境中，因为一些个性化的需求，往往涉及对原生的 Spark SQL 系统进行定制化的改造或新特性的添加，此过程需要开发人员对内部实现有深入的了解。然而笔者发现，目前业界在这方面的资料还比较缺乏，虽然已经涌现了一系列的文章和书籍，但内容通常都以 Spark 本身为主，或者停留在 API 使用和概括性介绍层面，难以满足开发人员的需求。

本书定位于弥补这方面的空白，对 Spark 的基本概念和功能（如 RDD 和调度等）不再展开讲解，而是将内容重点放在 SQL 内核实现的剖析上，旨在同读者一起“入于其中”，从源码实现上学习分布式计算和数据库领域的相关技术。

本书面向的读者

本书主要面向在企业中任职的系统架构师和软件开发人员，以及对大数据、分布式计算和数据库系统实现感兴趣的研究人员。需要注意的是，本书对读者在大数据系统和数据库方面的基础知识（SQL）上有一定的要求。对于初学者来说，最好能够首先参考相关资料，做到有所了解。

本书的主要内容

本书的内容可以分成 4 个部分：（1）背景和基础知识概述（第 1 ～ 2 章），这两章分别介绍 Spark SQL 的前生今世和与 Spark 相关的基础知识，对此熟悉的读者可以直接跳过；（2）Spark SQL 功能实现的各个阶段（第 3 ～ 6 章），这 4 章结合简单例子分别从整体和每个阶段的细节介绍内部机制，涉及 SQL 编译、逻辑计划和物理计划；（3）专题展开（第 7 ～ 10 章），这 4 章重点介绍 Spark SQL 中 Aggregation 和 Join 实现，深入分析 Tungsten 计划中的几项优化技术，以及 Spark SQL 连接 Hive 的实现；（4）实践部分（第 11 章），这一章分享 Spark SQL 系统在生产环境中的应用和一些改造优化经验。

一些约定和说明

- 相关术语：Spark 依赖于 JVM，主体采用 Scala 开发语言，部分功能也用到了 Java 语言来实现。本书没有严格地区分两种语言的术语，例如 Scala 语言中的 trait（特质）和 Java 语言中的 interface（接口）等，书中一般以 Java 语言的术语为主。此外，本书涉及的源码分析较多，不方便直接翻译的类或接口命名，以其英文命名为主。一些 SQL 关键词（例如 Select、Join 等）或 Spark 术语（例如 Shuffle、Partition、Executor 等）在不同上下文环境中也会出现大小写混用的情况。另外，Map 和 Reduce 虽然是来自 MapReduce 中的概念，但本书在介绍 Spark SQL 时也使用了这两个概念，分别用来表示 Shuffle 前的阶段和 Shuffle 后的阶段。
- 版本说明：本书使用的 Spark 版本是 Spark 2.X。笔者在写作时，以 2.1 版本和 2.2 版本中的实现机制为主。然而，Spark 社区活跃，版本的演化非常迅速（平均半年一个版本），读者在理解基本的框架和思路后可以结合 JIRA 上的相关 Issue 和对应的 Patch 进行跟踪。当然，后期最好的方式是参与到社区的贡献中。
- 推荐的阅读方式：本书内容涉及的实现细节较多，因此建议的阅读方式是结合代码进行理解。调试环境搭建好之后，在关键步骤插入日志信息，纵向（宏观）和横向（细节）分析交叉进行，最终做到在脑海中将上层的 SQL 语句映射为底层的 RDD 模型。

前沿技术的整理和分析并不是一件轻松的工作，从大纲的确定、内容的选择到最终出版得益于多方的大力支持。在此感谢电子工业出版社的各位编辑对本书出版提供的帮助，感谢马朋勃、马骉和邓飞等提出的宝贵修改建议。写作是一个不断学习并进行归纳和整理的过程，笔者在写作中也受到相关技术博客和论文思路的启发，在此一并感谢。

因笔者水平有限，本书的错漏和不足之处欢迎广大读者朋友批评指证。如果有更好的建议，也欢迎通过电子邮件联系几位作者：朱锋（wellfengzhu@gmail.com）、张韶全（shaoquan. zhang@hotmail.com）和黄明（andyyehoo@gmail.com）。

目　录

第1章 Spark SQL 背景

技术的诞生往往都有着特定的历史背景，而对技术来龙去脉的了解有助于我们从宏观层面把握全局方向。本章从大数据概念产生以来 10 多年的技术发展轨迹讲起，简要回顾 Spark SQL 的演化历程。

1.1 大数据与 Spark 系统

大数据一词，最早出现于 20 世纪 90 年代，由数据仓库之父 Bill Inmon 所提及。2008 年，*Nature* 杂志出版了大数据专刊“Big Data”，专门讨论海量数据对互联网、经济、环境和生物等各方面的影响与挑战。2011 年，*Science* 出版了如何应对数据洪流（Data deluge）的专刊“Dealing with Data”，指出如何利用大数据中宝贵的数据价值来推动人类社会的发展。迄今为止，大数据并没有统一的标准定义，业界和学术界通常用 5 个方面的属性（5V）来描述大数据的特点：Volume（体量大）、Velocity（时效高）、Variety（类型多）、Veracity（真实性）、Value（价值大）。

大数据一方面意味着巨大的信息价值，另一方面也带来了技术上的挑战，使得传统的计算机技术难以在合理的时间内达到数据存储、处理和分析的目的。大数据应用的爆发性增长，已经衍生出独特的架构，并直接推动了存储、网络和计算技术的研究。Google 公司于 2003 年在 SOSP 会议上发表论文介绍分布式文件系统 GFS[1]，于 2004 年在 OSDI 会议上发表论文介绍分布式大数据编程模型与处理框架 MapReduce[2]，于 2006 年再次在 OSDI 会议上发表论文介绍分布式数据库 BigTable[3] 的实现。以上三者统称为 Google 公司初期大数据技术的“三驾马车”，自此各种大数据存储与处理技术开始蓬勃发展。

Spark[4] 分布式计算框架是大数据处理领域的佼佼者，由美国加州大学伯克利分校的 AMP 实验室开发。相比于流行的 Hadoop[5] 系统，Spark 优势明显。Spark 一方面提供了更加灵活丰富的数据操作方式，有些需要分解成几轮 MapReduce 作业的操作，可以在 Spark 里一轮实现；另一方面，每轮的计算结果都可以分布式地存放在内存中，下一轮作业直接从内存中读取上一轮的数据，节省了大量的磁盘 IO 开销。因此，对于机器学习、模式识别等迭代型计算，Spark

在计算速度上通常可以获得几倍到几十倍的提升。得益于 Spark 对 Hadoop 计算的兼容，以及对迭代型计算的优异表现，成熟之后的 Spark 系统得到了广泛的应用。例如，在大部分公司中，典型的场景是将 Hadoop（HDFS）作为大数据存储的标准，而将 Spark 作为计算引擎的核心。

经过多年的发展，Spark 已成为目前大数据处理领域炙手可热的顶级开源项目。屈指一算，Spark 从诞生到 2018 年，已经走过了整整 9 个年头。如图 1.1 所示，第一个版本的 Spark 诞生在 2009 年，代码量仅有 3900 行左右，其中还包含 600 行的例子和 300 多行的测试代码。当时，Hadoop 在国外已经开始流行，但是其 MapReduce 编程模型较为笨拙和烦琐，Matei 借鉴 Scala 的 Collection 灵感，希望开发出一套能像操作本地集合一样简捷、高效操作远程大数据的框架，并能运行于 Mesos[6] 平台上，于是就有了 Spark 最初的 0.1 版本和 0.2 版本。后来，Reynold Xin 加入这个项目，在协助对 Core 模块进行开发的同时，在其之上启动了 Shark[7] 项目，希望能够让 Spark 更好地处理 SQL 任务，替代当时流行的 Hive[8]（基于 Hadoop 的数据仓库解决方案）。当然，这个版本的 Shark 在多个方面都存在先天的不足，Spark 在后来的发展过程中将其废弃，另起炉灶，从头来过，这也就是众所周知的 Spark SQL[9]，相关细节会在后面进一步详述。

图 1.1　Spark 发展历程

在此期间，Spark 经历了一个蓬勃的生长期，从 2012 年的 0.6 版本开始，Core 模块开始趋于稳定，接近生产级别，不再是实验室的产物。我国的阿里巴巴团队开始将其用于线上正式作业，并取得了比使用 MapReduce 更好的效果。同时，Intel 公司也开始投入力量到该项目的开发中。在 0.7 版本中，Tathagata Das 开始加入 Streaming 模块，使得 Spark 具备准实时的流处理能力。到了 0.8 版本，Spark 正式支持在 YARN[10] 上的部署运行，Yahoo 公司贡献了重要的代码，国内的阿里巴巴团队也开始将其正式部署到内部的云梯集群，搭建了 300 台专用 Spark 集群。在 0.9 版本中，图处理系统 GraphX[11] 正式成为独立模块，同年，Apache 接受 Spark 成为顶级项目，从孵化期到正式项目，只经历了短短半年时间，这种速度在 Apache 开源社区是非常难得的。到了 1.0 版本，孟祥瑞等人加入 DataBricks 公司，主导的 MLLib[12] 也成为正式模块。至此，各个主要模块形成了较完整的 Spark 技术栈（生态系统），如图 1.2 所示。

可以看出，在 2012—2014 年，Spark 经历了一个高速的发展过程，各个模块快速演进，各大公司和全球顶尖开发人员的加入，使得整个项目充满生命力和活力。DataBricks 公司成立后，

多数客户的需求集中在常用的数据处理方面，而这需要 Spark 系统有完善且强大的 SQL 能力。因此，在 2014—2017 年，Spark 技术栈重点关注 Spark SQL 子项目，在钨丝计划（Tungsten）的基础上，开始了 DataFrame 和 DataSet 为用户接口核心的 SQL 功能开发，使得 Spark SQL 项目发展迅速，整体内核也针对 SQL 做了很多优化。从 1.6 版本开始，社区的发展一步一个脚印，到如今功能完善的 2.x 版本，Spark SQL 已经非常成熟，完全达到了商用的程度。

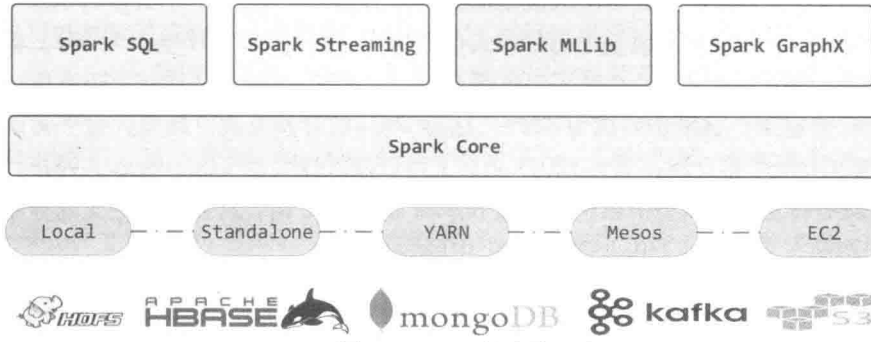

图 1.2 Spark 技术栈

1.2 关系模型与 SQL 语言

计算机软件系统离不开底层的数据，而数据的存储管理需求促进了数据库技术的发展。早期的数据库包含网状数据库和层次数据库等不同类型。在 20 世纪 70 年代，IBM 的研究员 E.F.Codd 提出关系模型[13]。因为具有严格的数学理论基础、较高的抽象级别，而且便于理解和使用，所以关系模型成为现代数据库产品的主流，并占据统治地位 40 多年。此外，IBM 公司研究人员将关系模型中的数学准则以关键字语法表现出来，于 1974 年里程碑式地提出了 SQL（Structured Query Language）语言[14]。SQL 语言是一种声明式的语言，提供了查询、操纵、定义和控制等数据库生命周期中的全部操作。另外，建立在关系模型之上并提供 SQL 语言支持的关系数据库管理系统（如 Oracle、DB2、SQL Server 等），已经成为企业通用的数据存储解决方案。对数据管理人员和应用开发人员来说，关系模型和 SQL 语言一直是必不可少的技术基础。

尽管非结构化数据和半结构化数据在数据量上占有绝大部分比例，但结构化数据和 SQL 需求仍旧具有举足轻重的作用。当 Hadoop 生态系统进入企业时（注：广义上 Spark 也可以看作是 Hadoop 生态系统中的一员），必须面对的一个问题就是怎样解决和应对传统成熟的信息架构。在企业内部，如何处理原有的结构化数据是企业进入大数据领域所面临的难题。Hadoop 生态系统的初衷在于解决日志文件分析、互联网点击流、倒排索引和大文件存储等非结构化数据的问题。在此背景的驱动下，面向 Hadoop 生态系统的 SQL 查询处理技术及框架（统称为

“SQL-on-Hadoop”[15-17]）应运而生。SQL-on-Hadoop 解决方案为用户提供关系模型和 SQL 查询接口，并透明地将存储与查询转换为 Hadoop 生态系统的对应技术来管理海量结构化数据。

作为数据分析领域重要的支撑，SQL-on-Hadoop 成为近几年来广受关注的热点，并涌现出大量的产品，如典型的 Hive、Impala[18] 和 Presto[19] 等。所以，从横向来看，Spark SQL 算是 SQL-on-Hadoop 解决方案大家庭中的重要一员。SQL-on-Hadoop 并非专指某一个特定的系统，而是借助于 Hadoop 生态系统完成 SQL 功能的解决方案的总称。因此，一个具体的 SQL-on-Hadoop 系统往往依赖于 Hadoop 生态系统中的分布式存储技术（如 HDFS、HBase[20] 等），或者利用分布式计算框架（如 MapReduce、Tez、Spark 等），具有高度的灵活性。例如，Hive 既可以转换为 MapReduce 计算框架，也能够转换为 Tez[21] 这种 DAG 的计算方式。此外，对于关系数据表的访问，Spark SQL 既支持直接存储在 HDFS 上，又可以通过连接器（Connector）连接存储在类似 HBase 这种具有特定数据模型的系统中。而 Impala 则将 SQL 语言转换为自定义的分布式计算框架，来访问存储在 HDFS 或 HBase 等 NoSQL 中的数据。

需要注意的是，Spark SQL 这类 SQL-on-Hadoop 解决方案和传统的 MPP 解决方案在架构上存在很大差异。根据 Hadoop 生态系统的特点，SQL-on-Hadoop 解决方案从架构上来看可以简单地划分为三层结构。最上层是应用（语言）层，应用层为用户提供数据管理查询的接口，不同的 SQL-on-Hadoop 系统往往提供各自的 SQL 语法特性，如 Hive 的 HiveQL、Pig 的 PigLatin[22] 和 Spark SQL 的 DataFrame 等。在大数据场景下，应用层也包含一些针对特别需求的接口，如 BlinkDB[23] 所支持的近似查询功能等。应用层之下是分布式执行层，SQL-on-Hadoop 系统通过一定的规则或策略将 SQL 语句转换为对应的计算模型。除 MapReduce、Spark 等通用的计算框架外，分布式执行层可能是某些系统自定义的计算单元，例如 Impala 中的 Query Exec Engine 等。分布式执行层通过接口访问数据存储层中的数据，并完成相应的计算任务。SQL-on-Hadoop 系统的底层是数据存储层，主要负责对关系数据表这样的逻辑视图进行存储与管理。目前，各种 SQL-on-Hadoop 数据存储层基本都支持分布式文件系统 HDFS 和分布式 NoSQL 数据库。总的来看，SQL-on-Hadoop 解决方案类似“堆积木”，各层之间松耦合，并可以灵活组合。数据存储层与分布式执行层之间通过特定的数据读写接口进行数据的交互。这种分层解耦的方式一方面具有通用性（各层可以分别分离为子系统）和灵活性（彼此能够互相组合）的优势，另一方面隔离了各层的特性，限制了深度集成优化的空间。

1.3 Spark SQL 发展历程

Spark SQL 的前身是 Shark[7]，即“Hive on Spark”，由 Reynold Xin 主导开发。Shark 项目最初启动于 2011 年，当时 Hive 几乎算是唯一的 SQL-on-Hadoop 选择方案。Hive 将 SQL 语句翻译为 MapReduce，正如前文所提到的，性能会受限于 MapReduce 计算模型，始终无法满足各种交