还纠结实时数仓选型，Spark +ClickHouse让你拍案叫绝！

dbaplus社群 5/13

The following article is from 爱奇艺技术产品团队 Author 爱奇艺后台研发部

爱奇艺技术产品团队
爱奇艺的技术产品团队

众所周知，爱奇艺拥有海量视频，在视频生产过程中产生的上千QPS的实时数据、T级别的数据存储。要支持这样的数据进行即席查询和多个大表的JOIN，是爱奇艺视频生产团队大数据应用的难点。

具体来说有以下几点：

- 实时性的要求，需要实时的解决方案。
- 生产数据更新频繁，OLAP 需支持更新。
- 生产需要大表 Join 方案。码流属性（亿级，百G）和节目属性（亿级，百G）经常放在一起做分析。

此外，爱奇艺视频生产数据还有一个特点，数据来源于OLTP 数据中台，其数据持久化在 Mongo，消息变动写入 Kafka， Kafka中：curData 是当前更新数据，oriData是历史为变动数据，这样的结构化数据为配置化开发提供了可能。

爱奇艺视频生产团队负责爱奇艺的视频生产，涵盖“素材、成片、运营流、图片”各个方面，并围绕生产进行了中台化建设、监控建设、数据报表建设等，旨在为视频生产提效，节省编辑精力，更快更好的产出优质视频。

针对以上痛点，爱奇艺视频生产团队进行了一系列努力。本文将详细叙述ClickHouse在爱奇艺视频生产实时数仓的应用：包括业务数据是如何通过 Spark / Spark Streaming 计算引擎处理，并将 HBase 作为维表数据存储，进行实时Join，最终写入ClickHouse，实现即席查询的。

最终的建设成果也比较显著，原本报表开发周期由天级缩短到小时级，满足了频繁更新的实时、离线可 Join 的报表需求。

一、背景及发展历史

选择Spark+ClickHouse实时数仓建设方案，基于爱奇艺视频生产的历史发展阶段及数据特点。

随着各种大数据技术蓬勃发展，爱奇艺视频生产的数据业务经历了两个阶段。

早期阶段一：团队基于公司内部 BabelX 离线数据同步工具，引入 Hive 技术，来做报表开发。

在阶段一中，缺点是每天跑全量数据，成本高，实时性低，修改纬度字段时，整条链路都要修改；ETL 完全依赖 Hive 内置函数，可复用性低，运维成本高。

早期阶段二：随着生产数据增多，Mysql 提供的可视化查询性能遇到瓶颈，且实效性要求提高，数据报表进入了第二阶段，引进 ClickHouse 进行实时报表开发。

在引进clickHouse的过程中，我们也研究了业界如druid、kudu等其他方案，结论是：Druid、kudu在用户视频数少，时间跨度大的情况下，性能表现还不错；当用户视频数超过1千万后，Druid会受聚合影响，速度大幅度降低，甚至会出现超时的情况。最终我们选择了clickHouse，通过它的引擎的选择，我们还支持了频繁的数据更新。

这个阶段其缺点是：不支持连表操作，业务库仅支持 JDBC/ODBC 类型，Merge引擎不支持更新，Mysql导入 ClickHouse再Truncate，期间数据存在丢失。

在此基础上，我们完善系统，最终形成了如下的新的架构体系。

二、Spark+ClickHouse实时数仓

话不多说，先上架构图

整体结构

ClickHouse 是面向列的数据库管理系统（DBMS），用于对查询进行联机分析处理（OLAP）。由俄罗斯IT公司 Yandex 为 Yandex.Metrica 网络分析服务开发的。允许分析实时更新的数据，该系统以高性能为目标，且储存明细数据。

Spark 是用于大规模数据处理的统一分析引擎，高效的支撑更多计算模式，包括交互式查询和流处理。一个主要特点是能够在内存中进行计算，即使依赖磁盘进行复杂的运算，Spark依然比MapReduce更加高效。Spark Streaming 是核心 Spark API 的扩展，可实现实时数据流的可伸缩，高吞吐量，容错流处理。其基于微批，和其他基于“一次处理一条记录”架构的系统相比，它的延迟会相对高一些，但是吞吐量也会有一定优势。而批量插入 ClickHouse，又是 ClickHouse 所推崇的。

结合 Spark/Spark Streaming 与 ClickHouse 的特性，这一方案优势也就显而易见了：

ClickHouse 支持更新且速度极快；Spark Streaming 微批，更适合写入clickHouse。

具体建设过程主要分为三个部分。

- **离线数据加工**

首先通过 Spark计算引擎，将 mongo 数据例行全量导入 Hive（担心业务库稳定性）。然后通过 Spark 计算引擎， 将 Hive 数据例行进行 ETL 处理，并离线导入 ClickHouse。

- **实时数据加工**

历史存量数据的处理是通过 Spark 计算引擎，将 Mongo 数据写入 ClickHouse（只执行一次，可以直接从业务库导。因为例行导入 Hive 表本身就是我们在做）。实时数据的处理就是Spark 技术引擎直接处理 Kafka 消息写入 ClickHouse 了。如果不需要历史存量数据，只需要消费 Kafka，实时计算导入 ClickHouse 就可以了。具体实时架构如下：

实时方案流程图

这里离线数据和实时数据连接点需要注意一下：ReplacingMergeTree 引擎由于幂等性质，可将 Kafka offset 向前多重置一些，保证最少一次。其他引擎存在误差数据。除非 Kafka 能够重放 Mongo 中历史所有数据。

- **Join需求**

存在 Join 需求时，由于两个表目前都是百G的存储，使用Redis、CB内存太浪费，我们最终选择了使用HBase。以 HBase 作为纬度表，在 Spark 计算引擎中，进行合并处理，并写入事实表。

大表Join方案流程图

除了以上工作，这里有一些注意事项：

1. 实时导入 ClickHouse，维表数据必须早于事实表产生。
2. 增量离线同步或者实时同步 ClickHouse 时，需保证 维表数据基本不变 或者 维表数据变化后，实时、离线增量数据也会发生变化。
3. 否则维表变化不会在 ClickHouse 输出表中体现。

看到这里，整体架构已经很清晰了。那么如何选择 ClickHouse引擎来支持频繁更新呢?

三、ClickHouse支持频繁更新

针对频繁更新请求，ClickHouse 可以选择 ReplacingMergeTree 和 VersionedCollapsingMergeTree 引擎：

- **ReplacingMergeTree（覆盖更新）**

以 id 作为主键，会删除相同的重复项。
不保证没有重复的数据出现。

- **VersionedCollapsingMergeTree（折叠更新）**

在数据块合并算法中添加了折叠行逻辑。

针对离线数据，有两种选择方案。

方案一：是用 ReplacingMergeTree 引擎的增量同步方案：先用 Spark 计算引擎将 Mongo 数据例行同步到 Hive，再用 Spark 计算引擎消费 Hive 增量数据写入 ClickHouse。其优点是增量同步，压力小。缺点是 Join 时，增量离线同步，需保证 维表数据基本不变 或者 维表数据变化后，实时表数据也会发生变化。否则维表变化不会再事实表中体现。

方案二：是用 MergeTree 引擎的全量同步方案：先用 Spark 计算引擎将 Mongo 数据定时同步到 Hive，然后Truncate ClickHouse 表，最后使用Spark 消费 Hive 近 N 天数据写入 ClickHouse。其优点是可解决方案一 Join 时问题。缺点是全量同步，仅保存近 N 天数据，压力大。

针对实时数据，也有两种选择方案。

方案一：是用 VersionedCollapsingMergeTree 引擎的增量同步方案：先用 Spark 计算引擎将 Mongo 存量数据一次性同步到 ClickHouse，再重置 Kafka 消费位置，将实时数据同步到 ClickHouse。其优点是即使有重复数据，也可使用变种 SQL 避免数据误差。缺点是实时数据强依赖 OLTP 数据中台 提供的 Kafka 消息（oriData、currData）准确性，且离线和实时数据连接点，存在无法折叠现象。

方案二：是用 ReplacingMergeTree 引擎的增量同步方案：先用 Spark 计算引擎将 Mongo 存量数据一次性同步到 ClickHouse，再重置 Kafka 消费位置，将实时数据同步到ClickHouse ReplacingMergeTree。其优点是相比与 VersionedCollapsingMergeTree 更简单，且离线和实时数据连接点，不存在异常。缺点是不保证没有重复的数据出现。

接下来介绍下数据的准确性保证。

四、数据准确性保证

离线数据的准确性保证方面，我们主要做了以下两点。

首先是离线重跑数据时，如果 ClickHouse 是 Merge 引擎，重跑时将 Drop 重跑分区。然后是离线全量重跑近 N 天数据，执行 Spark 任务前会先 Truncate 表。

而实时数据的数据准确性保证，首先是 在 Spark 消费 Kafka 时，offset不自动提交，待本次微批数据的所有业务逻辑均处理完成后，再手动提交 offset，以此达到最少一次消费的目的，保证不会丢数据，而 ClickHouse ReplacingMergeTree 引擎写入是幂等的。然后针对 ClickHouse，每间隔 time 时间主动进行 Merge，考虑服务器压力，只 Merge 最近 time * 2 时间段内修改的分区。目前 time 是 5 min。如下图：

自动Merge示意图

到此针对实时数仓的架构细节已经基本讲完了。

五、配置化开发

然而，面对源源而来的报表需求，每个需求花费几天去开发，不仅耗费人力，而且重复的工作也让开发人员无法抽身。考虑到爱艺奇视频生产都是结构化数据，这就为配置化开发提供了可能。

整个过程主要用到了程序参数解析器 - Apache Commons CLI，一款开源的命令行解析工具。它可以帮助开发者快速构建启动命令，并且帮助你组织命令的参数、以及输出列表等。

参数解析器结构图

六、价值与规划

爱奇艺视频生产实时数仓目前的建设方案完成后，我们基本实现了代码 0 开发，原本报表开发周期由天级缩短到小时级。满足频繁更新的实时、离线可 Join 的报表需求。目前已支持 4 个离线报表任务，3 个实时报表任务，其中 1 个离线 Join 需求，1 个实时 Join 需求，后续可能更多。

后续我们会在爱奇艺视频生产平台提供页面化操作，将同步工具产品化，首先与 Hive、HBase、ClickHouse 等打通，自动建表，然后将任务创建、运行、监控状态逻辑通过调度自动化 。通过技术创新去支持和落地新的业务场景，继续推动爱奇艺的数据和产品向着实时化的方向发展。

作者丨 爱奇艺后台研发部
来源丨公众号：爱奇艺技术产品团队（ID：iQIYI-TP）
dbaplus社群欢迎广大技术人员投稿，投稿邮箱：editor@dbaplus.cn