Apache Flink

基于Flink+Iceberg构建企业级实时数据湖

胡争 · 阿里巴巴 / 技术专家

Apache Flink China Meetup 深圳 – 2020年09月26日

Apache Flink

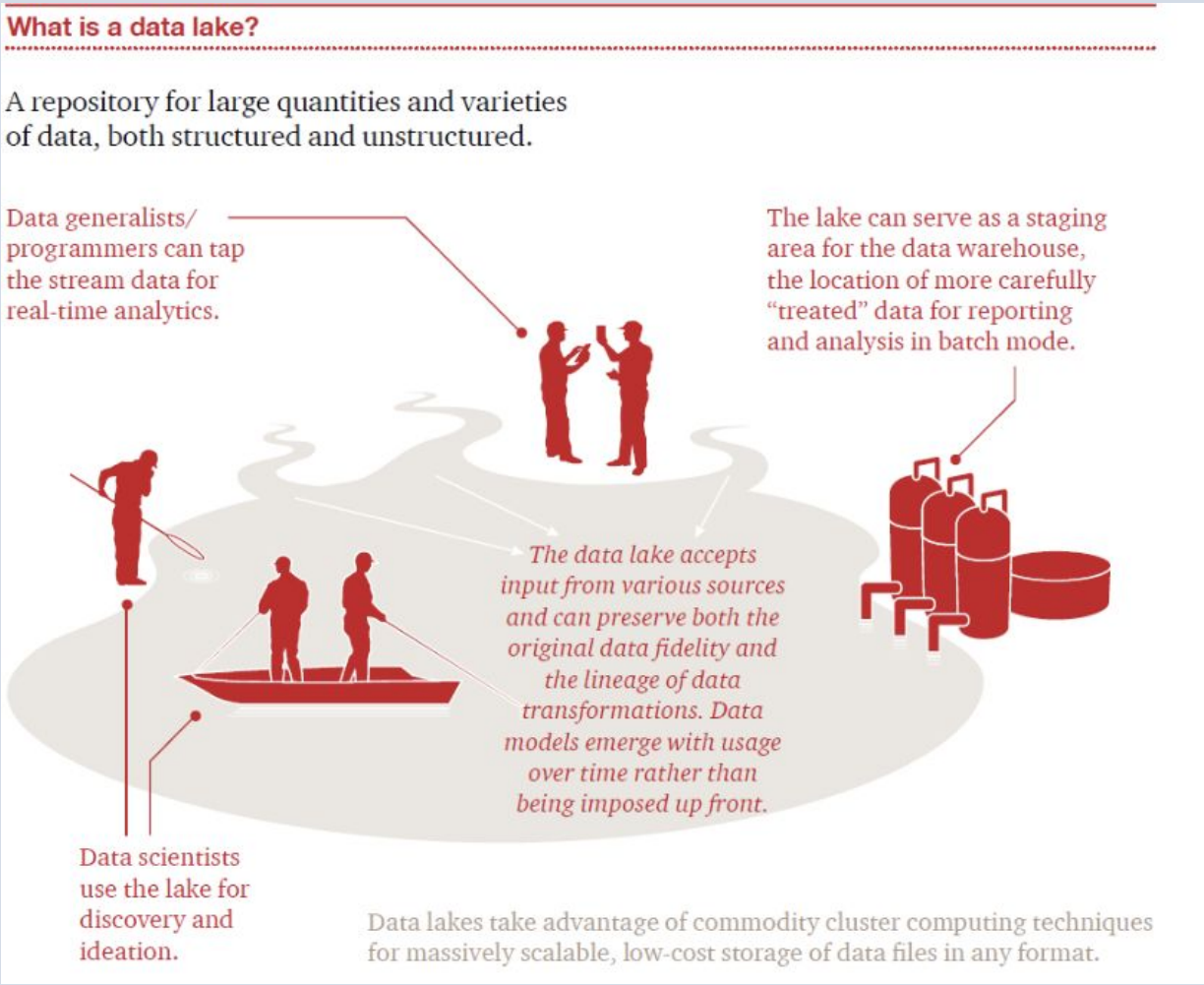
目录

Apache Flink

PART 01

数据湖背景介绍

数据湖



存储原始数据

结构化数据
半结构化数据
非结构化数据
二进制数据(图片等)

多种计算模型

批处理
流计算
交互式分析
机器学习

完善的数据管理

多种数据源接入
数据连接
Schema管理
权限管理

灵活的底层存储

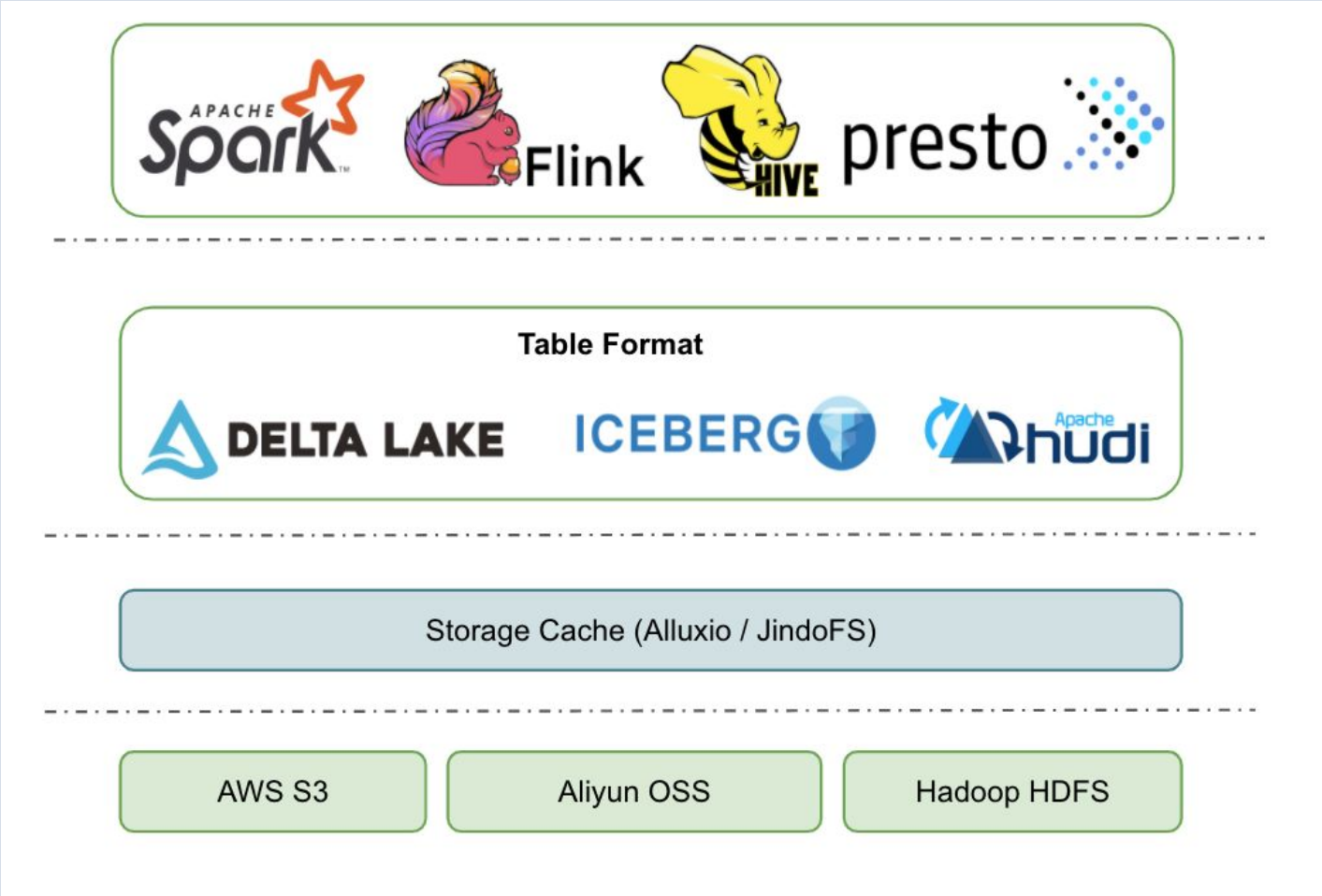
S3/OSS/HDFS
Parquet/Avro/Orc
数据缓存加速

Apache Flink

数据湖和数据仓库

特性	数据仓库	数据湖
数据	来自事务系统、运营数据库和业务线应用程序的关系数据	来自 IoT 设备、网站、移动应用程序、社交媒体和企业应用程序的非关系和关系数据
Schema	设计在数据仓库实施之前（写入型 Schema）	写入在分析时（读取型 Schema）
性价比	更快查询结果会带来较高存储成本	更快查询结果只需较低存储成本
数据质量	可作为重要事实依据的高度监管数据	任何可以或无法进行监管的数据（例如原始数据）
用户	业务分析师	数据科学家、数据开发人员和业务分析师（使用监管数据）
分析	批处理报告、BI 和可视化	机器学习、预测分析、数据发现和分析

开源数据湖架构



计算引擎层。多种计算引擎满足不同的分析需求。

Table Format层，提供面向用户的表级语义。

数据加速层，提供本地数据缓存和元数据加速服务。

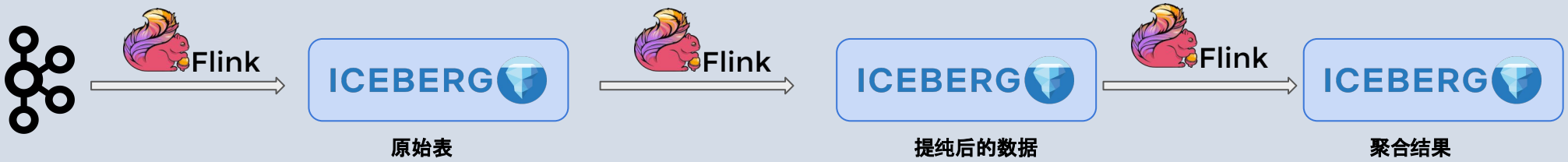
廉价、弹性可扩展的分布式文件系统层

Apache Flink

PART 02

Flink数据湖业务场景介绍

场景一:构建实时Data Pipeline



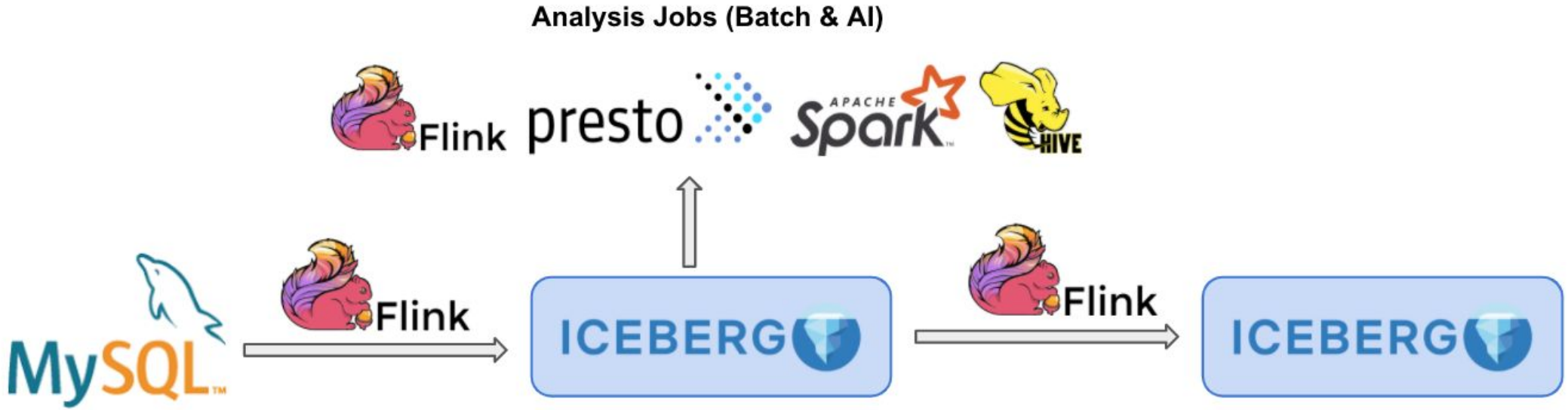
核心优势

- 可以借助flink实现数据exactly-once语义地入湖和出湖。
- 新写入数据可在checkpoint周期内可见。
- 可以方便地构建data pipeline，满足不同业务层的数据加工和分析需求。

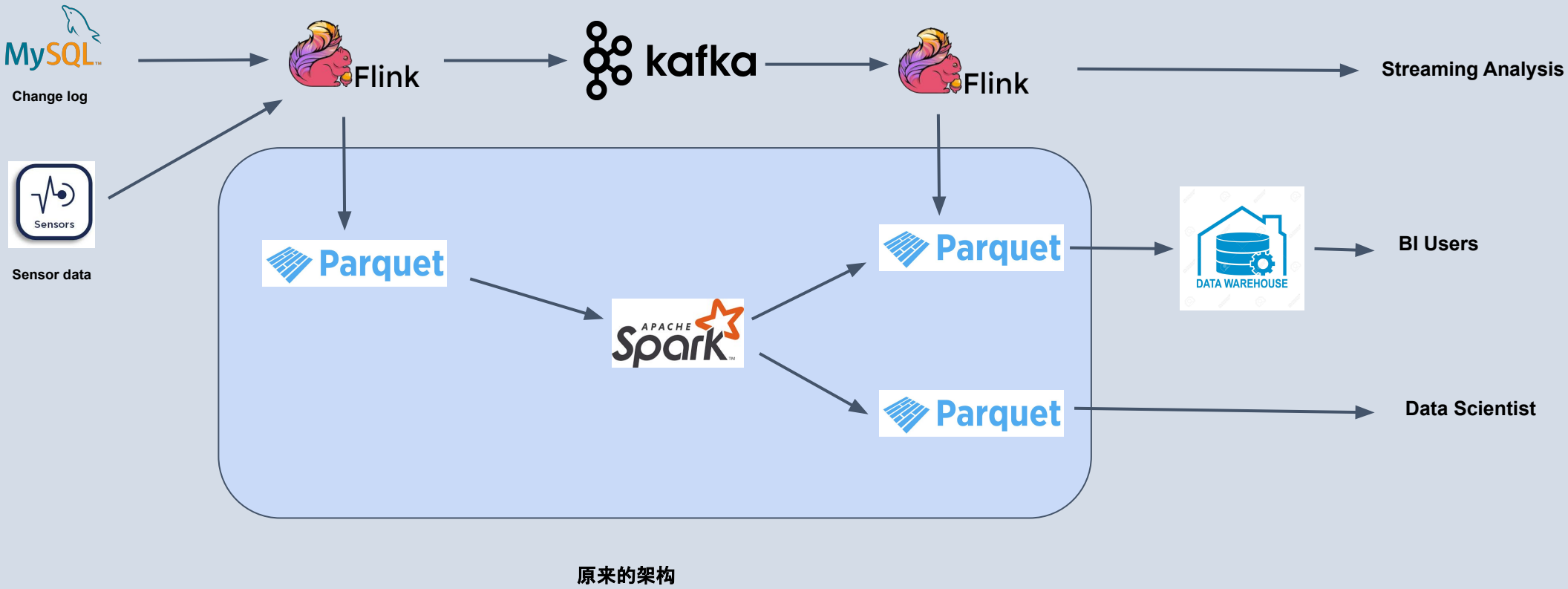
对比Hive方案:

- hive的增量写入以partition为单位，长期高频率的 checkpoint写入，会导致 hive partition膨胀。
- 本质上hive的增量写入和消 费粒度都太大，实时性无法比肩iceberg。

场景二:CDC数据实时摄入摄出



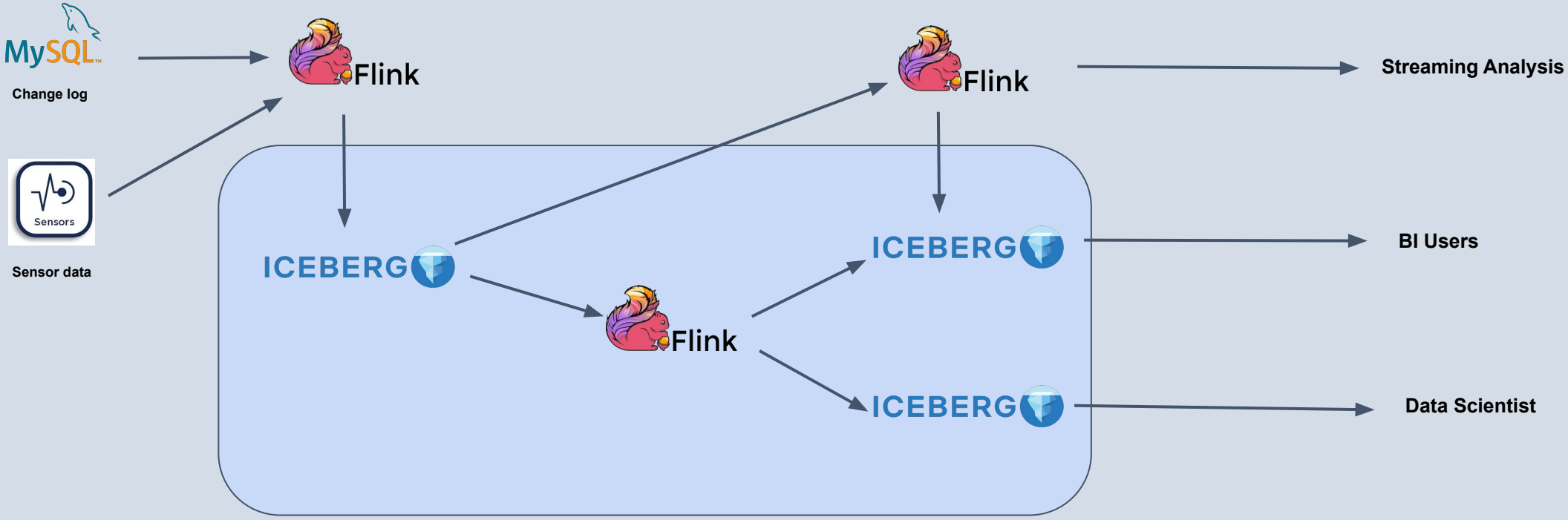
场景三:近实时场景的流批统一



原来的架构

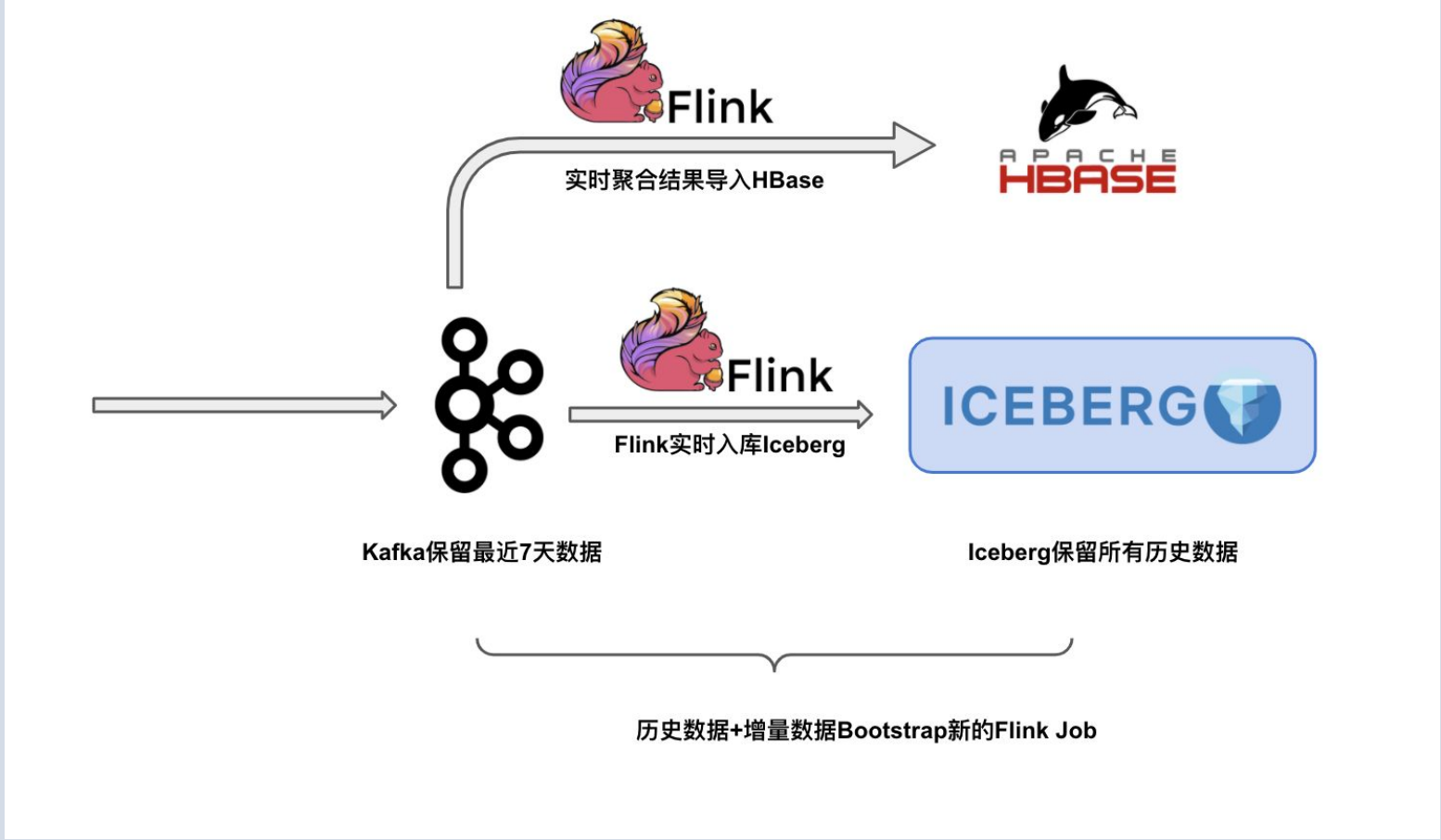
Apache Flink

场景三:近实时场景的流批统一



现有的架构

场景四：从Iceberg历史数据启动Flink任务



场景四：通过Iceberg数据来订正实时聚合结果

Apache Flink

PART 03

为何选择Iceberg？

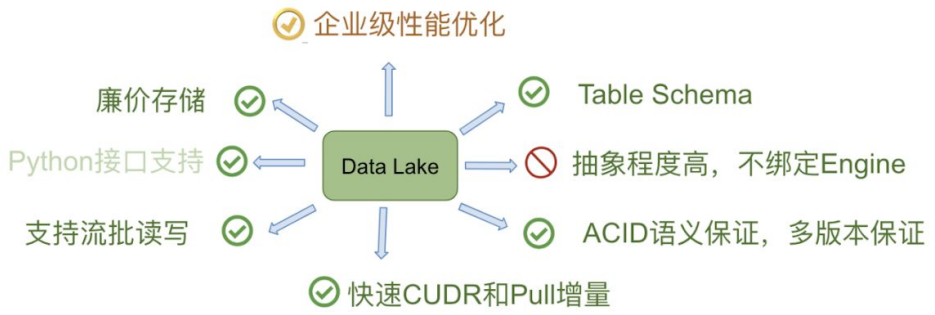
Apache Flink

Delta、Hudi、Iceberg三大开源项目对比

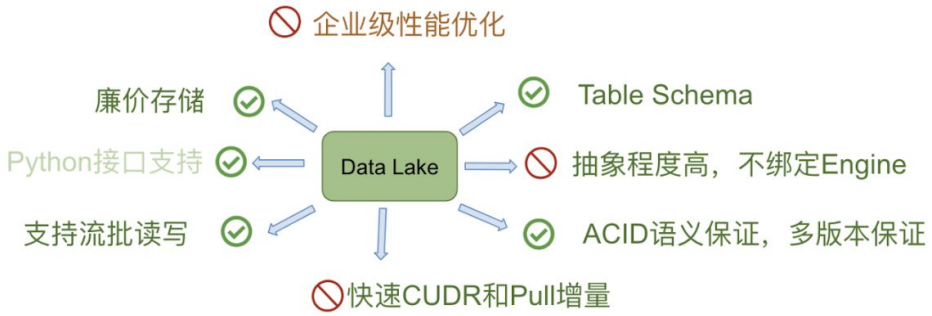
Iteams	Open Souce Delta	Apache Iceberg	Apache Hudi
Open Source Time	2019/04/12	2018/11/06(incubation)	2019/01/17(incubation)
Github Star	2800+	692	1400+
Releases	5	5	48
ACID	Yes	Yes	Yes
Isolation Level	Write/Snapshot serialization	Write Serialzation	Snapshot serialization
Time Travel	Yes	Yes	Yes
Row-level DELETE (batch)	Yes	Ongoing	No
Row-level DELETE (streaming)	No	Ongoing	Yes
Abstracted Schema	No	Yes	No
Engine Pluggable	No	Yes	No
Open File Format	Yes	Yes	Yes(Data) + No(Log)
Filter push down	No	Yes	No
Auto-Compaction	No	Ongoing	Yes
Python Support	Yes	Yes	No
File Encryption	No	Yes	No

Delta、Hudi、Iceberg三大开源项目对比

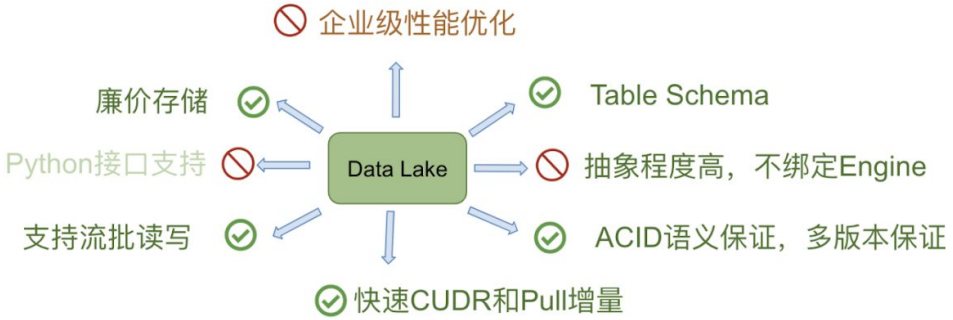
Databricks Delta



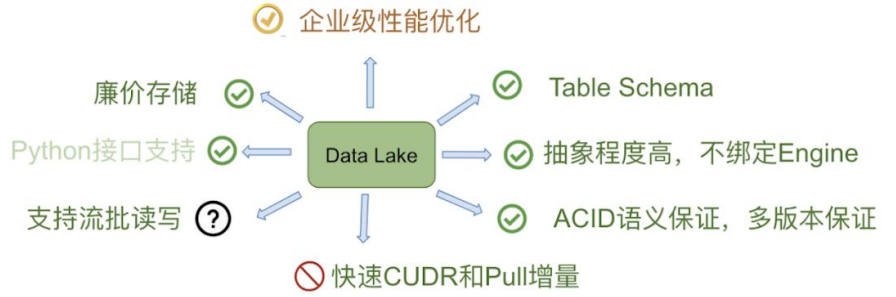
Open Source Delta



Apache Hudi



Apache Iceberg



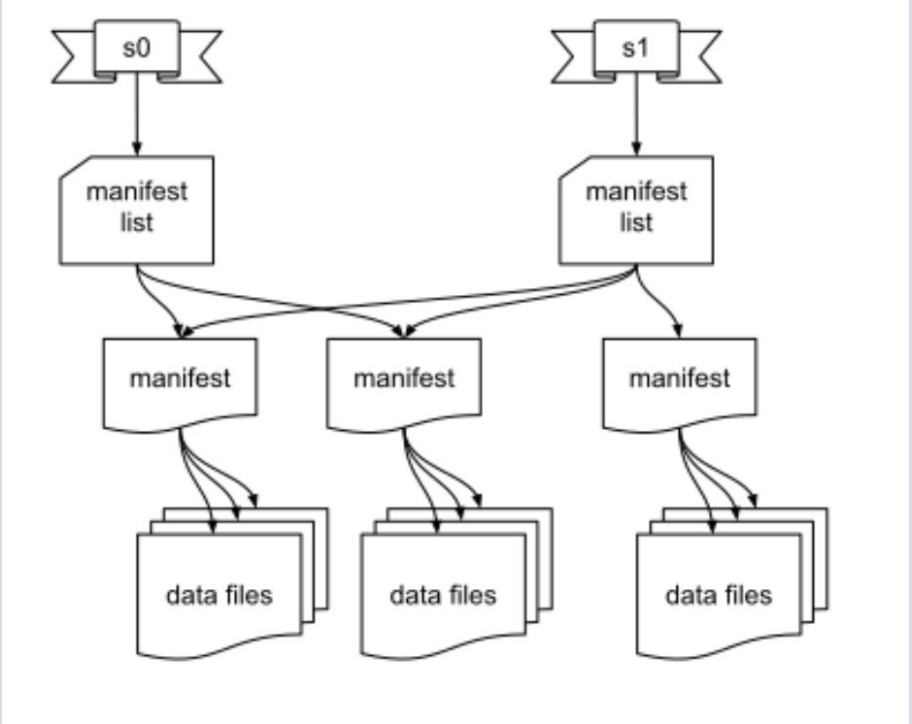
Flink: 为何选择Apache Iceberg

- Iceberg的设计和Flink数据湖的需求最匹配
  - 完美解耦计算引擎和文件格式两层。
    - 便于接入多样化的计算引擎和文件格式。
  - 正确地完成了Table Format这一层的功能需求。
  - 更容易成为Table Format层的事实标准
- 两个项目的长远规划相似
  - Apache Iceberg：打造流批一体的数据湖存储层。
  - Apache Flink: 打造流批一体的计算引擎。
  - 两者合力打造流批一体的数据湖架构。
- 强大的社区资源
  - Apache Iceberg始自Netflix。Netflix最早是All In Cloud的互联网巨头之一，也是最早在线上生产环境运行Flink/Spark+Iceberg这套数据湖方案的公司。
  - 支撑着Apple、LinkedIn、Adobe、腾讯、网易等多家互联网巨头PB级的生产数据。
  - 严苛的文档审核、代码审核及测试设计，有助于打造Format层的规范标准。
  - 最具影响力的Apache资深专家团队。拥有来自其他Apache项目的1个VP、7个PMC、4个Committer。

Apache Iceberg简介

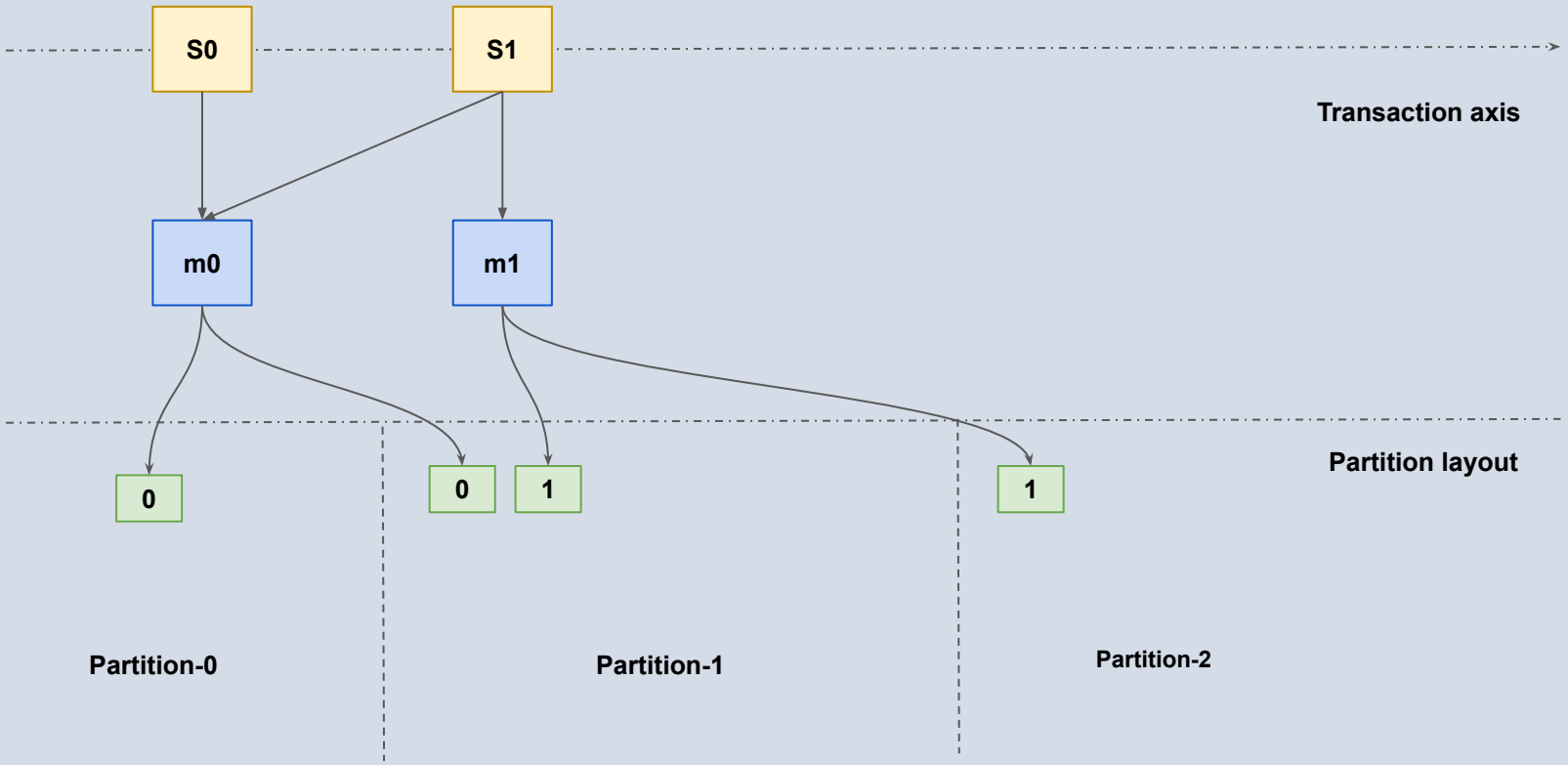
Apache Iceberg is an open table format for huge analytic datasets. Iceberg delivers high query performance for tables with tens of petabytes of data, along with atomic commits, concurrent writes, and SQL-compatible table evolution.

- Apache Iceberg
  - 在文件Format（parquet/avro/orc等）之上实现Table语义。
    - 支持定义和变更Schema
    - 支持Hidden Partition和Partition变更
    - ACID语义
    - 历史版本回溯
  - 特点
    - 借助partition和columns统计信息实现分区裁剪
    - 不绑定HDFS，可拓展到S3/OSS等
    - Serializable Isolation
    - 容许多个writer并发写入。乐观锁机制解决冲突。

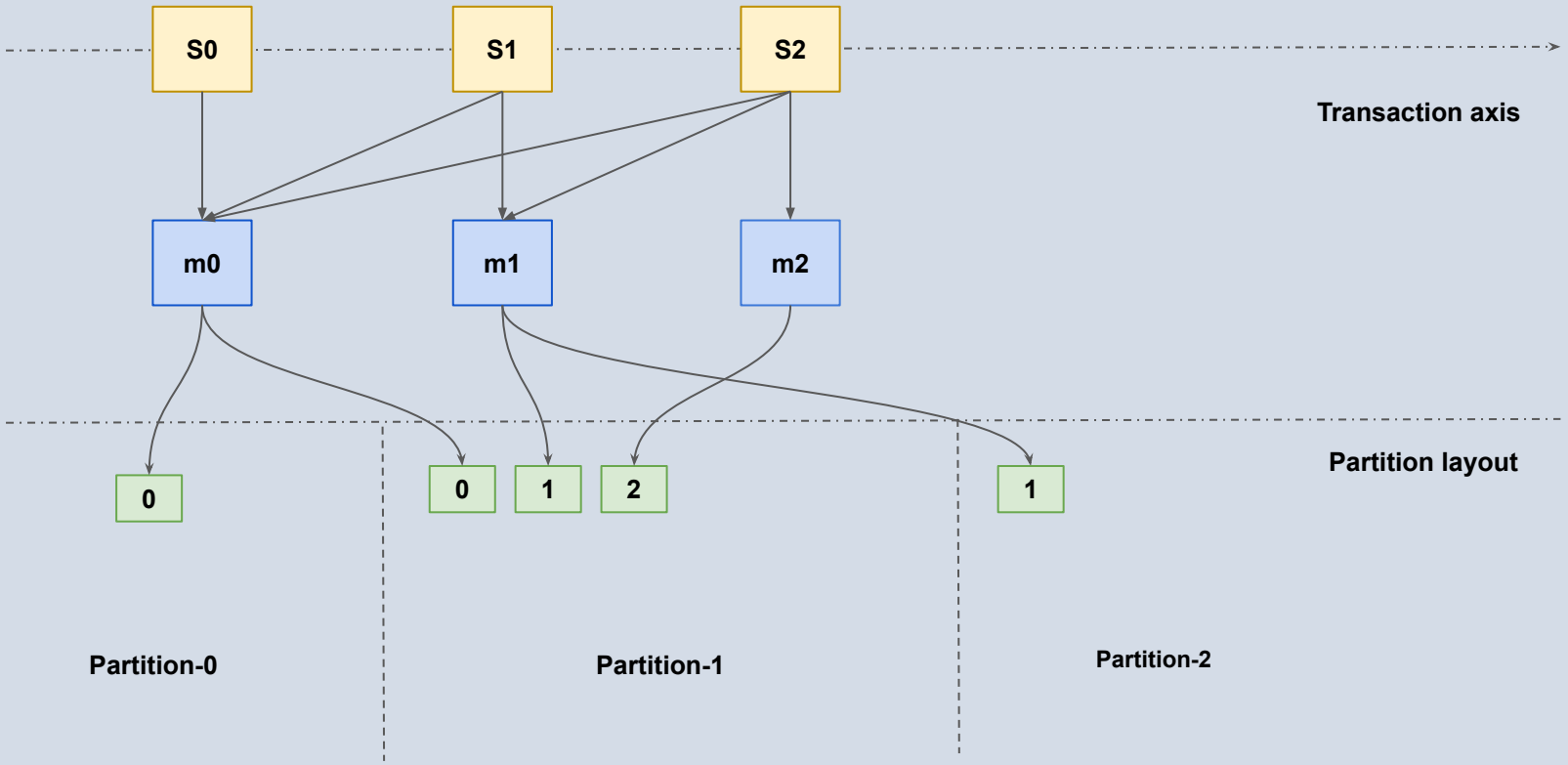


Apache Iceberg Layout

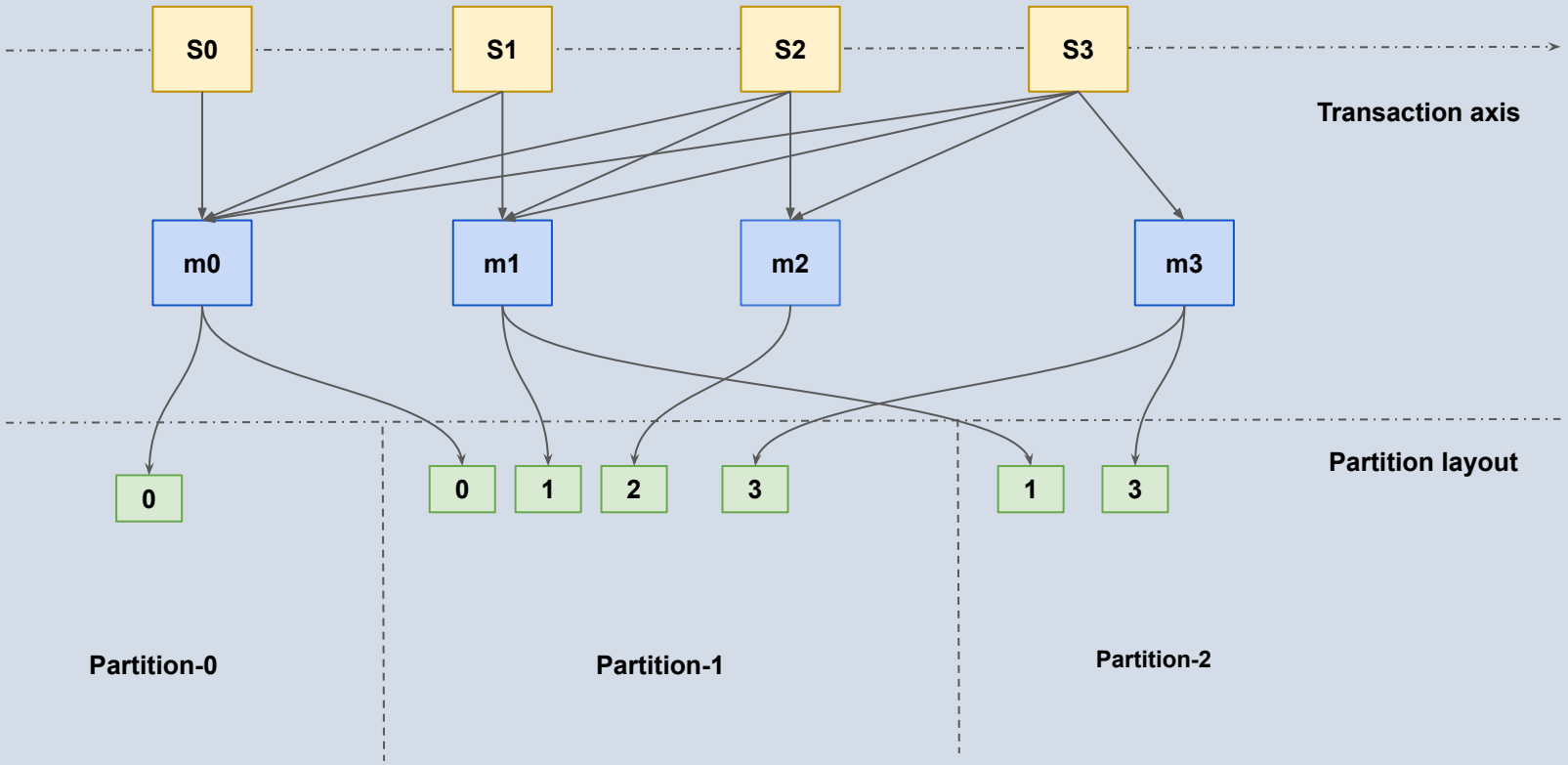
Apache Iceberg 读写原理



Apache Iceberg 读写原理



Apache Iceberg 读写原理



Apache Flink

PART 04

Flink+Iceberg流式入湖

启动Flink SQL Client

相关详细文档请参考:https://github.com/apache/iceberg/pull/1464/files

```
# HADOOP_HOME is your hadoop root directory after unpack the binary package.
export HADOOP_CLASSPATH=`$HADOOP_HOME/bin/hadoop classpath`

# wget the flink-sql-connector-hive-2.3.6_2.11-1.11.0.jar from the above bundled jar URL firstly.
wget https://repo.maven.apache.org/maven2/org/apache/flink/flink-sql-connector-hive-2.3.6_2.11/1.11.2/flink-sql-connector-hive-2.3.6_2

# open the SQL client.
./bin/sql-client.sh embedded \
    -j <flink-runtime-directory>/iceberg-flink-runtime-xxx.jar \
    -j <hive-bundlded-jar-directory>/flink-sql-connector-hive-2.3.6_2.11-1.11.0.jar \
    shell
```

注意:

1. iceberg-flink-runtime.jar 目前需要通过手动编译apache iceberg项目获得。等0.10.0发布之后，可以从Iceberg官网下载。
2. flink-sql-connector-hive-xxx-xxx.jar 可以从flink官网下载。https://ci.apache.org/projects/flink/flink-docs-stable/dev/table/hive。

第一步:创建Iceberg的Catalog

```
CREATE CATALOG hive_catalog WITH (
  'type'='iceberg',
  'catalog-type'='hive',
  'uri'='thrift://localhost:9083',
  'clients'='5',
  'property-version'='1'
);

USE CATALOG hive_catalog;
```

A. 创建Hive Catalog

- **type**: 表示flink connector的类型，这里直接选择 *iceberg*。
- **catalog-type**: 表示创建catalog的类型。目前支持*hive*和*hadoop*两种类型。
- **uri**: Hive metastore 的 thrift URI 地址。
- **clients**: Hive metastore 客户端线程池大小，默认值为2。
- **property-version**: Connector的属性值版本号，当前版本为1。这个值主要用来确保iceberg connector的property向后兼容性。

```
CREATE CATALOG hadoop_catalog WITH (
  'type'='iceberg',
  'catalog-type'='hadoop',
  'warehouse'='hdfs://nn:8020/warehouse/path',
  'property-version'='1'
);

USE CATALOG hadoop_catalog;
```

B. 创建Hadoop Catalog

- **warehouse**: 表示iceberg的metadata和data数据存放的文件路径。

Apache Flink

第二步:创建Iceberg表

```
CREATE TABLE hive_catalog.default.sample (
    id BIGINT COMMENT 'unique id',
    data STRING
) PARTITIONED BY (data);
```

- 支持 PARTITION BY子句。用来设置分区字段。
- 支持 COMMENT 'table document' 子句。
- 支持 WITH ('key'='value', ...) 子句。用来设置table级别的配置属性。

- 暂不支持 computed column。
- 暂不支持 Primary key。
- 暂不支持定义 watermark。

Apache Flink

第三步：通过Flink SQL写入数据到Iceberg表

借助流式作业（默认）写入数据到Apache Iceberg表

```
INSERT INTO hive_catalog.default.sample VALUES (1, 'a');

INSERT INTO hive_catalog.default.sample SELECT id, data from other_kafka_table;
```

通过设置 execution.type 来切换提交流作业和批作业

```
-- Execute the flink job in streaming mode for current session context
SET execution.type = streaming

-- Execute the flink job in batch mode for current session context
SET execution.type = batch
```

通过Flink SQL 来覆盖Apache Iceberg表中的数据。

```
-- 根据用户提供的数据，自动覆盖表中对应分区的数据
INSERT OVERWRITE sample VALUES (1, 'a');

-- 覆盖 data='a' 这个partition的数据
INSERT OVERWRITE hive_catalog.default.sample PARTITION(data='a') SELECT 6;
```

Apache Flink

第四步:将Row通过DataStream来写入Iceberg表

```
123   @Test
124   public void testWriteRowData() throws Exception {
125     List<Row> rows = Lists.newArrayList(
126         Row.of(1, "hello"),
127         Row.of(2, "world"),
128         Row.of(3, "foo")
129     );
130     DataStream<RowData> dataStream = env.addSource(new FiniteTestSource<>(rows), ROW_TYPE_INFO)
131         .map(CONVERTER::toInternal, RowDataTypeInfo.of(SimpleDataUtil.ROW_TYPE));
132
133     FlinkSink.forRowData(dataStream)
134         .table(table)
135         .tableLoader(tableLoader)
136         .hadoopConf(CONF)
137         .build();
138
139     // Execute the program.
140     env.execute("Test Iceberg DataStream");
141
142     // Assert the iceberg table's records. NOTICE: the FiniteTestSource will checkpoint the same rows twice, so it will
143     // commit the same row list into iceberg twice.
144     List<RowData> expectedRows = Lists.newArrayList(Iterables.concat(convertToRowData(rows), convertToRowData(rows)));
145     SimpleDataUtil.assertTableRows(tablePath, expectedRows);
146   }
```

Apache Flink

第五步:修改表属性

```
1 ALTER TABLE hive_catalog.default.sample SET ('write.format.default'='avro')
2
3 ALTER TABLE hive_catalog.default.sample RENAME TO hive_catalog.default.new_sample;
4
```

- Flink SQL目前仅支持修改iceberg表的相关属性。
- Flink SQL暂不支持添加列、修改列、删除列操作。但可以通过Iceberg Java API来完成。

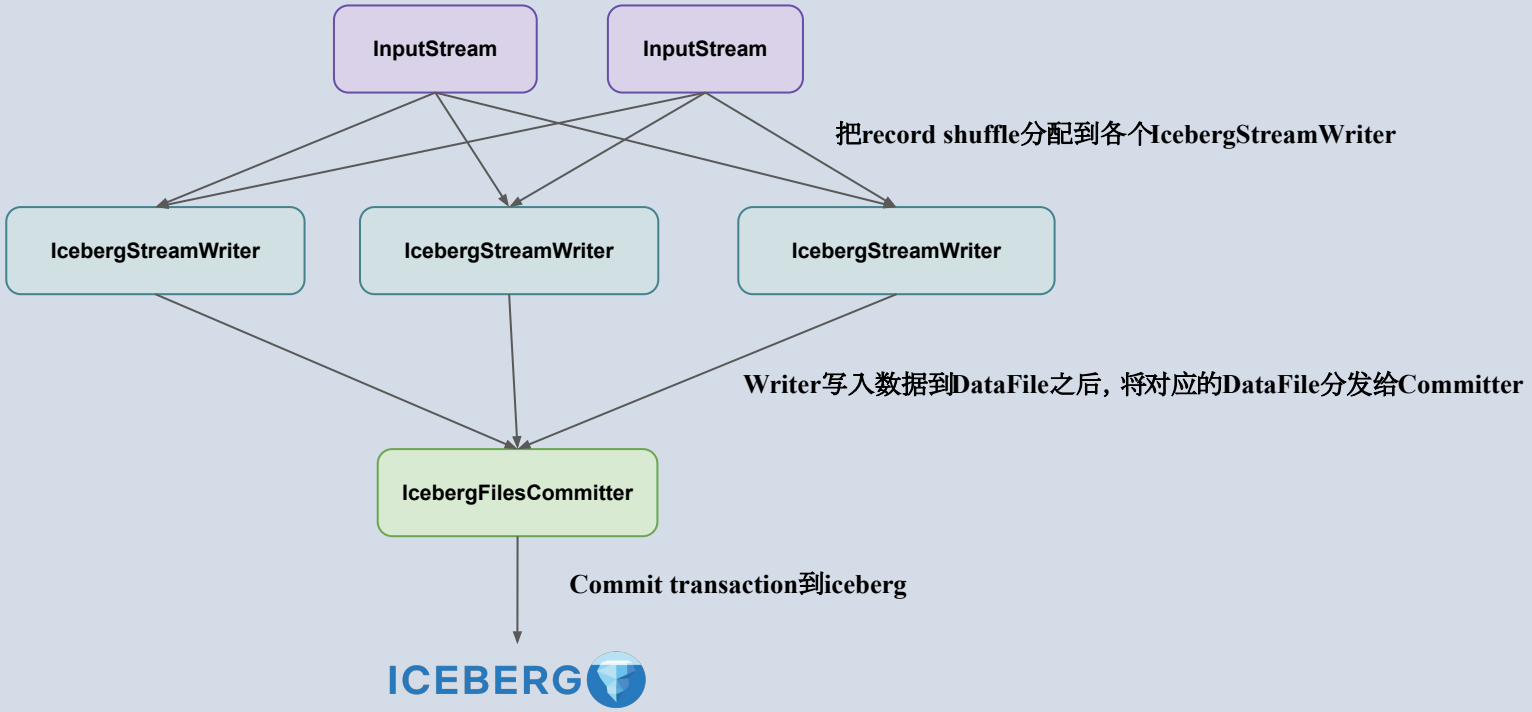
Apache Flink

第六步:删除表、库和Catalog

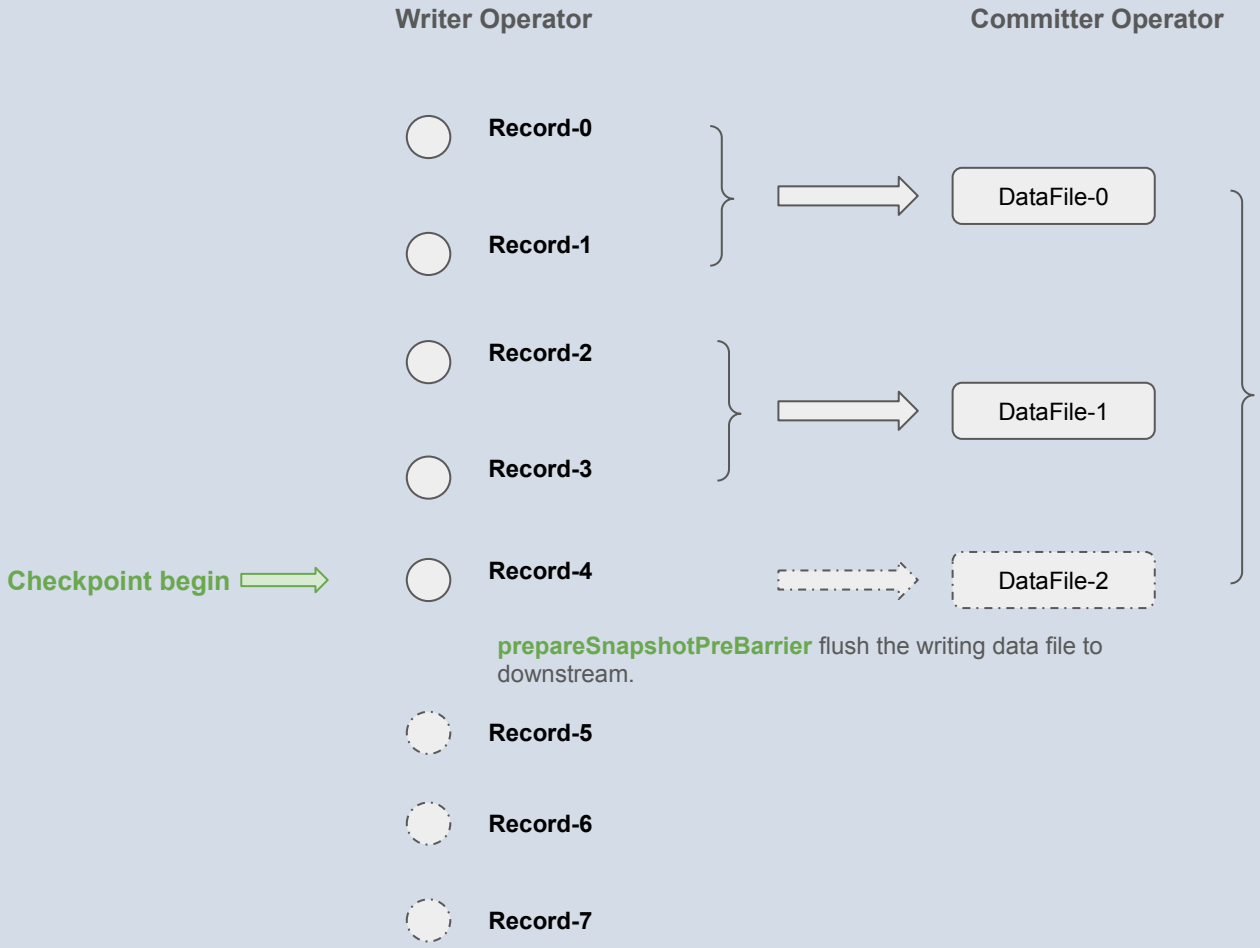
```
1 DROP TABLE hive_catalog.default.sample;
2
3 DROP DATABASE test_database;
4
5 DROP CATALOG hive_catalog;
6
```

- 支持正常的删表、删库、删Catalog操作。

Flink Sink原理



Flink Sink State设计



State: Map<Long, List<DataFile>>: 表示每个未成功提交的checkpoint对应的DataFile文件列表。

snapshotState: writer将当前所有的DataFiles发给Committer。Committer将所有的DataFiles写入到Flink Statebackend。

notifyCheckpointComplete: Committer将当前所有的DataFiles做成一个Transaction，提交到Iceberg表。

Apache Flink

Flink Sink State 改进

多个不同的Flink Job写入同一个Iceberg表时，如何保证写入数据正确性？

在云端环境下，metastore所在数据中心和日志数据中心是两个数据中心。这时候，可能导致Commit Transaction到Iceberg经常失败的现象，长期失败会导致state膨胀。如何解决这个问题？

Flink Sink: 小文件处理

- 小数据量的合并
  - 在IcebergFilesCommitter之后添加一个Compactor算子，用来实现少量小文件的频繁合并。

- 大数据量的合并
  - 设计Flink Batch作业，对接Iceberg的RewriteDataFilesAction来实现表内的大数据合并。

Apache Flink

PART 05

社区未来规划

Apache Flink

社区未来规划

1. 预计将在下个Apache Iceberg Release中支持：
    a. Flink Sink流式入湖和批量入湖。
    b. Flink Streaming Reader
    c. Flink Batch Reader
2. Flink+Iceberg对小文件的处理
    a. Committer任务之后添加Compactor算子，专门处理少量数据的compaction。
    b. 设计Flink批任务来处理大数据量的compaction。
3. 对接iceberg的row-level delete功能
    a. 通过Flink实现CDC日志的实时写入和分析。
    b. 通过Flink实现CDC日志的增量拉取。
    c. Flink+Iceberg支持批量的数据更新。
4. 更加完善的Flink SQL支持。
    a. 更富的Flink DDL支持，例如支持增删改column。
    b. 往Flink社区讨论支持hidden partition
5. 通过SQL extension来完成日常数据管理。
    a. 在iceberg内实现compaction语法和命令
    b. SQL查看history、snapshot、manifests、files文件。

THANKS

Apache Flink China Meetup · SHENZHEN

关注「Flink 中文社区」微信公众号

入门教程　　Meetup

应用案例　　源码解析

THANKS

Apache Flink China Meetup · SHENZHEN

扫码进入「数据湖技术交流」钉钉群
随时咨询讨论数据湖相关的技术问题