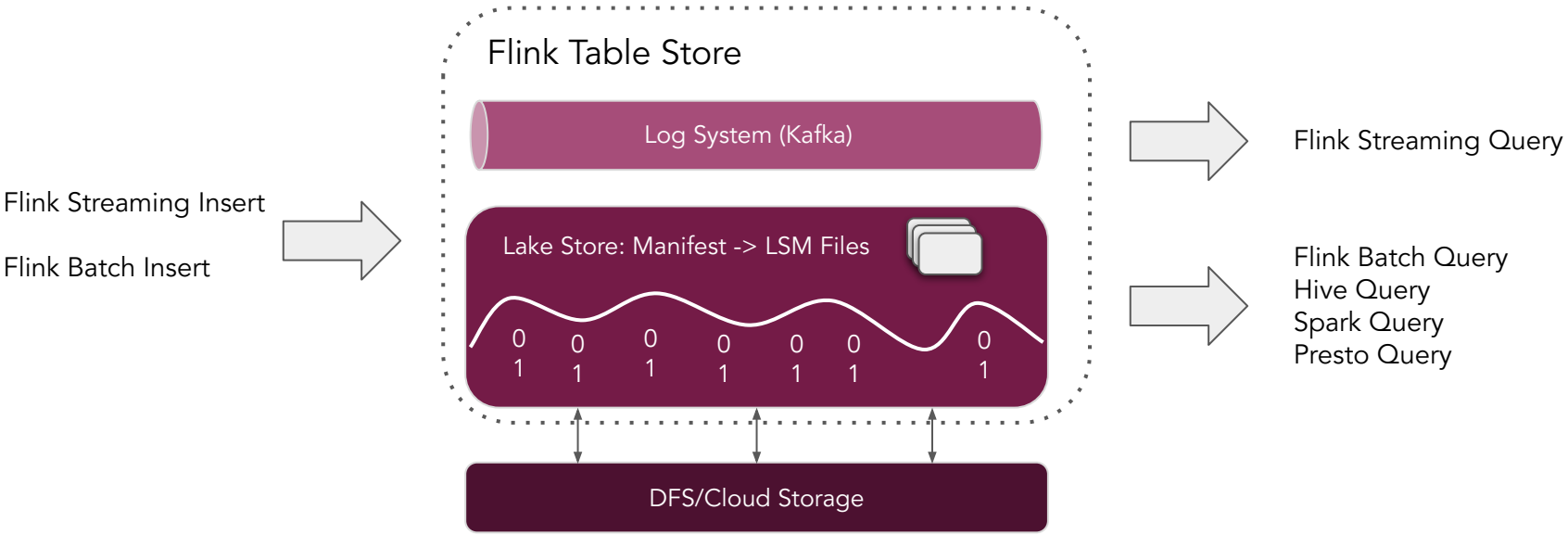
Flink Table Store v0.2 应用场景和核心功能

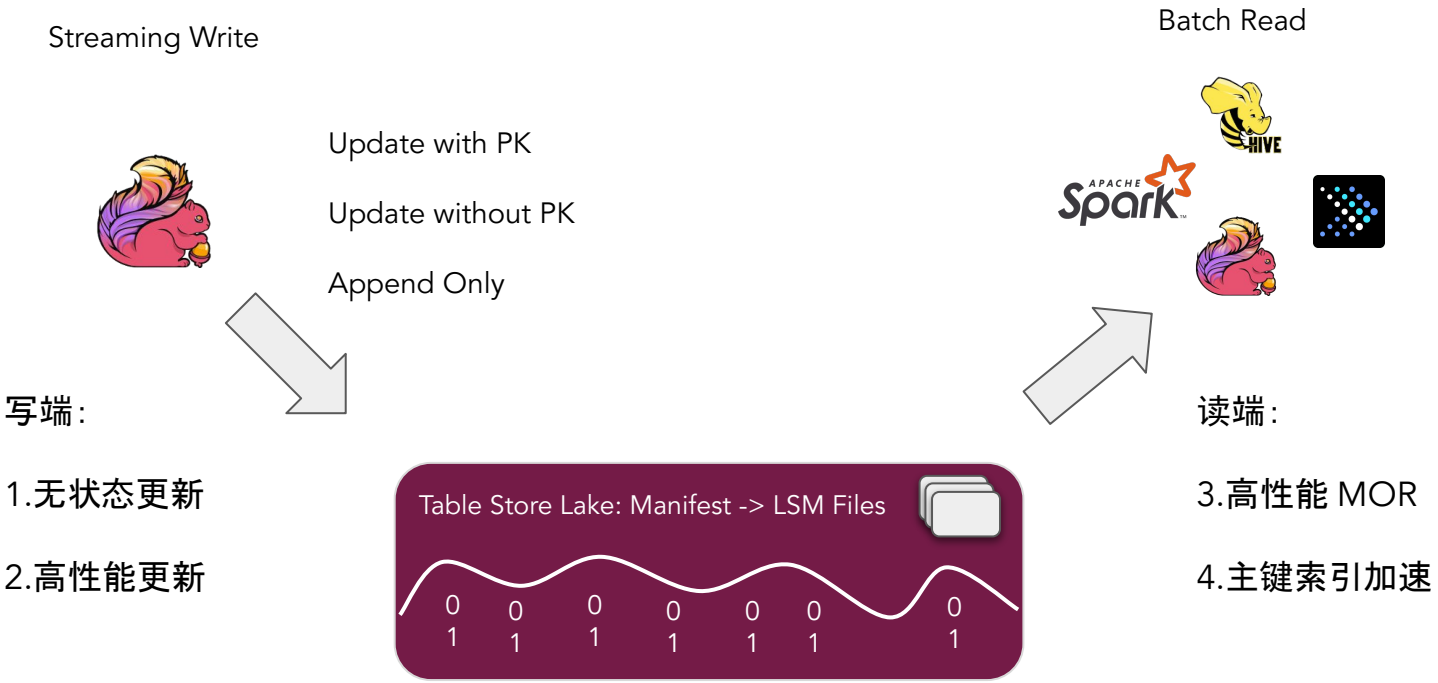
李劲松 阿里巴巴

目录

架构


Flink Table Store
Log System (Kafka)
Lake Store: Manifest -> LSM Files
0 1
DFS/Cloud Storage
Flink Streaming Insert
Flink Batch Insert
Flink Streaming Query
Flink Batch Query
Hive Query
Spark Query
Presto Query

场景一：离线数仓加速


Streaming Write
Update with PK
Update without PK
Append Only
Batch Read
HIVE
APACHE Spark
写端：
1.无状态更新
2.高性能更新
Table Store Lake: Manifest -> LSM Files
0 1
0 1
0 1
0 1
0 1
0 1
0 1
读端：
3.高性能 MOR
4.主键索引加速

场景二：Partial Update* (COALESCE)

Streaming Write

```
CREATE TABLE MyTable (
  pk BIGINT PRIMARY KEY NOT ENFORCED,
  column_1 DOUBLE,
  column_2 BIGINT
) WITH (
  'merge-engine' = 'partial-update'
);

INSERT INTO MyTable
SELECT pk, column_1, NULL FROM Src1
UNION ALL
SELECT pk, NULL, column_2 FROM Src2
```

Batch Read

写端：

1.无状态更新

2.高性能更新

基于主键打宽表

读端：

3.高性能 MOR

4.主键索引加速

场景三:预聚合 Rollup

Streaming Write

```
CREATE TABLE MyTable (
  pk BIGINT PRIMARY KEY NOT ENFORCED,
  column_1 DOUBLE,
  column_2 BIGINT
) WITH (
  'merge-engine' = 'aggregation',
  'column_1.aggregate' = 'sum',
  'column_2.aggregate' = 'max'
);
```

Batch Read

写端:

1.无状态更新

2.高性能更新

读端:

3.高性能 MOR

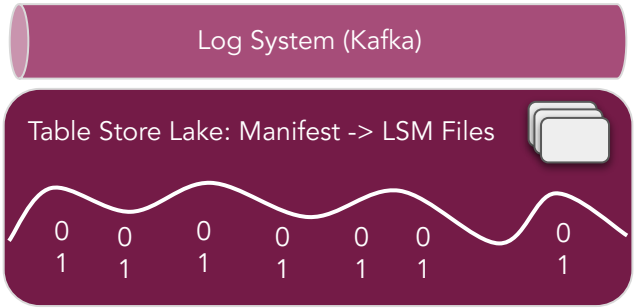
4.主键索引加速

场景四：实时数仓增强

Streaming Write

双写
记录 Offset

```
CREATE TABLE MyTable (
  column_1 DOUBLE,
  column_2 BIGINT,
  dt STRING
) PARTITIONED BY (dt)
WITH (
  'write-mode' = 'append-only',
  'log.system' = 'kafka',
  'log.topic' = 'my_topic',
  'log.kafka.bootstrap.servers' = '...'
);
```



Streaming Read

Hybrid：Backfill

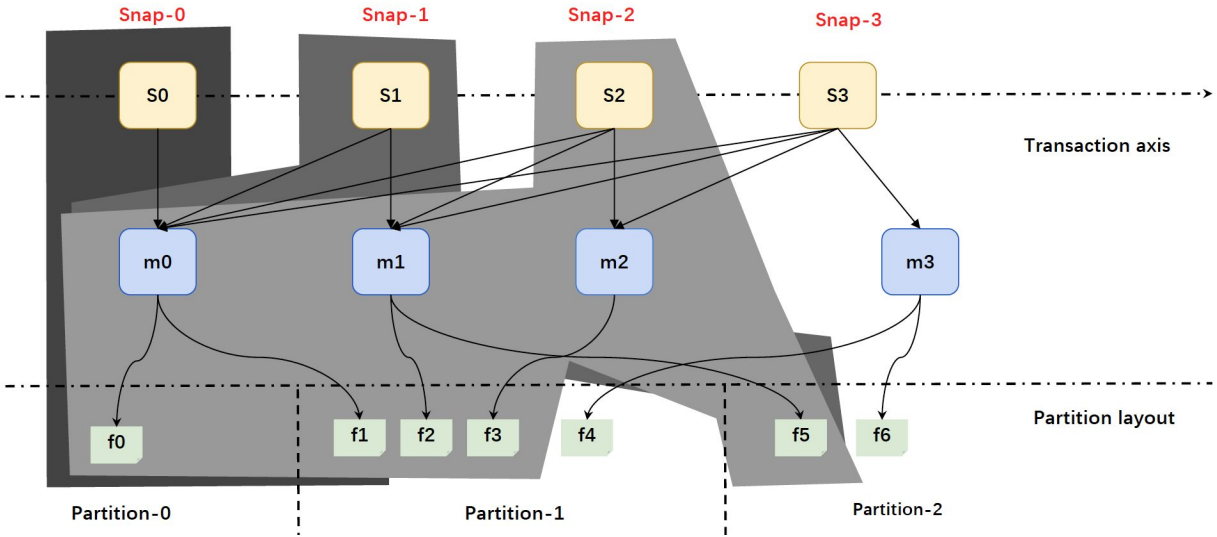
Query

中间表可查

AppendOnly：保证输入序

目录

Flink Table Store v0.1：湖存储结构



- Snapshot 级别的事务语义
- 对象存储上的大规模数据存储的支持

Flink Table Store v0.1：分区内部

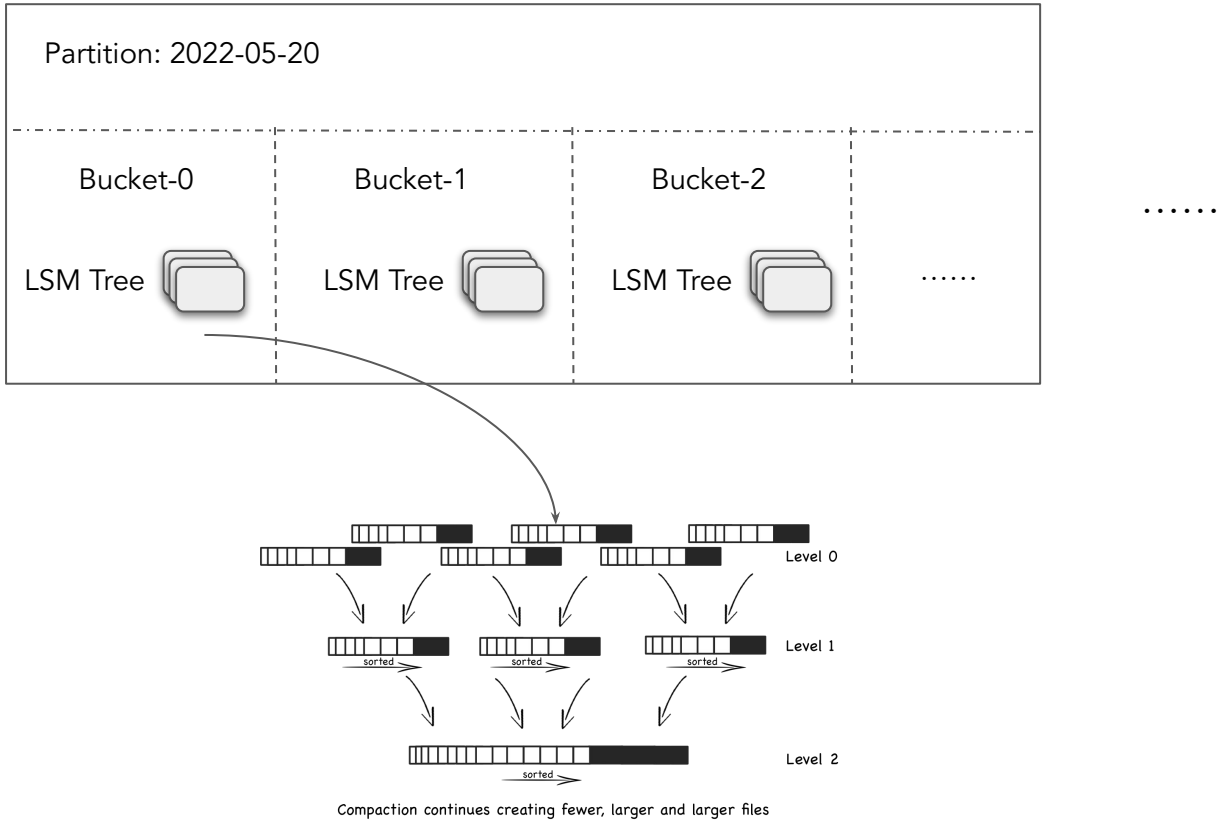


Table Store Catalog

Flink SQL:

```
CREATE CATALOG MyCatalog WITH (
  'type' = 'table-store',
  'root-path' = '...',
  'metastore.type' = 'hive',
  'metastore.uri' = '...'
);

USE CATALOG MyCatalog;

CREATE TABLE MyTable (
  pk BIGINT PRIMARY KEY NOT ENFORCED,
  column_1 DOUBLE,
  column_2 BIGINT
) WITH (
  'log.system' = 'kafka',
  'log.topic' = 'my_topic',
  'log.kafka.bootstrap.servers'= '...'
);
```

- 默认 Meta 保存在 FileSystem 上
- Metastore 配置为 Hive，Hive 引擎可直接读

Log 可选，需提供 Topic

生态

Hive SQL:

```
-- 创建外表
-- 已使用Hive Metastore的Catalog无需创建

CREATE EXTERNAL TABLE MyTable
STORED BY '...TableStoreStorageHandler'
LOCATION '.../table-path/';

SELECT * FROM MyTable;
```

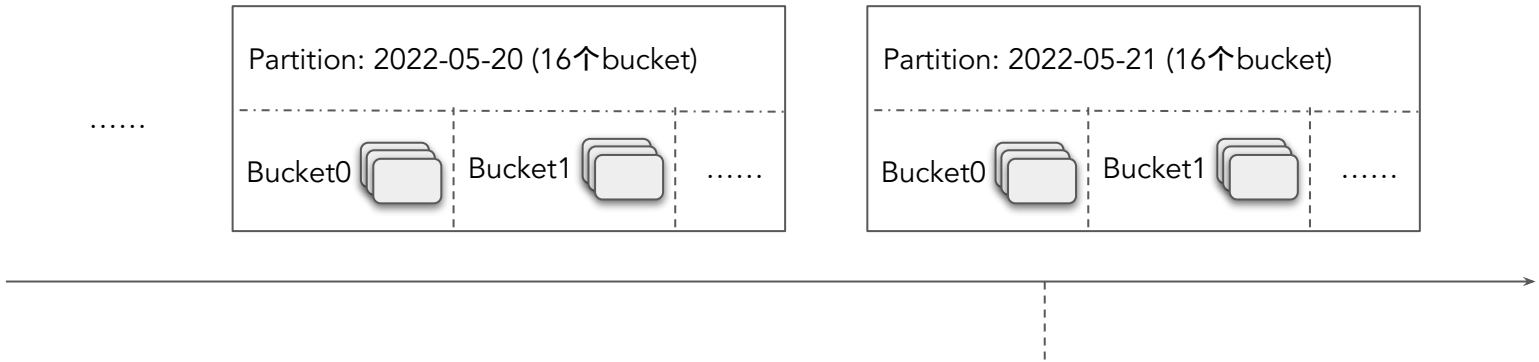
Spark SQL:

```
-- 创建映射表

CREATE TEMPORARY VIEW MyTable
USING tablestore
OPTIONS (
  path "..."
);

SELECT * FROM MyTable;
```

Change Bucket



发现当前 Bucket 太少，需要 Rescale：

1. 修改表的 Bucket 默认配置：ALTER TABLE … SET ('bucket' = '32');
2. 新分区使用新的 Bucket 个数：32，老分区保持不动
3. 暂停流写作业，使用 Batch 作业 Rescale 当前分区，恢复流写作业

Append Only 模式

- 低成本，没有合并：当做传统离线表来使用
- Kafka Tiered Storage：
  - 流读输入序，提供 Kafka 流读相同体验
  - 数据可查询
- 自动 Compaction，避免小文件

目录

Flink Table Store：满足 Flink SQL 对存储的需求

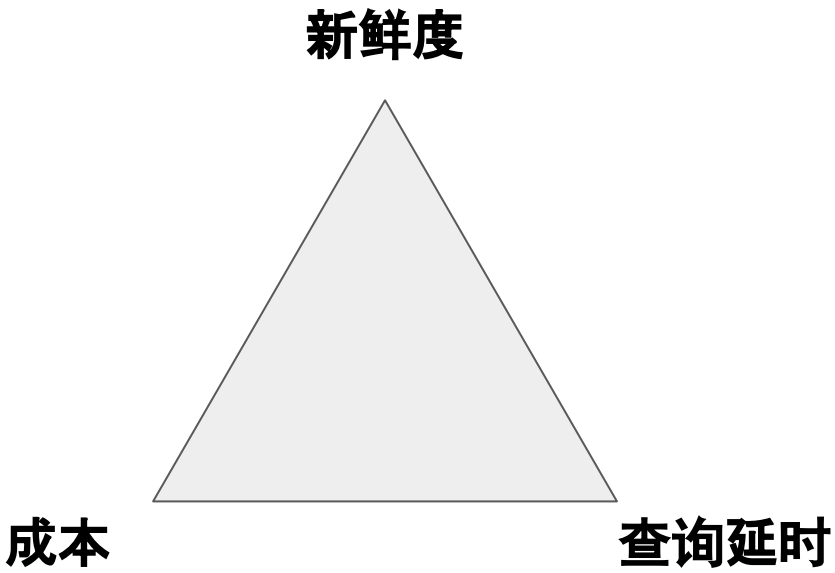
OLAP

Batch ETL

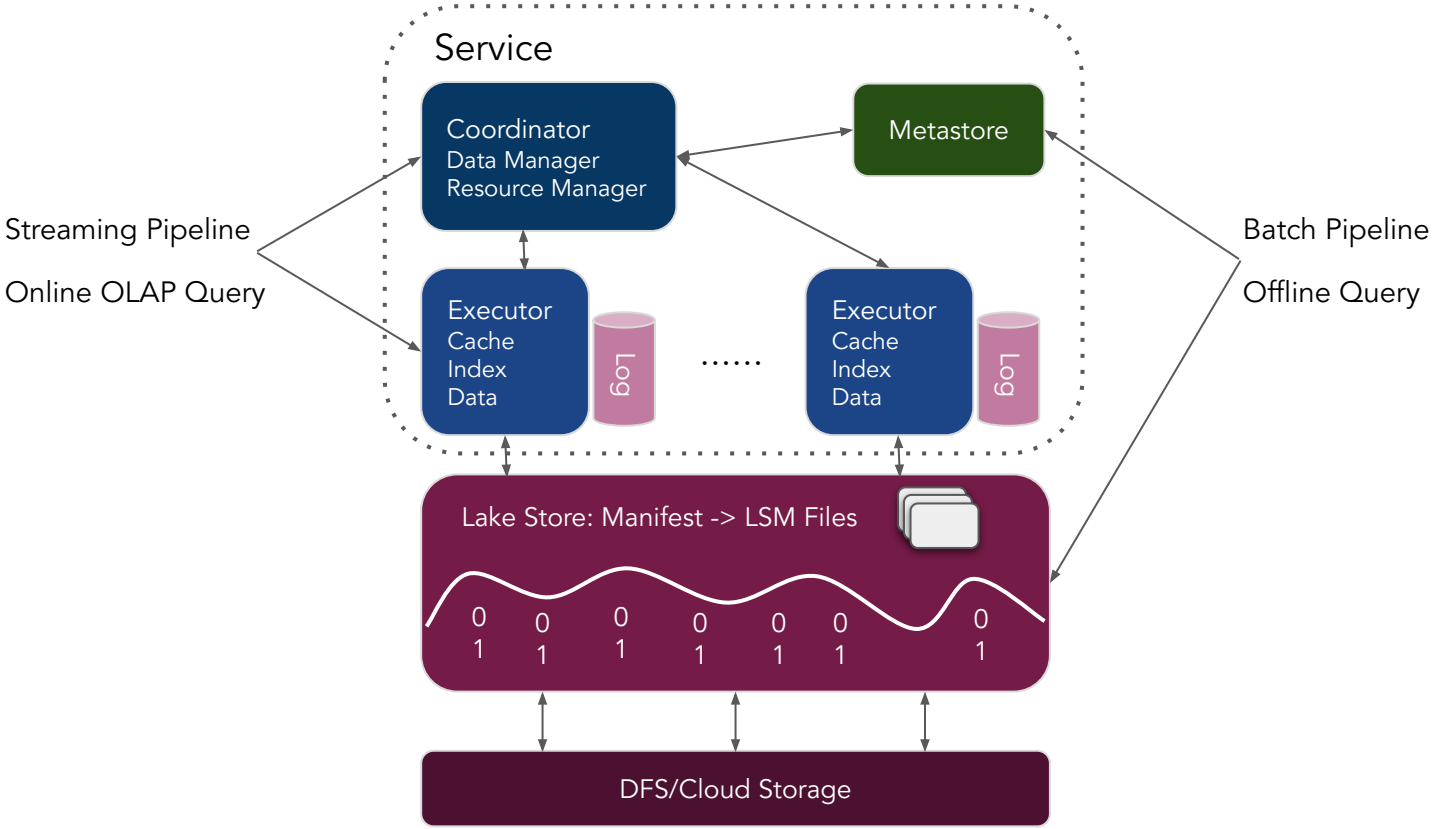
Stream ETL(Queue)

Dim Lookup

Flink Table Store：满足不同 Tradeoff 的选择



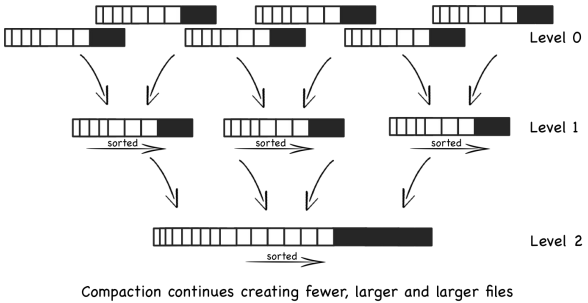
Flink Table Store 架构



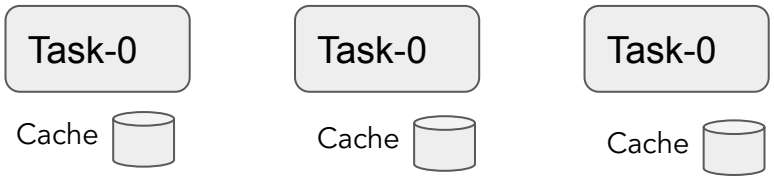
Flink Table Store：Dim Join，计算存储分离

DFS Bucketed LSM

Flink Streaming Insert



Flink Streaming Dim Join



类似于 HBase 计算存储分离

目录

Project

- Subproject of Apache Flink
- Github: https://github.com/apache/flink-table-store
- User Docs: https://nightlies.apache.org/flink/flink-table-store-docs-master/
- Mail list:
  - dev@flink.apache.org
  - user@flink.apache.org
  - user-zh@flink.apache.org
- Ding group

V0.2 **将在7月份发布！**
欢迎试用！

谢谢