镜舟

Powered by StarRocks

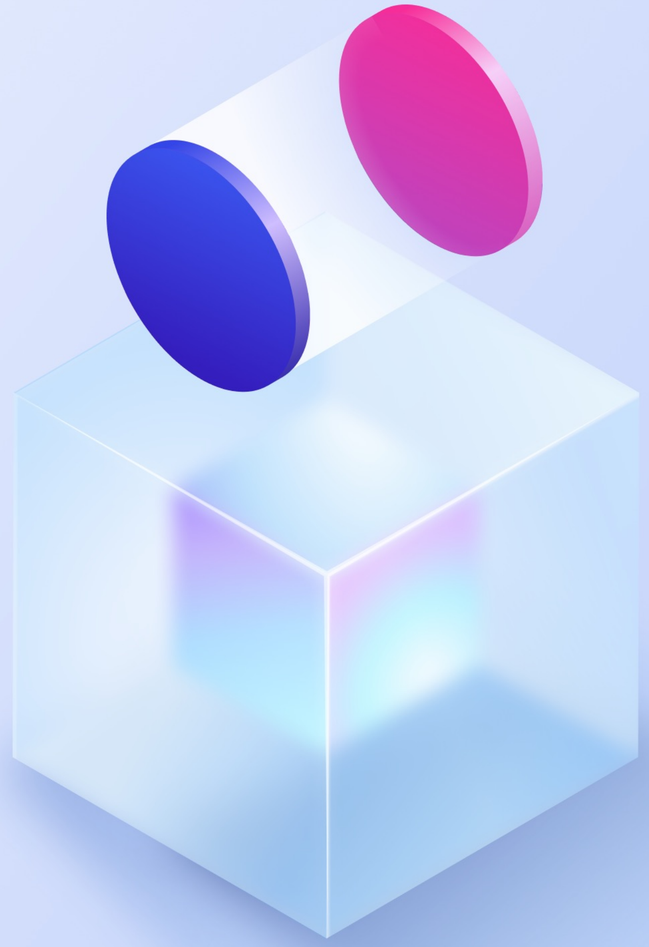
StarRocks实战系列

导入优化&问题排查

2023-02-20

演讲人：Eddie

镜舟科技 StarRocks DBA



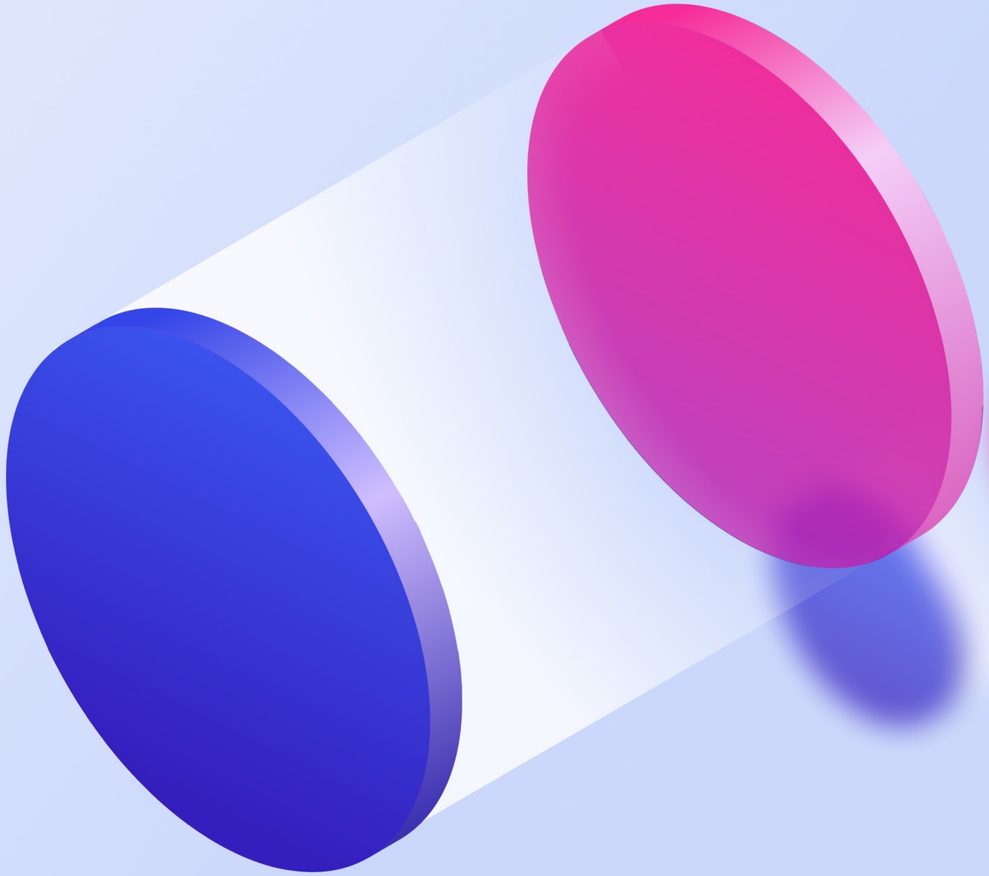
本次分享我将从三个模块拆分开来 进行讲解，分别为 导入整体介绍、导入的数据生态和重头戏 导入优化及问题排查。(翻页)

演讲大纲

01 **导入整体介绍**
Introduction to the overall presentation

02 导入的数据生态
Imported data ecology

03 导入优化及问题排查
Import optimization and problem solving



01 导入整体介绍

Introduction to the overall presentation

StarRocks导入方式概览

镜舟 Powered by StarRocks

导入方式	协议	业务场景	数据量（单任务）	数据源类型	数据格式	同步方式
Stream Load	HTTP	通过HTTP协议文本文件导入/通过程序导入的数据流	10GB以内	本地文件/流式数据	CSV、JSON	同步
Insert Into select	MYSQL	外表导入/StarRocks 数据表之间的数据导入	跟内存相关	StarRocks 表/外部表）	StarRocks 表	同步
Insert Into values	MYSQL	单条批量小数据量插入/通过 JDBC 等接口导入	简单测试用	程序/ETL 工具	SQL	同步

StarRocks导入方式概览

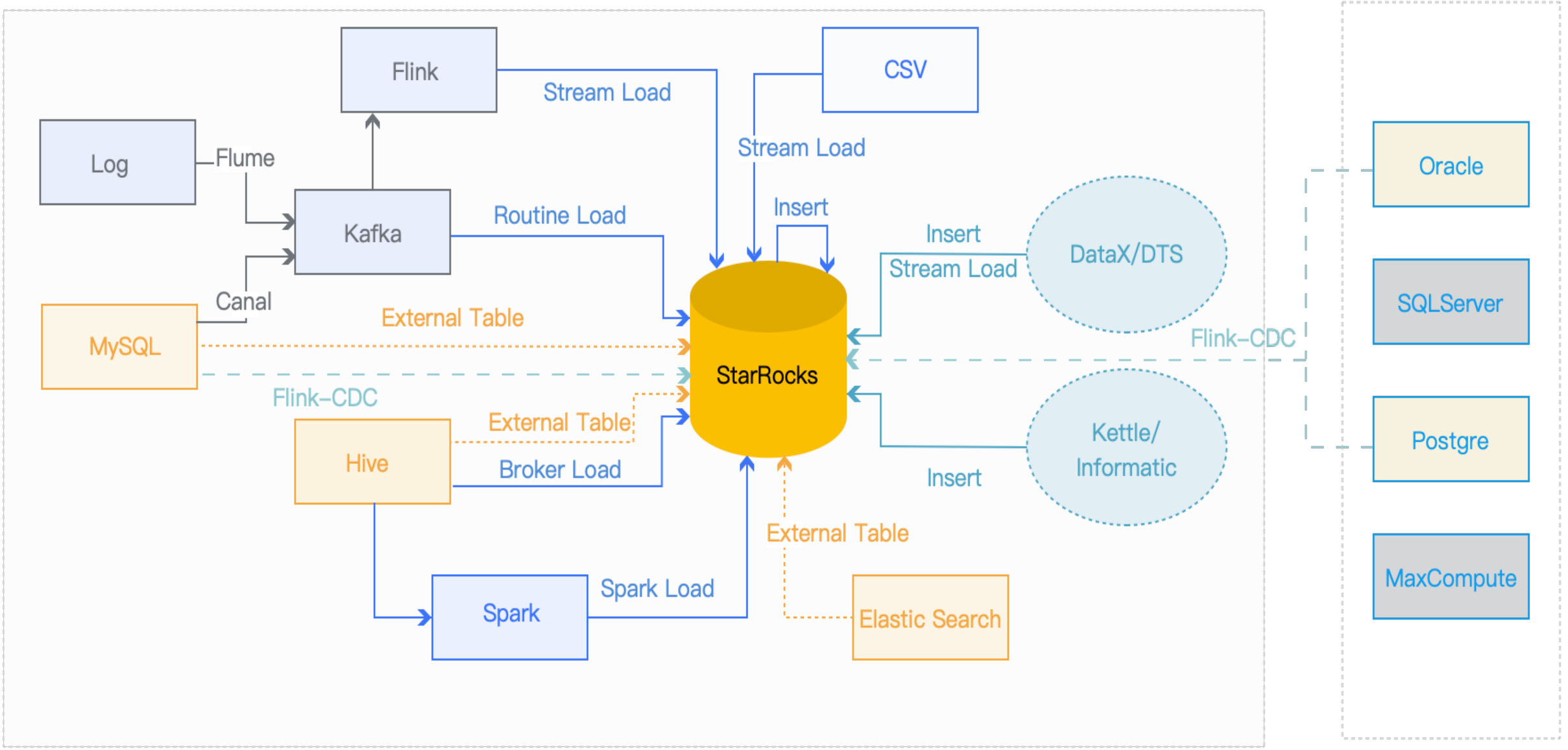
镜舟 Powered by StarRocks

导入方式	协议	使用场景	数据量（单任务）	数据源类型	格式	同步方式
Broker Load	MYSQL	从 HDFS 或外部云存储系统导入数据	数十GB到数百GB	HDFS、Amazon S3、Google GCS、阿里云 OSS、腾讯云 COS	CSV、Parquet、ORC	异步
Routine Load	MYSQL	从 Apache Kafka® 实时地导入数据流。	微批导入MB - GB	Kafka	CSV、JSON	异步
Spark Load	MYSQL	通过 Apache Spark™ 集群初次从 HDFS 或 Hive 迁移导入大量数据。需要做全局数据字典来精确去重	数十GB到TB级别	HDFS, Hive	CSV、 2.0 版本之后支持ORC、 Parquet	异步

02 导入的数据生态

Imported data ecology

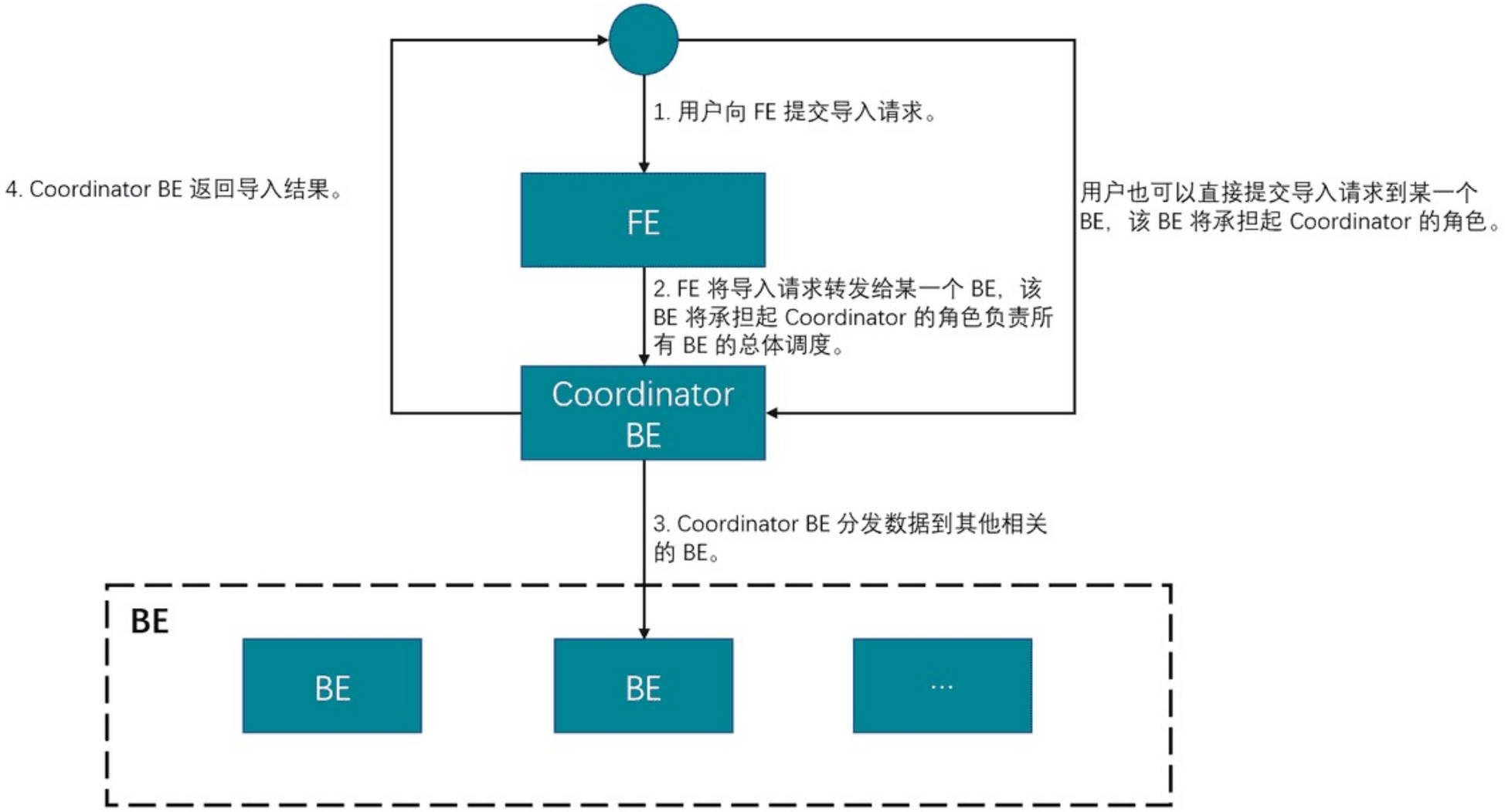
StarRocks支持的数据生态



03 导入优化及问题排查

Import optimization and problem solving

数据生态：Stream Load



数据生态：Stream Load

镜舟 Powered by StarRocks

StarRocksFE配置

stream_load_default_timeout_second = 10800

StarRocksBE配置

streaming_load_max_mb=102400

streaming_load_max_batch_size_mb=102400

flush_thread_num_per_store=8 // 每个盘的flush线程数，当用户盘比较少时可以设置较大，盘较多时设置较小，一般情况下 flush_thread_num_per_store * store_num < be_cpu_core_num / 2

olap_table_sink_send_interval_ms=1

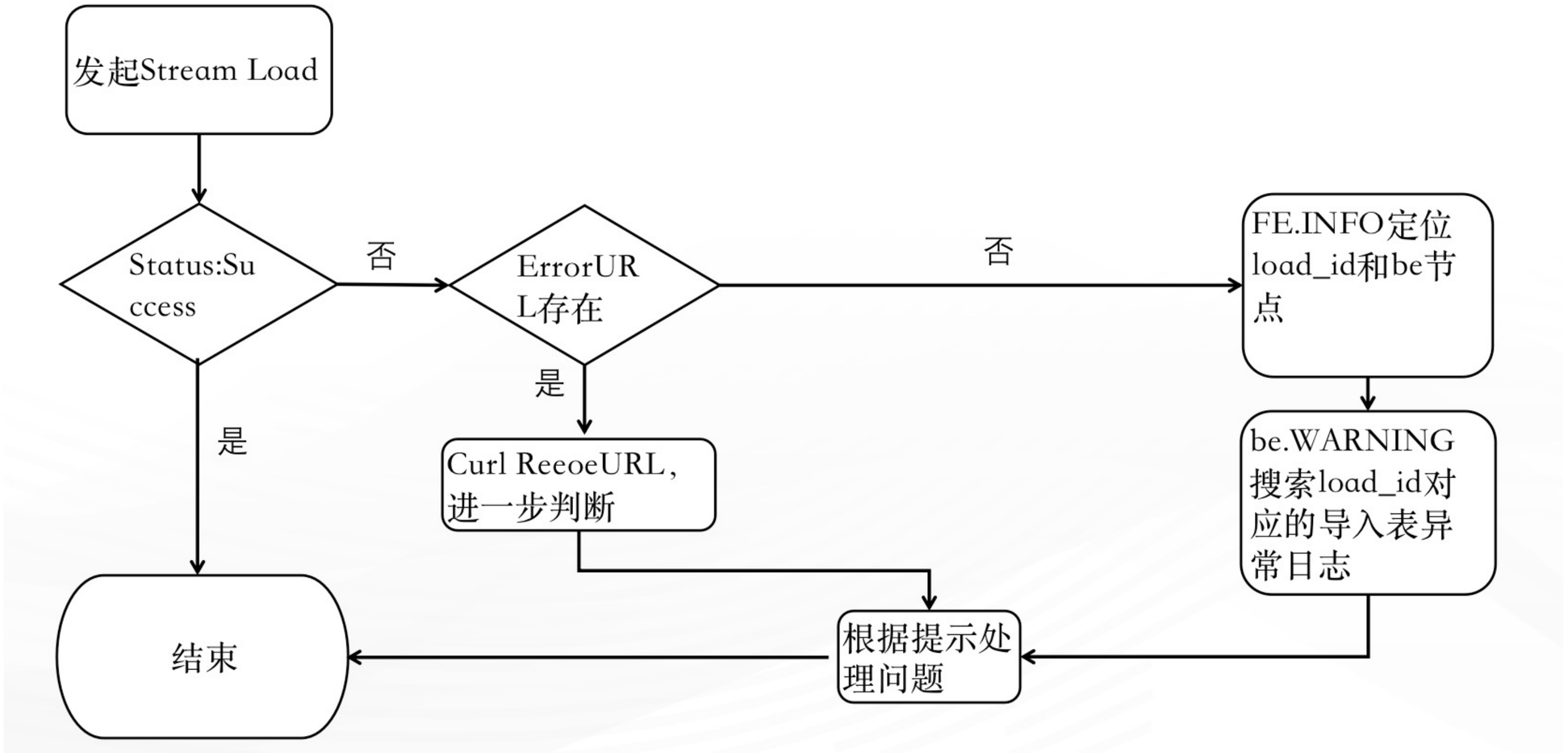
load_process_max_memory_limit_percent=50

tablet_max_versions = 20000

enable_new_load_on_memory_limit_exceeded=true

StreamLoad大数据量导入优化参数推荐配置

数据生态：Stream Load



Stream load内部调用链路

数据生态：Stream Load


```
{
    "Status":"Fail",
    "BeginTxnTimeMs":1,
    "Message":"too many filtered rows",
    "NumberUnselectedRows":0,
    "CommitAndPublishTimeMs":0,
    "Label":"4682d766-0e53-4fce-b111-56a8d8bef390",
    "LoadBytes":69238389,
    "StreamLoadPutTimeMs":4,
    "NumberTotalRows":7077604,
    "WriteDataTimeMs":4350,
    "TxnId":33,
    "LoadTimeMs":4356,
    "ErrorURL":""http://192.168.10.142:8040/api/_load_error_log?file=__shard_2/error_log_...",
    "ReadDataTimeMs":1961,
    "NumberLoadedRows":0,
    "NumberFilteredRows":7077604
}
```


发起导入后的返回值信息

数据生态：Stream Load

如果事务导入成功，则返回如下结果：

```
{"Status": "OK","Message": "","Label": "xxx","TxnId": 9032,"BeginTxnTimeMs": 0}
```

如果事务导入失败，则会返回如下等包含ErrorURL的结果：

```
{"Status": "FAILED","Message":... ","ErrorURL":"http://192.168.10.142:8040/api/_load..."}
```

这个时候我们则可以通过：
Curl ErrorURL命令来查看报错，例如

```
curl "  "ErrorURL":""http://192.168.10.142:8040/api/_load_error_log?file=__shard_2/error_log_...""
```

如果事务的标签重复，则返回如下结果：

```
{"Status": "LABEL_ALREADY_EXISTS","ExistingJobStatus": "RUNNING","Message": "Label [xxx] has already been used."}
```

如果发生标签重复以外的其他错误，则返回如下结果：

```
{"Status": "FAILED","Message": ""}
```

返回结果

数据生态：Stream Load


```
{
    "TxnId":2271727,
    "Label":"4682d766-0e53-4fce-b111-56a8d8bef2340",
    "Status":"Fail",
    "Message":"Failed to commit txn 2271727. Tablet [159816] success replica num 1 is less then quorum replica num 2 while error backends 10012",
    "NumberTotalRows":1,
    "NumberLoadedRows":1,
    "NumberFilteredRows":0,
    "NumberUnselectedRows":0,
    "LoadBytes":575,
    "LoadTimeMs":26,
    "BeginTxnTimeMs":0,
    "StreamLoadPutTimeMs":0,
    "ReadDataTimeMs":0,
    "WriteDataTimeMs":21,
    "CommitAndPublishTimeMs":0
}
```


不存在ErrorURL的返回

数据生态：Stream Load 排查思路

镜舟 Powered by StarRocks

1. 查看本次导入的load_id和调度到的BE节点IP
grep -w $TxnId fe.log|grep "load id"

2. 输出样例为：2023-1-30 20:48:50,169 INFO (thrift-server-pool-4|138) [FrontendServiceImpl.streamLoadPut():809] receive stream load put request. db:ssb, tbl: demo_test_1, txn id: 1580717, load id: 7a4d4384-1ad7-b798-f176-4ae9d7ea6b9d, backend: 172.26.92.155

3. 我们可以看到对应的BE节点IP，去到该节点上查看具体原因： grep $load_id be.INFO|less

4. 输出样例为：I0518 11:58:16.771597 4228 stream_load.cpp:202] new income streaming load request.id=f1481, job_id=-1, txn_id=-1, label=metrics_detail_16, db=starrocks, tbl=metrics_detail
I0518 11:58:16.7 4176 load_channel_mgr.cpp:186] Removing finished load channel load id=f181
I0518 11:58:16.7 4176 load_channel.cpp:40] load channel mem peak usage=1915984, info=limit: 16113540169; label: f181; all tracker size: 3; limit trackers size: 3; parent is null: false; , load_id=f181

5. 如果查不到具体原因，则可以继续查看线程上下文进行进一步跟踪定位，比如上文的 4176线程 ：
grep -w 4176 be.INFO|less 进一步分析即可。

数据生态：Insert Into

目前Insert into支持以下两种方式：

方式一：Insert into table values （）；

方式二：Insert into table1 xxx select xxx from table2

方式一（Insert into）不建议在线上使用，**频繁使用 INSERT 语句导入小批量数据会产生过多的数据版本**，从而影响查询性能，因此不建议您频繁使用 INSERT 语句导入数据或将其作为生产环境的日常例行导入作业。如果您的业务场景需要流式导入或者小批量多次导入数据，建议使用 Apache Kafka® 作为数据源并通过 Routine Load 方式进行导入作业。

方式二（Insert into select）可以通过 INSERT INTO SELECT 语句将源表中的数据导入至目标表中。源表可以是一张或多张内部表或者外部表。目标表必须是 StarRocks 的内表。执行该语句之后，系统将 SELECT 语句结果导入目标表。

注： insert into导入方式是同步的，执行完会立即返回结果。可以通过返回结果判断导入成功或失败。

数据生态：Insert Into

StarRocks SessionVariables（BE）：

1. flush_thread_num_per_store=8　// 每个盘的flush线程数，当用户盘比较少时可以设置较大，盘较多时设置较小，一般情况下 flush_thread_num_per_store * store_num < be_cpu_core_num / 2

2. olap_table_sink_send_interval_ms=1

3. load_process_max_memory_limit_percent=50

4. send_channel_buffer_limit = 67108864　// 默认值64MB，当用户导入的数据列较多、单行数据较大时，可以适当调大

5. number_tablet_writer_threads = 16　// 默认值16，[16-48]，一般设置为cpu核数的1/3左右。

6. 该配置同下述的Broker Load配置。

Insert Into大数据量导入优化参数推荐配置

数据生态：Insert Into

INSERT 导入作业会根据执行结果的不同，返回以下两种作业状态：

执行成功

如果导入执行成功，StarRocks 的返回如下：

Query OK, 2 rows affected, 2 warnings (0.05 sec)
{'label':'insert_load_wikipedia', 'status':'VISIBLE', 'txnId':'1006'}

执行失败

如果所有数据都无法被导入，则导入执行失败，StarRocks 将返回相应错误以及 tracking_url。您可以通过 tracking_url 查看错误相关的日志信息并排查问题。

失败样例

ERROR 1064 (HY000): Insert has filtered data in strict mode,

tracking_url=http://x.x.x.x:yyyy/api/_load_error_log?file=error_log_...

查看tracking_url ，里有具体报错，帮助我们去更好的去排查导入问题。

返回结果

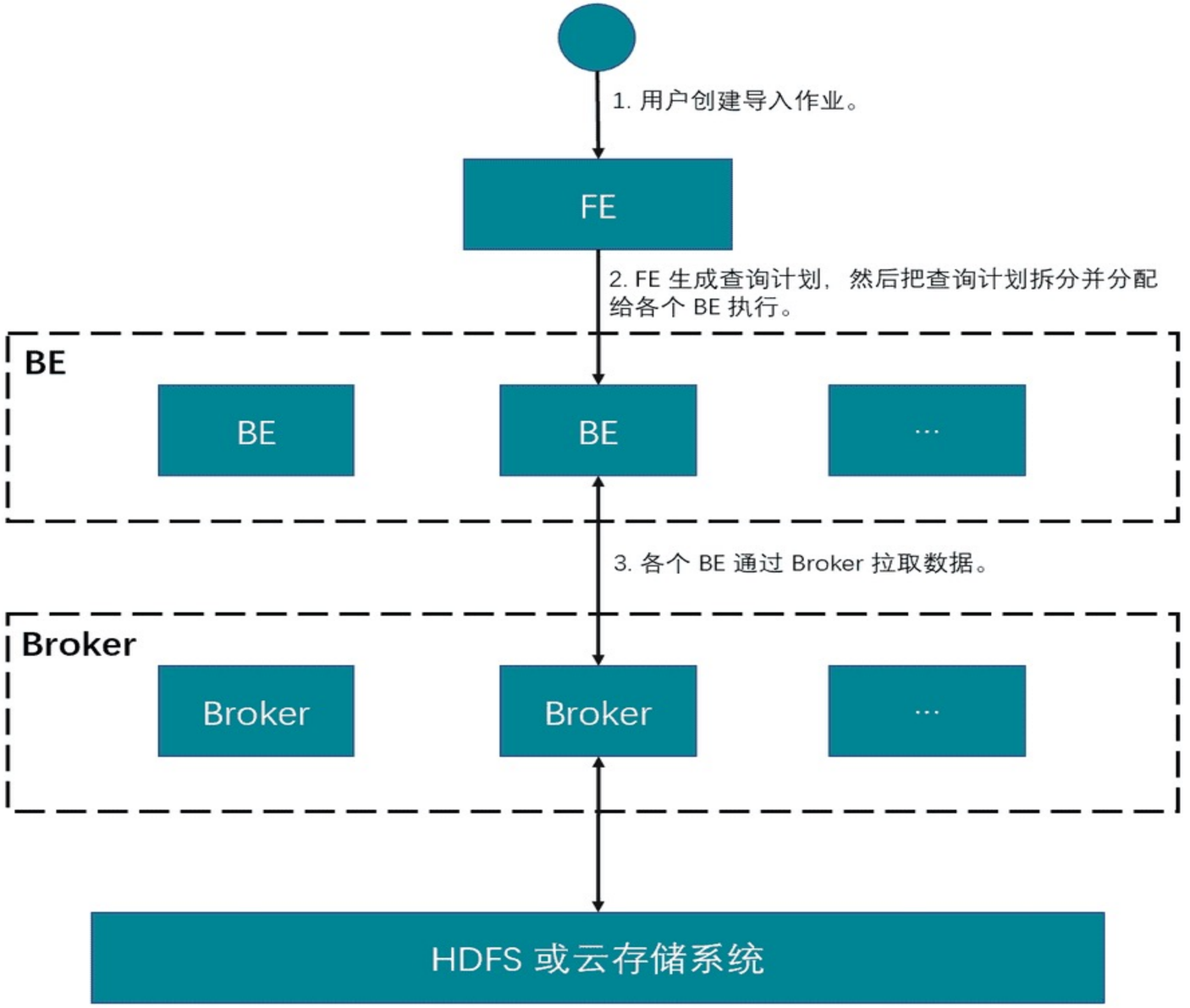
数据生态：Insert Into

通过 SHOW LOAD 语句查看您可以通过 SHOW LOAD 语句查看 INSERT 导入作业状态:

```
SHOW LOAD WHERE label="insert_load_wikipedia"\G
```

```
*************************** 1. row ***************************
         JobId: 13525
         Label: insert_load_wikipedia
         State: FINISHED
      Progress: ETL:100%; LOAD:100%
          Type: INSERT
       EtlInfo: NULL
      TaskInfo: cluster:N/A; timeout(s):3600; max_filter_ratio:0.0
      ErrorMsg: NULL
    CreateTime: 2022-08-02 11:41:26
  EtlStartTime: 2022-08-02 11:41:26
 EtlFinishTime: 2022-08-02 11:41:26
 LoadStartTime: 2022-08-02 11:41:26
LoadFinishTime: 2022-08-02 11:41:26
           URL:
    JobDetails: {"Unfinished backends":{},"ScannedRows":0,"TaskNumber":0,"All backends":{},"FileNumber":0,"FileSize":0}
```

数据生态：Broker Load



Broker Load导入流程

数据生态：Broker Load

BrokerLoad大数据量导入优化参数推荐配置

StarRocks SessionVariables（FE）：

1. load_transmission_compression_type = none　//默认值是none，当并发提高，cpu仍然打不满时，可以设置为LZ4_FRAME。

v2.4及之前版本：

1. load_parallel_instance_num=8　// 可以根据BE的cpu核数设置，一般设置区间为[cpu核数/2 - 64]

v2.5+版本 & session variable enable_adaptive_sink_dop=true：

1. pipeline_dop=0　// 自动设置BE核数的一半作为并行度，可以手动设置非0值，和查询的设置方式一致。

2. 注：　新部署的2.5集群自动设置enable_adaptive_sink_dop=true。从之前版本升级上来的集群会自动设置enable_adaptive_sink_dop=false，此时依然使用load_parallel_instance_num来配置并行度。

数据生态：Broker Load

BrokerLoad大数据量导入优化参数推荐配置

镜舟 Powered by StarRocks

StarRocks SessionVariables（BE）：

1. flush_thread_num_per_store=8 // 每个盘的flush线程数，当用户盘比较少时可以设置较大，盘较多时设置较小，一般情况下 flush_thread_num_per_store * store_num < be_cpu_core_num / 2

2. olap_table_sink_send_interval_ms=1

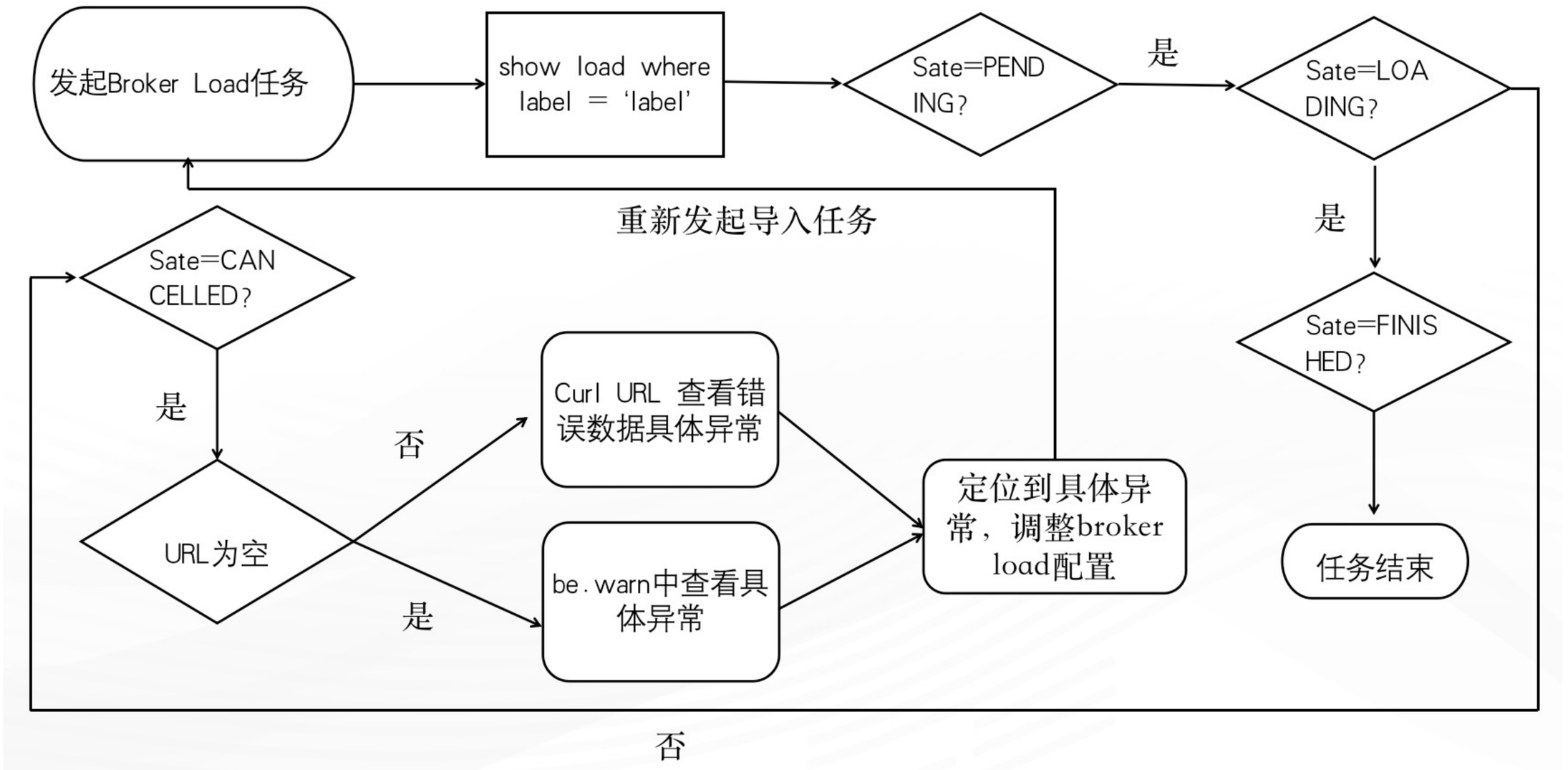
3. load_process_max_memory_limit_percent=50

4. send_channel_buffer_limit = 67108864 // 默认值64MB，当用户导入的数据列较多、单行数据较大时，可以适当调大

5. number_tablet_writer_threads = 16 // 默认值16，[16-48]，一般设置为cpu核数的1/3左右。

6. 该配置同上述Insert Into配置。

数据生态：Broker Load



数据生态：Broker Load 排查思路

1. Show load查看任务状态，状态为CANCELLED的时候进一步跟进排查。

2. 如果URL不为空，则通过curl $URL查看具体报错信息。

3. 如果URL为空，通过fe日志查看load id和be ip。

4. （检查hdfs文件路径是否指定正确，可以指定到具体文件也可以指定某目录下的所有文件。

5. （hdfs导入请检查一下是否有k8s认证,并进行配置： grep $JobnId fe.log

6. 去到对应的be中查看具体异常： grep $load_id be.INFO

7. 定位到具体错误，并调整broker load配置，尝试重新导入任务。

数据生态：Broker Load 排查思路

通过show load查看任务状态，当导入作业的状态为PENDING，LOADING或FINISHED时，failMsg该参数值为NULL。当导入作业的状态为CANCELLED时，该参数值会显示对应的错误信息，包括 type 和 msg 两部分：

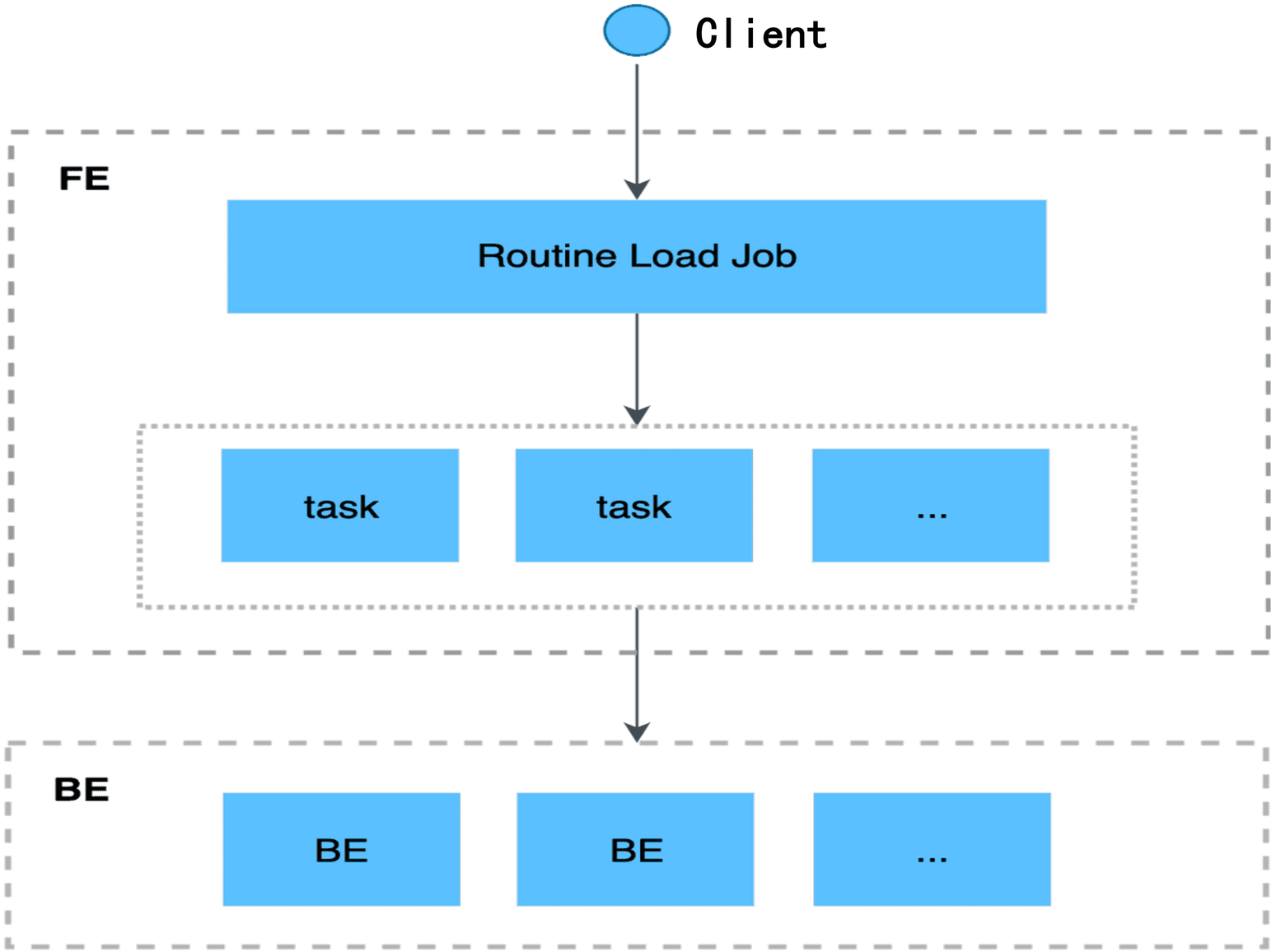
ErrorMsg中的type取值：

1. USER_CANCEL：导入作业被手动取消。
2. ETL_SUBMIT_FAIL：导入任务提交失败。
3. ETL-QUALITY-UNSATISFIED：数据质量不合格，即导入作业的错误数据率超过了 max-filter-ratio。
4. LOAD-RUN-FAIL：导入作业在 LOADING 状态失败。
5. TIMEOUT：导入作业未在允许的超时时间内完成。
6. UNKNOWN：未知的导入错误。

ErrorMsg中的msg ：

1. 显示有关失败原因的详细信息。

数据生态：Routine Load



Routine Load导入流程

数据生态：Routine Load

RoutineLoad 优化方式总览

任务调度周期

1. max_batch_interval

通过缩短任务调度周期加速数据消费。但是，更小的任务调度周期可能会带来更多的CPU资源消耗。
需要注意的是，任务调度周期最小值为5s。

任务级别调优

1. 降低导入QPS，集群总体的导入QPS尽量<10

计算方式： 集群routine_load_task_num / routine_load_task_consume_second

2. 增加单个导入事务的数据量，单个Routine Load Task导入的数据量 > 1G

需要同时调整max_routine_load_batch_size，routine_load_task_timeout_second来实现

3. 单个BE上并发导入任务routine_load_thread_pool_size尽量<be_core_num / 2

数据生态：Routine Load

RoutineLoad 优化方式总览

镜舟 Powered by StarRocks

任务并行度

1. max_routine_load_task_concurrent_num
2. desired_concurrent_number

(在partition数量和BE数量较多时，可以通过设置较大的该参数来加速任务执行。但是，更大的并行度可能会带来更多的CPU资源消耗)

任务批量大小

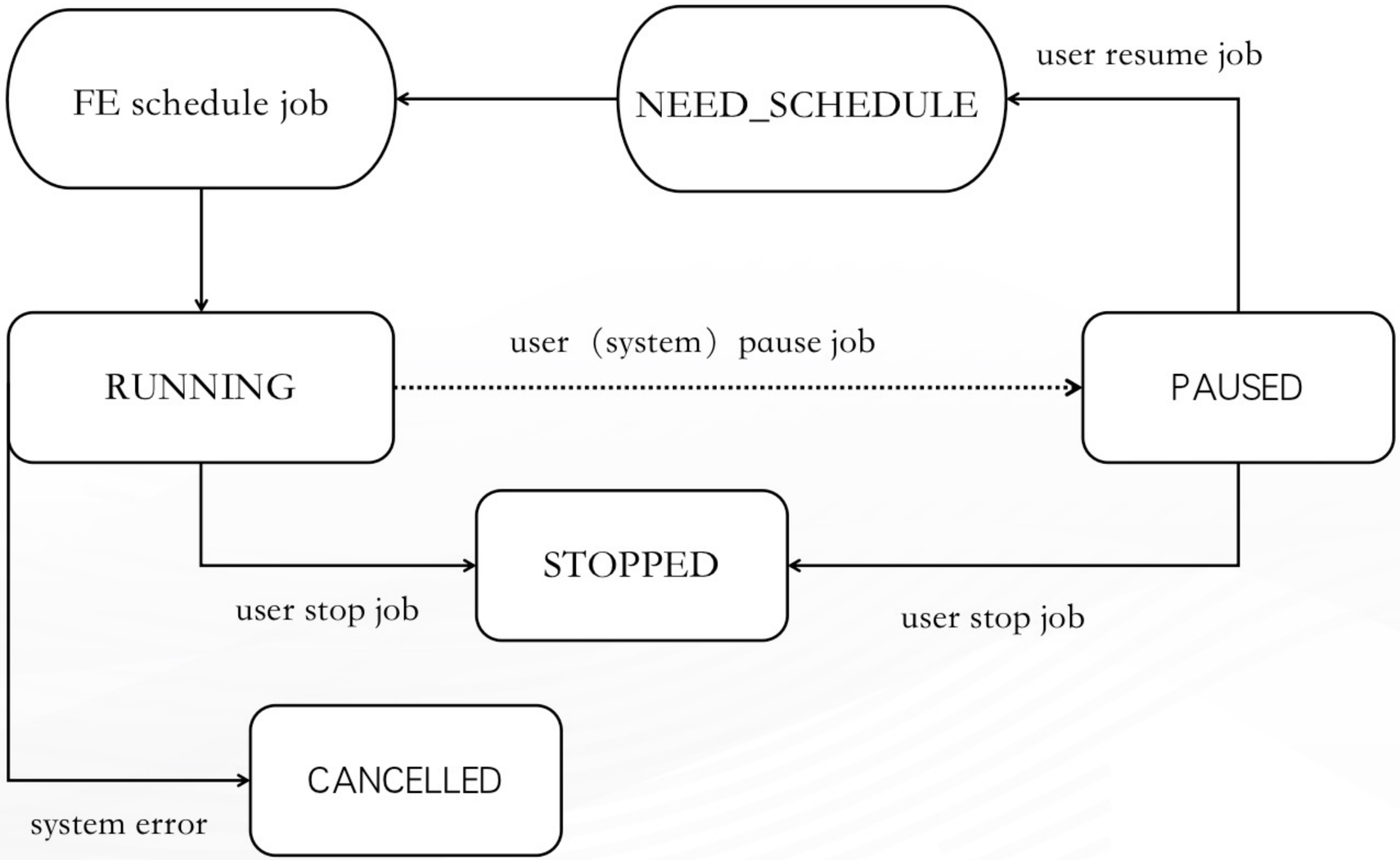
1. routine_load_task_consume_second

通过增大单次读取持续时间加速数据消费。

2. max_routine_load_batch_size

通过增大单次读取的数据量加速数据消费。

数据生态：Routine Load



数据生态：Routine Load

镜舟 Powered by StarRocks

1. 执行 SHOW ROUTINE LOAD 命令查看对应导入任务的状态：

```
SHOW ROUTINE LOAD FOR example1  \G
*************************** 1.row ***************************
                  Id:  63013
                Name: example1
          CreateTime: 2022-08-1017:09:00
           PauseTime: NULL
             EndTime: NULL
              DbName: default_cluster:example_db
           TableName: example_tbl2
               State: RUNNING
      DataSourceType: KAFKA
      CurrentTaskNum: 3
       JobProperties: {"partitions":"...}
DataSourceProperties: {"topic":}
  CustomProperties: {"kafka_default_offsets":"OFFSET_BEGINNING"...}
           Statistic: {"receivedBytes":230...}
            Progress: {"0":"1","1":"OFFSET_ZERO"...}
ReasonOfStateChanged:
        ErrorLogUrls:
            OtherMsg:
```

注意:StarRocks 只支持查看当前正在运行中的导入作业，不支持查看已经停止和未开始的导入作业。

数据生态：Routine Load 排查思路

当任务状态为PAUSED或者CANCELLED的时候需要介入排查

任务状态为PAUSED时：

1. 可以先查看ReasonOfStateChanged定位下原因，例如“Offset out of range”。

2. 若ReasonOfStateChanged为空，查看ErrorLogUrls可查看具体的报错信息：curl ${ErrorLogUrls}

3. 如果以上方法不能获取具体异常，可以执行以下命令查看（由于routine load是按周期调度的stream load任务，所以可以通过调度的任务查看任务的状态：

```
show routine load task where JobName="routine_load_wikipedia";
```

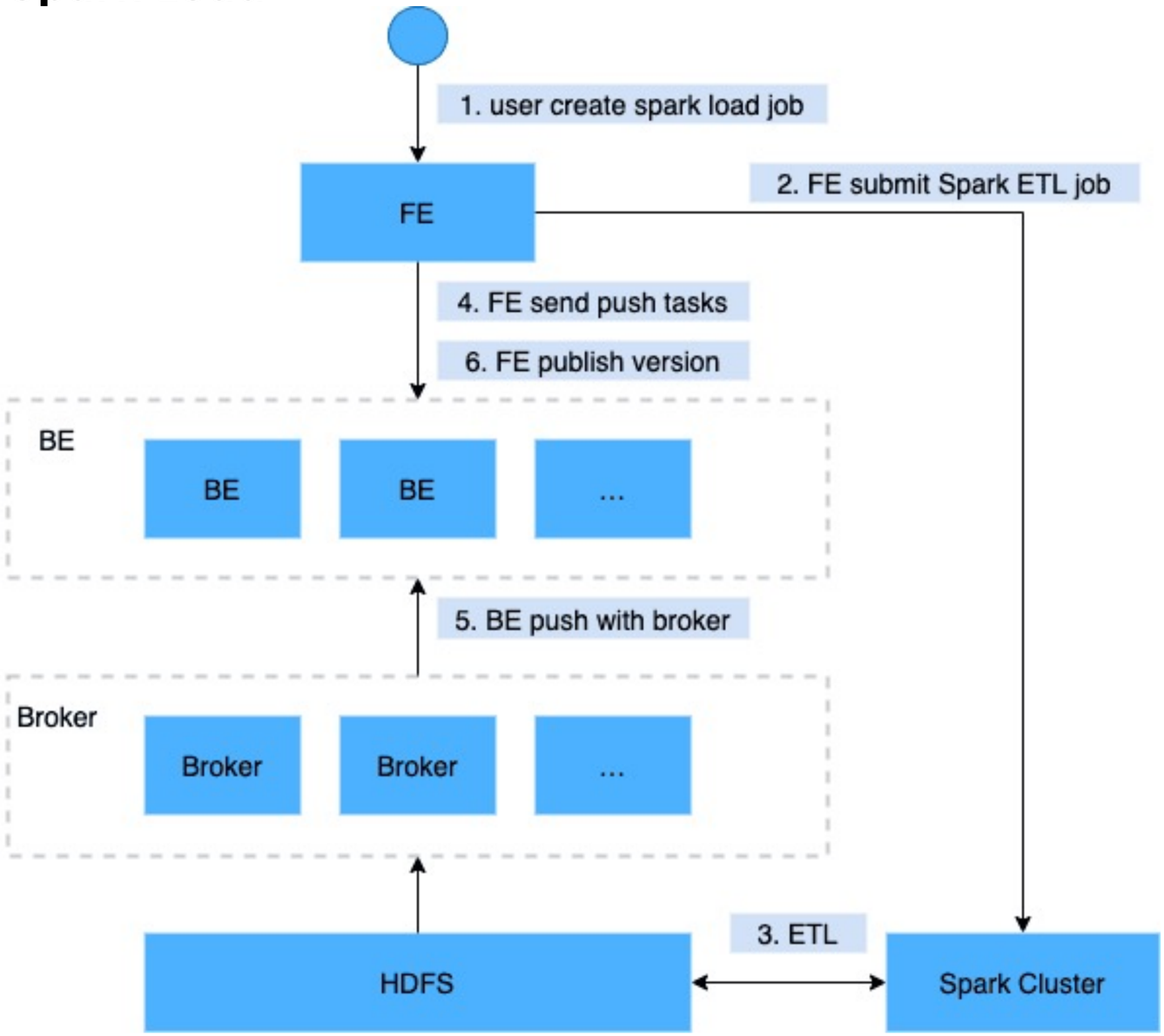
查看Message字段可以看到具体异常，如果以上方法都不能排查到问题，可以拿到job id在be.INFO日志中找到txn id，然后通过txn id在be.INFO中查看上下文有具体的任务信息。

任务状态为CANCELLED时：

1. 则可能为导入任务执行遇到异常（如表被删除）。

您可以参考ReasonOfStateChanged、ErrorLogUrls报错进行排查和修复。但是修复后，您无法恢复CANCELLED 状态的导入作业。

数据生态：Spark Load



Spark Load导入流程

数据生态：Spark Load

Spark Load 导入方式同 Broker Load 一样都是异步的，用户必须将创建导入的 Label 记录下来，并且在 SHOW LOAD 命令中使用 Label 来查看任务： show load order by createtime desc limit 1\G或者show load order where label="$label"\G;

*************************** 1. row ***************************

```
              JobId: 76391
              Label: label1
              State: FINISHED
           Progress: ETL:100%; LOAD:100%
               Type: SPARK
            EtlInfo: unselected.rows=4; dpp.abnorm.ALL=15; dpp.norm.ALL=28133376
           TaskInfo: cluster:cluster0; timeout(s):10800; max_filter_ratio:5.0E-5
           ErrorMsg: N/A
         CreateTime: 2019-07-27 11:46:42
       EtlStartTime: 2019-07-27 11:46:44
      EtlFinishTime: 2019-07-27 11:49:44
      LoadStartTime: 2019-07-27 11:49:44
     LoadFinishTime: 2019-07-27 11:50:16
              URL: http: //1.1.1.1:8089/proxy/application_1586619723848_0035/
       JobDetails: {"ScannedRows":28133395,"TaskNumber":1,"FileNumber":1,"FileSize":200000}
```

数据生态：Spark Load 排查思路

1. 先查看ErrorMsg，判断是否可直观定位出错误原因。

2. 如果上一步得不到具体异常，则可以去查看 Spark Launcher 的提交日志，spark 任务提交过程中产生的详细日志日志默认保存在 FE 根目录下： log/spark_launcher_log 路径下，并以 spark-launcher-{load-job-id}-{label}.log 命名，日志会在此目录下保存一段时间，当 FE 元数据中的导入信息被清理时，相应的日志也会被清理，默认保存时间为 3 天。

3. 如果根据第二步还查不到具体异常，可以在fe.WARNING日志中查找详细日志，搜索对应的 Label来进一步定位问题 。

4. 如果根据第三步也查不到异常，则访问spark ui查到对应executor日志，详细排查整个任务流程和配置打通等导入流程问题。

镜舟

Powered by StarRocks

谢谢观看 期待合作！