Apache Hudi集成Spark SQL抢先体验

原创 leesf ApacheHudi 5月6日

收录于话题

#Apache Hudi 36 #Spark SQL 1

1. 摘要

社区小伙伴一直期待的Hudi整合Spark SQL的[HUDI-1659](https://github.com/apache/hudi/pull/2645)正在积极Review中并已经快接近尾声，Hudi集成Spark SQL预计会在下个版本正式发布，在集成Spark SQL后，会极大方便用户对Hudi表的DDL/DML操作，下面来看看如何使用Spark SQL操作Hudi表。

2. 环境准备

首先需要将[HUDI-1659](https://github.com/apache/hudi/pull/2645)拉取到本地打包，生成 `SPARK_BUNDLE_JAR(hudi-spark-bundle_2.11-0.9.0-SNAPSHOT.jar)` 包

2.1 启动spark-sql

在配置完spark环境后可通过如下命令启动spark-sql

```
spark-sql --jars $PATH_TO_SPARK_BUNDLE_JAR  --conf 'spark.serializer=org.apache.spark
```

2.2 设置并发度

由于Hudi默认upsert/insert/delete的并发度是1500，对于演示的小规模数据集可设置更小的并发度。

```
set hoodie.upsert.shuffle.parallelism = 1;
set hoodie.insert.shuffle.parallelism = 1;
set hoodie.delete.shuffle.parallelism = 1;
```

同时设置不同步Hudi表元数据

```
set hoodie.datasource.meta.sync.enable=false;
```

3. Create Table

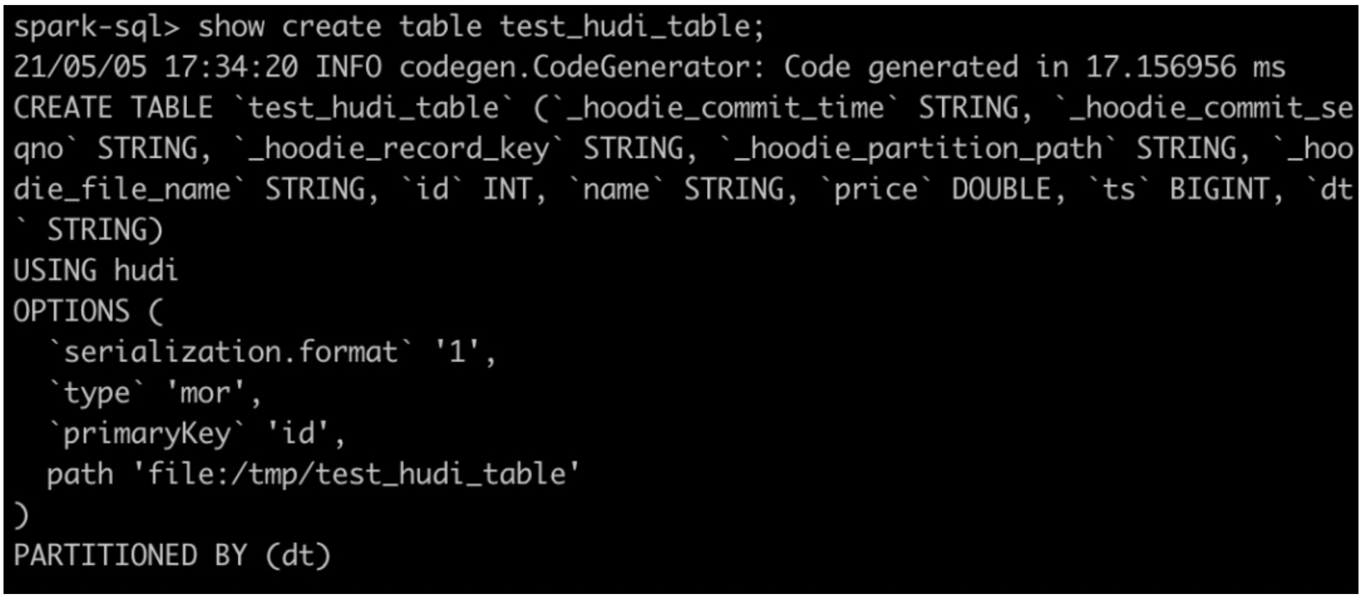
使用如下SQL创建表

```
create table test_hudi_table (
  id int,
  name string,
  price double,
  ts long,
  dt string
) using hudi
 partitioned by (dt)
 options (
  primaryKey = 'id',
  type = 'mor'
 )
 location 'file:///tmp/test_hudi_table'
```

说明：表类型为MOR，主键为id，分区字段为dt，合并字段默认为ts。

创建Hudi表后查看创建的Hudi表

```
show create table test_hudi_table
```



```
spark-sql> show create table test_hudi_table;
21/05/05 17:34:20 INFO codegen.CodeGenerator: Code generated in 17.156956 ms
CREATE TABLE `test_hudi_table` (`_hoodie_commit_time` STRING, `_hoodie_commit_se
qno` STRING, `_hoodie_record_key` STRING, `_hoodie_partition_path` STRING, `_hoo
die_file_name` STRING, `id` INT, `name` STRING, `price` DOUBLE, `ts` BIGINT, `dt
` STRING)
USING hudi
OPTIONS (
  `serialization.format` '1',
  `type` 'mor',
  `primaryKey` 'id',
  path 'file:/tmp/test_hudi_table'
)
PARTITIONED BY (dt)
```

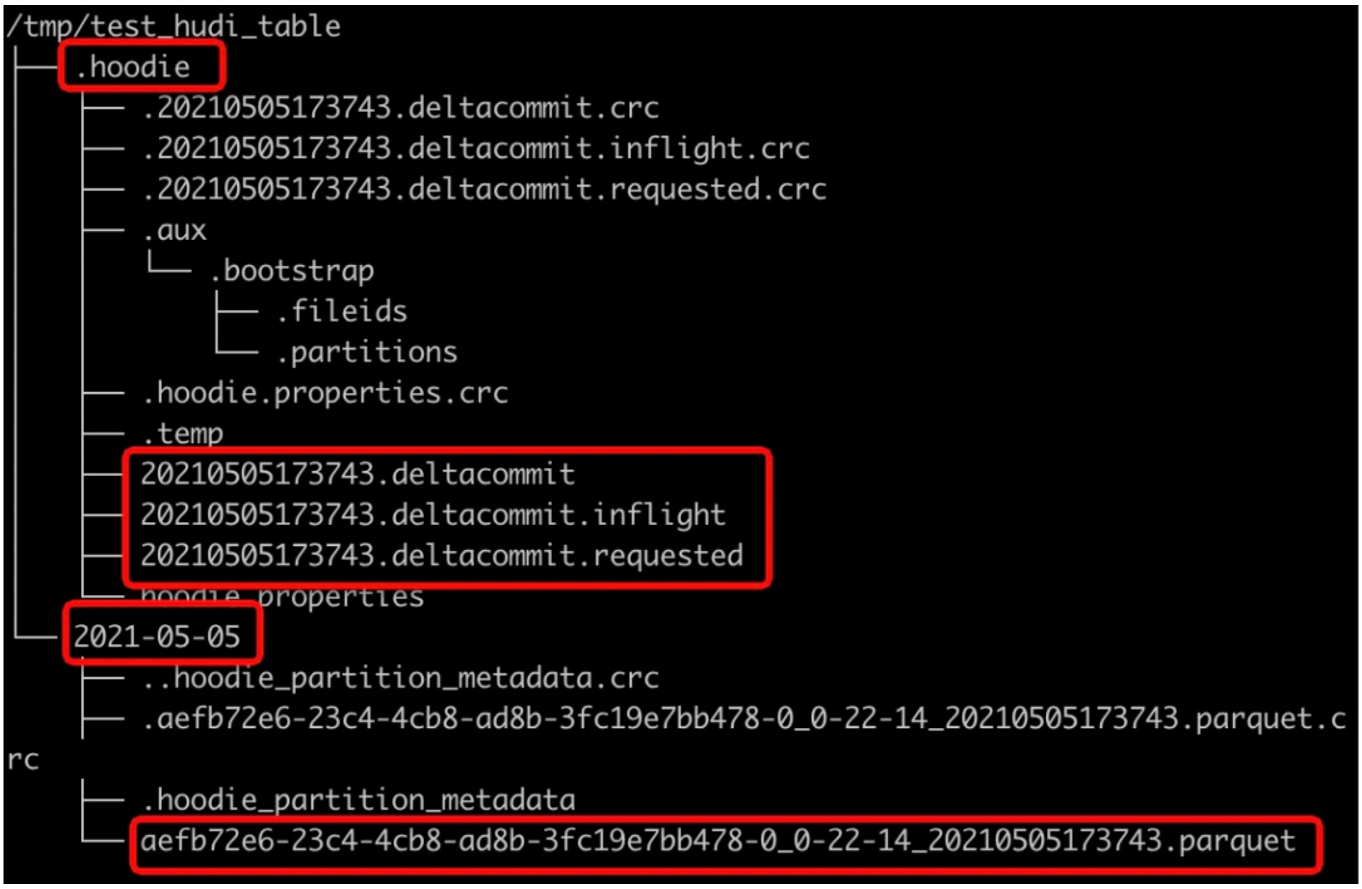
4. Insert Into

4.1 Insert

使用如下SQL插入一条记录

```
insert into test_hudi_table select 1 as id, 'hudi' as name, 10 as price, 1000 as ts,
```

insert完成后查看Hudi表本地目录结构，生成的元数据、分区和数据与Spark Datasource写入均相同。



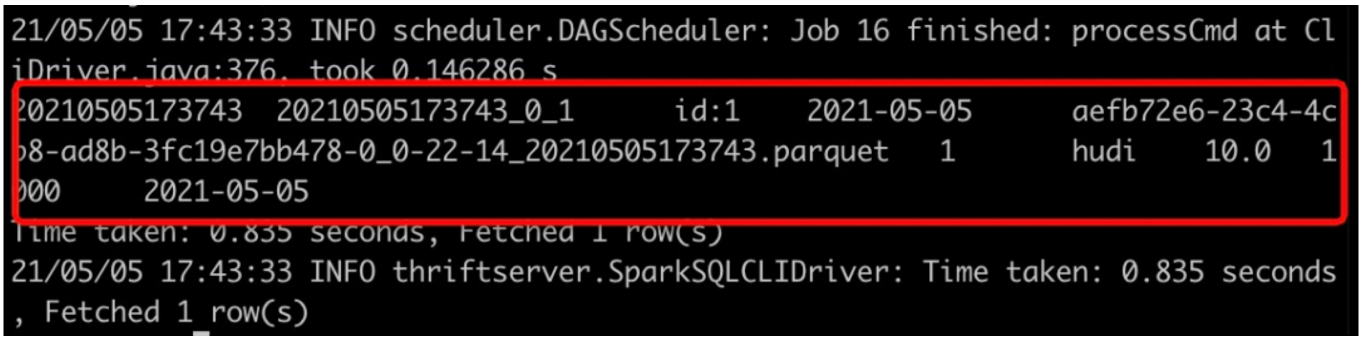
```
/tmp/test_hudi_table
├── .hoodie
│   ├── .20210505173743.deltacommit.crc
│   ├── .20210505173743.deltacommit.inflight.crc
│   ├── .20210505173743.deltacommit.requested.crc
│   ├── .aux
│   │   └── .bootstrap
│   │       ├── .fileids
│   │       └── .partitions
│   ├── .hoodie.properties.crc
│   ├── .temp
│   ├── 20210505173743.deltacommit
│   ├── 20210505173743.deltacommit.inflight
│   ├── 20210505173743.deltacommit.requested
│   └── hoodie.properties
└── 2021-05-05
    ├── ..hoodie_partition_metadata.crc
    ├── .aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_0-22-14_20210505173743.parquet.c
rc
    ├── .hoodie_partition_metadata
    └── aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_0-22-14_20210505173743.parquet
```

4.2 Select

使用如下SQL查询Hudi表数据

```
select * from test_hudi_table
```

查询结果如下



```
21/05/05 17:43:33 INFO scheduler.DAGScheduler: Job 16 finished: processCmd at Cl
iDriver.java:376, took 0.146286 s
20210505173743  20210505173743_0_1      id:1    2021-05-05      aefb72e6-23c4-4c
b8-ad8b-3fc19e7bb478-0_0-22-14_20210505173743.parquet   1       hudi    10.0    1
000     2021-05-05
Time taken: 0.835 seconds, Fetched 1 row(s)
21/05/05 17:43:33 INFO thriftserver.SparkSQLCLIDriver: Time taken: 0.835 seconds
, Fetched 1 row(s)
```

5. Update

5.1 Update

使用如下SQL将id为1的price字段值变更为20

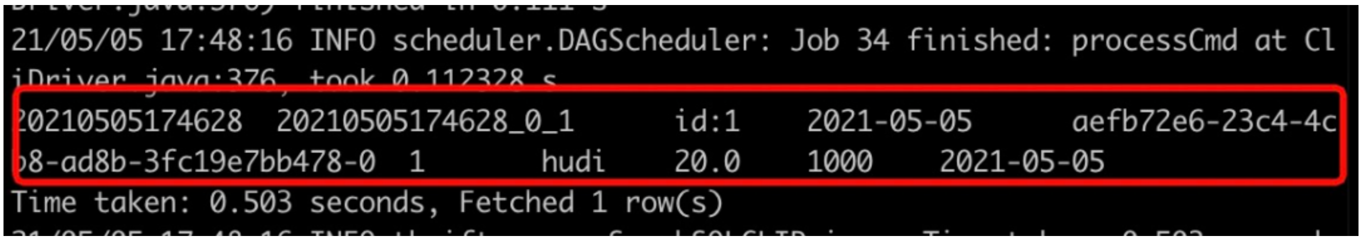
```
update test_hudi_table set price = 20.0 where id = 1
```

5.2 Select

再次查询Hudi表数据

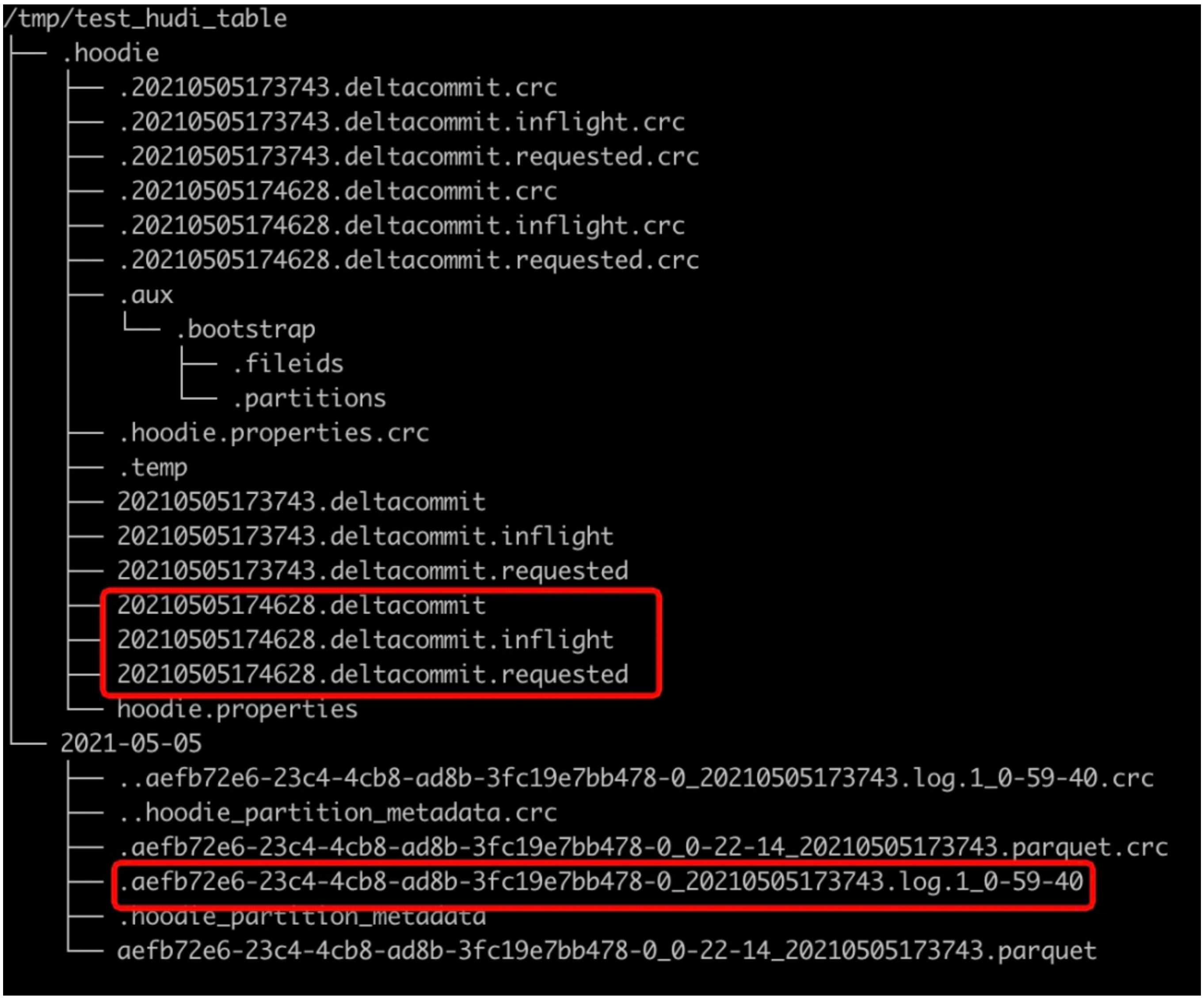
```
select * from test_hudi_table
```

查询结果如下，可以看到price已经变成了20.0



```
21/05/05 17:48:16 INFO scheduler.DAGScheduler: Job 34 finished: processCmd at Cl
iDriver.java:376, took 0.112328 s
20210505174628  20210505174628_0_1      id:1    2021-05-05      aefb72e6-23c4-4c
b8-ad8b-3fc19e7bb478-0  1       hudi    20.0    1000    2021-05-05
Time taken: 0.503 seconds, Fetched 1 row(s)
```

查看Hudi表的本地目录结构如下，可以看到在update之后又生成了一个 deltacommit ，同时生成了一个增量log文件。



```
/tmp/test_hudi_table
├── .hoodie
│   ├── .20210505173743.deltacommit.crc
│   ├── .20210505173743.deltacommit.inflight.crc
│   ├── .20210505173743.deltacommit.requested.crc
│   ├── .20210505174628.deltacommit.crc
│   ├── .20210505174628.deltacommit.inflight.crc
│   ├── .20210505174628.deltacommit.requested.crc
│   ├── .aux
│   │   └── .bootstrap
│   │       ├── .fileids
│   │       └── .partitions
│   ├── .hoodie.properties.crc
│   ├── .temp
│   ├── 20210505173743.deltacommit
│   ├── 20210505173743.deltacommit.inflight
│   ├── 20210505173743.deltacommit.requested
│   ├── 20210505174628.deltacommit
│   ├── 20210505174628.deltacommit.inflight
│   ├── 20210505174628.deltacommit.requested
│   └── hoodie.properties
└── 2021-05-05
    ├── ..aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_20210505173743.log.1_0-59-40.crc
    ├── ..hoodie_partition_metadata.crc
    ├── .aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_0-22-14_20210505173743.parquet.crc
    ├── .aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_20210505173743.log.1_0-59-40
    ├── .hoodie_partition_metadata
    └── aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_0-22-14_20210505173743.parquet
```

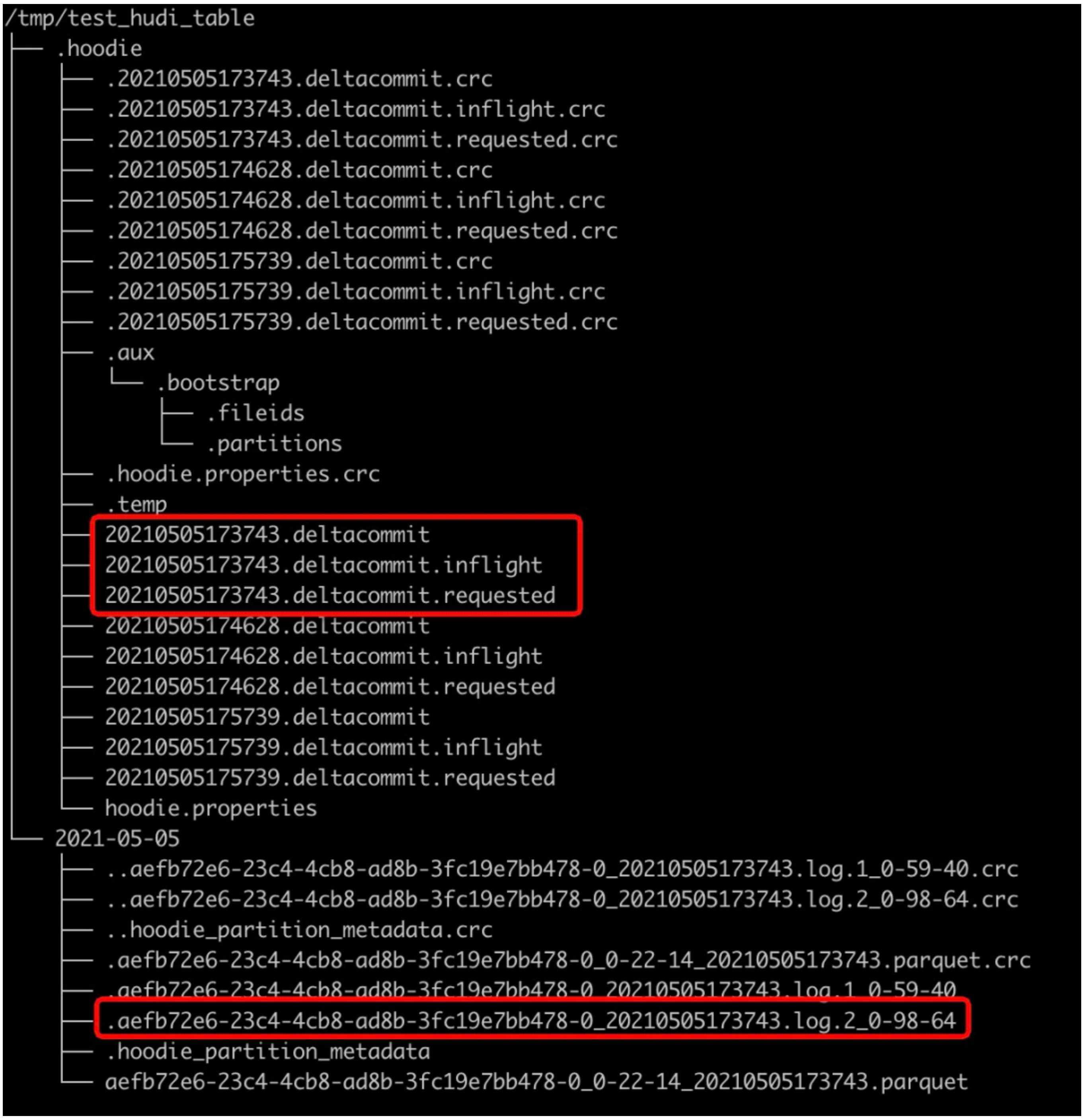
6. Delete

6.1 Delete

使用如下SQL将id=1的记录删除

```
delete from test_hudi_table where id = 1
```

查看Hudi表的本地目录结构如下，可以看到delete之后又生成了一个 deltacommit ，同时生成了一个增量log文件。



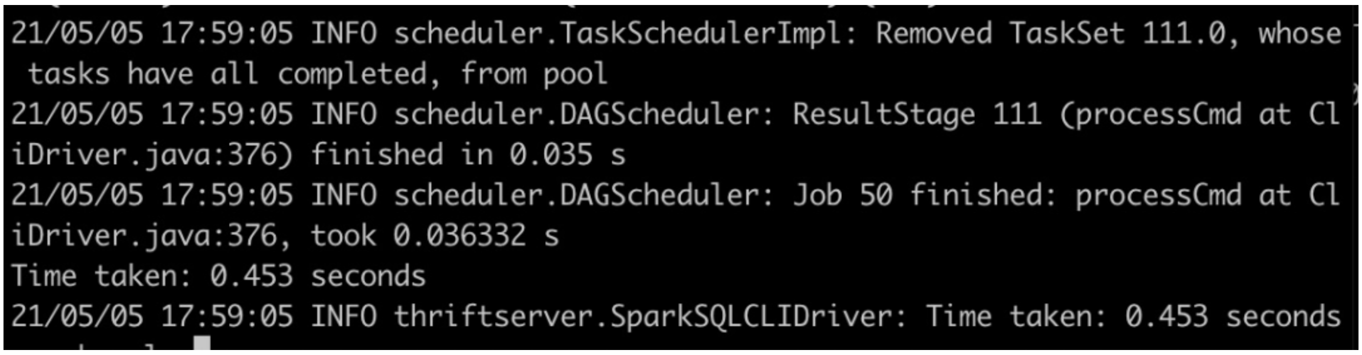
```
/tmp/test_hudi_table
├── .hoodie
│   ├── .20210505173743.deltacommit.crc
│   ├── .20210505173743.deltacommit.inflight.crc
│   ├── .20210505173743.deltacommit.requested.crc
│   ├── .20210505174628.deltacommit.crc
│   ├── .20210505174628.deltacommit.inflight.crc
│   ├── .20210505174628.deltacommit.requested.crc
│   ├── .20210505175739.deltacommit.crc
│   ├── .20210505175739.deltacommit.inflight.crc
│   ├── .20210505175739.deltacommit.requested.crc
│   ├── .aux
│   │   └── .bootstrap
│   │       ├── .fileids
│   │       └── .partitions
│   ├── .hoodie.properties.crc
│   ├── .temp
│   ├── 20210505173743.deltacommit
│   ├── 20210505173743.deltacommit.inflight
│   ├── 20210505173743.deltacommit.requested
│   ├── 20210505174628.deltacommit
│   ├── 20210505174628.deltacommit.inflight
│   ├── 20210505174628.deltacommit.requested
│   ├── 20210505175739.deltacommit
│   ├── 20210505175739.deltacommit.inflight
│   ├── 20210505175739.deltacommit.requested
│   └── hoodie.properties
└── 2021-05-05
    ├── ..aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_20210505173743.log.1_0-59-40.crc
    ├── ..aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_20210505173743.log.2_0-98-64.crc
    ├── ..hoodie_partition_metadata.crc
    ├── .aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_0-22-14_20210505173743.parquet.crc
    ├── .aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_20210505173743.log.1_0-59-40
    ├── .aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_20210505173743.log.2_0-98-64
    ├── .hoodie_partition_metadata
    └── aefb72e6-23c4-4cb8-ad8b-3fc19e7bb478-0_0-22-14_20210505173743.parquet
```

6.2 Select

再次查询Hudi表

```
select * from test_hudi_table;
```

查询结果如下，可以看到已经查询不到任何数据了，表明Hudi表中已经不存在任何记录了。



```
21/05/05 17:59:05 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 111.0, whose
 tasks have all completed, from pool
21/05/05 17:59:05 INFO scheduler.DAGScheduler: ResultStage 111 (processCmd at Cl
iDriver.java:376) finished in 0.035 s
21/05/05 17:59:05 INFO scheduler.DAGScheduler: Job 50 finished: processCmd at Cl
iDriver.java:376, took 0.036332 s
Time taken: 0.453 seconds
21/05/05 17:59:05 INFO thriftserver.SparkSQLCLIDriver: Time taken: 0.453 seconds
```

7. Merge Into

7.1 Merge Into Insert

使用如下SQL向 test_hudi_table 插入数据

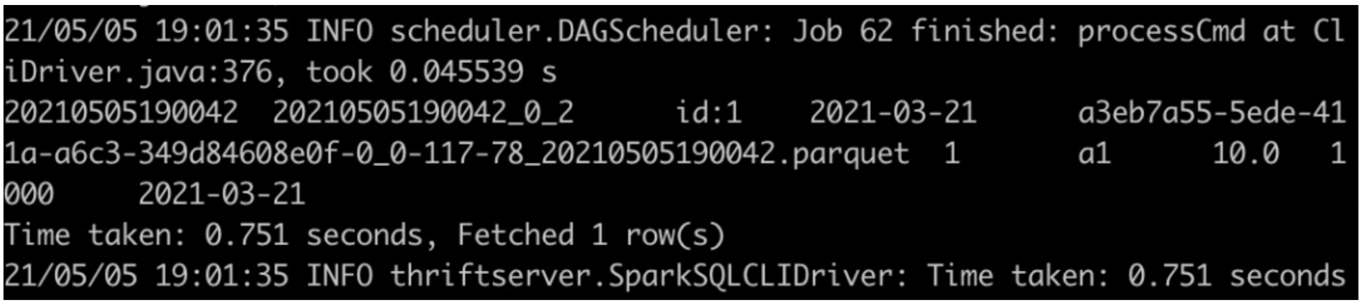
```
merge into test_hudi_table as t0
using (
 select 1 as id, 'a1' as name, 10 as price, 1000 as ts, '2021-03-21' as dt
) as s0
on t0.id = s0.id
when not matched and s0.id % 2 = 1 then insert *
```

7.2 Select

查询Hudi表数据

```
select * from test_hudi_table
```

查询结果如下，可以看到Hudi表中存在一条记录



```
21/05/05 19:01:35 INFO scheduler.DAGScheduler: Job 62 finished: processCmd at Cl
iDriver.java:376, took 0.045539 s
20210505190042  20210505190042_0_2      id:1    2021-03-21      a3eb7a55-5ede-41
1a-a6c3-349d84608e0f-0_0-117-78_20210505190042.parquet  1       a1      10.0    1
000     2021-03-21
Time taken: 0.751 seconds, Fetched 1 row(s)
21/05/05 19:01:35 INFO thriftserver.SparkSQLCLIDriver: Time taken: 0.751 seconds
```

7.4 Merge Into Update

使用如下SQL更新数据

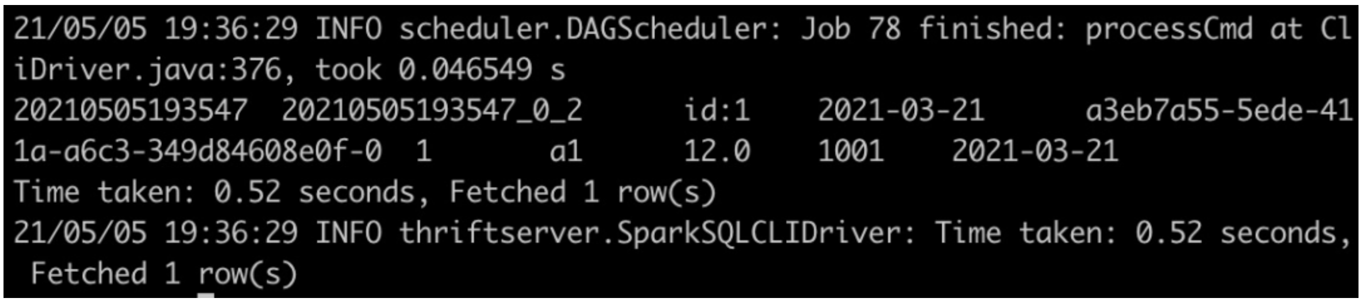
```
merge into test_hudi_table as t0
using (
 select 1 as id, 'a1' as name, 12 as price, 1001 as ts, '2021-03-21' as dt
) as s0
on t0.id = s0.id
when matched and s0.id % 2 = 1 then update set *
```

7.5 Select

查询Hudi表

```
select * from test_hudi_table
```

查询结果如下，可以看到Hudi表中的分区已经更新了



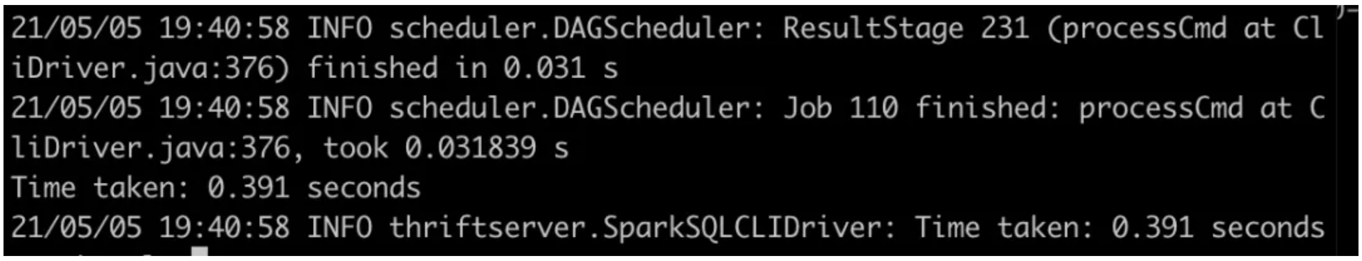
```
21/05/05 19:36:29 INFO scheduler.DAGScheduler: Job 78 finished: processCmd at Cl
iDriver.java:376, took 0.046549 s
20210505193547  20210505193547_0_2      id:1    2021-03-21      a3eb7a55-5ede-41
1a-a6c3-349d84608e0f-0  1       a1      12.0    1001    2021-03-21
Time taken: 0.52 seconds, Fetched 1 row(s)
21/05/05 19:36:29 INFO thriftserver.SparkSQLCLIDriver: Time taken: 0.52 seconds,
 Fetched 1 row(s)
```

7.6 Merge Into Delete

使用如下SQL删除数据

```
merge into test_hudi_table t0
 using (
  select 1 as s_id, 'a2' as s_name, 15 as s_price, 1001 as s_ts, '2021-03-21' as dt
 ) s0
 on t0.id = s0.s_id
 when matched and s_ts = 1001 then delete
```

查询结果如下，可以看到Hudi表中已经没有数据了



```
21/05/05 19:40:58 INFO scheduler.DAGScheduler: ResultStage 231 (processCmd at Cl
iDriver.java:376) finished in 0.031 s
21/05/05 19:40:58 INFO scheduler.DAGScheduler: Job 110 finished: processCmd at C
liDriver.java:376, took 0.031839 s
Time taken: 0.391 seconds
21/05/05 19:40:58 INFO thriftserver.SparkSQLCLIDriver: Time taken: 0.391 seconds
```

8. 删除表

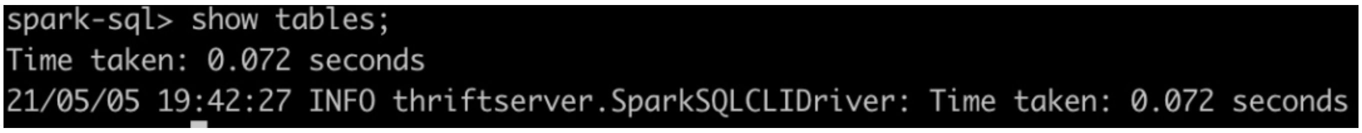
使用如下命令删除Hudi表

```
drop table test_hudi_table;
```

使用show tables查看表是否存在

```
show tables;
```

可以看到已经没有表了



```
spark-sql> show tables;
Time taken: 0.072 seconds
21/05/05 19:42:27 INFO thriftserver.SparkSQLCLIDriver: Time taken: 0.072 seconds
```

9. 总结

通过上面示例简单展示了通过Spark SQL Insert/Update/Delete Hudi表数据，通过SQL方式可以非常方便地操作Hudi表，降低了使用Hudi的门槛。另外Hudi集成Spark SQL工作将继续完善语法，尽量对标Snowflake和BigQuery的语法，如插入多张表（INSERT ALL WHEN condition1 INTO t1 WHEN condition2 into t2），变更Schema以及CALL Cleaner、CALL Clustering等Hudi表服务。

推荐阅读

提升50%+！Presto如何提升Hudi表查询性能？

在AWS Glue中使用Apache Hudi

致广大数据湖用户的一封信

Apache Hudi在Linkflow构建实时数据湖的生产实践

Apache Hudi C位！云计算一哥AWS EMR 2020年度回顾

喜欢此内容的人还喜欢

从linux源码看epoll

解Bug之路

JAVA线上故障排查全套解决方案

架构之家

Apache Impala 4.0 发布了！

Hadoop实操