Meetings

Meeting Notes for Etsy PoC
Notes - CLI runner and GCS bucket layout
(shared with Etsy) Zipline POC Setup

Deployment Strategy

- [Zipline] GCS Bucket for jars, prefixed by env(and possibly more)
    - Eg: gs://zipline-jars/prod/…
- [Zipline] GCS Bucket for temp data, prefixed by env
    - Eg: gs://zipline-warehouse/prod/…
- [Zipline] GCS Bucket for job configs, prefixed by env (and possibly more)
    - Eg: gs://zipline-confs/prod/…
- [Zipline]

```
None
gs://zipline-<customer_name>
       /bin
              <jars|wheels>
       /conf
              <conf_name>
       /data
              /tmp
              <dataproc_application>
```

Jars and configs could potentially reside in the same bucket.
We could also encode the region in the bucket naming (more future thinking)

- **Revised bucket layout: Notes - Sync on gcp stuff**

```
None
compile.py --conf file://path/to/python
CUSTOMER_ID=etsy run.py --conf file://path/to/conf/production/1.thriftjson
--gcp \
 --project-id \
 --region \
 --cluster-name
```

```
--- Customer VPC
gcs://zipline-warehouse-etsy
        /data
            ...
            ...
        /metadata
            1.thriftjson


--- Zipline VPC
gcs://zipline-canary
        /jars
            cloud_gcp-1.jar
            cloud_gcp-2.jar
            cloud_gcp-3.jar
            ...


gcs://zipline-artifacts-etsy
        /jars
            dataproc-submitter.jar (DataprocSubmitter.scala)
            cloud_gcp.jar (Driver.scala)
```

- [Zipline] Dataproc cluster
    - on 2.2
    - Configurations spark-bigquery, bigtable, flink
    - No HMS or federation configured
- [Zipline] BigTable cluster,
- [Zipline] Configure BigQuery -> BigTable exports
    - iam roles
    - BigTable profile
- [Zipline] Streaming side we need GCP managed kafka infra
- [Etsy] Bigquery project configured. This could be different from the dataproc project.

- [Etsy + Zipline] R/W access from dataproc to bigquery, using a service account

☐	703996152583-compute@developer.gserviceaccount.com	Compute Engine default service account	BigQuery Admin BigQuery Connection Admin BigQuery Data Owner BigQuery Data Viewer BigQuery Job User BigQuery User Bigtable User Dataproc Administrator Dataproc Metastore Admin Dataproc Metastore Editor Dataproc Metastore Metadata Editor Dataproc Metastore Metadata Mutate Admin Dataproc Metastore Metadata User Dataproc Service Agent Dataproc Worker Metastore Federation Accessor

- [Etsy + Zipline] Access from dataproc to bigtable using a service account

☐	dataproc@canary-443022.iam.gserviceaccount.com	Dataproc SA	BigQuery Admin	Advanced security insight
			BigQuery Connection Admin	Advanced security insight
			BigQuery Data Owner	Advanced security insight
			Bigtable User	Advanced security insight
			Dataproc Administrator	Advanced security insight
			Dataproc Metastore Admin	Advanced security insight
			Dataproc Metastore Editor	Advanced security insight
			Dataproc Metastore Metadata Editor	Advanced security insight
			Dataproc Metastore Metadata Mutate Admin	Advanced security insight
			Dataproc Service Agent	Advanced security insight
			Dataproc Worker	Advanced security insight
			Metastore Federation Accessor	Advanced security insight

- [Etsy + Zipline] Access from dataproc to GCS buckets using a service account

Open questions:
How does IAM work? How can we go from BQ to BT?

Work Findings

Gotchas

GCP dataproc versions are fixed to Java 11 across the board:
https://cloud.google.com/dataproc/docs/concepts/versioning/dataproc-release-2.2

We probably don't want to support BigLake, and push Etsy towards BigQuery on Iceberg.
https://cloud.google.com/bigquery/docs/biglake-intro

- BigLake tables are read-only. You cannot modify BigLake tables using DML statements or other methods.

Metastore
Should just use BigQuery as a metastore instead of the Dataproc metastore (basically HMS)

SparkSessionBuilder:
https://github.com/zipline-ai/chronon/blob/main/spark/src/main/scala/ai/chronon/spark/SparkSessionBuilder.scala#L71

https://github.com/GoogleCloudDataproc/spark-bigquery-connector/pull/649

```
Java
val spark = SparkSession.builder()
  .appName("IcebergBigQueryCatalogExample")
  .config("spark.sql.catalog.my_catalog",
"org.apache.iceberg.spark.SparkCatalog")
  .config("spark.sql.catalog.my_catalog.catalog-impl",
"org.apache.iceberg.gcp.bigquery.BigQueryCatalog")
  .config("spark.sql.catalog.my_catalog.warehouse",
"gs://my-iceberg-warehouse/")
  .config("spark.sql.catalog.my_catalog.project-id", "your-gcp-project-id")
  .getOrCreate()
```

The above won't work. We don't have a BigQueryCatalog. It got removed:
https://github.com/GoogleCloudDataproc/spark-bigquery-connector/pull/699

Maintainers decided standalone catalog impl won't be needed, most things are just baked into DataSource abstraction. Is that enough for us? We can't list partitions though.
https://github.com/GoogleCloudDataproc/spark-bigquery-connector?tab=readme-ov-file#configuring-partitioning

We do catalog operations:
https://github.com/zipline-ai/chronon/blob/main/spark/src/main/scala/ai/chronon/spark/TableUtils.scala#L231-L234

There's something called Dataplex which is a catalog layer on top of all your storage systems. It needs HMS access for spark job created tables, which dataproc metastore can provide. It can hook into BQ.

Even if we create our own Dataproc HMS, it'll be out of sync with BQ.
https://cloud.google.com/dataproc-metastore/docs/data-catalog-sync shows an explicit sync, but there's nothing to just directly rely on BQ

Dataproc has catalog federation!
https://cloud.google.com/dataproc-metastore/docs/hms-federation
Can use BigQuery or Dataplex Lakes https://cloud.google.com/dataplex/docs/create-lake .
Needs a BQ project with at least one dataset in it. Probably prefer this as Etsy has it. Dataplex is new, can always upgrade. This would be an integration at the infra level (terraform a new federation source).
This is actually just a Hive API on top. Nice!

BigQuery

Options

With federated endpoint, we don't need to expose DataPointer first. It can just be `database.table`.

Without federated endpoint, we may need to do that. Otherwise we'd do the fallback in the code ourselves (try catch waterfall).

Probably no federation? fewer gotchas. At the least, do the fallback ourselves in the code.

We can expose datapointer first, things are backwards compatible.

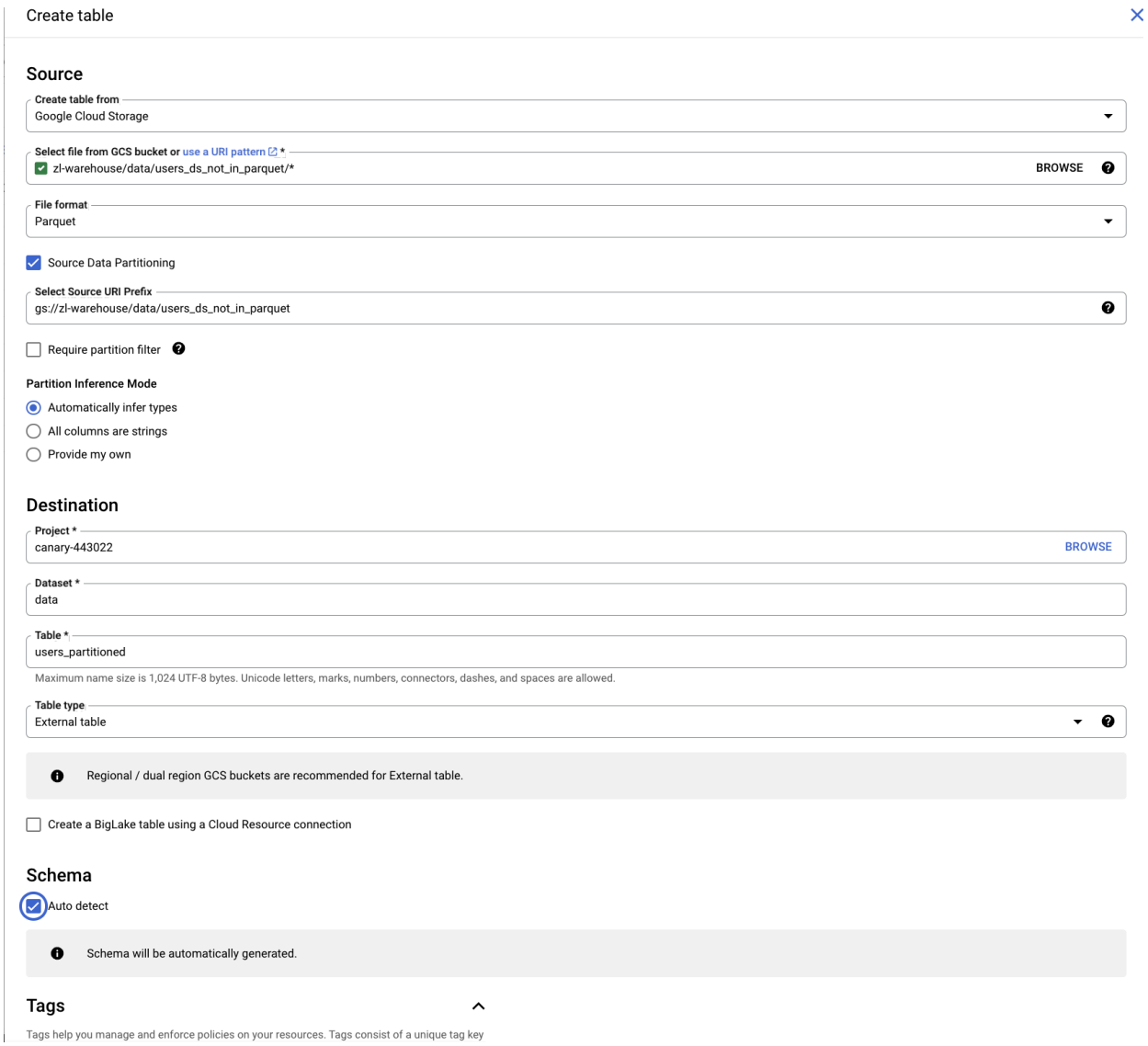
1. Expose the DataPointer
2. Users specify `database.table`
    a. Look it up in hive.
    b. Look it up in

Checking if an external table backed by GCS is partitioned?

```
None
SELECT * FROM `canary-443022.data.INFORMATION_SCHEMA.COLUMNS` where table_name
= <> and is_partitioning_column = 'YES'
```

Create external tables on partitioned data

https://cloud.google.com/bigquery/docs/external-data-cloud-storage#create-external-table-partitioned



Important things to note:

- **Source Data Partitioning**
- **Schema > Auto detect** checked
- BigQuery doesn't allow the partition in the gcs path to be present as a column in the raw data:

None

```
Failed to add partition key ds (type: TYPE_STRING) to schema, because another
column with the same name was already present. This is not allowed. Full
partition schema: [ds:TYPE_STRING].
```

Catalog Metadata from Spark

Python

```
scala> val ct = spark.sessionState.catalog.getTableMetadata(tableID)
ct: org.apache.spark.sql.catalyst.catalog.CatalogTable =
CatalogTable(
Catalog: spark_catalog
Database: data
Table: checkouts_native
Created Time: Fri Dec 13 05:51:22 UTC 2024
Last Access: Fri Dec 13 05:51:22 UTC 2024
Created By: Spark 2.2 or prior
Type: EXTERNAL
Provider: com.google.cloud.spark.bigquery
Table Properties:
[FEDERATION_BIGQUERY_TABLE_PROPERTY=canary-443022.data.checkouts_native,
federation.bigquery.table.type=MANAGED]
Serde Library: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat: org.apache.hadoop.mapred.TextInputFormat
OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Storage Properties: [project=canary-443022, dataset=data,
table=checkouts_native])


// External Tables

scala> val tableID = TableIdentifier("checkouts_parquet", Some("data"))
tableID: org.apache.spark.sql.catalyst.TableIdentifier =
`data`.`checkouts_parquet`

scala> val ct = spark.sessionState.catalog.getTableMetadata(tableID)
ct: org.apache.spark.sql.catalyst.catalog.CatalogTable =
CatalogTable(
Catalog: spark_catalog
```

```
Database: data
Table: checkouts_parquet
Created Time: Fri Dec 13 23:02:04 UTC 2024
Last Access: Fri Dec 13 23:02:04 UTC 2024
Created By: Spark 2.2 or prior
Type: EXTERNAL
Provider: com.google.cloud.spark.bigquery
Table Properties:
[FEDERATION_BIGQUERY_TABLE_PROPERTY=canary-443022.data.checkouts_parquet,
federation.bigquery.table.type=EXTERNAL]
Serde Library: org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe
InputFormat: org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat
OutputFormat: org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat)


scala> val ct = spark.sessionState.catalog.getTableMetadata(tableID)
ct: org.apache.spark.sql.catalyst.catalog.CatalogTable =
CatalogTable(
Catalog: spark_catalog
Database: default
Table: checkouts_dh
Owner: root
Created Time: Mon Dec 16 19:02:41 UTC 2024
Last Access: UNKNOWN
Created By: Spark 3.5.1
Type: MANAGED
Provider: hive
Table Properties: [transient_lastDdlTime=1734375761]
Location:
gs://gcs-bucket-service-baaa-35549b5d-c533-479b-a846-486147487b0f/hive-warehous
e/checkouts_dh
Serde Library: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat: org.apache.hadoop.mapred.TextInputFormat
OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Storage Properties: [serialization.format=1]
Partition Provider: Catalog
Partition Columns: [`ds`]
Schema: root
 |-- ts: long (nullable = true)
 |-- return_id: ...
```

Connecting to remote BigQuery

Set up a tunnel locally

```
None
gcloud compute ssh zipline-canary-cluster-m \
    --zone us-central1-c \
    -- -f -N -L 9083:localhost:9083
```

Dataproc

Steps to creating a dataproc cluster with federation

Federation info

Federation name *
federation-1560

Data location *
us-central1

Version *
3.1.2

Sources *

Add sources for this federated service. The order determines the collision priority. Use the handle to drag sources to reorder them.

New source

Source Type *

BigQuery

Dataplex PREVIEW

Dataproc Metastore

WSE

DONE

ADD A SOURCE

SUBMIT CANCEL

https://cloud.google.com/dataproc-metastore/docs/hms-federation#source_ordering

it looks like Federation requires at least one metastore instance.
https://cloud.google.com/dataproc-metastore/docs/hms-federation#metadata_sources
When you create a federation service, you must add a metadata source.

To create a cluster with federation hooked up:
First gotta give metastore access to dataproc cluster principal:
https://cloud.google.com/iam/docs/understanding-roles#metastore.editor

https://cloud.google.com/dataproc-metastore/docs/create-federation#attach-federation-cluster

None

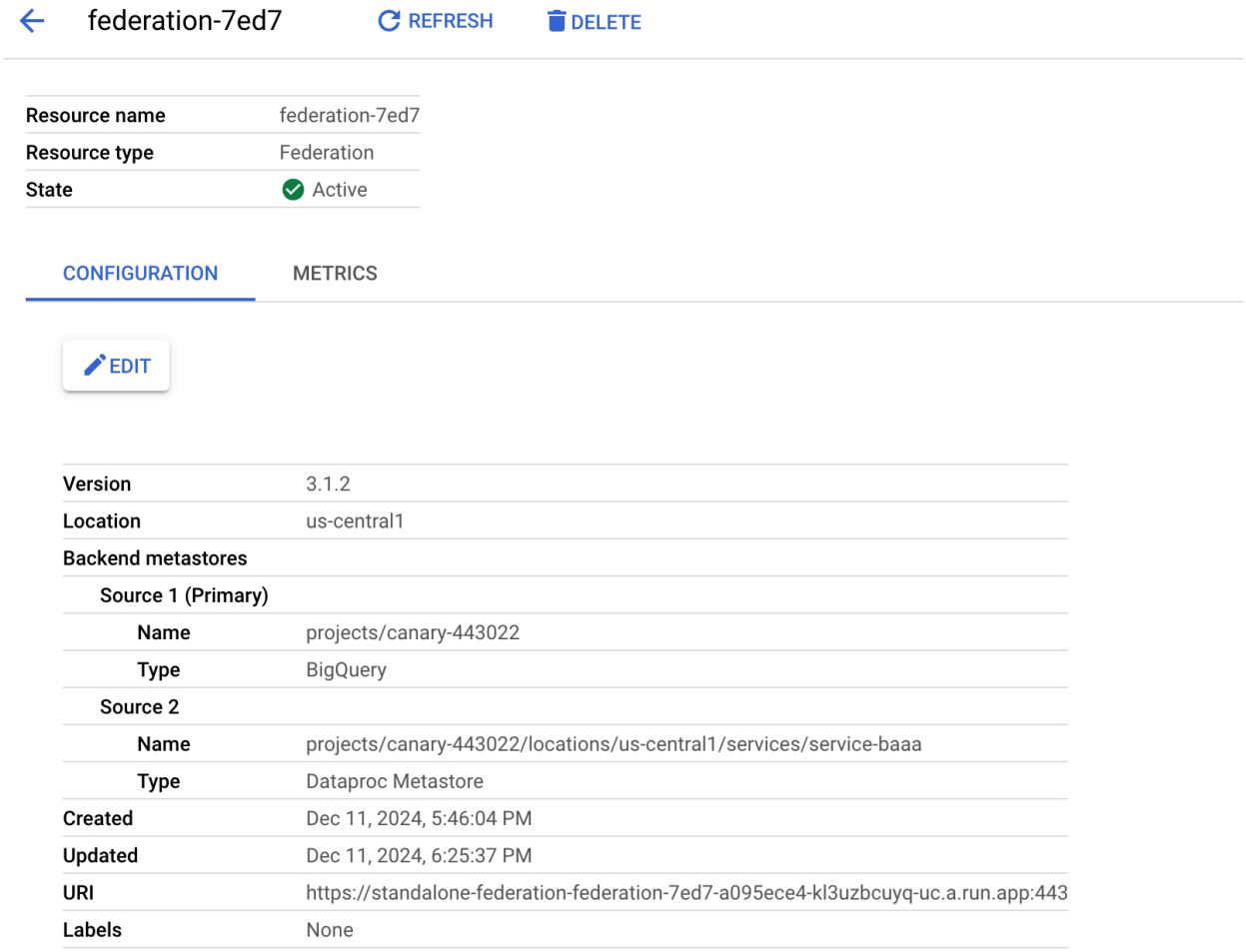
```
 gcloud dataproc clusters create canary-2 \
       --region=us-central1 \
       --project=canary-443022 \
       --scopes=https://www.googleapis.com/auth/cloud-platform \
       --image-version=2.2.39-debian12 \
       --service-account=703996152583-compute@developer.gserviceaccount.com \
       --enable-component-gateway \
       --optional-components=DOCKER,JUPYTER \

--initialization-actions=gs://metastore-init-actions/metastore-grpc-proxy/metas
tore-grpc-proxy.sh \
       --metadata="proxy-uri=https://standalone-federation-federation-7ed7-a095e
ce4-kl3uzbcuyq-uc.a.run.app:443,hive-version=3.1.2,SPARK_BQ_CONNECTOR_URL=gs://
spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.41.0.jar" \
       --properties="hive:hive.metastore.uris=thrift://localhost:9083,hive:hive.
metastore.warehouse.dir=gs://gcs-bucket-service-baaa-35549b5d-c533-479b-a846-48
6147487b0f/hive-warehouse"
```

Need the following roles:

Type	Principal	Name	Role	Security insights	Inheritance
	703996152583-compute@developer.gserviceaccount.com	Compute Engine default service account	BigQuery Admin	Advanced security insight	
			BigQuery Connection Admin	Advanced security insight	
			BigQuery Data Owner	Advanced security insight	
			Dataproc Administrator	Advanced security insight	
			Dataproc Metastore Admin	Advanced security insight	
			Dataproc Metastore Editor	Advanced security insight	
			Dataproc Metastore Metadata Editor	Advanced security insight	
			Dataproc Metastore Metadata Mutate Admin	Advanced security insight	
			Dataproc Metastore Metadata User	Advanced security insight	
			Dataproc Service Agent	Advanced security insight	
			Dataproc Worker	Advanced security insight	
			Metastore Federation Accessor	Advanced security insight	

Need to enable: https://cloud.google.com/bigquery/docs/reference/storage/#enabling_the_api

Make sure that BigQuery is the **primary** backend for the Federation endpoint:



Opening a Jupyter notebook on Dataproc

https://cloud.google.com/dataproc/docs/tutorials/jupyter-notebook#open_the_jupyter_and_jupyterlab_uis

***Warning**: you might have to open this via Safari. Chrome kept 502'ing when trying to open up Jupyter on Web Interfaces via Dataproc.*

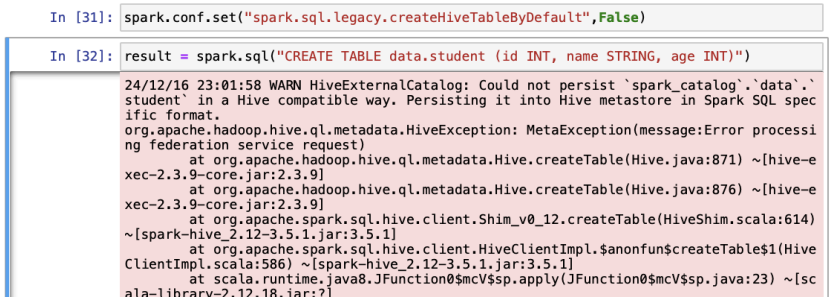
❌ Creating a new table in Spark under a BigQuery managed dataset

```
result = spark.sql("CREATE TABLE data.student (id INT, name STRING, age INT)")
```

```
24/12/16 22:58:12 WARN ResolveSessionCatalog: A Hive serde table will be created as there is no table provider spec
ified. You can set spark.sql.legacy.createHiveTableByDefault to false so that native data source table will be crea
ted instead.
```

```
---------------------------------------------------------------------------
AnalysisException                         Traceback (most recent call last)
Cell In[30], line 1
----> 1 result = spark.sql("CREATE TABLE data.student (id INT, name STRING, age INT)")

File /usr/lib/spark/python/pyspark/sql/session.py:1631, in SparkSession.sql(self, sqlQuery, args, **kwargs)
   1627         assert self._jvm is not None
   1628         litArgs = self._jvm.PythonUtils.toArray(
   1629             [_to_java_column(lit(v)) for v in (args or [])]
   1630         )
-> 1631     return DataFrame(self._jsparkSession.sql(sqlQuery, litArgs), self)
   1632 finally:
   1633     if len(kwargs) > 0:

File /usr/lib/spark/python/lib/py4j-0.10.9.7-src.zip/py4j/java_gateway.py:1322, in JavaMember.__call__(self, *args)
   1316 command = proto.CALL_COMMAND_NAME +\
   1317     self.command_header +\
   1318     args_command +\
   1319     proto.END_COMMAND_PART
   1321 answer = self.gateway_client.send_command(command)
-> 1322 return_value = get_return_value(
   1323     answer, self.gateway_client, self.target_id, self.name)
   1325 for temp_arg in temp_args:
   1326     if hasattr(temp_arg, "_detach"):

File /usr/lib/spark/python/pyspark/errors/exceptions/captured.py:185, in capture_sql_exception.<locals>.deco(*a, **
kw)
    181 converted = convert_exception(e.java_exception)
    182 if not isinstance(converted, UnknownException):
    183     # Hide where the exception came from that shows a non-Pythonic
    184     # JVM exception message.
--> 185     raise converted from None
    186 else:
    187     raise

AnalysisException: org.apache.hadoop.hive.ql.metadata.HiveException: MetaException(message:Error processing federat
ion service request)
```

Even flipping the flag to try to create native Spark tables didn't work.



```
In [31]: spark.conf.set("spark.sql.legacy.createHiveTableByDefault",False)
```

```
In [32]: result = spark.sql("CREATE TABLE data.student (id INT, name STRING, age INT)")
```

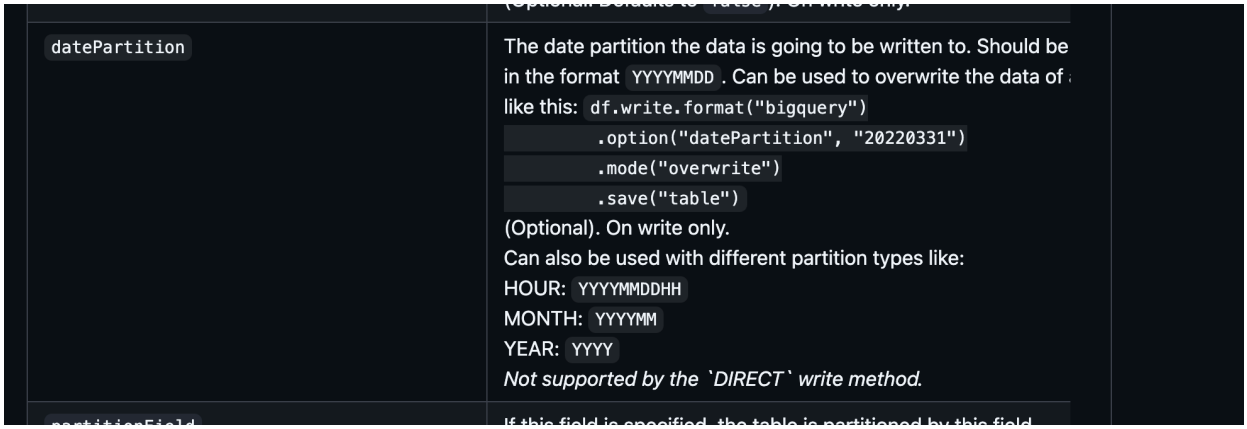
```
24/12/16 23:01:58 WARN HiveExternalCatalog: Could not persist `spark_catalog`.`data`.`
student` in a Hive compatible way. Persisting it into Hive metastore in Spark SQL spec
ific format.
org.apache.hadoop.hive.ql.metadata.HiveException: MetaException(message:Error processi
ng federation service request)
        at org.apache.hadoop.hive.ql.metadata.Hive.createTable(Hive.java:871) ~[hive-e
xec-2.3.9-core.jar:2.3.9]
        at org.apache.hadoop.hive.ql.metadata.Hive.createTable(Hive.java:876) ~[hive-e
xec-2.3.9-core.jar:2.3.9]
        at org.apache.spark.sql.hive.client.Shim_v0_12.createTable(HiveShim.scala:614)
~[spark-hive_2.12-3.5.1.jar:3.5.1]
        at org.apache.spark.sql.hive.client.HiveClientImpl.$anonfun$createTable$1(Hive
ClientImpl.scala:586) ~[spark-hive_2.12-3.5.1.jar:3.5.1]
        at scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23) ~[sc
ala-library-2.12.18.jar:?]
```

https://gist.github.com/david-zlai/36f890af8a0caf160c22f3cf68ecb10d

Spark Bigquery Connector

Partitioned Writes

Seems like the connector doesn't support partitioned writes "directly" through the storage API: https://github.com/GoogleCloudDataproc/spark-bigquery-connector?tab=readme-ov-file#properties



Trying: https://github.com/GoogleCloudDataproc/spark-bigquery-connector

Using:

```
Java
libraryDependencies += "com.google.cloud.spark" %%
s"spark-bigquery-with-dependencies" % "0.41.0"
```

But for some reason this isn't working:

There's a .properties file that isn't found by the jar.

```
None
Using warehouse dir: /tmp/chronon/spark-warehouse_caf479

24/12/12 18:03:25 INFO HiveConf: Found configuration file
file:/etc/hive/conf.dist/hive-site.xml
24/12/12 18:03:26 INFO SparkEnv: Registering MapOutputTracker
24/12/12 18:03:26 INFO SparkEnv: Registering BlockManagerMaster
24/12/12 18:03:26 INFO SparkEnv: Registering BlockManagerMasterHeartbeat
24/12/12 18:03:26 INFO SparkEnv: Registering OutputCommitCoordinator
24/12/12 18:03:27 INFO DataprocSparkPlugin: Registered 188 driver metrics
24/12/12 18:03:27 INFO DefaultNoHARMFailoverProxyProvider: Connecting to
ResourceManager at
canary-2-m.us-central1-f.c.canary-443022.internal./10.128.0.53:8032
24/12/12 18:03:27 INFO AHSProxy: Connecting to Application History server at
canary-2-m.us-central1-f.c.canary-443022.internal./10.128.0.53:10200
24/12/12 18:03:28 INFO Configuration: resource-types.xml not found
24/12/12 18:03:28 INFO ResourceUtils: Unable to find 'resource-types.xml'.
24/12/12 18:03:29 INFO YarnClientImpl: Submitted application
application_1733983961532_0009
```

```
24/12/12 18:03:30 INFO DefaultNoHARMFailoverProxyProvider: Connecting to
ResourceManager at
canary-2-m.us-central1-f.c.canary-443022.internal./10.128.0.53:8030
24/12/12 18:03:32 INFO GoogleCloudStorageImpl: Ignoring exception of type
GoogleJsonResponseException; verified object already exists with desired state.
Running join analysis for joins/quickstart.training_set.v1 ... ivysettings.xml
file not found in HIVE_HOME or
HIVE_CONF_DIR,/etc/hive/conf.dist/ivysettings.xml will be used Exception in
thread "main" org.sparkproject.guava.util.concurrent.ExecutionError:
java.util.ServiceConfigurationError:
org.apache.spark.sql.sources.DataSourceRegister: Provider
com.google.cloud.spark.bigquery.BigQueryRelationProvider could not be
instantiated  at
org.sparkproject.guava.cache.LocalCache$Segment.get(LocalCache.java:2053)
at org.sparkproject.guava.cache.LocalCache.get(LocalCache.java:3966)     at
org.sparkproject.guava.cache.LocalCache$LocalManualCache.get(LocalCache.java:48
63)    at
org.apache.spark.sql.catalyst.catalog.SessionCatalog.getCachedPlan(SessionCatal
og.scala:209)       at
org.apache.spark.sql.execution.datasources.FindDataSourceTable.org$apache$spark
$sql$execution$datasources$FindDataSourceTable$$readDataSourceTable(DataSourceS
trategy.scala:277)  at
org.apache.spark.sql.execution.datasources.FindDataSourceTable$$anonfun$apply$2
.applyOrElse(DataSourceStrategy.scala:323)     at
org.apache.spark.sql.execution.datasources.FindDataSourceTable$$anonfun$apply$2
.applyOrElse(DataSourceStrategy.scala:312)     at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.$anonfun$resolveOper
atorsDownWithPruning$2(AnalysisHelper.scala:170)     at
org.apache.spark.sql.catalyst.trees.CurrentOrigin$.withOrigin(origin.scala:76)
at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.$anonfun$resolveOper
atorsDownWithPruning$1(AnalysisHelper.scala:170)     at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper$.allowInvokingTransf
ormsInAnalyzer(AnalysisHelper.scala:323)       at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.resolveOperatorsDown
WithPruning(AnalysisHelper.scala:168)   at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.resolveOperatorsDown
WithPruning$(AnalysisHelper.scala:164)  at
org.apache.spark.sql.catalyst.plans.logical.LogicalPlan.resolveOperatorsDownWit
hPruning(LogicalPlan.scala:32)    at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.$anonfun$resolveOper
atorsDownWithPruning$4(AnalysisHelper.scala:175)     at
org.apache.spark.sql.catalyst.trees.UnaryLike.mapChildren(TreeNode.scala:1227)
at
```

```
org.apache.spark.sql.catalyst.trees.UnaryLike.mapChildren$(TreeNode.scala:1226)
      at
org.apache.spark.sql.catalyst.plans.logical.SubqueryAlias.mapChildren(basicLogi
calOperators.scala:1676)   at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.$anonfun$resolveOper
atorsDownWithPruning$1(AnalysisHelper.scala:175)     at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper$.allowInvokingTransf
ormsInAnalyzer(AnalysisHelper.scala:323)      at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.resolveOperatorsDown
WithPruning(AnalysisHelper.scala:168)   at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.resolveOperatorsDown
WithPruning$(AnalysisHelper.scala:164)  at
org.apache.spark.sql.catalyst.plans.logical.LogicalPlan.resolveOperatorsDownWit
hPruning(LogicalPlan.scala:32)   at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.resolveOperatorsWith
Pruning(AnalysisHelper.scala:99) at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.resolveOperatorsWith
Pruning$(AnalysisHelper.scala:96)       at
org.apache.spark.sql.catalyst.plans.logical.LogicalPlan.resolveOperatorsWithPru
ning(LogicalPlan.scala:32)       at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.resolveOperators(Ana
lysisHelper.scala:76)     at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.resolveOperators$(An
alysisHelper.scala:75)    at
org.apache.spark.sql.catalyst.plans.logical.LogicalPlan.resolveOperators(Logica
lPlan.scala:32)     at
org.apache.spark.sql.execution.datasources.FindDataSourceTable.apply(DataSource
Strategy.scala:312)       at
org.apache.spark.sql.execution.datasources.FindDataSourceTable.apply(DataSource
Strategy.scala:242)       at
org.apache.spark.sql.catalyst.rules.RuleExecutor.$anonfun$execute$2(RuleExecuto
r.scala:222) at
scala.collection.LinearSeqOptimized.foldLeft(LinearSeqOptimized.scala:126)
at scala.collection.LinearSeqOptimized.foldLeft$(LinearSeqOptimized.scala:122)
at scala.collection.immutable.List.foldLeft(List.scala:91)  at
org.apache.spark.sql.catalyst.rules.RuleExecutor.$anonfun$execute$1(RuleExecuto
r.scala:219) at
org.apache.spark.sql.catalyst.rules.RuleExecutor.$anonfun$execute$1$adapted(Rul
eExecutor.scala:211)      at
scala.collection.immutable.List.foreach(List.scala:431)     at
org.apache.spark.sql.catalyst.rules.RuleExecutor.execute(RuleExecutor.scala:211
)     at
org.apache.spark.sql.catalyst.analysis.Analyzer.org$apache$spark$sql$catalyst$a
nalysis$Analyzer$$executeSameContext(Analyzer.scala:226)    at
```

```
org.apache.spark.sql.catalyst.analysis.Analyzer.$anonfun$execute$1(Analyzer.sca
la:222)      at
org.apache.spark.sql.catalyst.analysis.AnalysisContext$.withNewAnalysisContext(
Analyzer.scala:173)       at
org.apache.spark.sql.catalyst.analysis.Analyzer.execute(Analyzer.scala:222)
at org.apache.spark.sql.catalyst.analysis.Analyzer.execute(Analyzer.scala:188)
at
org.apache.spark.sql.catalyst.rules.RuleExecutor.$anonfun$executeAndTrack$1(Rul
eExecutor.scala:182)       at
org.apache.spark.sql.catalyst.QueryPlanningTracker$.withTracker(QueryPlanningTr
acker.scala:89)     at
org.apache.spark.sql.catalyst.rules.RuleExecutor.executeAndTrack(RuleExecutor.s
cala:182)    at
org.apache.spark.sql.catalyst.analysis.Analyzer.$anonfun$executeAndCheck$1(Anal
yzer.scala:209)     at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper$.markInAnalyzer(Anal
ysisHelper.scala:330)      at
org.apache.spark.sql.catalyst.analysis.Analyzer.executeAndCheck(Analyzer.scala:
208)   at
org.apache.spark.sql.execution.QueryExecution.$anonfun$analyzed$1(QueryExecutio
n.scala:77)  at
org.apache.spark.sql.catalyst.QueryPlanningTracker.measurePhase(QueryPlanningTr
acker.scala:138)    at
org.apache.spark.sql.execution.QueryExecution.$anonfun$executePhase$2(QueryExec
ution.scala:219)    at
org.apache.spark.sql.execution.QueryExecution$.withInternalError(QueryExecution
.scala:547)  at
org.apache.spark.sql.execution.QueryExecution.$anonfun$executePhase$1(QueryExec
ution.scala:219)    at
org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:900)     at
org.apache.spark.sql.execution.QueryExecution.executePhase(QueryExecution.scala
:218)  at
org.apache.spark.sql.execution.QueryExecution.analyzed$lzycompute(QueryExecutio
n.scala:77)  at
org.apache.spark.sql.execution.QueryExecution.analyzed(QueryExecution.scala:74)
      at
org.apache.spark.sql.execution.QueryExecution.assertAnalyzed(QueryExecution.sca
la:66)       at
org.apache.spark.sql.Dataset$.$anonfun$ofRows$1(Dataset.scala:91)  at
org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:900)     at
org.apache.spark.sql.Dataset$.ofRows(Dataset.scala:89)      at
org.apache.spark.sql.DataFrameReader.table(DataFrameReader.scala:608)    at
org.apache.spark.sql.SparkSession.table(SparkSession.scala:602)    at
ai.chronon.spark.Extensions$DataPointerOps.toDf(Extensions.scala:367)    at
```

```
ai.chronon.spark.TableUtils.scanDfBase(TableUtils.scala:1017)      at
ai.chronon.spark.TableUtils.scanDf(TableUtils.scala:1051)   at
ai.chronon.spark.Analyzer.analyzeJoin(Analyzer.scala:288)   at
ai.chronon.spark.JoinBase.computeJoinOpt(JoinBase.scala:434)       at
ai.chronon.spark.JoinBase.computeJoin(JoinBase.scala:415)   at
ai.chronon.spark.Driver$JoinBackfill$.run(Driver.scala:296)        at
ai.chronon.spark.Driver$.main(Driver.scala:953)      at
ai.chronon.spark.Driver.main(Driver.scala)     at
java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at
java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAcce
ssorImpl.java:62)   at
java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMe
thodAccessorImpl.java:43)  at
java.base/java.lang.reflect.Method.invoke(Method.java:566)  at
org.apache.spark.deploy.JavaMainApplication.start(SparkApplication.scala:52)
at
org.apache.spark.deploy.SparkSubmit.org$apache$spark$deploy$SparkSubmit$$runMai
n(SparkSubmit.scala:1032)  at
org.apache.spark.deploy.SparkSubmit.doRunMain$1(SparkSubmit.scala:194)    at
org.apache.spark.deploy.SparkSubmit.submit(SparkSubmit.scala:217)  at
org.apache.spark.deploy.SparkSubmit.doSubmit(SparkSubmit.scala:91)        at
org.apache.spark.deploy.SparkSubmit$$anon$2.doSubmit(SparkSubmit.scala:1124)
at org.apache.spark.deploy.SparkSubmit$.main(SparkSubmit.scala:1133)      at
org.apache.spark.deploy.SparkSubmit.main(SparkSubmit.scala) Caused by:
java.util.ServiceConfigurationError:
org.apache.spark.sql.sources.DataSourceRegister: Provider
com.google.cloud.spark.bigquery.BigQueryRelationProvider could not be
instantiated  at java.base/java.util.ServiceLoader.fail(ServiceLoader.java:582)
at
java.base/java.util.ServiceLoader$ProviderImpl.newInstance(ServiceLoader.java:8
04)    at
java.base/java.util.ServiceLoader$ProviderImpl.get(ServiceLoader.java:722)
at java.base/java.util.ServiceLoader$3.next(ServiceLoader.java:1395)      at
scala.collection.convert.Wrappers$JIteratorWrapper.next(Wrappers.scala:46)
at scala.collection.Iterator.foreach(Iterator.scala:943)    at
scala.collection.Iterator.foreach$(Iterator.scala:943)      at
scala.collection.AbstractIterator.foreach(Iterator.scala:1431)     at
scala.collection.IterableLike.foreach(IterableLike.scala:74)       at
scala.collection.IterableLike.foreach$(IterableLike.scala:73)      at
scala.collection.AbstractIterable.foreach(Iterable.scala:56)       at
scala.collection.TraversableLike.filterImpl(TraversableLike.scala:303)    at
scala.collection.TraversableLike.filterImpl$(TraversableLike.scala:297)   at
scala.collection.AbstractTraversable.filterImpl(Traversable.scala:108)    at
```

```
scala.collection.TraversableLike.filter(TraversableLike.scala:395)       at
scala.collection.TraversableLike.filter$(TraversableLike.scala:395)      at
scala.collection.AbstractTraversable.filter(Traversable.scala:108)       at
org.apache.spark.sql.execution.datasources.DataSource$.lookupDataSource(DataSou
rce.scala:629)      at
org.apache.spark.sql.execution.datasources.DataSource.providingClass$lzycompute
(DataSource.scala:100)     at
org.apache.spark.sql.execution.datasources.DataSource.providingClass(DataSource
.scala:99)    at
org.apache.spark.sql.execution.datasources.DataSource.providingInstance(DataSou
rce.scala:113)      at
org.apache.spark.sql.execution.datasources.DataSource.resolveRelation(DataSourc
e.scala:341)  at
org.apache.spark.sql.execution.datasources.FindDataSourceTable.$anonfun$readDat
aSourceTable$1(DataSourceStrategy.scala:293)   at
org.sparkproject.guava.cache.LocalCache$LocalManualCache$1.load(LocalCache.java
:4868)       at
org.sparkproject.guava.cache.LocalCache$LoadingValueReference.loadFuture(LocalC
ache.java:3533)     at
org.sparkproject.guava.cache.LocalCache$Segment.loadSync(LocalCache.java:2282)
at
org.sparkproject.guava.cache.LocalCache$Segment.lockedGetOrLoad(LocalCache.java
:2159)       at
org.sparkproject.guava.cache.LocalCache$Segment.get(LocalCache.java:2049)
... 85 more Caused by: java.lang.IllegalStateException: This connector was made
for Scala null, it was not meant to run on Scala 2.12       at
com.google.cloud.spark.bigquery.BigQueryUtilScala$.validateScalaVersionCompatib
ility(BigQueryUtil.scala:37)     at
com.google.cloud.spark.bigquery.BigQueryRelationProvider.<init>(BigQueryRelatio
nProvider.scala:42)       at
com.google.cloud.spark.bigquery.BigQueryRelationProvider.<init>(BigQueryRelatio
nProvider.scala:49)       at
java.base/jdk.internal.reflect.NativeConstructorAccessorImpl.newInstance0(Nativ
e Method)    at
java.base/jdk.internal.reflect.NativeConstructorAccessorImpl.newInstance(Native
ConstructorAccessorImpl.java:62) at
java.base/jdk.internal.reflect.DelegatingConstructorAccessorImpl.newInstance(De
legatingConstructorAccessorImpl.java:45)      at
java.base/java.lang.reflect.Constructor.newInstance(Constructor.java:490)
at
java.base/java.util.ServiceLoader$ProviderImpl.newInstance(ServiceLoader.java:7
80)    ... 111 more
24/12/12 18:03:38 ERROR TransportRequestHandler: Error while invoking
RpcHandler#receive() for one-way message. org.apache.spark.SparkException:
```

```
Could not find CoarseGrainedScheduler.  at
org.apache.spark.rpc.netty.Dispatcher.postMessage(Dispatcher.scala:178)
~[spark-core_2.12-3.5.1.jar:3.5.1]      at
org.apache.spark.rpc.netty.Dispatcher.postOneWayMessage(Dispatcher.scala:150)
~[spark-core_2.12-3.5.1.jar:3.5.1]      at
org.apache.spark.rpc.netty.NettyRpcHandler.receive(NettyRpcEnv.scala:690)
~[spark-core_2.12-3.5.1.jar:3.5.1]      at
org.apache.spark.network.server.TransportRequestHandler.processOneWayMessage(Tr
ansportRequestHandler.java:274) [spark-network-common_2.12-3.5.1.jar:3.5.1]
at
org.apache.spark.network.server.TransportRequestHandler.handle(TransportRequest
Handler.java:111) [spark-network-common_2.12-3.5.1.jar:3.5.1]      at
org.apache.spark.network.server.TransportChannelHandler.channelRead0(TransportC
hannelHandler.java:140) [spark-network-common_2.12-3.5.1.jar:3.5.1]      at
org.apache.spark.network.server.TransportChannelHandler.channelRead0(TransportC
hannelHandler.java:53) [spark-network-common_2.12-3.5.1.jar:3.5.1]       at
io.netty.channel.SimpleChannelInboundHandler.channelRead(SimpleChannelInboundHa
ndler.java:99) [netty-transport-4.1.100.Final.jar:4.1.100.Final]   at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChanne
lHandlerContext.java:444) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChanne
lHandlerContext.java:420) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelH
andlerContext.java:412) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.handler.timeout.IdleStateHandler.channelRead(IdleStateHandler.java:286
) [netty-handler-4.1.100.Final.jar:4.1.100.Final]    at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChanne
lHandlerContext.java:442) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChanne
lHandlerContext.java:420) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelH
andlerContext.java:412) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.handler.codec.MessageToMessageDecoder.channelRead(MessageToMessageDeco
der.java:103) [netty-codec-4.1.100.Final.jar:4.1.100.Final]       at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChanne
lHandlerContext.java:444) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChanne
```

```
lHandlerContext.java:420) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelH
andlerContext.java:412) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
org.apache.spark.network.util.TransportFrameDecoder.channelRead(TransportFrameD
ecoder.java:102) [spark-network-common_2.12-3.5.1.jar:3.5.1]       at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChanne
lHandlerContext.java:444) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChanne
lHandlerContext.java:420) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelH
andlerContext.java:412) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.DefaultChannelPipeline$HeadContext.channelRead(DefaultChannelP
ipeline.java:1410) [netty-transport-4.1.100.Final.jar:4.1.100.Final]     at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChanne
lHandlerContext.java:440) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChanne
lHandlerContext.java:420) [netty-transport-4.1.100.Final.jar:4.1.100.Final]
at
io.netty.channel.DefaultChannelPipeline.fireChannelRead(DefaultChannelPipeline.
java:919) [netty-transport-4.1.100.Final.jar:4.1.100.Final]       at
io.netty.channel.nio.AbstractNioByteChannel$NioByteUnsafe.read(AbstractNioByteC
hannel.java:166) [netty-transport-4.1.100.Final.jar:4.1.100.Final]       at
io.netty.channel.nio.NioEventLoop.processSelectedKey(NioEventLoop.java:788)
[netty-transport-4.1.100.Final.jar:4.1.100.Final]    at
io.netty.channel.nio.NioEventLoop.processSelectedKeysOptimized(NioEventLoop.jav
a:724) [netty-transport-4.1.100.Final.jar:4.1.100.Final]    at
io.netty.channel.nio.NioEventLoop.processSelectedKeys(NioEventLoop.java:650)
[netty-transport-4.1.100.Final.jar:4.1.100.Final]    at
io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:562)
[netty-transport-4.1.100.Final.jar:4.1.100.Final]    at
io.netty.util.concurrent.SingleThreadEventExecutor$4.run(SingleThreadEventExecu
tor.java:997) [netty-common-4.1.100.Final.jar:4.1.100.Final]       at
io.netty.util.internal.ThreadExecutorMap$2.run(ThreadExecutorMap.java:74)
[netty-common-4.1.100.Final.jar:4.1.100.Final]       at
io.netty.util.concurrent.FastThreadLocalRunnable.run(FastThreadLocalRunnable.ja
va:30) [netty-common-4.1.100.Final.jar:4.1.100.Final]       at
java.base/java.lang.Thread.run(Thread.java:829) [?:?]
```

Including it in our jar doesn't work, excluding it and separately specifying it using `--jars` doesn't work either. Both show the above exception. Are we using the right jar?

Some historical reports:
https://github.com/GoogleCloudDataproc/spark-bigquery-connector/issues?q=This+connector+was+made+for+Scala+null

Gonna try bare-bones straight on the master node:

```
None
spark-submit \
--class ai.chronon.spark.Driver \
--files training_set.v1 \
cloud_gcp-assembly-0.1.0-SNAPSHOT.jar \
join --end-date=2024-12-10 --conf-path=training_set.v1
```

Storage API

The Spark BigQuery Connector uses BigQuery's Storage API (SO) however external tables aren't supported in the Storage API. Therefore, the spark bigquery connector cannot query BQ *external* tables.

```
None
24/12/13 05:15:11 ERROR Join: An unexpected error occurred during validation.
com.google.cloud.spark.bigquery.repackaged.io.grpc.StatusRuntimeException:
INVALID_ARGUMENT: request failed: Only external tables with connections can be
read with the Storage API.
```

Two options:

1. Just pass the prefix (might require DataPointer)
2. Use bigLake: https://cloud.google.com/bigquery/docs/reference/storage#limitations

HMS first, BQ fallback

Some errors encountered:

```Java
24/12/13 05:31:07 ERROR Join: An unexpected error occurred during validation.
Fail to resolve data source for the table
`spark_catalog`.`data`.`checkouts_native` since the table serde property has
the duplicated key table with extra options specified for this scan operation.
To fix this, you can rollback to the legacy behavior of ignoring the extra
options by setting the config spark.sql.legacy.extraOptionsBehavior.enabled to
`false`, or address the conflicts of the same config.
```

```None
ivysettings.xml file not found in HIVE_HOME or
HIVE_CONF_DIR,/etc/hive/conf.dist/ivysettings.xml will be used
24/12/13 01:02:18 ERROR Join: An unexpected error occurred during validation.
com.google.cloud.spark.bigquery.repackaged.com.google.inject.ProvisionException
: Unable to provision, see the following errors:

1) [Guice/ErrorInCustomProvider]: IllegalArgumentException: No table has been
specified
  at
SparkBigQueryConnectorModule.provideSparkBigQueryConfig(SparkBigQueryConnectorM
odule.java:102)
  while locating SparkBigQueryConfig

Learn more:
  https://github.com/google/guice/wiki/ERROR_IN_CUSTOM_PROVIDER

1 error

======================
Full classname legend:
======================
SparkBigQueryConfig:
"com.google.cloud.spark.bigquery.SparkBigQueryConfig"
SparkBigQueryConnectorModule:
"com.google.cloud.spark.bigquery.SparkBigQueryConnectorModule"
========================
End of classname legend:
========================
```

Dammit, the federation service allows us to connect to BQ but the BQ tables are treated as external tables (should've guessed):

```
None
CREATE EXTERNAL TABLE `checkouts`(
  `ds` date COMMENT '',
  `ts` bigint COMMENT '',
  `return_id` string COMMENT '',
  `user_id` bigint COMMENT '',
  `product_id` bigint COMMENT '',
  `refund_amt` bigint COMMENT '')
PARTITIONED BY (
  `ds` timestamp COMMENT '')
ROW FORMAT SERDE
  'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'
WITH SERDEPROPERTIES (
  'dataset'='data',
  'project'='canary-443022',
  'table'='checkouts')
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  '/data'
TBLPROPERTIES (
  'FEDERATION_BIGQUERY_TABLE_PROPERTY'='canary-443022.data.checkouts',
  'federation.bigquery.table.type'='MANAGED',
  'spark.sql.sources.provider'='com.google.cloud.spark.bigquery')
```

```
None
scala> spark.catalog.listTables("data").show
+-----------------+-------------+---------+-----------+---------+-----------+
|             name|      catalog|namespace|description|tableType|isTemporary|
+-----------------+-------------+---------+-----------+---------+-----------+
|        checkouts|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
```

```
| checkouts_native|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
|checkouts_parquet|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
|        purchases|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
| purchases_native|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
|purchases_parquet|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
|          returns|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
|   returns_native|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
|  returns_parquet|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
|            users|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
|     users_native|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
|    users_parquet|spark_catalog|   [data]|       NULL| EXTERNAL|      false|
+-----------------+-------------+---------+-----------+---------+-----------+



scala> spark.sql("describe formatted data.checkouts").show(truncate=false)
+----------------------------+----------------------------------------------------
-----------------------------------------------------------+-------+
|col_name                    |data_type
|comment|
+----------------------------+----------------------------------------------------
-----------------------------------------------------------+-------+
|ds                          |date
|NULL   |
|ts                          |bigint
|NULL   |
|return_id                   |string
|NULL   |
|user_id                     |bigint
|NULL   |
|product_id                  |bigint
|NULL   |
|refund_amt                  |bigint
|NULL   |
|                            |
|       |
|# Detailed Table Information|
|       |
|Catalog                     |spark_catalog
|       |
|Database                    |data
|       |
|Table                       |checkouts
|       |
```

```
|Created Time                |Fri Dec 13 21:28:44 UTC 2024
|       |
|Last Access                 |Fri Dec 13 23:07:38 UTC 2024
|       |
|Created By                  |Spark 2.2 or prior
|       |
|Type                        |EXTERNAL
|       |
|Provider                    |com.google.cloud.spark.bigquery
|       |
|Table Properties
|[FEDERATION_BIGQUERY_TABLE_PROPERTY=canary-443022.data.checkouts,
federation.bigquery.table.type=MANAGED]|       |
|Serde Library
|org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
|       |
|InputFormat                 |org.apache.hadoop.mapred.TextInputFormat
|       |
|OutputFormat
|org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
|       |
+----------------------------+--------------------------------------------------
----------------------------------------------------------+-------+
only showing top 20 rows
```

None

```
scala> val ct = spark.sessionState.catalog.getTableMetadata(ti)
ct: org.apache.spark.sql.catalyst.catalog.CatalogTable =
CatalogTable(
Catalog: spark_catalog
Database: data
Table: checkouts
Created Time: Fri Dec 13 21:28:44 UTC 2024
Last Access: Fri Dec 13 23:07:38 UTC 2024
Created By: Spark 2.2 or prior
Type: EXTERNAL
Provider: com.google.cloud.spark.bigquery
Table Properties:
[FEDERATION_BIGQUERY_TABLE_PROPERTY=canary-443022.data.checkouts,
federation.bigquery.table.type=MANAGED]
Serde Library: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
```

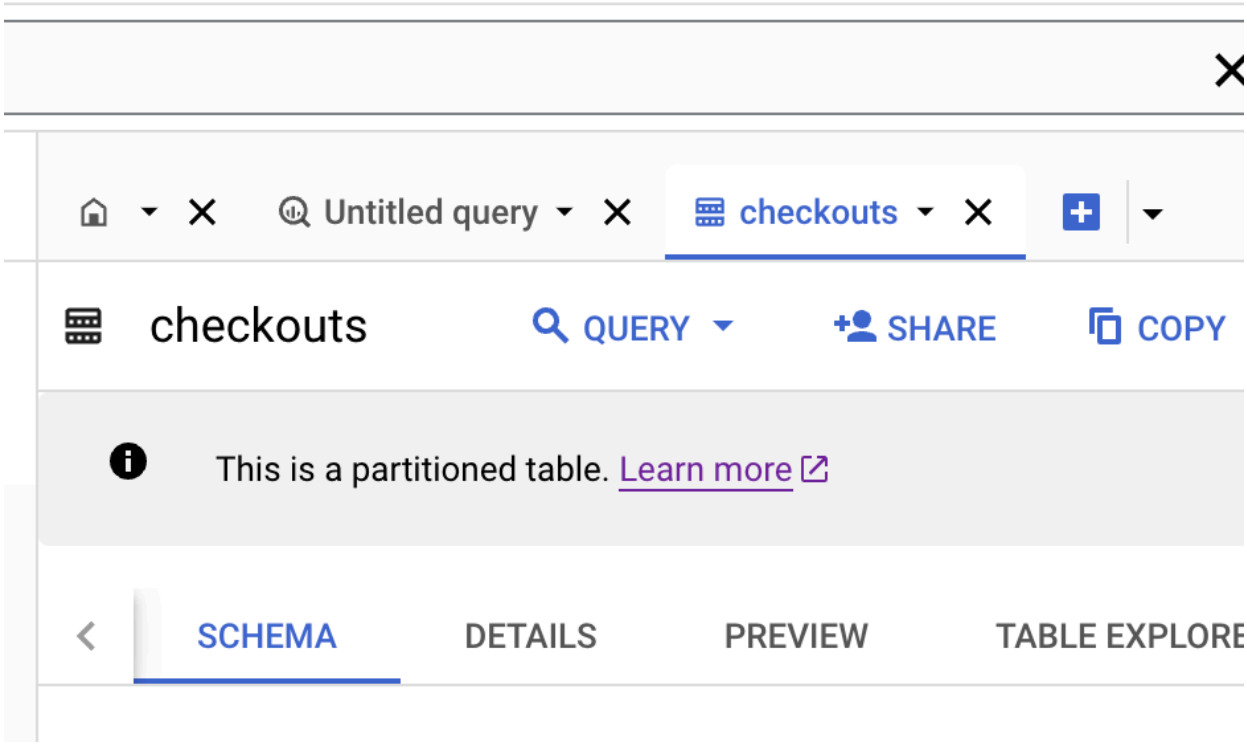
```
InputFormat: org.apache.hadoop.mapred.TextInputFormat
OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Storage Properties: [project=canary-443022, dataset=data, table=checkouts])

scala> spark.version
res20: String = 3.5.1

scala> ct.partitionColumnNames
res21: Seq[String] = List()
```

BigQuery on Iceberg

https://cloud.google.com/bigquery/docs/iceberg-tables#create-iceberg-tables

Why are partitioned tables on Dataproc federated metastore not partitioned but the underlying BigQuery table is?

Table is partitioned in BigQuery



Tried running the join job:
https://console.cloud.google.com/dataproc/jobs/b03d6419-4923-4a01-ba6f-98a85c40be6f/configuration?region=us-central1&project=canary-443022

```
None
Exception in thread "main" org.apache.spark.sql.AnalysisException:
[INVALID_PARTITION_OPERATION.PARTITION_SCHEMA_IS_EMPTY] The partition command
is invalid. Table `spark_catalog`.`data`.`checkouts` is not partitioned.
```

Opened up a Jupyter notebook and we can run some spark sql commands to inspect:

```
In [20]: results = spark.sql("show create table data.checkouts")
         results.show(truncate=False)

+-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------+
|createtab_stmt
|
+-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------+
|CREATE TABLE spark_catalog.data.checkouts (\n  )\nUSING com.google.cloud.spark.bigquery\nOPTIONS (\n  'dataset' =
'data',\n  'project' = 'canary-443022',\n  'table' = 'checkouts')\nTBLPROPERTIES (\n  'FEDERATION_BIGQUERY_TABLE_PR
OPERTY' = 'canary-443022.data.checkouts',\n  'federation.bigquery.table.type' = 'MANAGED')\n|
+-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------+
```

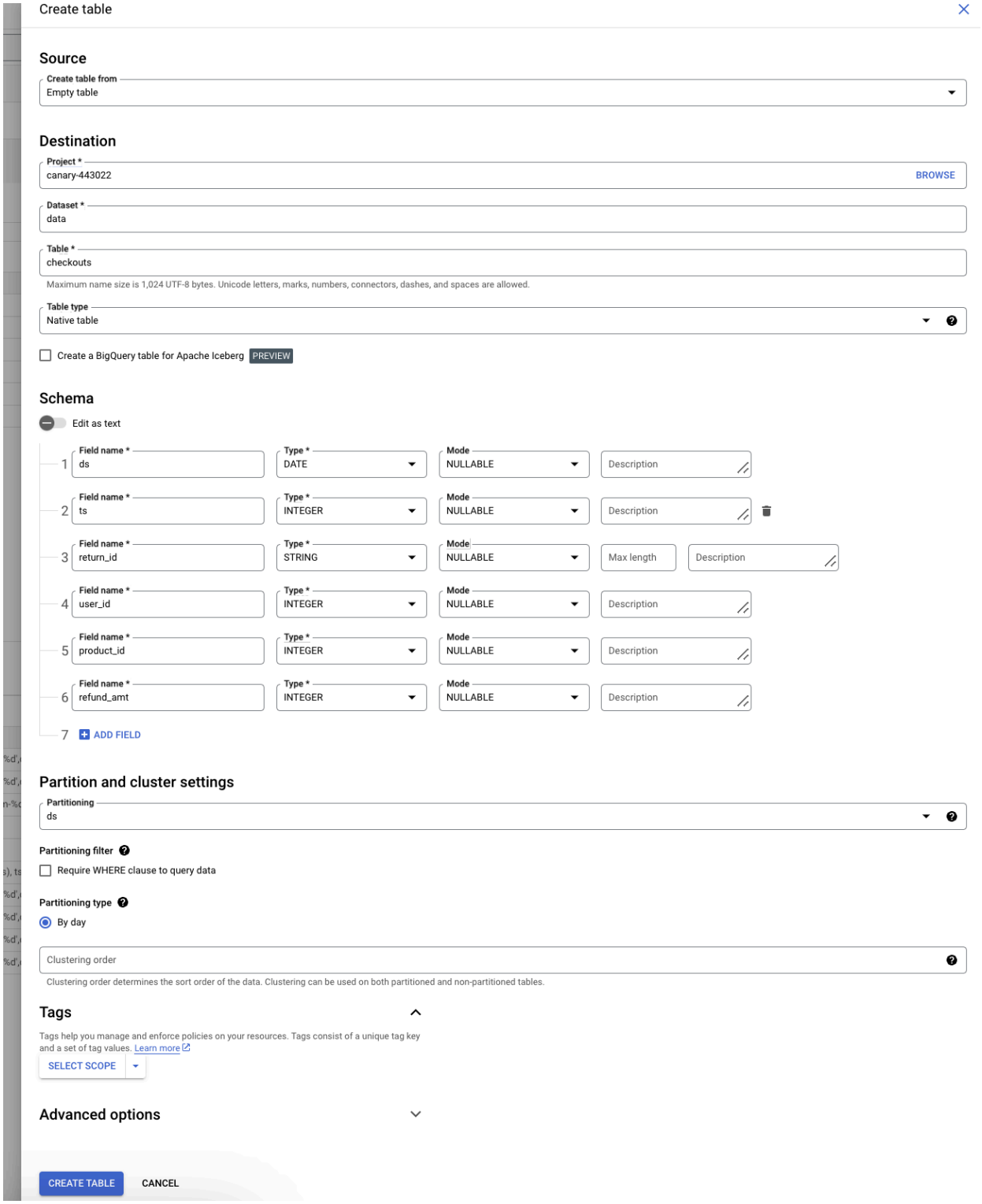
```
In [21]: TION_BIGQUERY_TABLE_PROPERTY' = 'canary-443022.data.checkouts',\n  'federation.bigquery.table.type' = 'MANAGED')\n")

CREATE TABLE spark_catalog.data.checkouts (
  )
USING com.google.cloud.spark.bigquery
OPTIONS (
  'dataset' = 'data',
  'project' = 'canary-443022',
  'table' = 'checkouts')
TBLPROPERTIES (
  'FEDERATION_BIGQUERY_TABLE_PROPERTY' = 'canary-443022.data.checkouts',
  'federation.bigquery.table.type' = 'MANAGED')
```

None

```
CREATE TABLE spark_catalog.data.checkouts (
  )
USING com.google.cloud.spark.bigquery
OPTIONS (
  'dataset' = 'data',
  'project' = 'canary-443022',
  'table' = 'checkouts')
TBLPROPERTIES (
  'FEDERATION_BIGQUERY_TABLE_PROPERTY' =
'canary-443022.data.checkouts',
  'federation.bigquery.table.type' = 'MANAGED')
```

Setting up BigQuery native partitioned tables from Quickstart

Create the table as an `Empty Table` first but mark `ds` as a `DATE` type. In `Partition and cluster settings`, choose `ds` as the partition column.



Then insert into the table with some modified sql like so (uncomment the one you need):

```
None
-- insert into `canary-443022.data.checkouts`
-- (ds, ts, return_id, user_id, product_id, refund_amt)
-- select parse_date('%Y-%m-%d',ds), ts, return_id, user_id, product_id,
refund_amt from `canary-443022.data.checkouts_parquet`


-- insert into `canary-443022.data.purchases`
-- (ds, ts, purchase_id, user_id, product_id, purchase_price)
-- select parse_date('%Y-%m-%d',ds), ts, purchase_id, user_id, product_id,
purchase_price from `canary-443022.data.purchases_parquet`


-- insert into `canary-443022.data.returns`
-- (ds, ts, return_id, user_id, product_id, refund_amt)
-- select parse_date('%Y-%m-%d',ds), ts, return_id, user_id, product_id,
refund_amt from `canary-443022.data.returns_parquet`


-- insert into `canary-443022.data.users`
-- (ds, user_id, account_created_ds, email_verified)
-- select parse_date('%Y-%m-%d',ds), user_id, account_created_ds,
email_verified from `canary-443022.data.users_parquet`
```

A series of exception traces

```
Java
Exception in thread "main" java.lang.ClassCastException: class java.sql.Date
cannot be cast to class java.lang.String (java.sql.Date is in module java.sql
of loader 'platform'; java.lang.String is in module java.base of loader
'bootstrap')
       at org.apache.spark.sql.Row.getString(Row.scala:277)
       at org.apache.spark.sql.Row.getString$(Row.scala:277)
       at
org.apache.spark.sql.catalyst.expressions.GenericRow.getString(rows.scala:27)
       at
ai.chronon.spark.Extensions$DfWithStats$.$anonfun$apply$1(Extensions.scala:93)
       at
scala.collection.TraversableLike.$anonfun$map$1(TraversableLike.scala:286)
       at
scala.collection.IndexedSeqOptimized.foreach(IndexedSeqOptimized.scala:36)
```

```
	at
scala.collection.IndexedSeqOptimized.foreach$(IndexedSeqOptimized.scala:33)
	at scala.collection.mutable.ArrayOps$ofRef.foreach(ArrayOps.scala:198)
	at scala.collection.TraversableLike.map(TraversableLike.scala:286)
	at scala.collection.TraversableLike.map$(TraversableLike.scala:279)
	at scala.collection.mutable.ArrayOps$ofRef.map(ArrayOps.scala:198)
	at ai.chronon.spark.Extensions$DfWithStats$.apply(Extensions.scala:93)
	at
ai.chronon.spark.Extensions$DataframeOps.withStats(Extensions.scala:126)
	at ai.chronon.spark.Join.computeRange(Join.scala:261)
	at
ai.chronon.spark.JoinBase.$anonfun$computeJoinOpt$15(JoinBase.scala:530)
	at
ai.chronon.spark.JoinBase.$anonfun$computeJoinOpt$15$adapted(JoinBase.scala:527
)
	at scala.Option.map(Option.scala:230)
	at
ai.chronon.spark.JoinBase.$anonfun$computeJoinOpt$14(JoinBase.scala:527)
	at scala.collection.immutable.List.foreach(List.scala:431)
	at ai.chronon.spark.JoinBase.computeJoinOpt(JoinBase.scala:522)
	at ai.chronon.spark.JoinBase.computeJoin(JoinBase.scala:415)
	at ai.chronon.spark.Driver$JoinBackfill$.run(Driver.scala:313)
	at ai.chronon.spark.Driver$.main(Driver.scala:970)
	at ai.chronon.spark.Driver.main(Driver.scala)
	at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native
Method)
	at
java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAcce
ssorImpl.java:62)
	at
java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMe
thodAccessorImpl.java:43)
	at java.base/java.lang.reflect.Method.invoke(Method.java:566)
	at
org.apache.spark.deploy.JavaMainApplication.start(SparkApplication.scala:52)
	at
org.apache.spark.deploy.SparkSubmit.org$apache$spark$deploy$SparkSubmit$$runMai
n(SparkSubmit.scala:1032)
	at org.apache.spark.deploy.SparkSubmit.doRunMain$1(SparkSubmit.scala:194)
	at org.apache.spark.deploy.SparkSubmit.submit(SparkSubmit.scala:217)
	at org.apache.spark.deploy.SparkSubmit.doSubmit(SparkSubmit.scala:91)
	at
org.apache.spark.deploy.SparkSubmit$$anon$2.doSubmit(SparkSubmit.scala:1124)
	at org.apache.spark.deploy.SparkSubmit$.main(SparkSubmit.scala:1133)
```

```
	at org.apache.spark.deploy.SparkSubmit.main(SparkSubmit.scala)
```

None

```
Caused by:
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.jso
n.GoogleJsonResponseException: 400 Bad Request
POST
https://bigquery.googleapis.com/bigquery/v2/projects/canary-443022/jobs?prettyP
rint=false
{
  "code": 400,
  "errors": [
    {
      "domain": "global",
      "message": "The field specified for time partitioning can only be of type
TIMESTAMP, DATE or DATETIME. The type found is: STRING.",
      "reason": "invalid"
    }
  ],
  "message": "The field specified for time partitioning can only be of type
TIMESTAMP, DATE or DATETIME. The type found is: STRING.",
  "status": "INVALID_ARGUMENT"
}
	at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.jso
n.GoogleJsonResponseException.from(GoogleJsonResponseException.java:146)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.json.AbstractGoogleJsonClientRequest.newExceptionOnError(AbstractGoogleJs
onClientRequest.java:118)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.json.AbstractGoogleJsonClientRequest.newExceptionOnError(AbstractGoogleJs
onClientRequest.java:37)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.AbstractGoogleClientRequest$3.interceptResponse(AbstractGoogleClientReque
st.java:479)
```

```
	at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.http.HttpReque
st.execute(HttpRequest.java:1111)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.AbstractGoogleClientRequest.executeUnparsed(AbstractGoogleClientRequest.j
ava:565)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.AbstractGoogleClientRequest.executeUnparsed(AbstractGoogleClientRequest.j
ava:506)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.AbstractGoogleClientRequest.execute(AbstractGoogleClientRequest.java:616)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.spi.v2.Htt
pBigQueryRpc.create(HttpBigQueryRpc.java:235)
	... 72 more
2025-01-04 23:34:46 INFO  TableUtils:396 - Cleared the dataframe cache after
Computing left parts for bootstrap table - start @ 2025-01-04 23:34:07 end @
2025-01-04 23:34:46
Exception in thread "Join-2023-11-30-2023-11-30"
com.google.cloud.bigquery.connector.common.BigQueryConnectorException: Failed
to write to BigQuery
	at
com.google.cloud.spark.bigquery.write.BigQueryWriteHelper.writeDataFrameToBigQu
ery(BigQueryWriteHelper.java:157)
	at
com.google.cloud.spark.bigquery.write.BigQueryDeprecatedIndirectInsertableRelat
ion.insert(BigQueryDeprecatedIndirectInsertableRelation.java:43)
	at
com.google.cloud.spark.bigquery.write.CreatableRelationProviderHelper.createRel
ation(CreatableRelationProviderHelper.java:54)
	at
com.google.cloud.spark.bigquery.BigQueryRelationProvider.createRelation(BigQuer
yRelationProvider.scala:107)
	at
org.apache.spark.sql.execution.datasources.SaveIntoDataSourceCommand.run(SaveIn
toDataSourceCommand.scala:48)
	at
org.apache.spark.sql.execution.command.ExecutedCommandExec.sideEffectResult$lzy
compute(commands.scala:75)
```

```
	at
org.apache.spark.sql.execution.command.ExecutedCommandExec.sideEffectResult(com
mands.scala:73)
	at
org.apache.spark.sql.execution.command.ExecutedCommandExec.executeCollect(comma
nds.scala:84)
	at
org.apache.spark.sql.execution.QueryExecution$$anonfun$eagerlyExecuteCommands$1
.$anonfun$applyOrElse$1(QueryExecution.scala:107)
	at
org.apache.spark.sql.execution.SQLExecution$.$anonfun$withNewExecutionId$6(SQLE
xecution.scala:125)
	at
org.apache.spark.sql.execution.SQLExecution$.withSQLConfPropagated(SQLExecution
.scala:201)
	at
org.apache.spark.sql.execution.SQLExecution$.$anonfun$withNewExecutionId$1(SQLE
xecution.scala:108)
	at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:900)
	at
org.apache.spark.sql.execution.SQLExecution$.withNewExecutionId(SQLExecution.sc
ala:66)
	at
org.apache.spark.sql.execution.QueryExecution$$anonfun$eagerlyExecuteCommands$1
.applyOrElse(QueryExecution.scala:107)
	at
org.apache.spark.sql.execution.QueryExecution$$anonfun$eagerlyExecuteCommands$1
.applyOrElse(QueryExecution.scala:98)
	at
org.apache.spark.sql.catalyst.trees.TreeNode.$anonfun$transformDownWithPruning$
1(TreeNode.scala:473)
	at
org.apache.spark.sql.catalyst.trees.CurrentOrigin$.withOrigin(origin.scala:76)
	at
org.apache.spark.sql.catalyst.trees.TreeNode.transformDownWithPruning(TreeNode.
scala:473)
	at
org.apache.spark.sql.catalyst.plans.logical.LogicalPlan.org$apache$spark$sql$ca
talyst$plans$logical$AnalysisHelper$$super$transformDownWithPruning(LogicalPlan
.scala:32)
	at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.transformDownWithPru
ning(AnalysisHelper.scala:267)
```

```
	at
org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper.transformDownWithPru
ning$(AnalysisHelper.scala:263)
	at
org.apache.spark.sql.catalyst.plans.logical.LogicalPlan.transformDownWithPrunin
g(LogicalPlan.scala:32)
	at
org.apache.spark.sql.catalyst.plans.logical.LogicalPlan.transformDownWithPrunin
g(LogicalPlan.scala:32)
	at
org.apache.spark.sql.catalyst.trees.TreeNode.transformDown(TreeNode.scala:449)
	at
org.apache.spark.sql.execution.QueryExecution.eagerlyExecuteCommands(QueryExecu
tion.scala:98)
	at
org.apache.spark.sql.execution.QueryExecution.commandExecuted$lzycompute(QueryE
xecution.scala:85)
	at
org.apache.spark.sql.execution.QueryExecution.commandExecuted(QueryExecution.sc
ala:83)
	at
org.apache.spark.sql.execution.QueryExecution.assertCommandExecuted(QueryExecut
ion.scala:142)
	at
org.apache.spark.sql.DataFrameWriter.runCommand(DataFrameWriter.scala:859)
	at
org.apache.spark.sql.DataFrameWriter.saveToV1Source(DataFrameWriter.scala:388)
	at
org.apache.spark.sql.DataFrameWriter.saveInternal(DataFrameWriter.scala:361)
	at org.apache.spark.sql.DataFrameWriter.save(DataFrameWriter.scala:240)
	at
ai.chronon.spark.Extensions$DataPointerAwareDataFrameWriter.$anonfun$save$1(Ext
ensions.scala:320)
	at
ai.chronon.spark.Extensions$DataPointerAwareDataFrameWriter.$anonfun$save$1$ada
pted(Extensions.scala:308)
	at scala.Option.map(Option.scala:230)
	at
ai.chronon.spark.Extensions$DataPointerAwareDataFrameWriter.save(Extensions.sca
la:308)
	at
ai.chronon.spark.TableUtils.repartitionAndWriteInternal(TableUtils.scala:491)
```

```
	at
ai.chronon.spark.TableUtils.$anonfun$repartitionAndWrite$1(TableUtils.scala:416
)
	at scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
	at
ai.chronon.spark.TableUtils.$anonfun$wrapWithCache$3(TableUtils.scala:399)
	at scala.util.Try$.apply(Try.scala:213)
	at ai.chronon.spark.TableUtils.wrapWithCache(TableUtils.scala:398)
	at ai.chronon.spark.TableUtils.repartitionAndWrite(TableUtils.scala:414)
	at ai.chronon.spark.TableUtils.insertPartitions(TableUtils.scala:318)
	at ai.chronon.spark.Extensions$DataframeOps.save(Extensions.scala:159)
	at
ai.chronon.spark.JoinBase.$anonfun$computeRightTable$5(JoinBase.scala:200)
	at
ai.chronon.spark.JoinBase.$anonfun$computeRightTable$5$adapted(JoinBase.scala:1
89)
	at scala.collection.immutable.List.foreach(List.scala:431)
	at ai.chronon.spark.JoinBase.computeRightTable(JoinBase.scala:189)
	at ai.chronon.spark.Join.$anonfun$computeRange$4(Join.scala:337)
	at scala.concurrent.Future$.$anonfun$apply$1(Future.scala:659)
	at scala.util.Success.$anonfun$map$1(Try.scala:255)
	at scala.util.Success.map(Try.scala:213)
	at scala.concurrent.Future.$anonfun$map$1(Future.scala:292)
	at scala.concurrent.impl.Promise.liftedTree1$1(Promise.scala:33)
	at scala.concurrent.impl.Promise.$anonfun$transform$1(Promise.scala:33)
	at scala.concurrent.impl.CallbackRunnable.run(Promise.scala:64)
	at
java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.
java:1128)
	at
java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor
.java:628)
	at java.base/java.lang.Thread.run(Thread.java:829)
Caused by:
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.BigQueryEx
ception: The field specified for time partitioning can only be of type
TIMESTAMP, DATE or DATETIME. The type found is: STRING.
	at
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.spi.v2.Htt
pBigQueryRpc.translate(HttpBigQueryRpc.java:116)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.spi.v2.Htt
pBigQueryRpc.create(HttpBigQueryRpc.java:237)
```

```
	at
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.BigQueryIm
pl$5.call(BigQueryImpl.java:411)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.BigQueryIm
pl$5.call(BigQueryImpl.java:400)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.api.gax.retrying.DirectRe
tryingExecutor.submit(DirectRetryingExecutor.java:102)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.BigQueryRe
tryHelper.run(BigQueryRetryHelper.java:86)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.BigQueryRe
tryHelper.runWithRetries(BigQueryRetryHelper.java:49)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.BigQueryIm
pl.create(BigQueryImpl.java:399)
	at
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.BigQueryIm
pl.create(BigQueryImpl.java:364)
	at
com.google.cloud.bigquery.connector.common.BigQueryClient.createAndWaitFor(BigQ
ueryClient.java:513)
	at
com.google.cloud.bigquery.connector.common.BigQueryClient.createAndWaitFor(BigQ
ueryClient.java:508)
	at
com.google.cloud.bigquery.connector.common.BigQueryClient.loadDataIntoTable(Big
QueryClient.java:809)
	at
com.google.cloud.spark.bigquery.write.BigQueryWriteHelper.loadDataToBigQuery(Bi
gQueryWriteHelper.java:179)
	at
com.google.cloud.spark.bigquery.write.BigQueryWriteHelper.writeDataFrameToBigQu
ery(BigQueryWriteHelper.java:153)
	... 60 more
Caused by:
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.jso
n.GoogleJsonResponseException: 400 Bad Request
POST
https://bigquery.googleapis.com/bigquery/v2/projects/canary-443022/jobs?prettyP
rint=false
{
```

```
  "code": 400,
  "errors": [
    {
      "domain": "global",
      "message": "The field specified for time partitioning can only be of type
TIMESTAMP, DATE or DATETIME. The type found is: STRING.",
      "reason": "invalid"
    }
  ],
  "message": "The field specified for time partitioning can only be of type
TIMESTAMP, DATE or DATETIME. The type found is: STRING.",
  "status": "INVALID_ARGUMENT"
}
       at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.jso
n.GoogleJsonResponseException.from(GoogleJsonResponseException.java:146)
       at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.json.AbstractGoogleJsonClientRequest.newExceptionOnError(AbstractGoogleJs
onClientRequest.java:118)
       at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.json.AbstractGoogleJsonClientRequest.newExceptionOnError(AbstractGoogleJs
onClientRequest.java:37)
       at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.AbstractGoogleClientRequest$3.interceptResponse(AbstractGoogleClientReque
st.java:479)
       at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.http.HttpReque
st.execute(HttpRequest.java:1111)
       at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.AbstractGoogleClientRequest.executeUnparsed(AbstractGoogleClientRequest.j
ava:565)
       at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.AbstractGoogleClientRequest.executeUnparsed(AbstractGoogleClientRequest.j
ava:506)
       at
com.google.cloud.spark.bigquery.repackaged.com.google.api.client.googleapis.ser
vices.AbstractGoogleClientRequest.execute(AbstractGoogleClientRequest.java:616)
```

```
	at
com.google.cloud.spark.bigquery.repackaged.com.google.cloud.bigquery.spi.v2.Htt
pBigQueryRpc.create(HttpBigQueryRpc.java:235)
	... 72 more
```

Resources

https://app.excalidraw.com/s/2zSIL5ELLgM/3WwsOLFuxx1