Spark SQL Cheat Sheet with Iceberg

iomete

SPARK DDL

DATABASE MANAGEMENT

```
-- Managed database is saved in the Hive metastore. Default database is named
"default".
DROP DATABASE IF EXISTS dbName;
CREATE DATABASE dbName;
USE dbName; -- This command avoids having to specify dbName.tableName every time
instead of just tableName.
```

TABLE MANAGEMENT

```
-- Basic syntax
CREATE TABLE dbName.sample (
    id bigint COMMENT 'unique id',
    data string);

-- Create table with partitions
CREATE TABLE dbName.sample (
    id bigint,
    data string,
    category string,
    ts timestamp)
USING iceberg
PARTITIONED BY (bucket(16, id), days(ts), category);

-- Create table with iceberg properties
REPLACE TABLE dbName.sample
USING iceberg
TBLPROPERTIES ('key'='value')
AS SELECT ...

DROP TABLE dbName.sample;

-- ALTER TABLE
ALTER TABLE dbName.sample RENAME TO dbName.new_name;

ALTER TABLE dbName.sample SET TBLPROPERTIES ('read.split.target-size'='268435456');
ALTER TABLE dbName.sample UNSET TBLPROPERTIES ('read.split.target-size');

ALTER TABLE dbName.sample ADD COLUMNS (new_column string);

ALTER TABLE dbName.sample ADD COLUMN point struct<x: double, y: double>;
-- add a field to the struct
ALTER TABLE dbName.sample ADD COLUMN point.z double;
ALTER TABLE dbName.sample ADD COLUMN new_col bigint [AFTER other_col | FIRST];
ALTER TABLE dbName.sample DROP COLUMN id;
ALTER TABLE dbName.sample DROP COLUMN point.z;

ALTER TABLE dbName.sample RENAME COLUMN data TO payload;
ALTER TABLE dbName.sample RENAME COLUMN location.lat TO latitude;

ALTER TABLE dbName.sample ALTER COLUMN measurement TYPE int COMMENT 'unit in bps';
ALTER TABLE dbName.sample ALTER COLUMN id DROP NOT NULL;

ALTER TABLE dbName.sample ADD PARTITION FIELD catalog;
ALTER TABLE dbName.sample DROP PARTITION FIELD catalog;

ALTER TABLE dbName.sample WRITE ORDERED BY category ASC, id DESC;

ALTER TABLE dbName.sample WRITE DISTRIBUTED BY PARTITION LOCALLY ORDERED BY
category, id;
```

READS & WRITES

SPARK QUERIES

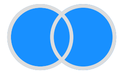
```
SELECT * FROM spark_catalog.db.table; -- catalog [optional]: spark_catalog
SELECT * FROM tbl_1 [LEFT | RIGHT | FULL OUTER] JOIN tbl_2 ON ...
```

INNER JOIN

LEFT JOIN

RIGHT JOIN

FULL OUTER JOIN



INSPECT TABLES

```
SELECT * FROM db.table.history;
SELECT * FROM db.table.snapshots;
SELECT * FROM db.table.files;
SELECT * FROM db.table.manifests;
SELECT * FROM db.table.partitions;
```

INSERT INTO

```
INSERT INTO db.table VALUES (1, 'a'), (2, 'b');
INSERT INTO db.table SELECT ...
```

UPDATE

```
UPDATE db.table
SET c1 = 'update_c1', c2 = 'update_c2'
WHERE ts >= '2020-05-01 00:00:00' and ts < '2020-06-01 00:00:00'

UPDATE db.orders AS t1
SET order_status = 'returned'
WHERE EXISTS (SELECT oid FROM db.returned_orders WHERE t1.oid = oid)
```

MERGE INTO

```
MERGE INTO db.target t   -- a target table
USING (SELECT ...) s          -- the source updates
ON t.id = s.id                -- condition to find updates for target rows
WHEN MATCHED AND s.op = 'delete' THEN DELETE
WHEN MATCHED AND s.op = 'increment' THEN UPDATE SET t.count = t.count + 1
WHEN NOT MATCHED THEN INSERT *
```

DELETE FROM

```
DELETE FROM db.table
WHERE ts >= '2020-05-01 00:00:00' and ts < '2020-06-01 00:00:00';
DELETE FROM db.orders AS t1
WHERE EXISTS (SELECT oid FROM db.returned_orders WHERE t1.oid = oid)
```

TIME TRAVEL

```
SELECT * FROM db.table TIMESTAMP AS OF '1986-10-26 01:21:00';
SELECT * FROM db.table VERSION AS OF 10963874102873; -- time travel to snapshot
```

SPARK PROCEDURES

rollback_to_snapshot

argument name	type	required?	description
table	string	yes	Name of the table to update
snapshot_id	long	yes	Snapshot ID to rollback to

```
CALL spark_catalog.system.rollback_to_snapshot('db.sample', 1)
```

rollback_to_timestamp

argument name	type	required?	description
table	string	yes	Name of the table to update
timestamp	timestamp	yes	A timestamp to rollback to

```
CALL spark_catalog.system.rollback_to_timestamp('db.sample', TIMESTAMP '2021-06-30
00:00:00.000')
```

expire_snapshots

argument name	type	required?	description
table	string	yes	Name of the table to update
older_than	timestamp	no	Default: 5 days ago
retain_last	int	no	Default: 1

```
CALL spark_catalog.system.expire_snapshots('db.sample')
```

remove_orphan_files

argument name	type	required?	description
table	string	yes	Name of the table to update
older_than	timestamp	no	Defaults to 3 days ago

```
CALL spark_catalog.system.remove_orphan_files(table => 'db.sample')
```

rewrite_data_files

argument name	type	required?	description
table	string	yes	Name of the table to update
strategy	string	no	binpack or sort. Default: binpack
sort_order	string	no	...
options	map<string, string>	no	Options to be used for actions

```
CALL spark_catalog.system.rewrite_data_files(table => 'db.sample', options =>
map('min-input-files','2'))
```

rewrite_manifests

argument name	type	required?	description
table	string	yes	Name of the table to update

```
CALL spark_catalog.system.rewrite_manifests('db.sample')
```

** Please note that some arguments are not listed here. This is just a brief overview. More information could be found on apache iceberg documentation.*