List Partitioning

Let us understand how we can take care of list partitioning of tables.

- It is primarily used to create partitions based up on the values.
- Here are the steps involved in creating table using list partitioning strategy.
  - Create table using `PARTITION BY LIST`
  - Add default and value specific partitions
  - Validate by inserting data into the table
- We can detach as well as drop the partitions from the table.

Create Partitioned Table

Let us create partitioned table with name `users_part`.

- It contains same columns as `users`.
- We will partition based up on `user_role` field.

```
%load_ext sql
```

```
%env DATABASE_URL=postgresql://itversity_sms_user:sms_password@localhost:5432/itversity_sms_db
```

```
env:
DATABASE_URL=postgresql://itversity_sms_user:sms_password@localhost:5432/itversity_sms_db
```

```
%sql DROP TABLE IF EXISTS users
```

```
Done.
```

```
[]
```

```
%%sql

CREATE TABLE users (
    user_id SERIAL PRIMARY KEY,
    user_first_name VARCHAR(30) NOT NULL,
    user_last_name VARCHAR(30) NOT NULL,
    user_email_id VARCHAR(50) NOT NULL,
    user_email_validated BOOLEAN DEFAULT FALSE,
    user_password VARCHAR(200),
    user_role VARCHAR(1) NOT NULL DEFAULT 'U', --U and A
    is_active BOOLEAN DEFAULT FALSE,
    created_dt DATE DEFAULT CURRENT_DATE,
    last_updated_ts TIMESTAMP DEFAULT CURRENT_TIMESTAMP
)
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
Done.
```

```
[]
```

```
%sql DROP TABLE IF EXISTS users_part
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
Done.
```

```
[]
```

```
%%sql

CREATE TABLE users_part (
    user_id SERIAL,
    user_first_name VARCHAR(30) NOT NULL,
    user_last_name VARCHAR(30) NOT NULL,
    user_email_id VARCHAR(50) NOT NULL,
    user_email_validated BOOLEAN DEFAULT FALSE,
    user_password VARCHAR(200),
    user_role VARCHAR(1) NOT NULL DEFAULT 'U', --U and A
    is_active BOOLEAN DEFAULT FALSE,
    created_dt DATE DEFAULT CURRENT_DATE,
    last_updated_ts TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (user_role, user_id)
) PARTITION BY LIST(user_role)
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
Done.
```

```
[]
```

Note

Additional indexes on the users_part table.

```
%%sql

CREATE INDEX users_part_email_id_idx
    ON users_part(user_email_id)
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
Done.
```

```
[]
```

Error

Below `INSERT` statement will fail as we have not added any partitions to the table `users_part` even though it is created as partitioned table.

```
%%sql

INSERT INTO users_part (user_first_name, user_last_name, user_email_id)
VALUES
    ('Scott', 'Tiger', 'scott@tiger.com'),
    ('Donald', 'Duck', 'donald@duck.com'),
    ('Mickey', 'Mouse', 'mickey@mouse.com')
```

```
 * postgresql://itversity_sms_user:***@localhost:5432/itversity_sms_db
```

```
---------------------------------------------------------------------------
CheckViolation                            Traceback (most recent call last)
/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sqlalchemy/engine/base.py in
_execute_context(self, dialect, constructor, statement, parameters, *args)
   1276                     self.dialect.do_execute(
-> 1277                         cursor, statement, parameters, context
   1278                     )

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sqlalchemy/engine/default.py in
do_execute(self, cursor, statement, parameters, context)
    592     def do_execute(self, cursor, statement, parameters, context=None):
--> 593         cursor.execute(statement, parameters)
    594

CheckViolation: no partition of relation "users_part" found for row
DETAIL:  Partition key of the failing row contains (user_role) = (U).


The above exception was the direct cause of the following exception:

IntegrityError                            Traceback (most recent call last)
<ipython-input-23-b06b3c83cab2> in <module>
----> 1 get_ipython().run_cell_magic('sql', '', "\nINSERT INTO users_part (user_first_name,
user_last_name, user_email_id)\nVALUES \n    ('Scott', 'Tiger', 'scott@tiger.com'),\n
('Donald', 'Duck', 'donald@duck.com'),\n    ('Mickey', 'Mouse', 'mickey@mouse.com')\n")

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/IPython/core/interactiveshell.py in
run_cell_magic(self, magic_name, line, cell)
   2369             with self.builtin_trap:
   2370                 args = (magic_arg_s, cell)
-> 2371                 result = fn(*args, **kwargs)
   2372             return result
   2373

<decorator-gen-135> in execute(self, line, cell, local_ns)

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/IPython/core/magic.py in <lambda>(f,
*a, **k)
    185     # but it's overkill for just that one bit of state.
    186     def magic_deco(arg):
--> 187         call = lambda f, *a, **k: f(*a, **k)
    188
    189         if callable(arg):

<decorator-gen-134> in execute(self, line, cell, local_ns)

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/IPython/core/magic.py in <lambda>(f,
*a, **k)
    185     # but it's overkill for just that one bit of state.
    186     def magic_deco(arg):
--> 187         call = lambda f, *a, **k: f(*a, **k)
    188
    189         if callable(arg):

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sql/magic.py in execute(self, line,
cell, local_ns)
    215
    216         try:
--> 217             result = sql.run.run(conn, parsed["sql"], self, user_ns)
    218
    219             if (

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sql/run.py in run(conn, sql, config,
user_namespace)
    365             else:
    366                 txt = sqlalchemy.sql.text(statement)
--> 367                 result = conn.session.execute(txt, user_namespace)
    368             _commit(conn=conn, config=config)
    369             if result and config.feedback:

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sqlalchemy/engine/base.py in
execute(self, object_, *multiparams, **params)
   1009             )
   1010         else:
-> 1011             return meth(self, multiparams, params)
   1012
   1013     def _execute_function(self, func, multiparams, params):

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sqlalchemy/sql/elements.py in
_execute_on_connection(self, connection, multiparams, params)
    296     def _execute_on_connection(self, connection, multiparams, params):
    297         if self.supports_execution:
--> 298             return connection._execute_clauseelement(self, multiparams, params)
    299         else:
    300             raise exc.ObjectNotExecutableError(self)

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sqlalchemy/engine/base.py in
_execute_clauseelement(self, elem, multiparams, params)
```

```
   1128             distilled_params,
   1129             compiled_sql,
-> 1130             distilled_params,
   1131         )
   1132         if self._has_events or self.engine._has_events:

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sqlalchemy/engine/base.py in
_execute_context(self, dialect, constructor, statement, parameters, *args)
   1315         except BaseException as e:
   1316             self._handle_dbapi_exception(
-> 1317                 e, statement, parameters, cursor, context
   1318             )
   1319

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sqlalchemy/engine/base.py in
_handle_dbapi_exception(self, e, statement, parameters, cursor, context)
   1509             elif should_wrap:
   1510                 util.raise_(
-> 1511                     sqlalchemy_exception, with_traceback=exc_info[2], from_=e
   1512                 )
   1513             else:

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sqlalchemy/util/compat.py in
raise_(***failed resolving arguments***)
    180
    181         try:
--> 182             raise exception
    183         finally:
    184             # credit to

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sqlalchemy/engine/base.py in
_execute_context(self, dialect, constructor, statement, parameters, *args)
   1275                 if not evt_handled:
   1276                     self.dialect.do_execute(
-> 1277                         cursor, statement, parameters, context
   1278                     )
   1279

/opt/anaconda3/envs/beakerx/lib/python3.6/site-packages/sqlalchemy/engine/default.py in
do_execute(self, cursor, statement, parameters, context)
    591
    592     def do_execute(self, cursor, statement, parameters, context=None):
--> 593         cursor.execute(statement, parameters)
    594
    595     def do_execute_no_params(self, cursor, statement, context=None):

IntegrityError: (psycopg2.errors.CheckViolation) no partition of relation "users_part" found
for row
DETAIL:  Partition key of the failing row contains (user_role) = (U).

[SQL: INSERT INTO users_part (user_first_name, user_last_name, user_email_id)
VALUES
    ('Scott', 'Tiger', 'scott@tiger.com'),
    ('Donald', 'Duck', 'donald@duck.com'),
    ('Mickey', 'Mouse', 'mickey@mouse.com')]
(Background on this error at: http://sqlalche.me/e/13/gkpj)
```

By Durga Gadiraju