

# L1 E1 - Step 4

June 10, 2021

## 1 STEP4 : Creating Facts & Dimensions

Start by connecting to the database by running the cells below. If you are coming back to this exercise, then uncomment and run the first cell to recreate the database. If you recently completed steps 1 and 2, then skip to the second cell.

```
In [1]: !PGPASSWORD=student createdb -h 127.0.0.1 -U student pagila
        !PGPASSWORD=student psql -q -h 127.0.0.1 -U student -d pagila -f Data/pagila-schema.sql
        !PGPASSWORD=student psql -q -h 127.0.0.1 -U student -d pagila -f Data/pagila-data.sql
```

```
setval
-----
      200
(1 row)
```

```
setval
-----
      605
(1 row)
```

```
setval
-----
       16
(1 row)
```

```
setval
-----
      600
(1 row)
```

```
setval
-----
      109
(1 row)
```

```
setval
-----
```

```
599
(1 row)
```

```
setval
-----
1000
(1 row)
```

```
setval
-----
4581
(1 row)
```

```
setval
-----
6
(1 row)
```

```
setval
-----
32098
(1 row)
```

```
setval
-----
16049
(1 row)
```

```
setval
-----
2
(1 row)
```

```
setval
-----
2
(1 row)
```

```
In [2]: %load_ext sql
```

```
DB_ENDPOINT = "127.0.0.1"
DB = 'pagila'
DB_USER = 'student'
DB_PASSWORD = 'student'
DB_PORT = '5432'
```

```
# postgresql://username:password@host:port/database
conn_string = "postgresql://{username}:{password}@{host}:{port}/{database}" \
               .format(DB_USER, DB_PASSWORD, DB_ENDPOINT, DB_PORT, DB)

print(conn_string)
%sql $conn_string
```

```
postgresql://student:student@127.0.0.1:5432/pagila
```

```
Out[2]: 'Connected: student@pagila'
```

### 1.0.1 Star Schema - Entity Relationship Diagram

**Create the first dimension table** TODO: Create the dimDate dimension table with the fields and data types shown in the ERD above.

```
In [5]: %%sql
CREATE TABLE dimDate
(
    date_key integer NOT NULL PRIMARY KEY,
    date date NOT NULL,
    year smallint NOT NULL,
    quarter smallint NOT NULL,
    month smallint NOT NULL,
    day smallint NOT NULL,
    week smallint NOT NULL,
    is_weekend boolean
);

* postgresql://student:***@127.0.0.1:5432/pagila
Done.
```

```
Out[5]: []
```

To check your work, run the following query to see a table with the field names and data types. The output should match the table below.

```
In [6]: %%sql
SELECT column_name, data_type
FROM information_schema.columns
WHERE table_name = 'dimdate'

* postgresql://student:***@127.0.0.1:5432/pagila
8 rows affected.
```

```
Out[6]: [('date_key', 'integer'),
         ('date', 'date'),
         ('year', 'smallint'),
         ('quarter', 'smallint'),
         ('month', 'smallint'),
         ('day', 'smallint'),
         ('week', 'smallint'),
         ('is_weekend', 'boolean')]
```

```
<tbody><tr>
  <th>column_name</th>
  <th>data_type</th>
</tr>
<tr>
  <td>date_key</td>
  <td>integer</td>
</tr>
<tr>
  <td>date</td>
  <td>date</td>
</tr>
<tr>
  <td>year</td>
  <td>smallint</td>
</tr>
<tr>
  <td>quarter</td>
  <td>smallint</td>
</tr>
<tr>
  <td>month</td>
  <td>smallint</td>
</tr>
<tr>
  <td>day</td>
  <td>smallint</td>
</tr>
<tr>
  <td>week</td>
  <td>smallint</td>
</tr>
<tr>
  <td>is_weekend</td>
  <td>boolean</td>
</tr>
```

Run the cell below to create the rest of the dimension tables.

```
In [7]: %%sql
```

```

CREATE TABLE dimCustomer
(
    customer_key SERIAL PRIMARY KEY,
    customer_id  smallint NOT NULL,
    first_name   varchar(45) NOT NULL,
    last_name    varchar(45) NOT NULL,
    email        varchar(50),
    address      varchar(50) NOT NULL,
    address2     varchar(50),
    district     varchar(20) NOT NULL,
    city         varchar(50) NOT NULL,
    country      varchar(50) NOT NULL,
    postal_code  varchar(10),
    phone        varchar(20) NOT NULL,
    active       smallint NOT NULL,
    create_date  timestamp NOT NULL,
    start_date   date NOT NULL,
    end_date     date NOT NULL
);

CREATE TABLE dimMovie
(
    movie_key      SERIAL PRIMARY KEY,
    film_id        smallint NOT NULL,
    title          varchar(255) NOT NULL,
    description     text,
    release_year   year,
    language       varchar(20) NOT NULL,
    original_language varchar(20),
    rental_duration smallint NOT NULL,
    length         smallint NOT NULL,
    rating         varchar(5) NOT NULL,
    special_features varchar(60) NOT NULL
);

CREATE TABLE dimStore
(
    store_key      SERIAL PRIMARY KEY,
    store_id       smallint NOT NULL,
    address        varchar(50) NOT NULL,
    address2       varchar(50),
    district       varchar(20) NOT NULL,
    city          varchar(50) NOT NULL,
    country        varchar(50) NOT NULL,
    postal_code    varchar(10),
    manager_first_name varchar(45) NOT NULL,
    manager_last_name  varchar(45) NOT NULL,
    start_date     date NOT NULL,
    end_date       date NOT NULL
);

```

```

);

* postgresql://student:***@127.0.0.1:5432/pagila
Done.
Done.
Done.

```

```
Out[7]: []
```

**Create the fact table** TODO: Create the factSales table with the fields and data types shown in the ERD above.

**Note on REFERENCES constraints:** The demo video does not cover the REFERENCES constraint. When building a fact table, you use the REFERENCES constrain to identify which table and column a foreign key is connected to. This ensures that the fact table does not refer to items that do not appear in the respective dimension tables. You can read more [here](#). Here's an example of the syntax on a different schema:

```

CREATE TABLE orders (
    order_id integer PRIMARY KEY,
    product_no integer REFERENCES products (product_no),
    quantity integer
);

In [8]: %%sql
        CREATE TABLE factSales
        (
            sales_key integer,
            date_key integer REFERENCES dimDate (date_key),
            customer_key integer REFERENCES dimCustomer (customer_key),
            movie_key integer REFERENCES dimMovie (movie_key),
            store_key integer REFERENCES dimStore (store_key),
            sales_amount numeric
        );

* postgresql://student:***@127.0.0.1:5432/pagila
Done.

```

```
Out[8]: []
```

To check your work, run the following query to see a table with the field names and data types. The output should match the table below.

```

In [9]: %%sql
        SELECT column_name, data_type
        FROM information_schema.columns
        WHERE table_name = 'factsales'

```

```
* postgresql://student:***@127.0.0.1:5432/pagila
6 rows affected.
```

```
Out[9]: [('sales_key', 'integer'),
         ('date_key', 'integer'),
         ('customer_key', 'integer'),
         ('movie_key', 'integer'),
         ('store_key', 'integer'),
         ('sales_amount', 'numeric')]
```

```
<tbody><tr>
  <th>column_name</th>
  <th>data_type</th>
</tr>
<tr>
  <td>sales_key</td>
  <td>integer</td>
</tr>
<tr>
  <td>date_key</td>
  <td>integer</td>
</tr>
<tr>
  <td>customer_key</td>
  <td>integer</td>
</tr>
<tr>
  <td>movie_key</td>
  <td>integer</td>
</tr>
<tr>
  <td>store_key</td>
  <td>integer</td>
</tr>
<tr>
  <td>sales_amount</td>
  <td>numeric</td>
</tr>
```

If you need to delete the table and start over, use the DROP TABLE command: DROP TABLE <table\_name>