L1 E2 - 2 - Roll up and Drill Down

January 10, 2023

1 Exercise 02 - OLAP Cubes - Roll Up and Drill Down

All the databases table in this demo are based on public database samples and transformations - `Sakila` is a sample database created by `MySql` Link - The postgresql version of it is called `Pagila` Link - The facts and dimension tables design is based on O'Reilly's public dimensional modelling tutorial schema Link

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 the slicing and dicing exercise, then skip to the second cell.

```
In [2]:  !PGPASSWORD=student createdb -h 127.0.0.1 -U student pagila_star
         !PGPASSWORD=student psql -q -h 127.0.0.1 -U student -d pagila_star -f Data/pagila-star.
```

```
 set_config
------------

(1 row)

 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
--------
      1
(1 row)

 setval
--------
      1
(1 row)

 setval
--------
      1
(1 row)

 setval
--------
      1
(1 row)

 setval
--------
  16049
(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)
```

1.0.1 Connect to the local database where Pagila is loaded

```
In [3]: import sql
        %load_ext sql

        DB_ENDPOINT = "127.0.0.1"
        DB = 'pagila_star'
        DB_USER = 'student'
        DB_PASSWORD = 'student'
        DB_PORT = '5432'

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

        print(conn_string)
        %sql $conn_string
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
postgresql://student:student@127.0.0.1:5432/pagila_star


Out[3]: 'Connected: student@pagila_star'
```

1.0.2 Star Schema

1.1 Roll-up

- Stepping up the level of aggregation to a large grouping
- e.g.`city` is summed as `country`

TODO: Write a query that calculates revenue (sales_amount) by day, rating, and country. Sort the data by revenue in descending order, and limit the data to the top 20 results. The first few rows of your output should match the table below.

```
In [17]: %%time
         %%sql
         SELECT d.day, m.rating, c.country, SUM(sales_amount) AS revenue
         FROM factsales f
         JOIN dimdate d
           ON f.date_key = d.date_key
         JOIN dimmovie m
           ON f.movie_key = m.movie_key
         JOIN dimcustomer c
           ON f.customer_key = c.customer_key
         GROUP BY d.day, m.rating, c.country
         ORDER BY revenue DESC
         LIMIT 20

 * postgresql://student:***@127.0.0.1:5432/pagila_star
20 rows affected.
CPU times: user 4.18 ms, sys: 0 ns, total: 4.18 ms
Wall time: 26.3 ms


Out[17]: [(30, 'G', 'China', Decimal('169.67')),
          (30, 'PG', 'India', Decimal('156.67')),
          (30, 'NC-17', 'India', Decimal('153.64')),
          (30, 'PG-13', 'China', Decimal('146.67')),
          (30, 'R', 'China', Decimal('145.66')),
          (30, 'R', 'India', Decimal('143.68')),
          (30, 'G', 'India', Decimal('137.67')),
          (18, 'NC-17', 'India', Decimal('135.75')),
          (30, 'PG', 'China', Decimal('131.72')),
          (21, 'PG-13', 'India', Decimal('128.74')),
          (18, 'PG-13', 'India', Decimal('121.72')),
          (18, 'PG', 'India', Decimal('119.76')),
          (30, 'PG-13', 'India', Decimal('116.72')),
          (21, 'NC-17', 'China', Decimal('115.77')),
          (21, 'R', 'India', Decimal('115.75')),
          (27, 'NC-17', 'India', Decimal('115.72')),
          (17, 'PG-13', 'China', Decimal('111.75')),
          (17, 'NC-17', 'United States', Decimal('109.76')),
```

```
          (30, 'NC-17', 'China', Decimal('108.77')),
          (20, 'PG-13', 'India', Decimal('101.79'))]
```

```
<tbody><tr>
    <th>day</th>
    <th>rating</th>
    <th>country</th>
    <th>revenue</th>
</tr>
<tr>
    <td>30</td>
    <td>G</td>
    <td>China</td>
    <td>169.67</td>
</tr>
<tr>
    <td>30</td>
    <td>PG</td>
    <td>India</td>
    <td>156.67</td>
</tr>
<tr>
    <td>30</td>
    <td>NC-17</td>
    <td>India</td>
    <td>153.64</td>
</tr>
<tr>
    <td>30</td>
    <td>PG-13</td>
    <td>China</td>
    <td>146.67</td>
</tr>
<tr>
    <td>30</td>
    <td>R</td>
    <td>China</td>
    <td>145.66</td>
</tr>
```

1.2 Drill-down

- Breaking up one of the dimensions to a lower level.
- e.g.`city` is broken up into `districts`

TODO: Write a query that calculates revenue (sales_amount) by day, rating, and district. Sort the data by revenue in descending order, and limit the data to the top 20 results. The first few rows of your output should match the table below.

```
In [15]: %%time
         %%sql
         SELECT d.day, m.rating, c.district, SUM(f.sales_amount) AS revenue
         FROM factsales f
         JOIN dimdate d
           ON f.date_key = d.date_key
         JOIN dimmovie m
           ON f.movie_key = m.movie_key
         JOIN dimcustomer c
           ON c.customer_key = f.customer_key
         GROUP BY d.day, m.rating, c.district
         ORDER BY revenue DESC
         LIMIT 20

 * postgresql://student:***@127.0.0.1:5432/pagila_star
20 rows affected.
CPU times: user 2.19 ms, sys: 2.29 ms, total: 4.48 ms
Wall time: 31.5 ms


Out[15]: [(30, 'PG-13', 'Southern Tagalog', Decimal('53.88')),
          (30, 'G', 'Inner Mongolia', Decimal('38.93')),
          (30, 'G', 'Shandong', Decimal('36.93')),
          (30, 'NC-17', 'West Bengali', Decimal('36.92')),
          (17, 'PG-13', 'Shandong', Decimal('34.95')),
          (1, 'PG', 'California', Decimal('32.94')),
          (18, 'NC-17', 'So Paulo', Decimal('32.93')),
          (30, 'NC-17', 'Buenos Aires', Decimal('31.93')),
          (21, 'R', 'So Paulo', Decimal('31.93')),
          (30, 'PG', 'Southern Tagalog', Decimal('30.94')),
          (30, 'PG', 'So Paulo', Decimal('30.93')),
          (30, 'R', 'Buenos Aires', Decimal('30.92')),
          (30, 'G', 'California', Decimal('29.95')),
          (18, 'NC-17', 'Maharashtra', Decimal('29.95')),
          (21, 'PG-13', 'Uttar Pradesh', Decimal('29.94')),
          (20, 'PG', 'Shandong', Decimal('29.93')),
          (10, 'R', 'Maharashtra', Decimal('29.93')),
          (18, 'R', 'Sumy', Decimal('28.97')),
          (21, 'PG-13', 'Southern Tagalog', Decimal('28.96')),
          (29, 'NC-17', 'So Paulo', Decimal('28.95'))]

<tbody><tr>
    <th>day</th>
    <th>rating</th>
    <th>district</th>
    <th>revenue</th>
</tr>
<tr>
```

```
    <td>30</td>
    <td>PG-13</td>
    <td>Southern Tagalog</td>
    <td>53.88</td>
</tr>
<tr>
    <td>30</td>
    <td>G</td>
    <td>Inner Mongolia</td>
    <td>38.93</td>
</tr>
<tr>
    <td>30</td>
    <td>G</td>
    <td>Shandong</td>
    <td>36.93</td>
</tr>
<tr>
    <td>30</td>
    <td>NC-17</td>
    <td>West Bengali</td>
    <td>36.92</td>
</tr>
<tr>
    <td>17</td>
    <td>PG-13</td>
    <td>Shandong</td>
    <td>34.95</td>
</tr>
```