L1 E2 - 1 - Slicing and Dicing

June 13, 2021

1 Exercise 02 - OLAP Cubes - Slicing and Dicing

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 creating and connecting to the database by running the cells below.

```
In [3]: # !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
```

1.0.1 Connect to the local database where Pagila is loaded

```
In [4]: 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
```

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

```
Out[4]: 'Connected: student@pagila_star'
```

1.0.2 Star Schema

2 Start with a simple cube

TODO: Write a query that calculates the revenue (sales_amount) by day, rating, and city. Remember to join with the appropriate dimension tables to replace the keys with the dimension labels. Sort by revenue in descending order and limit to the first 20 rows. The first few rows of your output should match the table below.

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

        select dimDate.day, dimMovie.rating, dimCustomer.city, sum(sales_amount) as revenue
        from factSales
        join dimMovie on dimMovie.movie_key = factSales.movie_key
        join dimDate on dimDate.date_key = factSales.date_key
        join dimCustomer on dimCustomer.customer_key = factSales.customer_key
        group by 1,2,3
        order by 4 desc
        limit 5;
```

```
 * postgresql://student:***@127.0.0.1:5432/pagila_star
5 rows affected.
CPU times: user 4.29 ms, sys: 0 ns, total: 4.29 ms
Wall time: 35.7 ms
```

```
Out[7]: [(30, 'G', 'San Bernardino', Decimal('24.97')),
         (30, 'NC-17', 'Apeldoorn', Decimal('23.95')),
         (21, 'NC-17', 'Belm', Decimal('22.97')),
         (30, 'PG-13', 'Zanzibar', Decimal('21.97')),
         (28, 'R', 'Mwanza', Decimal('21.97'))]
```

```
<tbody><tr>
    <th>day</th>
    <th>rating</th>
    <th>city</th>
    <th>revenue</th>
</tr>
<tr>
    <td>30</td>
    <td>G</td>
    <td>San Bernardino</td>
    <td>24.97</td>
</tr>
<tr>
    <td>30</td>
    <td>NC-17</td>
    <td>Apeldoorn</td>
    <td>23.95</td>
</tr>
<tr>
    <td>21</td>
    <td>NC-17</td>
    <td>Belm</td>
    <td>22.97</td>
</tr>
<tr>
    <td>30</td>
    <td>PG-13</td>
    <td>Zanzibar</td>
    <td>21.97</td>
</tr>
<tr>
    <td>28</td>
    <td>R</td>
    <td>Mwanza</td>
    <td>21.97</td>
</tr>
```

2.1 Slicing

Slicing is the reduction of the dimensionality of a cube by 1 e.g. 3 dimensions to 2, fixing one of the dimensions to a single value. In the example above, we have a 3-dimensional cube on day, rating, and country.

TODO: Write a query that reduces the dimensionality of the above example by limiting the results to only include movies with a `rating` of "PG-13". Again, sort by revenue in descending order and limit to the first 20 rows. The first few rows of your output should match the table below.

```
In [8]: %%time
        %%sql

        select dimDate.day, dimMovie.rating, dimCustomer.city, sum(sales_amount) as revenue
        from factSales
        join dimMovie on dimMovie.movie_key = factSales.movie_key
        join dimDate on dimDate.date_key = factSales.date_key
        join dimCustomer on dimCustomer.customer_key = factSales.customer_key
        where dimMovie.rating = 'PG-13'
        group by 1,2,3
        order by 4 desc
        limit 5;
```

```
 * postgresql://student:***@127.0.0.1:5432/pagila_star
5 rows affected.
CPU times: user 4.56 ms, sys: 0 ns, total: 4.56 ms
Wall time: 14.7 ms
```

```
Out[8]: [(30, 'PG-13', 'Zanzibar', Decimal('21.97')),
         (28, 'PG-13', 'Dhaka', Decimal('19.97')),
         (30, 'PG-13', 'Osmaniye', Decimal('18.97')),
         (29, 'PG-13', 'Shimoga', Decimal('18.97')),
         (21, 'PG-13', 'Asuncin', Decimal('18.95'))]
```

```
<tbody><tr>
    <th>day</th>
    <th>rating</th>
    <th>city</th>
    <th>revenue</th>
</tr>
<tr>
    <td>30</td>
    <td>PG-13</td>
    <td>Zanzibar</td>
    <td>21.97</td>
</tr>
<tr>
    <td>28</td>
    <td>PG-13</td>
    <td>Dhaka</td>
    <td>19.97</td>
</tr>
<tr>
    <td>29</td>
    <td>PG-13</td>
    <td>Shimoga</td>
    <td>18.97</td>
</tr>
<tr>
    <td>30</td>
    <td>PG-13</td>
    <td>Osmaniye</td>
    <td>18.97</td>
</tr>
<tr>
    <td>21</td>
    <td>PG-13</td>
    <td>Asuncin</td>
    <td>18.95</td>
</tr>
```

2.2 Dicing

Dicing is creating a subcube with the same dimensionality but fewer values for two or more dimensions.

TODO: Write a query to create a subcube of the initial cube that includes moves with: * ratings of PG or PG-13 * in the city of Bellevue or Lancaster * day equal to 1, 15, or 30

The first few rows of your output should match the table below.

```
In [10]: %%time
         %%sql

         select dimDate.day, dimMovie.rating, dimCustomer.city, sum(sales_amount) as revenue
         from factSales
         join dimMovie on dimMovie.movie_key = factSales.movie_key
         join dimDate on dimDate.date_key = factSales.date_key
         join dimCustomer on dimCustomer.customer_key = factSales.customer_key
         where dimMovie.rating in ('PG-13','PG')
         and dimCustomer.city in ('Bellevue', 'Lancaster')
         and dimDate.day in ('1', '15', '30')
         group by 1,2,3
         order by 4 desc
         limit 5;
```

```
 * postgresql://student:***@127.0.0.1:5432/pagila_star
5 rows affected.
CPU times: user 5.24 ms, sys: 159 ţs, total: 5.4 ms
Wall time: 9.64 ms
```

```
Out[10]: [(30, 'PG', 'Lancaster', Decimal('12.98')),
          (1, 'PG-13', 'Lancaster', Decimal('5.99')),
          (30, 'PG-13', 'Bellevue', Decimal('3.99')),
          (30, 'PG-13', 'Lancaster', Decimal('2.99')),
          (15, 'PG-13', 'Bellevue', Decimal('1.98'))]
```

```
<tbody><tr>
    <th>day</th>
    <th>rating</th>
    <th>city</th>
    <th>revenue</th>
</tr>
<tr>
    <td>30</td>
    <td>PG</td>
    <td>Lancaster</td>
    <td>12.98</td>
</tr>
<tr>
    <td>1</td>
    <td>PG-13</td>
    <td>Lancaster</td>
    <td>5.99</td>
</tr>
```

```
<tr>
    <td>30</td>
    <td>PG-13</td>
    <td>Bellevue</td>
    <td>3.99</td>
</tr>
<tr>
    <td>30</td>
    <td>PG-13</td>
    <td>Lancaster</td>
    <td>2.99</td>
</tr>
<tr>
    <td>15</td>
    <td>PG-13</td>
    <td>Bellevue</td>
    <td>1.98</td>
</tr>
```