L1 E1 - Step 3

May 29, 2021

1 STEP3: Perform some simple data analysis

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 [3]: !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 [4]: %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://{}:{}@{}:{}/{}" \
                                .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
```

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

1.0.1 3NF - Entity Relationship Diagram

1.1 3.1 Insight 1: Top Grossing Movies

- Payments amounts are in table `payment`
- Movies are in table `film`
- They are not directly linked, `payment` refers to a `rental`, `rental` refers to an `inventory` item and `inventory` item refers to a `film`
- `payment rental inventory film`

1.1.1 3.1.1 Films

```
In [5]: %%sql
        select film_id, title, release_year, rental_rate, rating  from film limit 5;
```

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

```
Out[5]: [(1, 'ACADEMY DINOSAUR', 2006, Decimal('0.99'), 'PG'),
         (2, 'ACE GOLDFINGER', 2006, Decimal('4.99'), 'G'),
         (3, 'ADAPTATION HOLES', 2006, Decimal('2.99'), 'NC-17'),
         (4, 'AFFAIR PREJUDICE', 2006, Decimal('2.99'), 'G'),
         (5, 'AFRICAN EGG', 2006, Decimal('2.99'), 'G')]
```

1.1.2 3.1.2 Payments

```
In [6]: %%sql
        select * from payment limit 5;
```

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

```
Out[6]: [(16050, 269, 2, 7, Decimal('1.99'), datetime.datetime(2017, 1, 24, 21, 40, 19, 996577,
         (16051, 269, 1, 98, Decimal('0.99'), datetime.datetime(2017, 1, 25, 15, 16, 50, 996577,
         (16052, 269, 2, 678, Decimal('6.99'), datetime.datetime(2017, 1, 28, 21, 44, 14, 996577
         (16053, 269, 2, 703, Decimal('0.99'), datetime.datetime(2017, 1, 29, 0, 58, 2, 996577,
         (16054, 269, 1, 750, Decimal('4.99'), datetime.datetime(2017, 1, 29, 8, 10, 6, 996577,
```

1.1.3 3.1.3 Inventory

```
In [7]: %%sql
        select * from inventory limit 5;

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


Out[7]: [(1, 1, 1, datetime.datetime(2017, 2, 15, 10, 9, 17, tzinfo=psycopg2.tz.FixedOffsetTimez
         (2, 1, 1, datetime.datetime(2017, 2, 15, 10, 9, 17, tzinfo=psycopg2.tz.FixedOffsetTimez
         (3, 1, 1, datetime.datetime(2017, 2, 15, 10, 9, 17, tzinfo=psycopg2.tz.FixedOffsetTimez
         (4, 1, 1, datetime.datetime(2017, 2, 15, 10, 9, 17, tzinfo=psycopg2.tz.FixedOffsetTimez
         (5, 1, 2, datetime.datetime(2017, 2, 15, 10, 9, 17, tzinfo=psycopg2.tz.FixedOffsetTimez
```

1.1.4 3.1.4 Get the movie of every payment

```
In [8]: %%sql
        SELECT f.title, p.amount, p.payment_date, p.customer_id
        FROM payment p
        JOIN rental r    ON ( p.rental_id = r.rental_id )
        JOIN inventory i ON ( r.inventory_id = i.inventory_id )
        JOIN film f ON ( i.film_id = f.film_id)
        limit 5;

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


Out[8]: [('SWARM GOLD', Decimal('1.99'), datetime.datetime(2017, 1, 24, 21, 40, 19, 996577, tzin
         ('PACKER MADIGAN', Decimal('0.99'), datetime.datetime(2017, 1, 25, 15, 16, 50, 996577,
         ('SOMETHING DUCK', Decimal('6.99'), datetime.datetime(2017, 1, 28, 21, 44, 14, 996577,
         ('DRACULA CRYSTAL', Decimal('0.99'), datetime.datetime(2017, 1, 29, 0, 58, 2, 996577, t
         ('CLOSER BANG', Decimal('4.99'), datetime.datetime(2017, 1, 29, 8, 10, 6, 996577, tzinf
```

1.1.5 3.1.5 sum movie rental revenue

TODO: Write a query that displays the amount of revenue from each title. Limit the results to the top 10 grossing titles. Your results should match the table below.

```
In [9]: %%sql
        SELECT        f.title, sum(p.amount) as revenue
        from        payment p
```

```
        join        rental r
        on                p.rental_id = r.rental_id
        join        inventory i
        on                r.inventory_id = i.inventory_id
        join        film f
        on                i.film_id = f.film_id
        group by title
        order by revenue desc
        limit 10;

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

Out[9]:
```
[('TELEGRAPH VOYAGE', Decimal('231.73')),
 ('WIFE TURN', Decimal('223.69')),
 ('ZORRO ARK', Decimal('214.69')),
 ('GOODFELLAS SALUTE', Decimal('209.69')),
 ('SATURDAY LAMBS', Decimal('204.72')),
 ('TITANS JERK', Decimal('201.71')),
 ('TORQUE BOUND', Decimal('198.72')),
 ('HARRY IDAHO', Decimal('195.70')),
 ('INNOCENT USUAL', Decimal('191.74')),
 ('HUSTLER PARTY', Decimal('190.78'))]
```

```
<tbody><tr>
    <th>title</th>
    <th>revenue</th>
</tr>
<tr>
    <td>TELEGRAPH VOYAGE</td>
    <td>231.73</td>
</tr>
<tr>
    <td>WIFE TURN</td>
    <td>223.69</td>
</tr>
<tr>
    <td>ZORRO ARK</td>
    <td>214.69</td>
</tr>
<tr>
    <td>GOODFELLAS SALUTE</td>
    <td>209.69</td>
</tr>
<tr>
    <td>SATURDAY LAMBS</td>
    <td>204.72</td>
```

```
</tr>
<tr>
    <td>TITANS JERK</td>
    <td>201.71</td>
</tr>
<tr>
    <td>TORQUE BOUND</td>
    <td>198.72</td>
</tr>
<tr>
    <td>HARRY IDAHO</td>
    <td>195.70</td>
</tr>
<tr>
    <td>INNOCENT USUAL</td>
    <td>191.74</td>
</tr>
<tr>
    <td>HUSTLER PARTY</td>
    <td>190.78</td>
</tr>
```

1.2 3.2 Insight 2: Top grossing cities

- Payments amounts are in table `payment`
- Cities are in table `cities`
- `payment customer address city`

1.2.1 3.2.1 Get the city of each payment

```
In [10]: %%sql
         SELECT p.customer_id, p.rental_id, p.amount, ci.city
         FROM payment p
         JOIN customer c  ON ( p.customer_id = c.customer_id )
         JOIN address a ON ( c.address_id = a.address_id )
         JOIN city ci ON ( a.city_id = ci.city_id )
         order by p.payment_date
         limit 10;

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


Out[10]: [(130, 1, Decimal('2.99'), 'guas Lindas de Gois'),
          (459, 2, Decimal('2.99'), 'Qomsheh'),
          (408, 3, Decimal('3.99'), 'Jaffna'),
          (333, 4, Decimal('4.99'), 'Baku'),
          (222, 5, Decimal('6.99'), 'Jaroslavl'),
          (549, 6, Decimal('0.99'), 'Santiago de Compostela'),
```

```
          (269, 7, Decimal('1.99'), 'Salinas'),
          (239, 8, Decimal('4.99'), 'Ciomas'),
          (126, 9, Decimal('4.99'), 'Po'),
          (399, 10, Decimal('5.99'), 'Okara')]
```

1.2.2 3.2.2 Top grossing cities

TODO: Write a query that returns the total amount of revenue by city as measured by the `amount` variable in the `payment` table. Limit the results to the top 10 cities. Your result should match the table below.

```
In [11]: %%sql
         SELECT        ci.city, sum(p.amount) as revenue
         from        payment p
         join        customer c
         on                p.customer_id = c.customer_id
         join        address a
         on                c.address_id = a.address_id
         join        city ci
         on                a.city_id = ci.city_id
         group by ci.city_id
         order by revenue desc
         limit 10;

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


Out[11]: [('Cape Coral', Decimal('221.55')),
          ('Saint-Denis', Decimal('216.54')),
          ('Aurora', Decimal('198.50')),
          ('Molodetno', Decimal('195.58')),
          ('Apeldoorn', Decimal('194.61')),
          ('Santa Brbara dOeste', Decimal('194.61')),
          ('Qomsheh', Decimal('186.62')),
          ('London', Decimal('180.52')),
          ('Ourense (Orense)', Decimal('177.60')),
          ('Bijapur', Decimal('175.61'))]

<tbody><tr>
    <th>city</th>
    <th>revenue</th>
</tr>
<tr>
    <td>Cape Coral</td>
    <td>221.55</td>
</tr>
<tr>
    <td>Saint-Denis</td>
```

```
    <td>216.54</td>
</tr>
<tr>
    <td>Aurora</td>
    <td>198.50</td>
</tr>
<tr>
    <td>Molodetno</td>
    <td>195.58</td>
</tr>
<tr>
    <td>Apeldoorn</td>
    <td>194.61</td>
</tr>
<tr>
    <td>Santa Brbara dOeste</td>
    <td>194.61</td>
</tr>
<tr>
    <td>Qomsheh</td>
    <td>186.62</td>
</tr>
<tr>
    <td>London</td>
    <td>180.52</td>
</tr>
<tr>
    <td>Ourense (Orense)</td>
    <td>177.60</td>
</tr>
<tr>
    <td>Bijapur</td>
    <td>175.61</td>
</tr>
```

1.3 3.3 Insight 3 : Revenue of a movie by customer city and by month

1.3.1 3.3.1 Total revenue by month

```
In [12]: %%sql
         SELECT sum(p.amount) as revenue, EXTRACT(month FROM p.payment_date) as month
         from payment p
         group by month
         order by revenue desc
         limit 10;
```

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

```
Out[12]: [(Decimal('28559.46'), 4.0),
          (Decimal('23886.56'), 3.0),
          (Decimal('9631.88'), 2.0),
          (Decimal('4824.43'), 1.0),
          (Decimal('514.18'), 5.0)]
```

1.3.2 3.3.2 Each movie by customer city and by month (data cube)

```
In [13]: %%sql
         SELECT f.title, p.amount, p.customer_id, ci.city, p.payment_date,EXTRACT(month FROM p.p
         FROM payment p
         JOIN rental r    ON ( p.rental_id = r.rental_id )
         JOIN inventory i ON ( r.inventory_id = i.inventory_id )
         JOIN film f ON ( i.film_id = f.film_id)
         JOIN customer c  ON ( p.customer_id = c.customer_id )
         JOIN address a ON ( c.address_id = a.address_id )
         JOIN city ci ON ( a.city_id = ci.city_id )
         order by p.payment_date
         limit 10;

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


Out[13]: [('BLANKET BEVERLY', Decimal('2.99'), 130, 'guas Lindas de Gois', datetime.datetime(201
          ('FREAKY POCUS', Decimal('2.99'), 459, 'Qomsheh', datetime.datetime(2017, 1, 24, 21, 2
          ('GRADUATE LORD', Decimal('3.99'), 408, 'Jaffna', datetime.datetime(2017, 1, 24, 21, 3
          ('LOVE SUICIDES', Decimal('4.99'), 333, 'Baku', datetime.datetime(2017, 1, 24, 21, 33,
          ('IDOLS SNATCHERS', Decimal('6.99'), 222, 'Jaroslavl', datetime.datetime(2017, 1, 24,
          ('MYSTIC TRUMAN', Decimal('0.99'), 549, 'Santiago de Compostela', datetime.datetime(20
          ('SWARM GOLD', Decimal('1.99'), 269, 'Salinas', datetime.datetime(2017, 1, 24, 21, 40,
          ('LAWLESS VISION', Decimal('4.99'), 239, 'Ciomas', datetime.datetime(2017, 1, 24, 22,
          ('MATRIX SNOWMAN', Decimal('4.99'), 126, 'Po', datetime.datetime(2017, 1, 24, 22, 29,
          ('HANGING DEEP', Decimal('5.99'), 399, 'Okara', datetime.datetime(2017, 1, 24, 22, 30,
```

1.3.3 3.3.3 Sum of revenue of each movie by customer city and by month

TODO: Write a query that returns the total amount of revenue for each movie by customer city and by month. Limit the results to the top 10 movies. Your result should match the table below.

```
In [15]: %%sql
         SELECT        f.title, ci.city, EXTRACT (month FROM p.payment_date) as month, sum(p.amo
         FROM         payment p
         JOIN         rental r
         ON                 p.rental_id = r.rental_id
         JOIN         inventory i
         ON                 r.inventory_id = i.inventory_id
         JOIN         film f
         ON                i.film_id = f.film_id
```

```
        JOIN         customer c
        ON                p.customer_id = c.customer_id
        JOIN         address a
        ON                c.address_id = a.address_id
        JOIN         city ci
        ON                a.city_id = ci.city_id
        group by f.title, ci.city, month
        order by month, revenue desc
        limit 10;

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


Out[15]: [('SHOW LORD', 'Mannheim', 1.0, Decimal('11.99')),
          ('AMERICAN CIRCUS', 'Callao', 1.0, Decimal('10.99')),
          ('CASUALTIES ENCINO', 'Warren', 1.0, Decimal('10.99')),
          ('TELEGRAPH VOYAGE', 'Naala-Porto', 1.0, Decimal('10.99')),
          ('KISSING DOLLS', 'Toulon', 1.0, Decimal('10.99')),
          ('MILLION ACE', 'Bergamo', 1.0, Decimal('9.99')),
          ('TITANS JERK', 'Kimberley', 1.0, Decimal('9.99')),
          ('DARKO DORADO', 'Bhilwara', 1.0, Decimal('9.99')),
          ('SUNRISE LEAGUE', 'Nagareyama', 1.0, Decimal('9.99')),
          ('MILLION ACE', 'Gaziantep', 1.0, Decimal('9.99'))]

<tbody><tr>
    <th>title</th>
    <th>city</th>
    <th>month</th>
    <th>revenue</th>
</tr>
<tr>
    <td>SHOW LORD</td>
    <td>Mannheim</td>
    <td>1.0</td>
    <td>11.99</td>
</tr>
<tr>
    <td>AMERICAN CIRCUS</td>
    <td>Callao</td>
    <td>1.0</td>
    <td>10.99</td>
</tr>
<tr>
    <td>CASUALTIES ENCINO</td>
    <td>Warren</td>
    <td>1.0</td>
    <td>10.99</td>
```

```
</tr>
<tr>
    <td>TELEGRAPH VOYAGE</td>
    <td>Naala-Porto</td>
    <td>1.0</td>
    <td>10.99</td>
</tr>
<tr>
    <td>KISSING DOLLS</td>
    <td>Toulon</td>
    <td>1.0</td>
    <td>10.99</td>
</tr>
<tr>
    <td>MILLION ACE</td>
    <td>Bergamo</td>
    <td>1.0</td>
    <td>9.99</td>
</tr>
<tr>
    <td>TITANS JERK</td>
    <td>Kimberley</td>
    <td>1.0</td>
    <td>9.99</td>
</tr>
<tr>
    <td>DARKO DORADO</td>
    <td>Bhilwara</td>
    <td>1.0</td>
    <td>9.99</td>
</tr>
<tr>
    <td>SUNRISE LEAGUE</td>
    <td>Nagareyama</td>
    <td>1.0</td>
    <td>9.99</td>
</tr>
<tr>
    <td>MILLION ACE</td>
    <td>Gaziantep</td>
    <td>1.0</td>
    <td>9.99</td>
</tr>
```