Investigate a Relational Database - Project

Rawhi Al Rae

Question Set # 1

Set 1 – Question 1

We want to understand more about the movies that families are watching. The following categories are considered family movies: Animation, Children, Classics, Comedy, Family and Music.

- **Create a query that lists each movie, the film category it is classified in, and the number of times it has been rented out.**
- For this query, you will need 5 tables: Category, Film_Category, Inventory, Rental and Film. Your solution should have three columns: Film title, Category name and Count of Rentals.

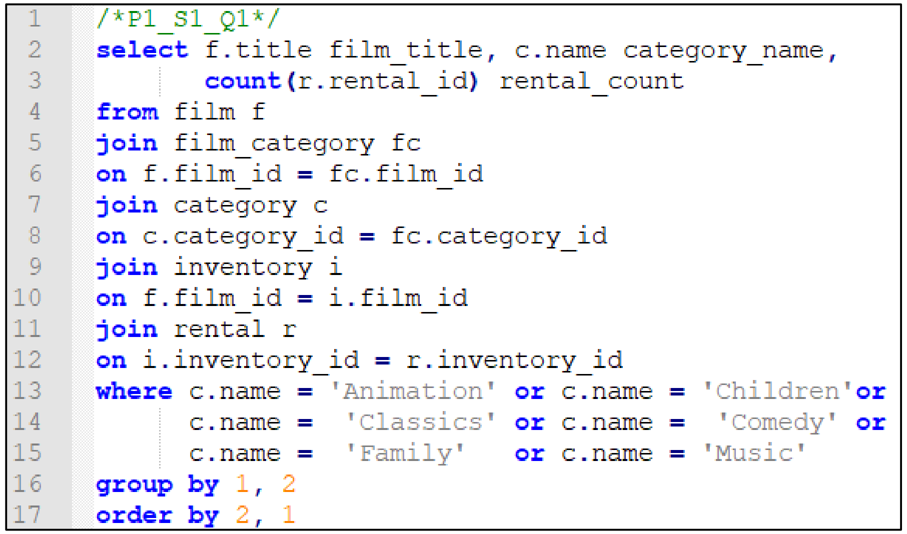
Set 1 – Question 1 – Solution

After Running the query the resulting table consists of three columns, which are:

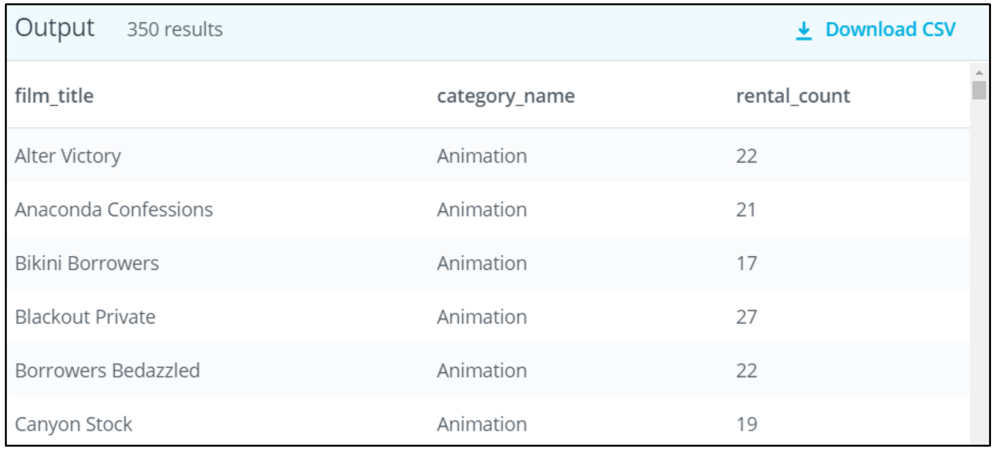
- film_title
- category_Name
- rental_count

As shown in the output figure:

- the number of records are 350
- The records are sorted in ascending order by category_name then by the film_title.



```
/*P1_S1_Q1*/
select f.title film_title, c.name category_name,
       count(r.rental_id) rental_count
from film f
join film_category fc
on f.film_id = fc.film_id
join category c
on c.category_id = fc.category_id
join inventory i
on f.film_id = i.film_id
join rental r
on i.inventory_id = r.inventory_id
where c.name = 'Animation' or c.name = 'Children'or
      c.name =  'Classics' or c.name =  'Comedy' or
      c.name =  'Family'   or c.name = 'Music'
group by 1, 2
order by 2, 1
```

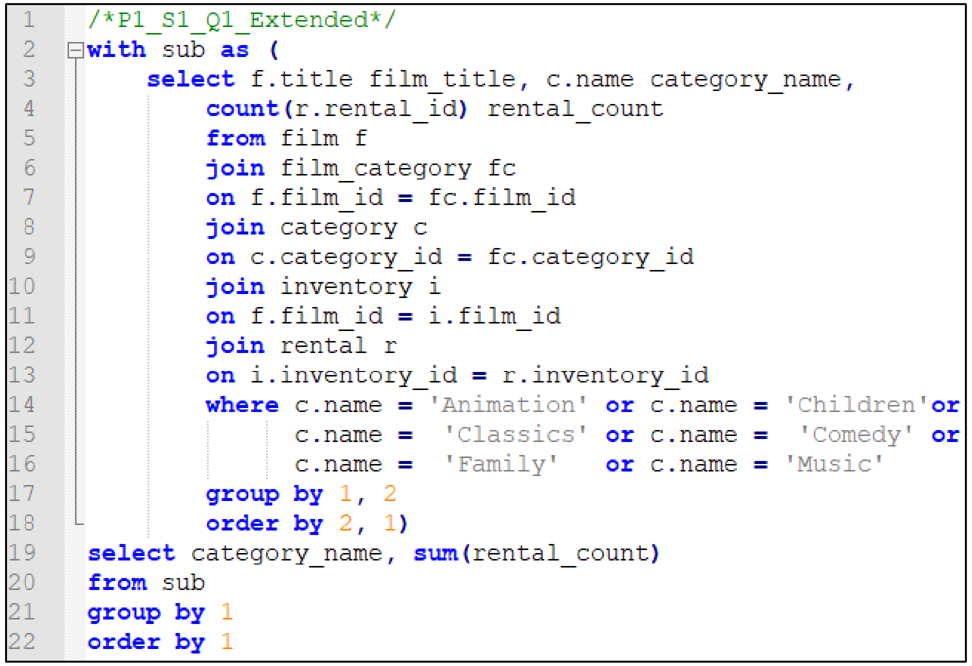


film_title	category_name	rental_count
Alter Victory	Animation	22
Anaconda Confessions	Animation	21
Bikini Borrowers	Animation	17
Blackout Private	Animation	27
Borrowers Bedazzled	Animation	22
Canyon Stock	Animation	19

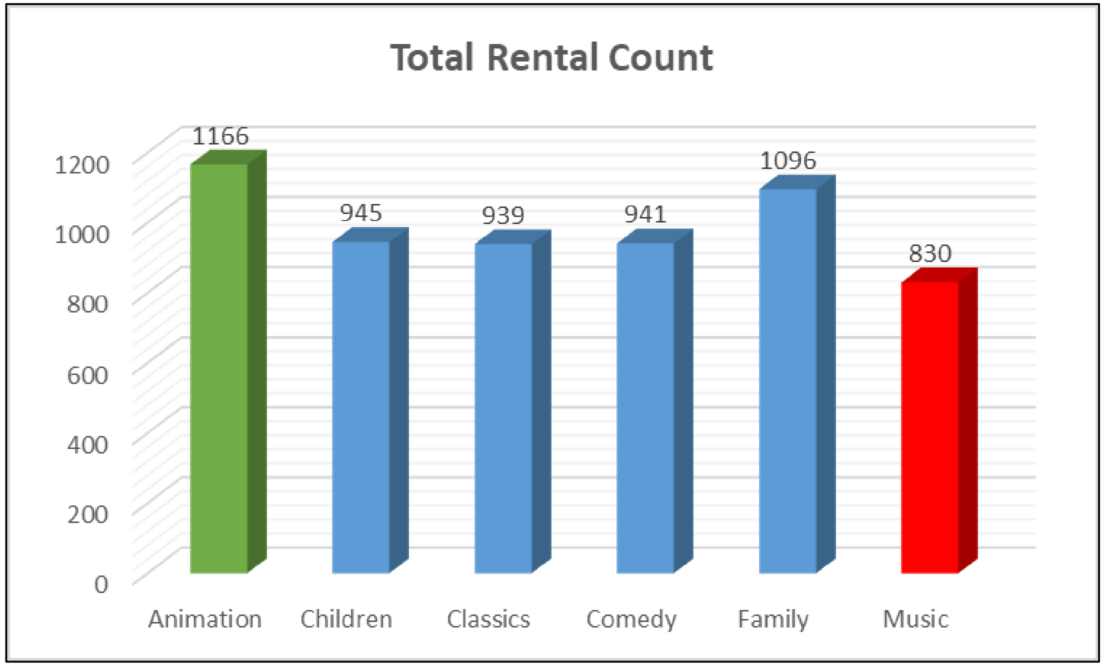
Code Part: P1_S1_Q1

Set 1 – Question 1 – Solution

- Since the number of records are huge to be illustrated, the total Rental count for each category were calculated using the presented query (that used the previous query in the previous slide as a subquery). The result illustrated by the presented chart. The chart shows that Animation category was the highest in total rental, when the music category was the lowest.



```sql
/*P1_S1_Q1_Extended*/
with sub as (
    select f.title film_title, c.name category_name,
        count(r.rental_id) rental_count
        from film f
        join film_category fc
        on f.film_id = fc.film_id
        join category c
        on c.category_id = fc.category_id
        join inventory i
        on f.film_id = i.film_id
        join rental r
        on i.inventory_id = r.inventory_id
        where c.name = 'Animation' or c.name = 'Children'or
              c.name =  'Classics' or c.name =  'Comedy' or
              c.name =  'Family'   or c.name = 'Music'
        group by 1, 2
        order by 2, 1)
select category_name, sum(rental_count)
from sub
group by 1
order by 1
```



Code Part: P1_S1_Q1_Extended

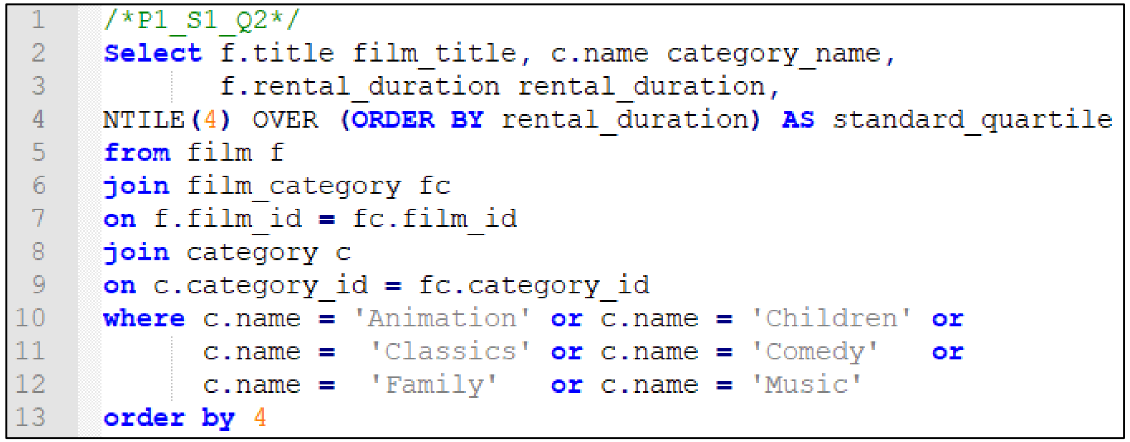
Set 1 – Question 2

- Now we need to know how the length of rental duration of these family-friendly movies compares to the duration that all movies are rented for.
- **Can you provide a table with the movie titles and divide them into 4 levels (first_quarter, second_quarter, third_quarter, and final_quarter) based on the quartiles (25%, 50%, 75%) of the rental duration for movies across all categories?** Make sure to also indicate the category that these family-friendly movies fall into.
- You should only need the category, film_category, and film tables to answer this and the next questions.

Set 1 – Question 2 – Solution

After Running the query the resulting table consists of four columns, which are:

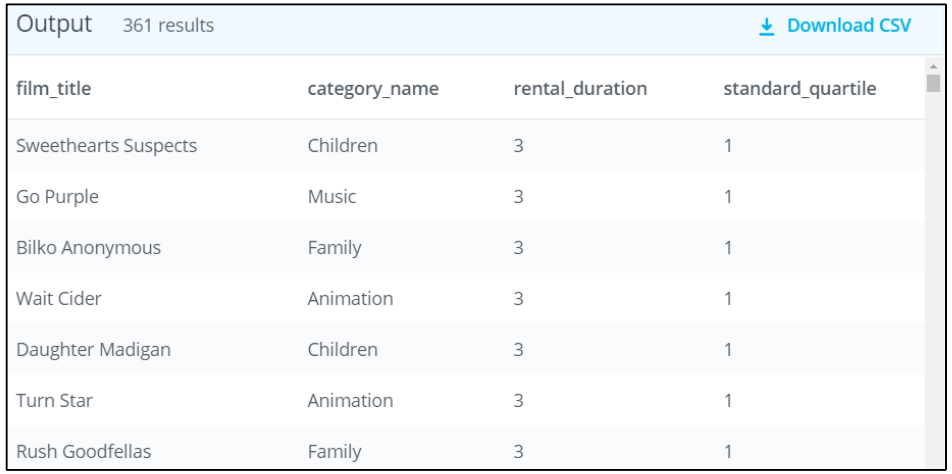
- film_title
- category_Name
- rental_duration
- standard_quartile



```
/*P1_S1_Q2*/
Select f.title film_title, c.name category_name,
       f.rental_duration rental_duration,
NTILE(4) OVER (ORDER BY rental_duration) AS standard_quartile
from film f
join film_category fc
on f.film_id = fc.film_id
join category c
on c.category_id = fc.category_id
where c.name = 'Animation' or c.name = 'Children' or
      c.name =  'Classics' or c.name = 'Comedy'   or
      c.name =  'Family'   or c.name = 'Music'
order by 4
```

As shown in the output figure:

- the number of records are 361
- The records are sorted in ascending order by standard_quartile



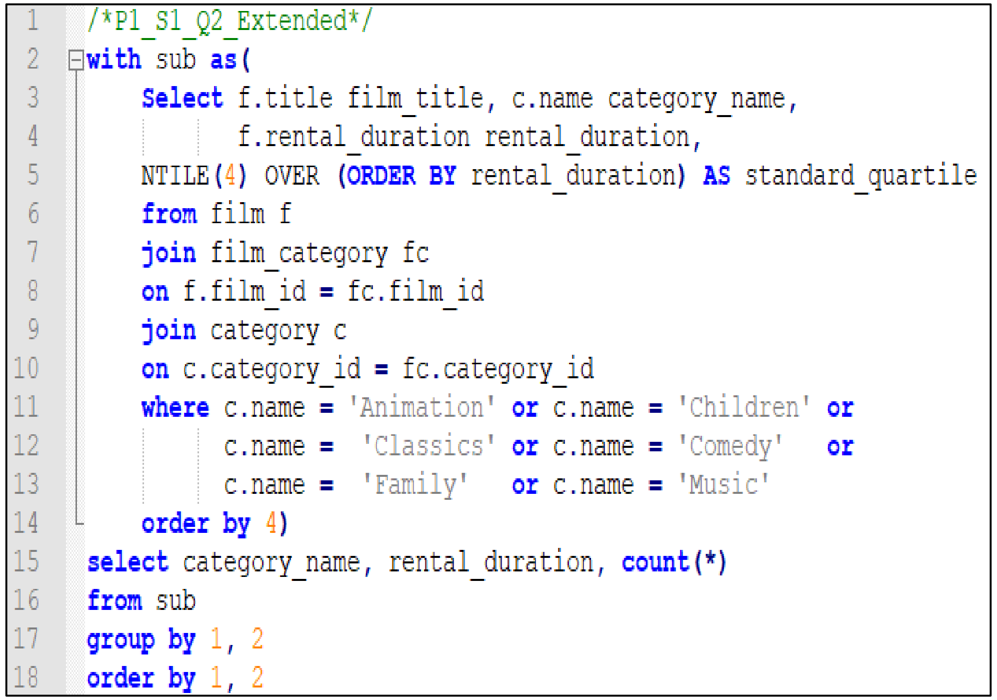
Output 361 results

film_title	category_name	rental_duration	standard_quartile
Sweethearts Suspects	Children	3	1
Go Purple	Music	3	1
Bilko Anonymous	Family	3	1
Wait Cider	Animation	3	1
Daughter Madigan	Children	3	1
Turn Star	Animation	3	1
Rush Goodfellas	Family	3	1

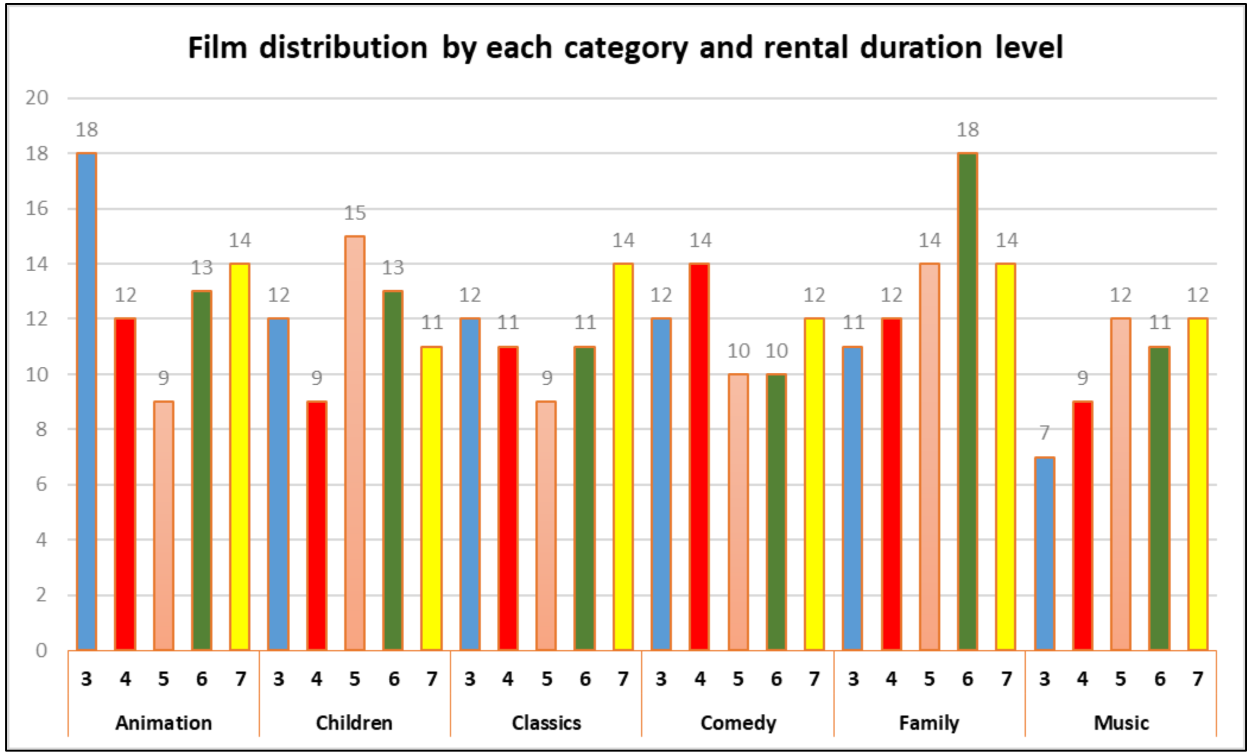
Code Part: P1_S1_Q2

Set 1 – Question 2 – Solution

- Since the number of records are huge to be illustrated, the film distribution per category for each duration level had been calculated using the presented query (that used the previous query in the previous slide as a subquery).



```
/*P1_S1_Q2_Extended*/
with sub as(
    Select f.title film_title, c.name category_name,
           f.rental_duration rental_duration,
    NTILE(4) OVER (ORDER BY rental_duration) AS standard_quartile
    from film f
    join film_category fc
    on f.film_id = fc.film_id
    join category c
    on c.category_id = fc.category_id
    where c.name = 'Animation' or c.name = 'Children' or
          c.name =  'Classics' or c.name = 'Comedy'   or
          c.name =  'Family'   or c.name = 'Music'
    order by 4)
select category_name, rental_duration, count(*)
from sub
group by 1, 2
order by 1, 2
```



Code Part: P1_S1_Q2_Extended

Set 1 – Question 3

- Finally, provide a table with the family-friendly film category, each of the quartiles, and the corresponding count of movies within each combination of film category for each corresponding rental duration category. The resulting table should have three columns:
  - Category
  - Rental length category
  - Count
- The Count column should be sorted first by Category and then by Rental Duration category.

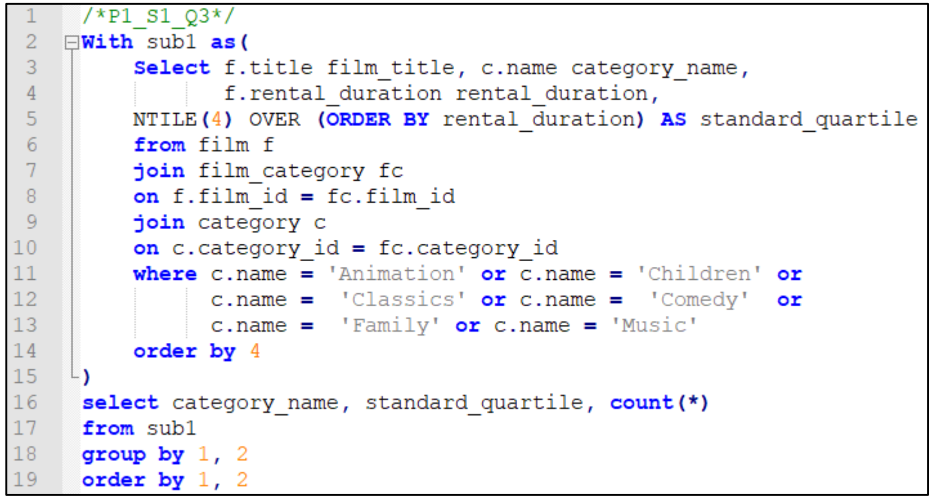
Set 1 – Question 3 – Solution

After Running the query the resulting table consists of three columns, which are:

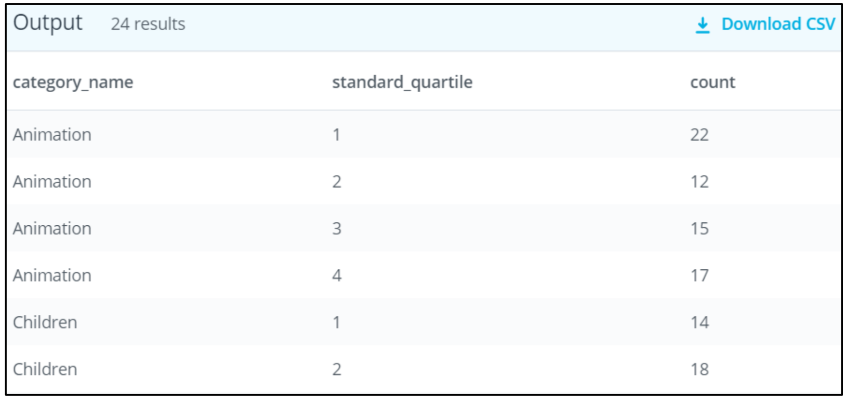
- category_Name
- standard_quartile
- count

As shown in the output figure:

- the number of records are 24
- The records are sorted in ascending order by category_Name then standard_quartile



```sql
/*P1_S1_Q3*/
With sub1 as(
    Select f.title film_title, c.name category_name,
           f.rental_duration rental_duration,
    NTILE(4) OVER (ORDER BY rental_duration) AS standard_quartile
    from film f
    join film_category fc
    on f.film_id = fc.film_id
    join category c
    on c.category_id = fc.category_id
    where c.name = 'Animation' or c.name = 'Children' or
          c.name =  'Classics' or c.name =  'Comedy'  or
          c.name =  'Family' or c.name = 'Music'
    order by 4
)
select category_name, standard_quartile, count(*)
from sub1
group by 1, 2
order by 1, 2
```

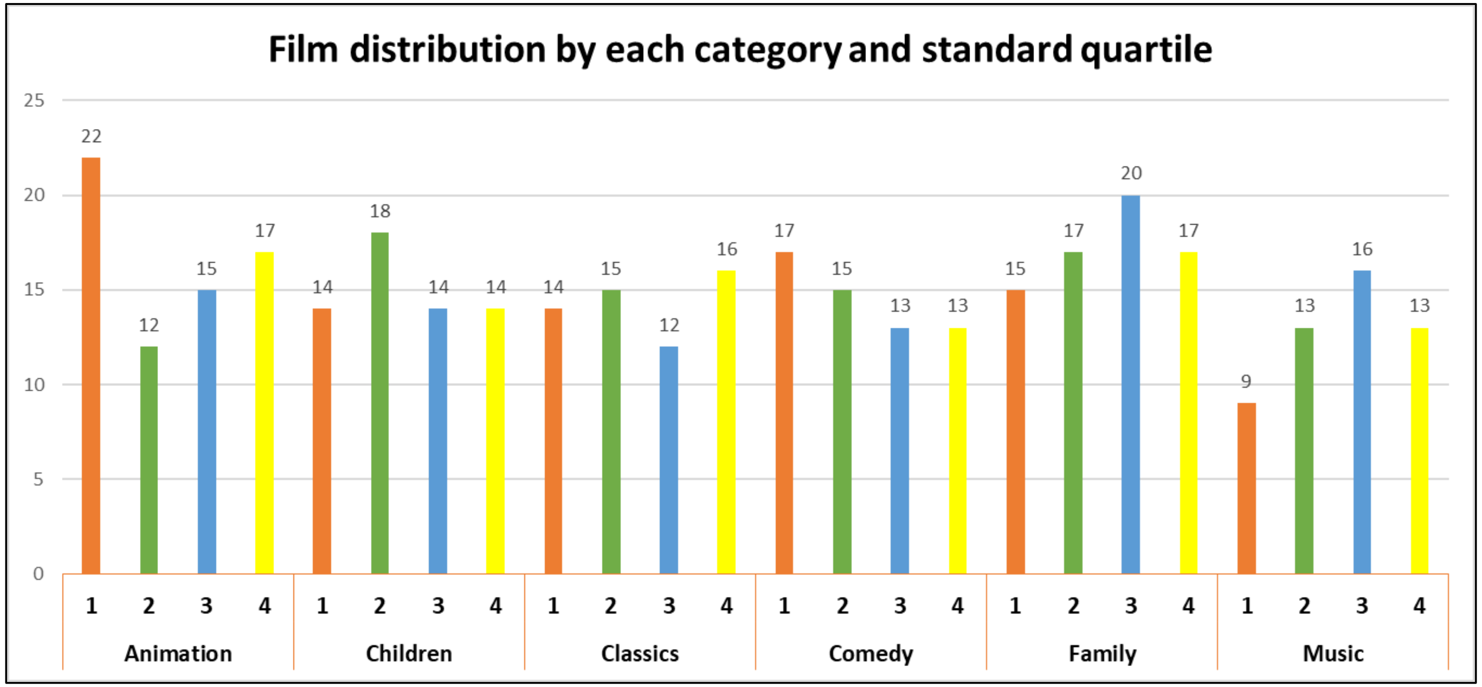


category_name	standard_quartile	count
Animation	1	22
Animation	2	12
Animation	3	15
Animation	4	17
Children	1	14
Children	2	18

Code Part: P1_S1_Q3

Set 1 – Question 3 – Solution

- The figure shows the film distribution over the categories and the standard quartile. The standard quartile refer by 4 levels (first_quarter, second_quarter, third_quarter, and final_quarter) based on the quartiles (25%, 50%, 75%) of the rental duration for movies across all categories. So, when "Animation" is the highest for quartiles 1 and 4, "Children" is the highest for quartile 2, and "Family" is the highest for quartile 3.



Code Part: P1_S1_Q3

Question Set # 2

Set 2 – Question 1

- We want to find out how the two stores compare in their count of rental orders during every month for all the years we have data for.
- **Write a query that returns the store ID for the store, the year and month and the number of rental orders each store has fulfilled for that month.**
- **Your table should include a column for each of the following: year, month, store ID and count of rental orders fulfilled during that month.**
- The count of rental orders is sorted in descending order.

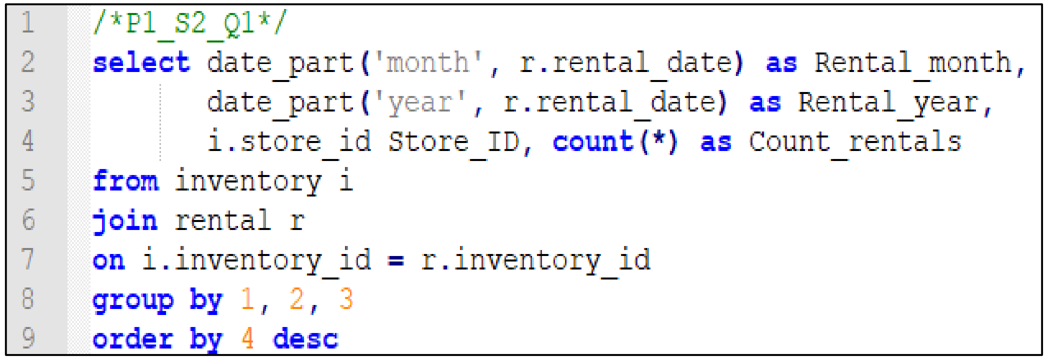
Set 2 – Question 1 – Solution

After Running the query the resulting table consists of four columns, which are:

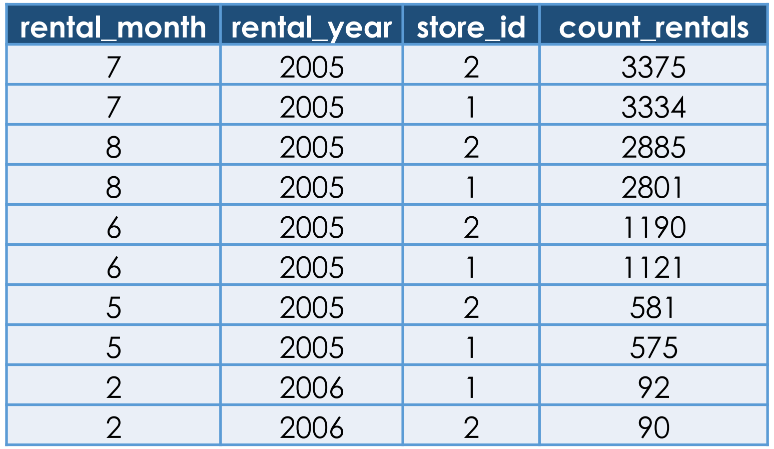
- rental_month
- rental_year
- store_id
- count_rentals

As shown in the output figure:

- the number of records are 12
- The records are sorted in descending order by count_rentals



```
/*P1_S2_Q1*/
select date_part('month', r.rental_date) as Rental_month,
       date_part('year', r.rental_date) as Rental_year,
       i.store_id Store_ID, count(*) as Count_rentals
from inventory i
join rental r
on i.inventory_id = r.inventory_id
group by 1, 2, 3
order by 4 desc
```

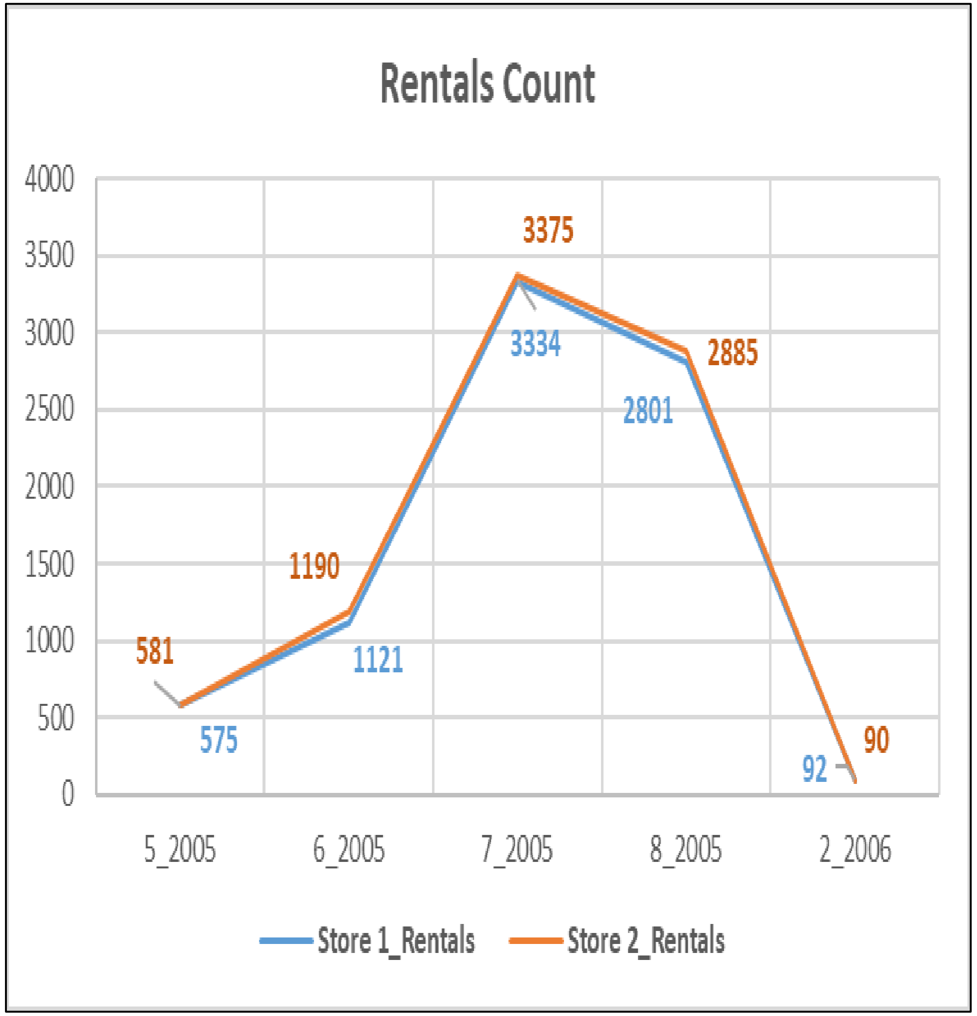


rental_month	rental_year	store_id	count_rentals
7	2005	2	3375
7	2005	1	3334
8	2005	2	2885
8	2005	1	2801
6	2005	2	1190
6	2005	1	1121
5	2005	2	581
5	2005	1	575
2	2006	1	92
2	2006	2	90

Code Part: P1_S2_Q1

Set 2 – Question 1 – Solution

- The figure shows that:
  - the highest rentals counts were on July 2005 for both stores.
  - The lowest rentals counts were on February 2006 for both stores.
  - The rentals counts for both stores are almost identical over the five months.



Code Part: P1_S2_Q1

Set 2 – Question 2

- We would like to know who were our top 10 paying customers, how many payments they made on a monthly basis during 2007, and what was the amount of the monthly payments.
- **Can you write a query to capture the customer name, month and year of payment, and total payment amount for each month by these top 10 paying customers?**
- The results are sorted first by customer name and then for each month. As you can see, total amounts per month are listed for each customer.

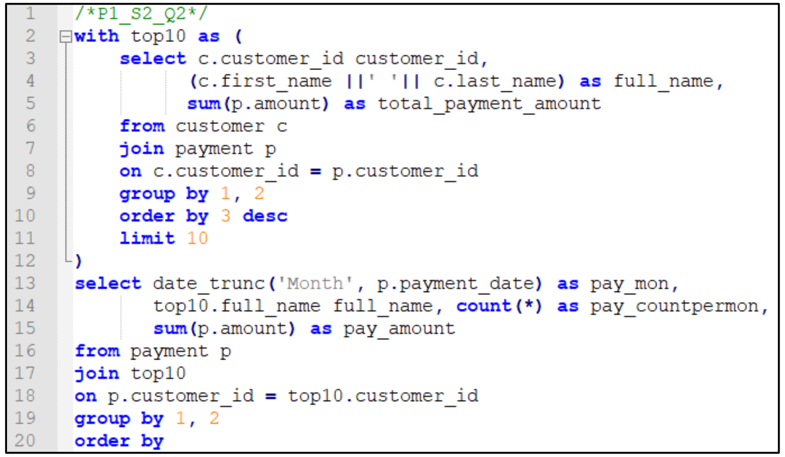
Set 2 – Question 2 – Solution

After Running the query the resulting table consists of four columns, which are:

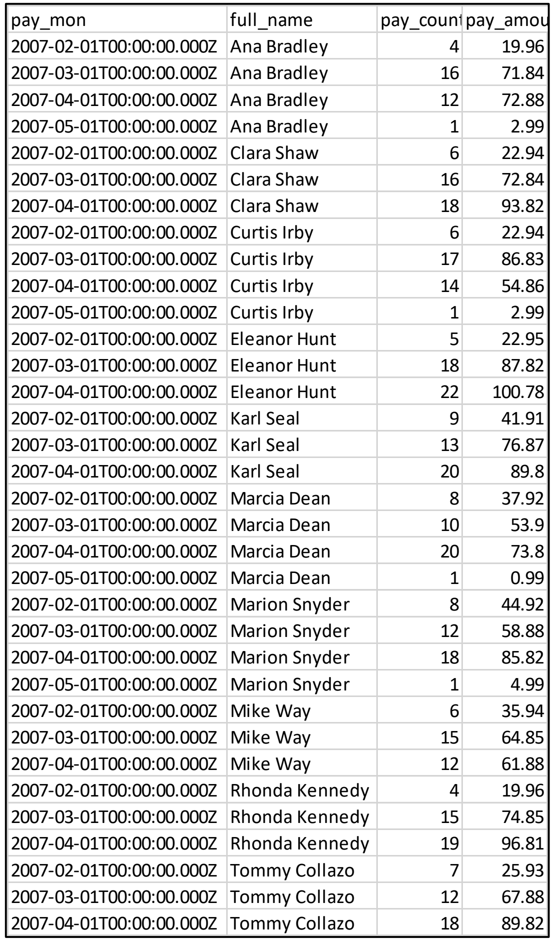
- pay_mon: payment month
- full_name: first and last name of payer
- pay_count: payment count/month
- count_rentals: payment/month

As shown in the output figure:

- the number of records are 34
- The records are sorted in ascending order by full_name then pay_mon
- The records represent payment count and amount per month for the top 10 payers based on the total payment done during 2007.



```
/*P1_S2_Q2*/
with top10 as (
    select c.customer_id customer_id,
           (c.first_name ||' '|| c.last_name) as full_name,
           sum(p.amount) as total_payment_amount
    from customer c
    join payment p
    on c.customer_id = p.customer_id
    group by 1, 2
    order by 3 desc
    limit 10
)
select date_trunc('Month', p.payment_date) as pay_mon,
       top10.full_name full_name, count(*) as pay_countpermon,
       sum(p.amount) as pay_amount
from payment p
join top10
on p.customer_id = top10.customer_id
group by 1, 2
order by
```

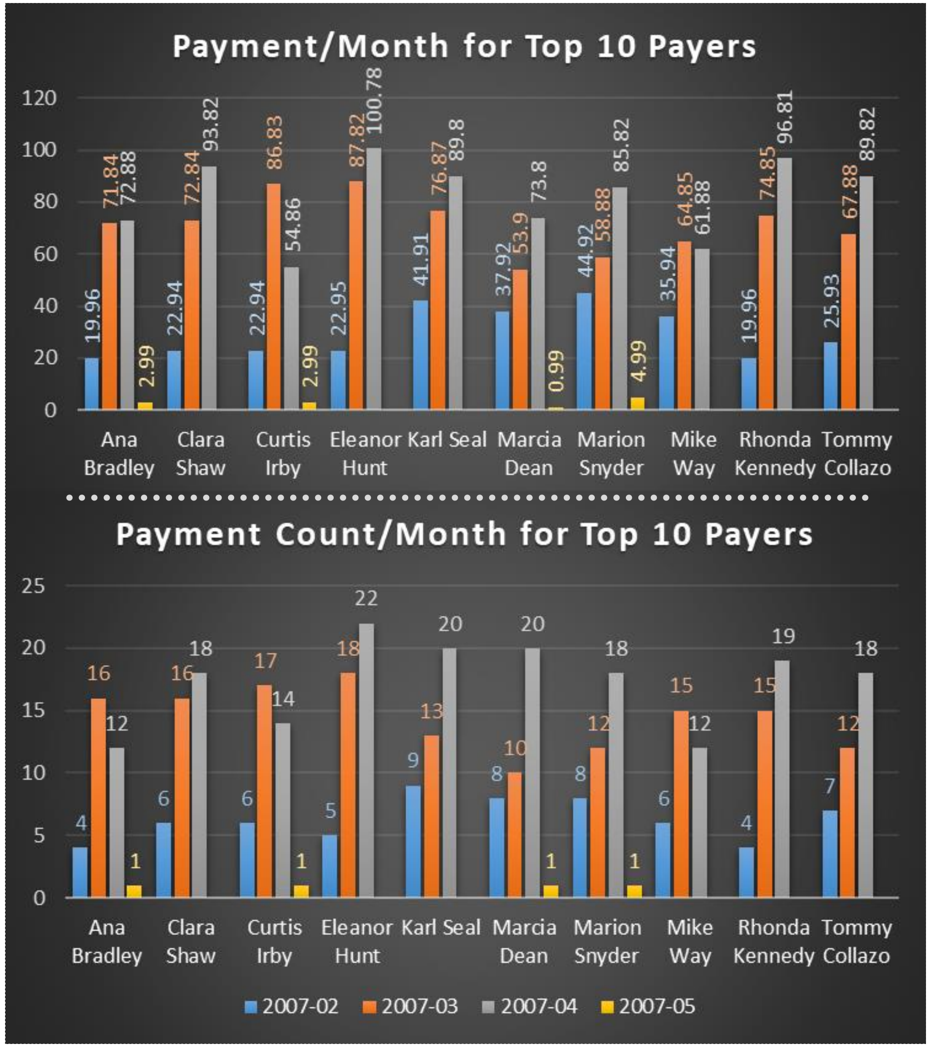


pay_mon	full_name	pay_count	pay_amou
2007-02-01T00:00:00.000Z	Ana Bradley	4	19.96
2007-03-01T00:00:00.000Z	Ana Bradley	16	71.84
2007-04-01T00:00:00.000Z	Ana Bradley	12	72.88
2007-05-01T00:00:00.000Z	Ana Bradley	1	2.99
2007-02-01T00:00:00.000Z	Clara Shaw	6	22.94
2007-03-01T00:00:00.000Z	Clara Shaw	16	72.84
2007-04-01T00:00:00.000Z	Clara Shaw	18	93.82
2007-02-01T00:00:00.000Z	Curtis Irby	6	22.94
2007-03-01T00:00:00.000Z	Curtis Irby	17	86.83
2007-04-01T00:00:00.000Z	Curtis Irby	14	54.86
2007-05-01T00:00:00.000Z	Curtis Irby	1	2.99
2007-02-01T00:00:00.000Z	Eleanor Hunt	5	22.95
2007-03-01T00:00:00.000Z	Eleanor Hunt	18	87.82
2007-04-01T00:00:00.000Z	Eleanor Hunt	22	100.78
2007-02-01T00:00:00.000Z	Karl Seal	9	41.91
2007-03-01T00:00:00.000Z	Karl Seal	13	76.87
2007-04-01T00:00:00.000Z	Karl Seal	20	89.8
2007-02-01T00:00:00.000Z	Marcia Dean	8	37.92
2007-03-01T00:00:00.000Z	Marcia Dean	10	53.9
2007-04-01T00:00:00.000Z	Marcia Dean	20	73.8
2007-05-01T00:00:00.000Z	Marcia Dean	1	0.99
2007-02-01T00:00:00.000Z	Marion Snyder	8	44.92
2007-03-01T00:00:00.000Z	Marion Snyder	12	58.88
2007-04-01T00:00:00.000Z	Marion Snyder	18	85.82
2007-05-01T00:00:00.000Z	Marion Snyder	1	4.99
2007-02-01T00:00:00.000Z	Mike Way	6	35.94
2007-03-01T00:00:00.000Z	Mike Way	15	64.85
2007-04-01T00:00:00.000Z	Mike Way	12	61.88
2007-02-01T00:00:00.000Z	Rhonda Kennedy	4	19.96
2007-03-01T00:00:00.000Z	Rhonda Kennedy	15	74.85
2007-04-01T00:00:00.000Z	Rhonda Kennedy	19	96.81
2007-02-01T00:00:00.000Z	Tommy Collazo	7	25.93
2007-03-01T00:00:00.000Z	Tommy Collazo	12	67.88
2007-04-01T00:00:00.000Z	Tommy Collazo	18	89.82

Code Part: P1_S2_Q2

Set 2 – Question 2 – Solution

- The figure shows:
  - The name of the ten top payers for 2007.
  - The top part is the payment amount for every payer/month.
  - The bottom part is the payment count for every payer/month.
  - The highest count and amount of payment were during March and April of 2007.
  - The lowest count and amount of payment were during February and May of 2007.
  - The payments are dramatically low during May in comparison to other months.



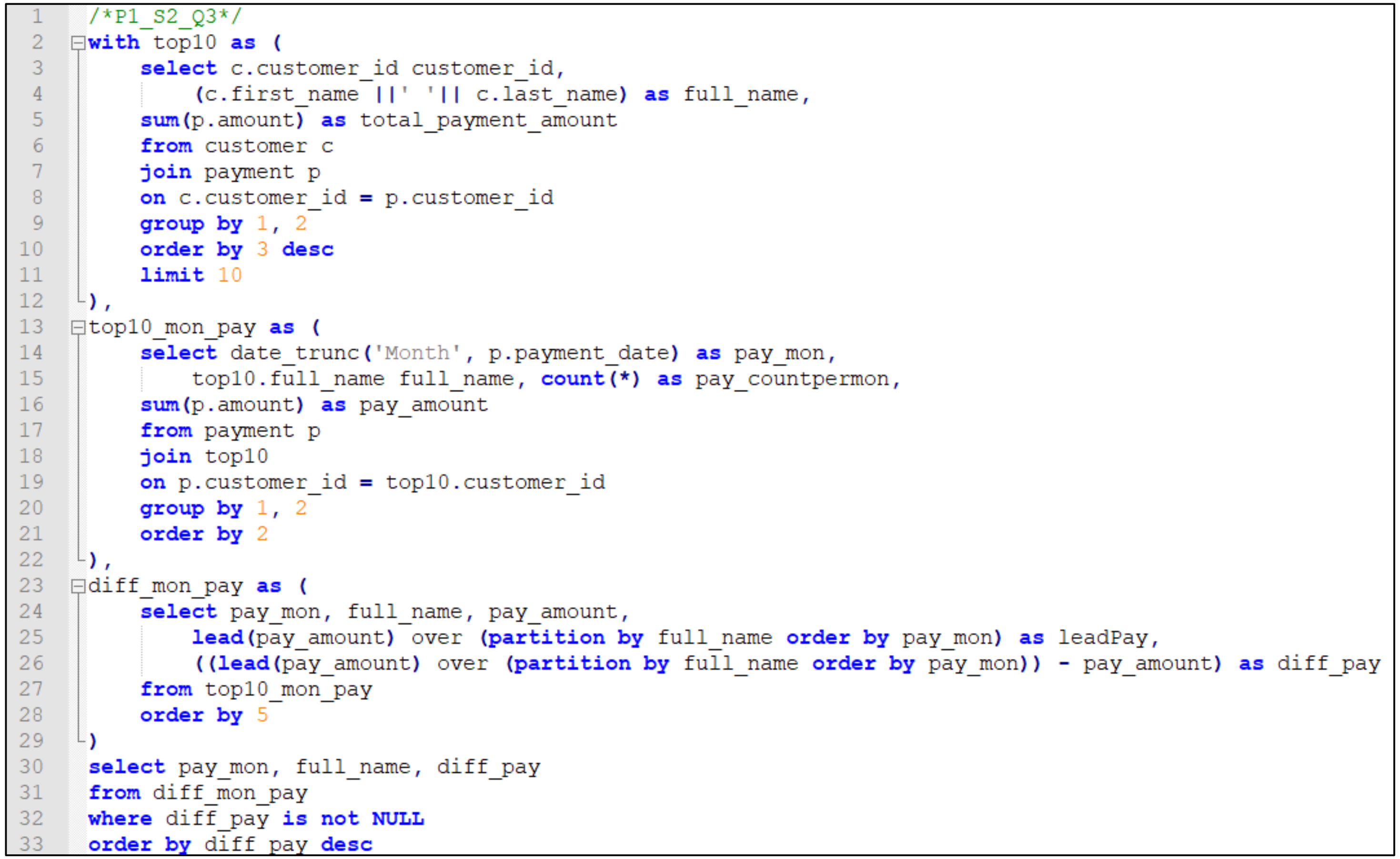
Code Part: P1_S2_Q2

Set 2 – Question 3

- Finally, for each of these top 10 paying customers, I would like to find out the difference across their monthly payments during 2007.
- Please go ahead and **write a query to compare the payment amounts in each successive month.**
- Repeat this for each of these 10 paying customers. Also, it will be tremendously helpful if you can identify the customer name who paid the most difference in terms of payments.

Set 2 – Question 3 – Solution

- The way followed to solve the problem is through several stages of sub-queries, which are:
  1. Find top 10 payers:
     a) customer_id
     b) full_name: first_name + last_name
     c) total_payment_amount
  2. Find payment/month for every of top 10 payers:
     a) pay_mon: payment month
     b) full_name
     c) pay_coutpermon: payments count
     d) pay_amount: payments amount
  3. Find difference of payment/month for every of top 10 payers.:
     a) pay_mon: payment month
     b) full_name
     c) pay_amount
     d) leadPay: payment of next record for a payer
     e) diff_pay: month payment difference for payer
  4. Define a final table, to take out NULLL values:
     a) pay_mon: payment month
     b) full_name
     c) diff_pay

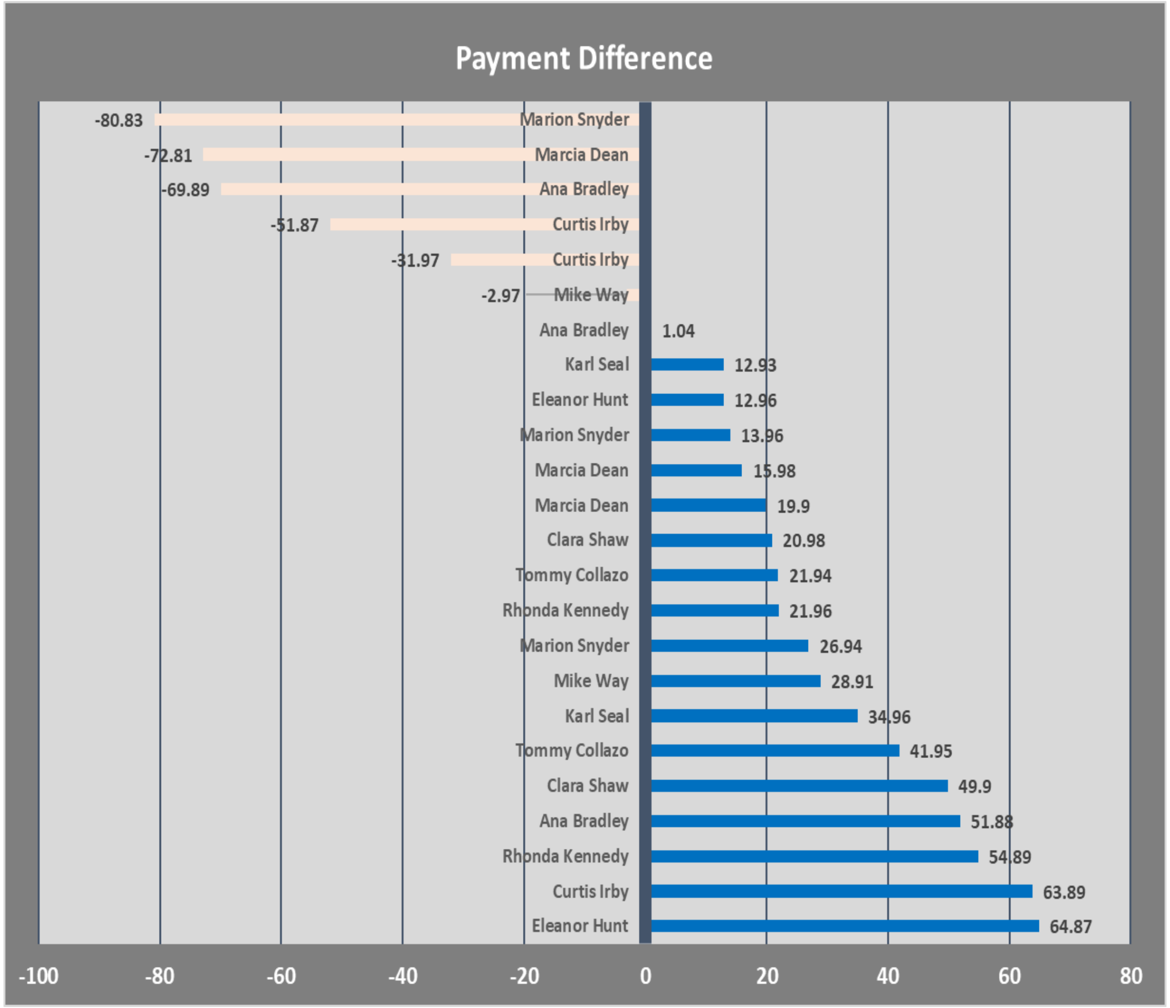


```sql
/*P1_S2_Q3*/
with top10 as (
    select c.customer_id customer_id,
        (c.first_name ||' '|| c.last_name) as full_name,
    sum(p.amount) as total_payment_amount
    from customer c
    join payment p
    on c.customer_id = p.customer_id
    group by 1, 2
    order by 3 desc
    limit 10
),
top10_mon_pay as (
    select date_trunc('Month', p.payment_date) as pay_mon,
        top10.full_name full_name, count(*) as pay_countpermon,
    sum(p.amount) as pay_amount
    from payment p
    join top10
    on p.customer_id = top10.customer_id
    group by 1, 2
    order by 2
),
diff_mon_pay as (
    select pay_mon, full_name, pay_amount,
        lead(pay_amount) over (partition by full_name order by pay_mon) as leadPay,
        ((lead(pay_amount) over (partition by full_name order by pay_mon)) - pay_amount) as diff_pay
    from top10_mon_pay
    order by 5
)
select pay_mon, full_name, diff_pay
from diff_mon_pay
where diff_pay is not NULL
order by diff_pay desc
```

Code Part: P1_S2_Q3

Set 2 – Question 3 – Solution

- The figure defines the negative differences on top, and the positive differences on the bottom.
- The negative difference indicates the decreasing in payment for the payer between 2 months. The payer who has the highest decrement in payment is “Martion Snyder” for $ -80.83
- The positive difference indicates the increasing in payment for the payer between 2 months. The payer who has the highest increament in payment is “Eleanor Hunt” for $ +64.87
- The difference that close to zero indicate low change on payment, like the case for “Ana Bradly” and “Mike Way”.



Code Part: P1_S2_Q3