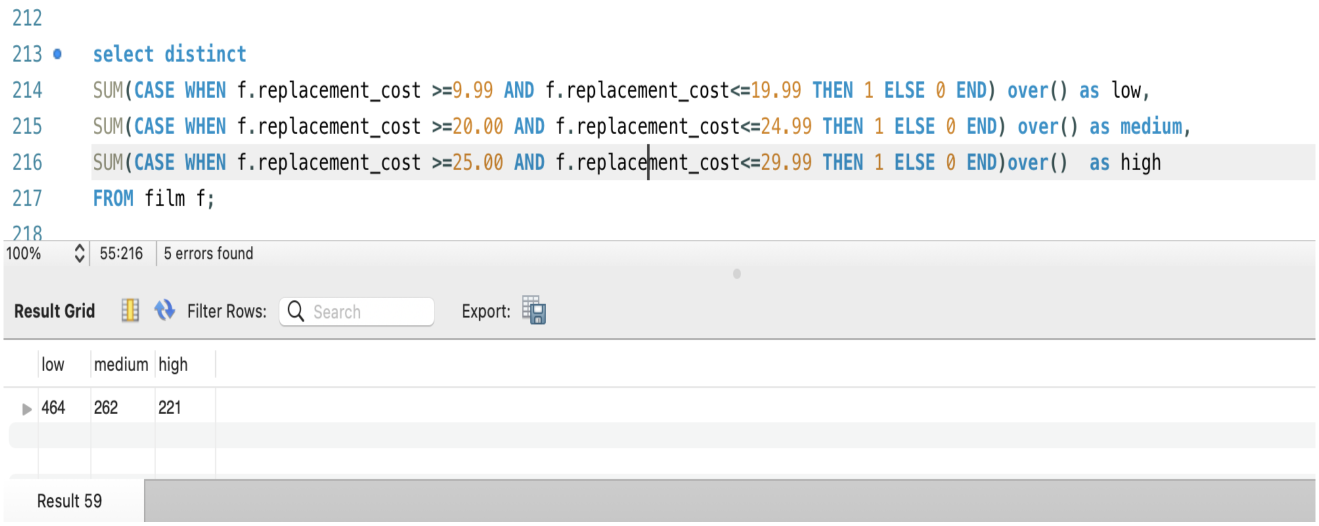
ADVANCE SQL ASSIGNMENT

Q1. Write a query that gives an overview of how many films have replacements costs in the following cost ranges

low: 9.99 - 19.99
medium: 20.00 - 24.99
high: 25.00 - 29.99

Soln:-

Query screenshot with Output:-



```sql
select distinct
SUM(CASE WHEN f.replacement_cost >=9.99 AND f.replacement_cost<=19.99 THEN 1 ELSE 0 END) over() as low,
SUM(CASE WHEN f.replacement_cost >=20.00 AND f.replacement_cost<=24.99 THEN 1 ELSE 0 END) over() as medium,
SUM(CASE WHEN f.replacement_cost >=25.00 AND f.replacement_cost<=29.99 THEN 1 ELSE 0 END)over()  as high
FROM film f;
```

low	medium	high
464	262	221

Approach:-

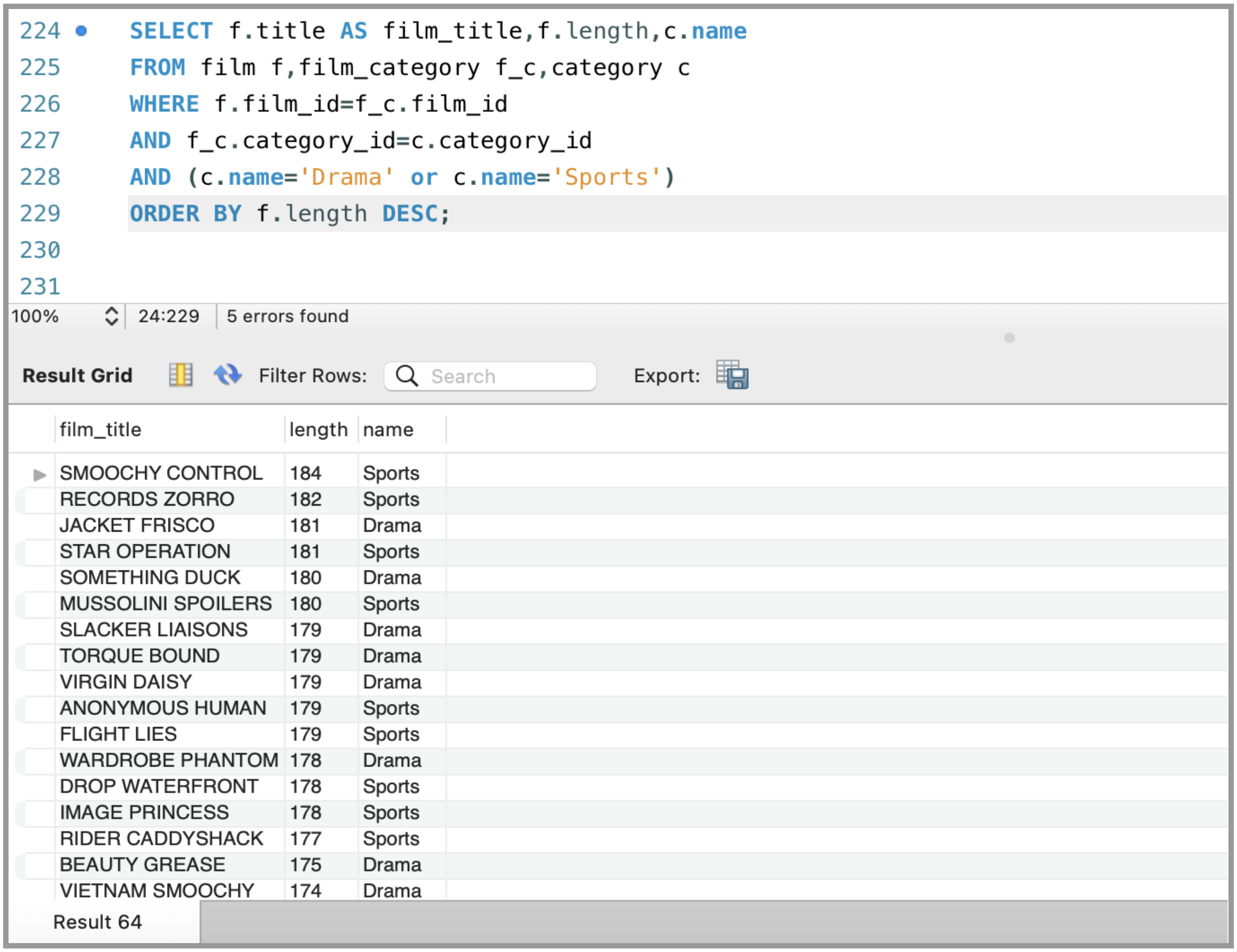
- Here in this particular question we have been asked to give the overview of the films where the replacement_cost is in the given range categorizing it into low,medium and high
- So I have used window function where I have used CASE statement for filtering the data according to the given range of replacement cost and i categorize those filtered data for each case statement as low,medium and high
- Logic in the case statement is if the particular replacement_cost is within the specified range for the particular category then returning 1 else 0 which we are summing up using SUM() function which results in total number of films in that particular range

Q2. Write a query to create a list of the film titles including their film title, film length and film category name ordered descendingly by the film length. Filter the results to only the movies in the category 'Drama' or 'Sports'.

Eg.	"STAR OPERATION"	"Sports"	181
	"JACKET FRISCO"	"Drama"	181

Soln:-

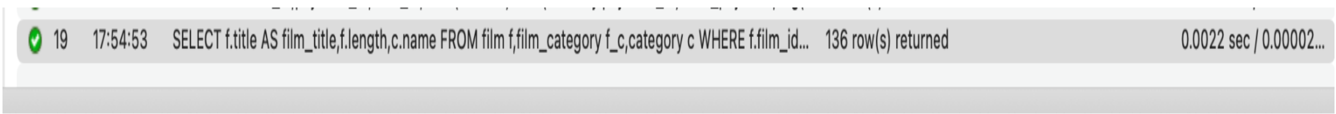
Query screenshot with Output:-



```sql
SELECT f.title AS film_title,f.length,c.name
FROM film f,film_category f_c,category c
WHERE f.film_id=f_c.film_id
AND f_c.category_id=c.category_id
AND (c.name='Drama' or c.name='Sports')
ORDER BY f.length DESC;
```

film_title	length	name
SMOOCHY CONTROL	184	Sports
RECORDS ZORRO	182	Sports
JACKET FRISCO	181	Drama
STAR OPERATION	181	Sports
SOMETHING DUCK	180	Drama
MUSSOLINI SPOILERS	180	Sports
SLACKER LIAISONS	179	Drama
TORQUE BOUND	179	Drama
VIRGIN DAISY	179	Drama
ANONYMOUS HUMAN	179	Sports
FLIGHT LIES	179	Sports
WARDROBE PHANTOM	178	Drama
DROP WATERFRONT	178	Sports
IMAGE PRINCESS	178	Sports
RIDER CADDYSHACK	177	Sports
BEAUTY GREASE	175	Drama
VIETNAM SMOOCHY	174	Drama

Total Rows Fetched: 136



```
19    17:54:53    SELECT f.title AS film_title,f.length,c.name FROM film f,film_category f_c,category c WHERE f.film_id...    136 row(s) returned    0.0022 sec / 0.00002...
```

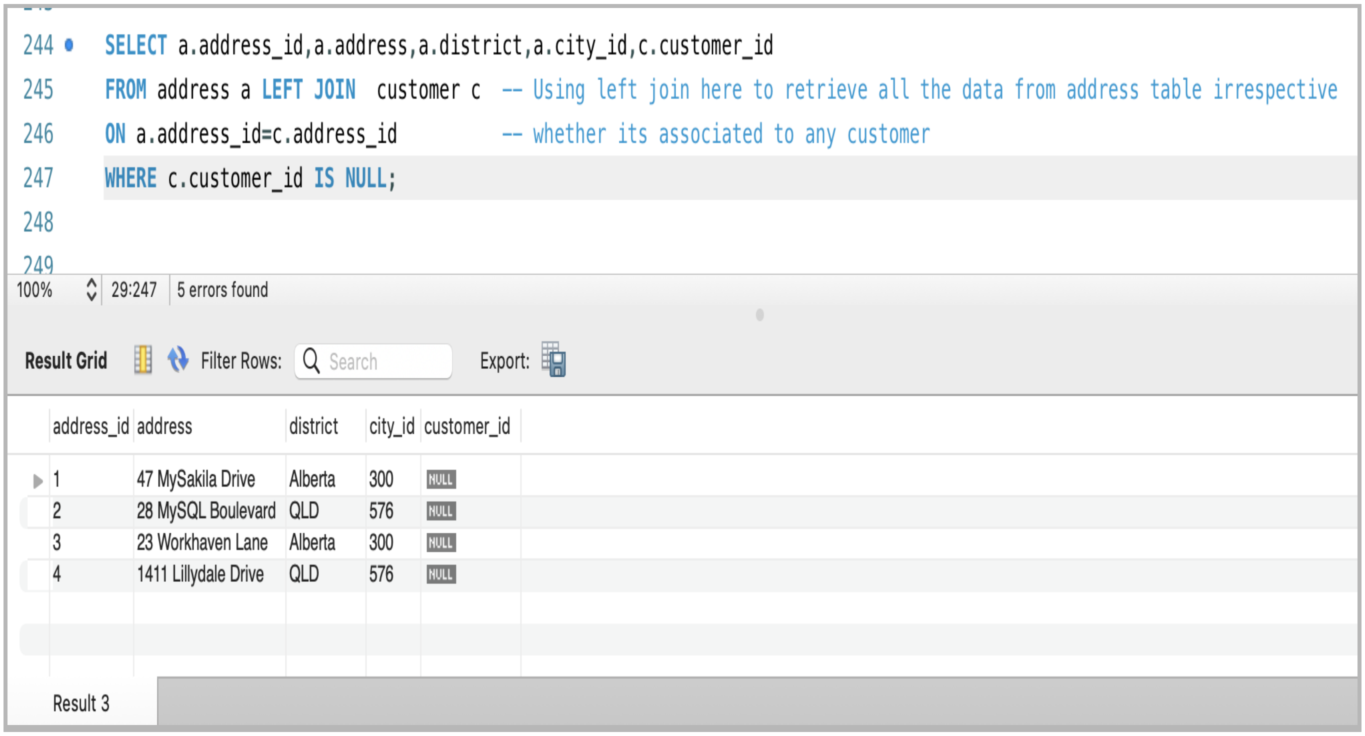
Approach:-

- Here we have to create a list of the film titles including their film title, film length and film category name ordered descendingly by the film length for the category “sports” and “drama”
- So for that I have joined three tables i.e., film, film_category, category and extracted the film_title, film_length and film_category by applying the filter for category name to be only “Sports” and “Drama” ordering by the length of the film in descending order.

Q3. Write a query to create a list of the addresses that are not associated to any customer.

Soln:-

Query screenshot with Output:-



```sql
SELECT a.address_id,a.address,a.district,a.city_id,c.customer_id
FROM address a LEFT JOIN  customer c  -- Using left join here to retrieve all the data from address table irrespective
ON a.address_id=c.address_id          -- whether its associated to any customer
WHERE c.customer_id IS NULL;
```

5 errors found

address_id	address	district	city_id	customer_id
1	47 MySakila Drive	Alberta	300	NULL
2	28 MySQL Boulevard	QLD	576	NULL
3	23 Workhaven Lane	Alberta	300	NULL
4	1411 Lillydale Drive	QLD	576	NULL

Total Rows Fetched: 4

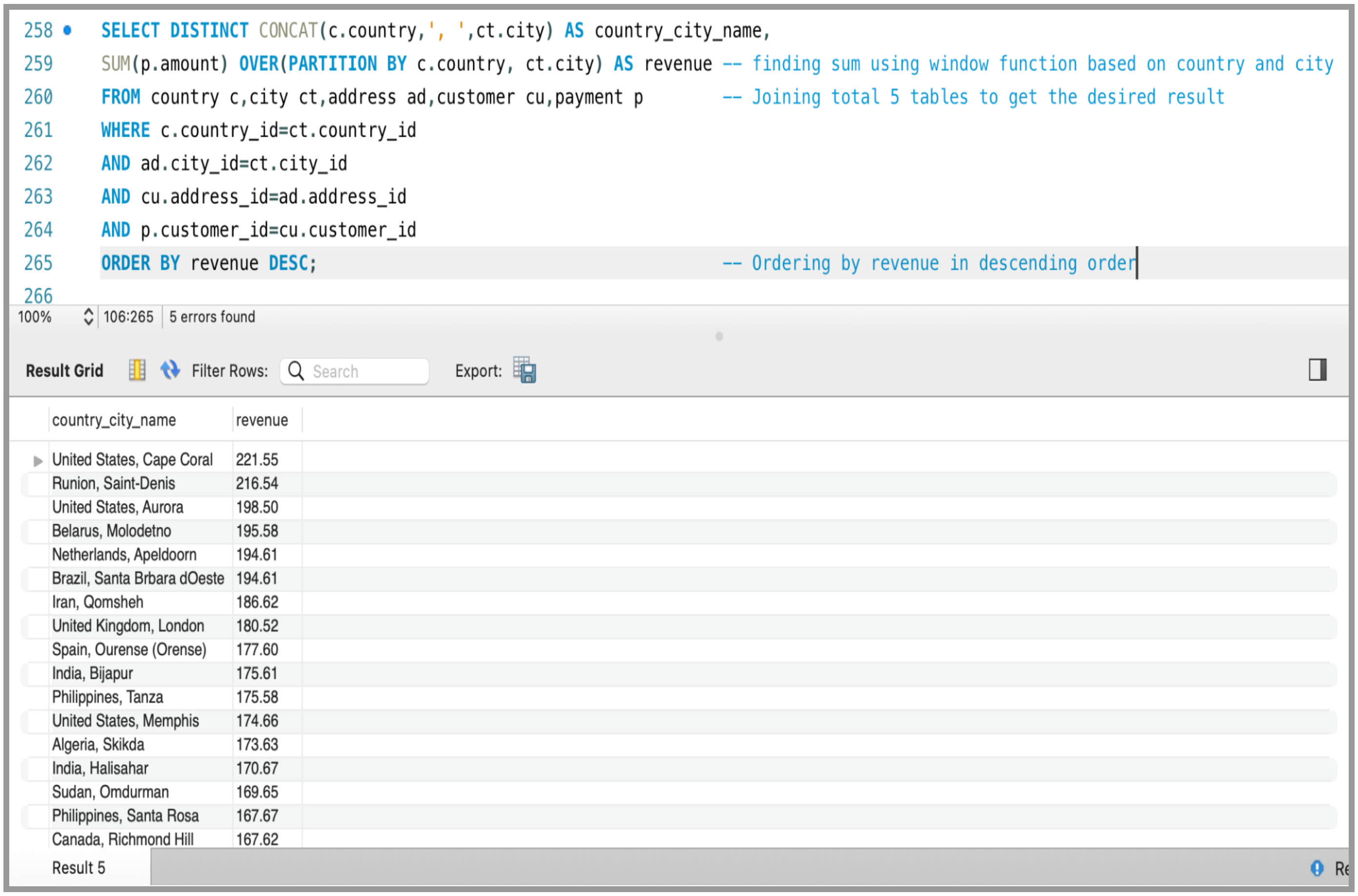
Approach:-

- We have been asked to create a list of the addresses that are not associated to any customer
- So For that I have used two tables i.e., address and customer to extract address_id, address, district, city_id and customer_id
- I have used left join above to retrieve all the data from the address table irrespective of whether it is present in the customer table. So whichever customer_id is not associated with any address will result in null values which i am using in the where clause to filter out the data where customer_id is null which will give me all the address which are not related to any custome.

Q4. Write a query to create a list of the revenue (sum of amount) grouped by a column in the format "country, city" ordered in decreasing amount of revenue.
eg. "Poland, Bydgoszcz" 52.88

Soln:

Query screenshot with Output:-

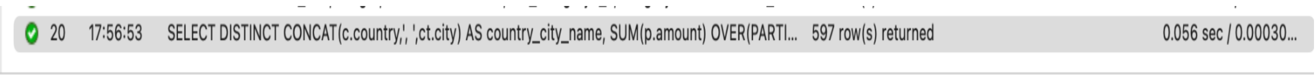


```
SELECT DISTINCT CONCAT(c.country,', ',ct.city) AS country_city_name,
SUM(p.amount) OVER(PARTITION BY c.country, ct.city) AS revenue -- finding sum using window function based on country and city
FROM country c,city ct,address ad,customer cu,payment p       -- Joining total 5 tables to get the desired result
WHERE c.country_id=ct.country_id
AND ad.city_id=ct.city_id
AND cu.address_id=ad.address_id
AND p.customer_id=cu.customer_id
ORDER BY revenue DESC;                                         -- Ordering by revenue in descending order
```

```
100%  106:265  5 errors found
```

country_city_name	revenue
United States, Cape Coral	221.55
Runion, Saint-Denis	216.54
United States, Aurora	198.50
Belarus, Molodetno	195.58
Netherlands, Apeldoorn	194.61
Brazil, Santa Brbara dOeste	194.61
Iran, Qomsheh	186.62
United Kingdom, London	180.52
Spain, Ourense (Orense)	177.60
India, Bijapur	175.61
Philippines, Tanza	175.58
United States, Memphis	174.66
Algeria, Skikda	173.63
India, Halisahar	170.67
Sudan, Omdurman	169.65
Philippines, Santa Rosa	167.67
Canada, Richmond Hill	167.62

Total Rows Fetched: 597



```
20  17:56:53  SELECT DISTINCT CONCAT(c.country,', ',ct.city) AS country_city_name, SUM(p.amount) OVER(PARTI...  597 row(s) returned  0.056 sec / 0.00030...
```

Approach:-

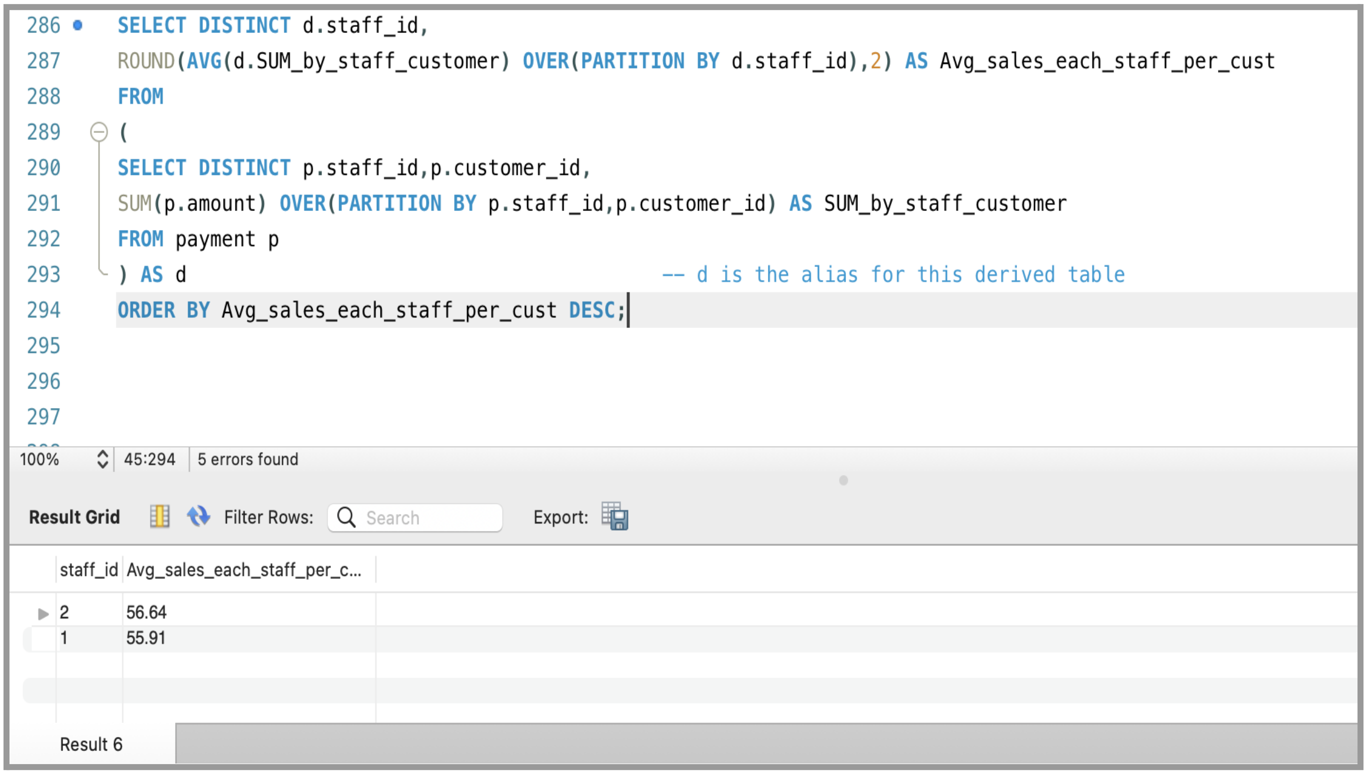
- Here we need to create a list of the revenue based on country, city according to decreasing amount of revenue
- So for getting the desired result I have joined 4 tables i.e., country, city, address and payment to fetch country name, city name which i am the concatenating using CONCAT() function
- Also calculated the revenue by using window function which is calculating the sum of amount partitioning by Country,City and then ordering revenue by descending order

Q5. Write a query to create a list with the average of the sales amount each staff_id has per customer.

result: 2 56.64
1 55.91

Soln:-

Query screenshot with Output:-



```
SELECT DISTINCT d.staff_id,
ROUND(AVG(d.SUM_by_staff_customer) OVER(PARTITION BY d.staff_id),2) AS Avg_sales_each_staff_per_cust
FROM
(
SELECT DISTINCT p.staff_id,p.customer_id,
SUM(p.amount) OVER(PARTITION BY p.staff_id,p.customer_id) AS SUM_by_staff_customer
FROM payment p
) AS d                                          -- d is the alias for this derived table
ORDER BY Avg_sales_each_staff_per_cust DESC;
```

staff_id	Avg_sales_each_staff_per_c...
2	56.64
1	55.91

Total Rows Fetched: 2

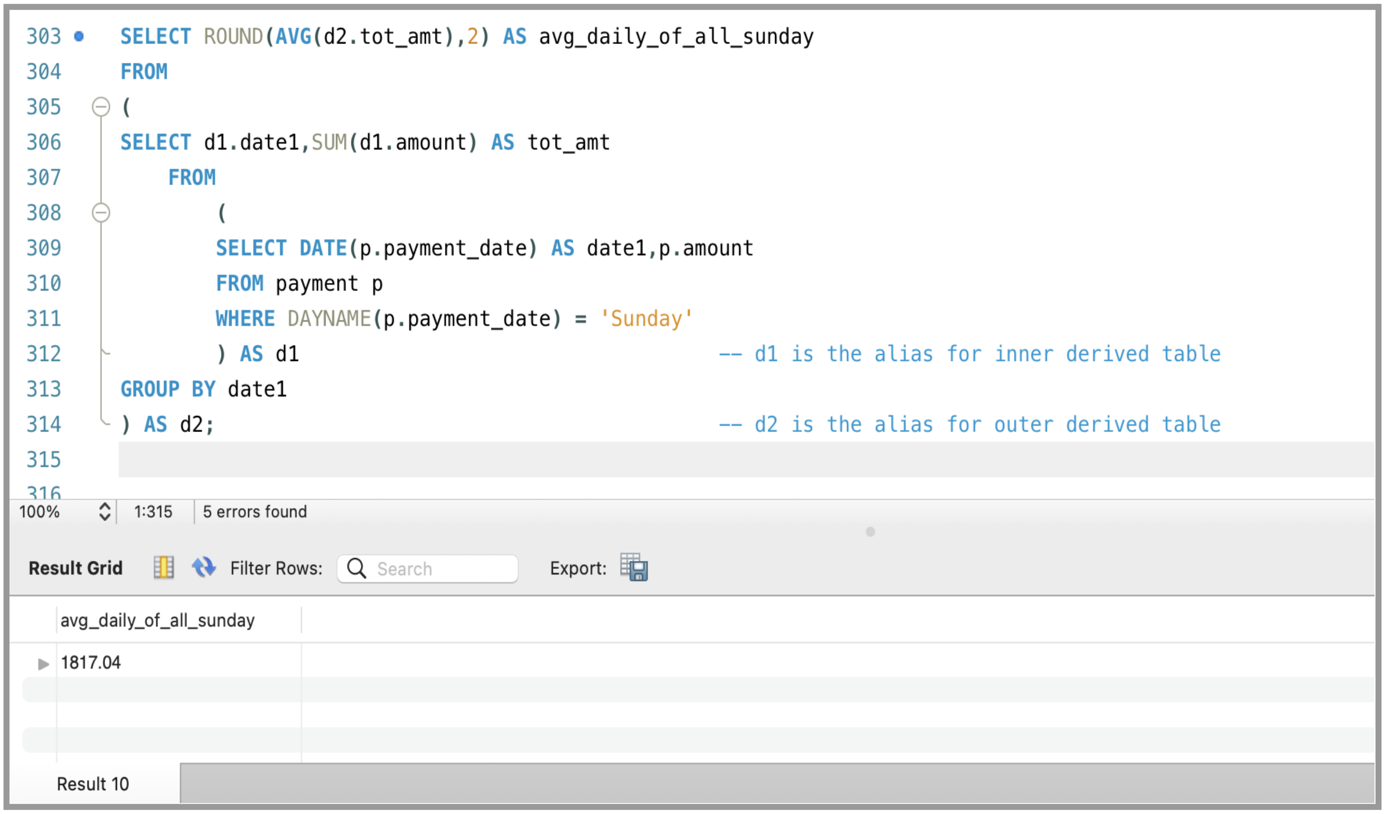
Approach:-

- Here we are asked to create a list with the average of sales per customer with each staff id
- Here each customer has multiple transaction with particular staff id, so first we need to find total amount of transaction each customer had with particular staff id, so I am calculating that In the derived table (d) above using window function which is calculating the sum of amount partitioning by staff_id,customer_id

- Now using derived table I am extracting staff_id and calculating average of the total_amount per customer calculated in derived table to find out the average amount of sales each staff had

Q6. Write a query that shows average daily revenue of all Sundays.

Query screenshot with Output:-



```sql
SELECT ROUND(AVG(d2.tot_amt),2) AS avg_daily_of_all_sunday
FROM
(
SELECT d1.date1,SUM(d1.amount) AS tot_amt
    FROM
        (
        SELECT DATE(p.payment_date) AS date1,p.amount
        FROM payment p
        WHERE DAYNAME(p.payment_date) = 'Sunday'
        ) AS d1                                   -- d1 is the alias for inner derived table
GROUP BY date1
) AS d2;                                          -- d2 is the alias for outer derived table
```

avg_daily_of_all_sunday
1817.04

Total Rows Fetched: 1

Approach:-

- We have been asked to show average daily revenue of all Sundays.
- So for that I have used single payment table wherein in the inner derived table I am extracting the payment_date and amount by filtering the data where the date is of sunday by using DAYNAME() function in where clause
- In each sunday their can be multiple transaction so in order to find out the average, first we need to calculate the total of all the transaction in every sunday.So in outer derived table I am calculating the sum of amount

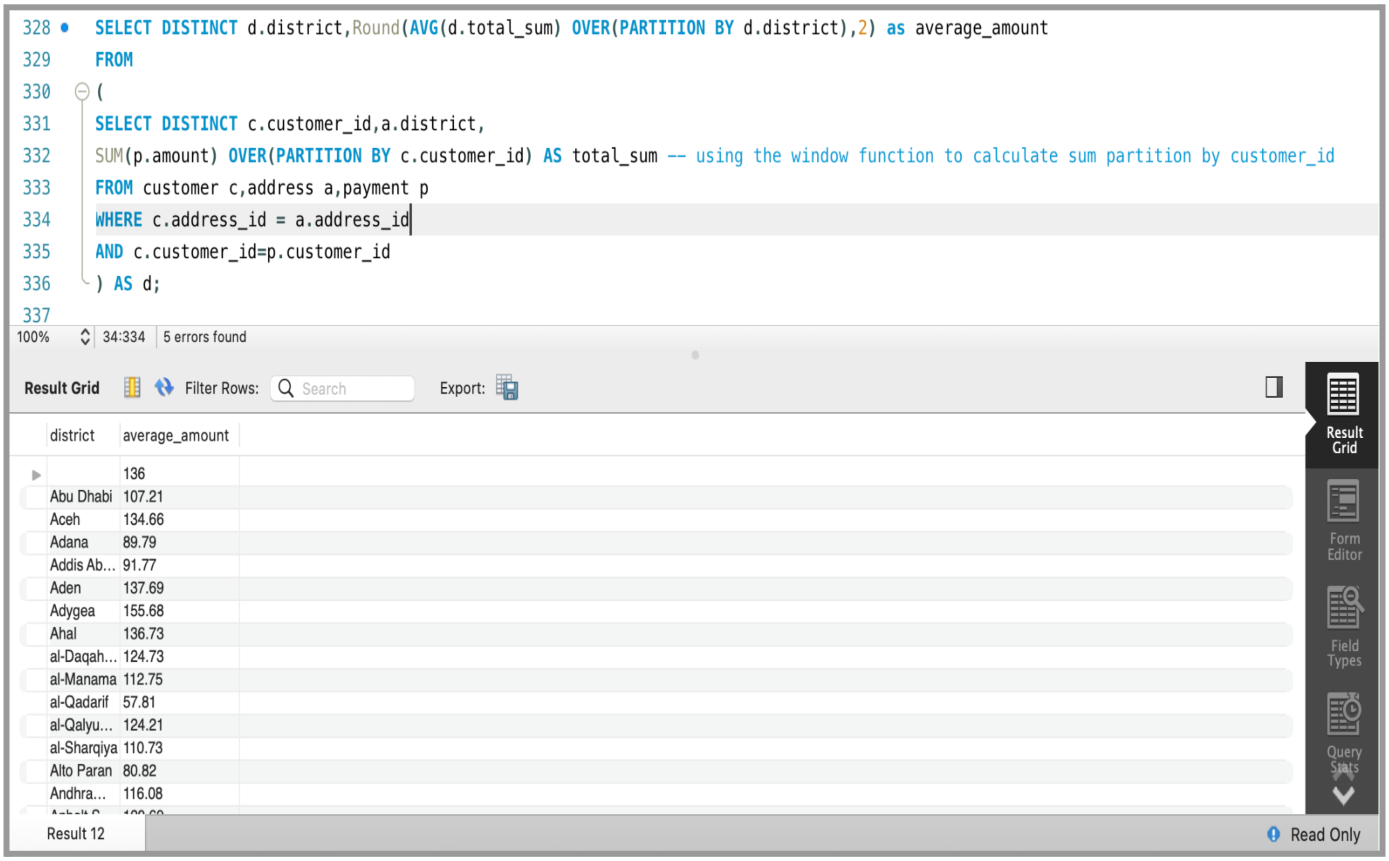
grouping by the date(obtain from inner derived table by filtering out on the basis of sunday) which will give me the total transaction in all sundays

- Now that i have got total amount ,so i can find the average of that total amount which will give me the desired result that is average daily revenue of all sundays.

Q7. Write a query to create a list that shows how much the average customer spent in total (customer life-time value) grouped by the different districts.

Soln:-

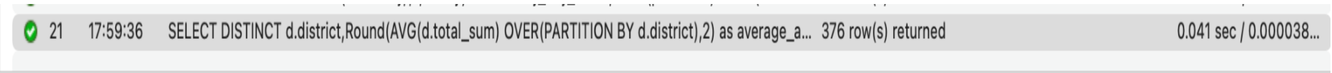
Query screenshot with Output:-



```
SELECT DISTINCT d.district,Round(AVG(d.total_sum) OVER(PARTITION BY d.district),2) as average_amount
FROM
(
SELECT DISTINCT c.customer_id,a.district,
SUM(p.amount) OVER(PARTITION BY c.customer_id) AS total_sum -- using the window function to calculate sum partition by customer_id
FROM customer c,address a,payment p
WHERE c.address_id = a.address_id
AND c.customer_id=p.customer_id
) AS d;
```

district	average_amount
	136
Abu Dhabi	107.21
Aceh	134.66
Adana	89.79
Addis Ab...	91.77
Aden	137.69
Adygea	155.68
Ahal	136.73
al-Daqah...	124.73
al-Manama	112.75
al-Qadarif	57.81
al-Qalyu...	124.21
al-Sharqiya	110.73
Alto Paran	80.82
Andhra...	116.08

Total Rows Fetched: 376



```
21    17:59:36    SELECT DISTINCT d.district,Round(AVG(d.total_sum) OVER(PARTITION BY d.district),2) as average_a...    376 row(s) returned    0.041 sec / 0.000038...
```

Approach:-

- Here we need to find out the average amount of all customer spent in total based on each district
- For that I have used three tables i.e., customer, address and payment and extracted customer_id, district and the total amount for each customer as each customer have multiple transaction in each district
- Now making above as derived table I extracted district and Average of total_sum calculated for each customer in the derived table partitioning b the district.So this way I am getting the district and the average of customers total spent grouped by each district

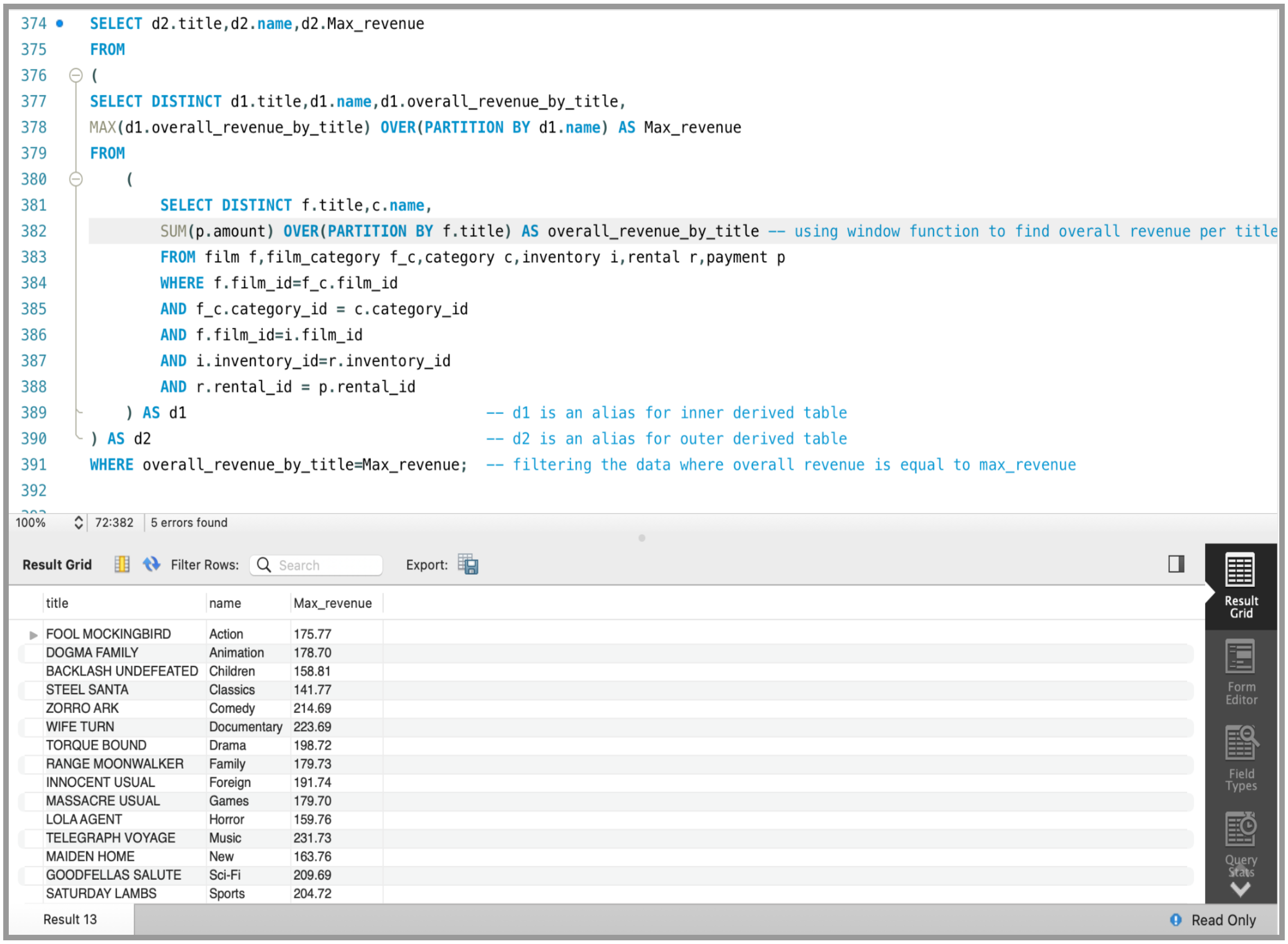
Q8. Write a query to list down the highest overall revenue collected (sum of amount per title) by a film in each category. Result should display the film title, category name and total revenue.

eg.

"FOOL MOCKINGBIRD"	"Action"	175.77
"DOGMA FAMILY"	"Animation"	178.7
"BACKLASH UNDEFEATED"	"Children"	158.81

Soln:-

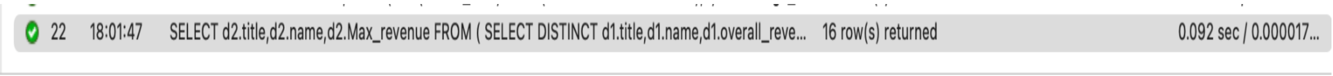
Query screenshot with Output:-



```
SELECT d2.title,d2.name,d2.Max_revenue
FROM
(
SELECT DISTINCT d1.title,d1.name,d1.overall_revenue_by_title,
MAX(d1.overall_revenue_by_title) OVER(PARTITION BY d1.name) AS Max_revenue
FROM
    (
        SELECT DISTINCT f.title,c.name,
        SUM(p.amount) OVER(PARTITION BY f.title) AS overall_revenue_by_title -- using window function to find overall revenue per title
        FROM film f,film_category f_c,category c,inventory i,rental r,payment p
        WHERE f.film_id=f_c.film_id
        AND f_c.category_id = c.category_id
        AND f.film_id=i.film_id
        AND i.inventory_id=r.inventory_id
        AND r.rental_id = p.rental_id
    ) AS d1                                      -- d1 is an alias for inner derived table
) AS d2                                          -- d2 is an alias for outer derived table
WHERE overall_revenue_by_title=Max_revenue;   -- filtering the data where overall revenue is equal to max_revenue
```

title	name	Max_revenue
FOOL MOCKINGBIRD	Action	175.77
DOGMA FAMILY	Animation	178.70
BACKLASH UNDEFEATED	Children	158.81
STEEL SANTA	Classics	141.77
ZORRO ARK	Comedy	214.69
WIFE TURN	Documentary	223.69
TORQUE BOUND	Drama	198.72
RANGE MOONWALKER	Family	179.73
INNOCENT USUAL	Foreign	191.74
MASSACRE USUAL	Games	179.70
LOLA AGENT	Horror	159.76
TELEGRAPH VOYAGE	Music	231.73
MAIDEN HOME	New	163.76
GOODFELLAS SALUTE	Sci-Fi	209.69
SATURDAY LAMBS	Sports	204.72

Total Rows Fetched: 16



```
22   18:01:47   SELECT d2.title,d2.name,d2.Max_revenue FROM ( SELECT DISTINCT d1.title,d1.name,d1.overall_reve...   16 row(s) returned   0.092 sec / 0.000017...
```

Approach:-

- Here we need to list out the film_title and highest overall revenue collected (sum of amount per title) by a film for each category
- For this I have used 6 tables i.e., film, film_category ,category, inventory, rental, payment and from that i have extracted film_title, film category and sum of amount partition by film_title, so this will give me the overall revenue for each film title
- Now from inner derived table I have the data of film_title, film category and overall revenue generated from each film.So in outer derived table I extracted the film_title, category and Maximum of overall revenue for each

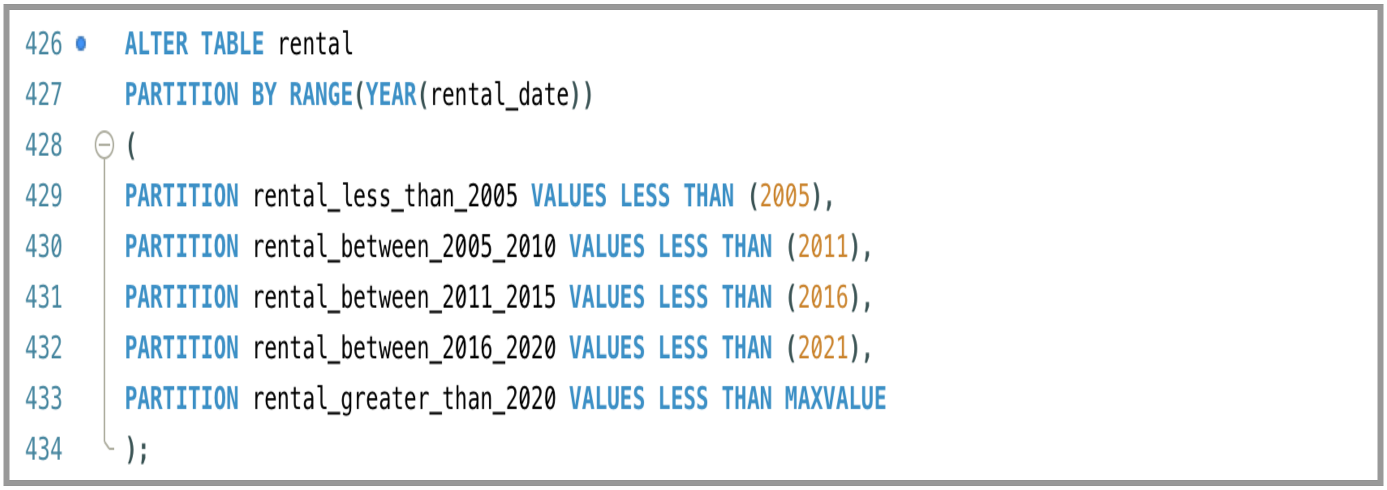
category by using window function for finding MAX of overall_revenue partition by category.

- Now from outer derived table I extracted the film_title,category name and the max revenue where my max_revenue= overall_revenue. This way I am getting those film title who has generated highest revenue in each category

Q9. Modify the table "rental" to be partitioned using PARTITION command based on ‘rental_date’ in below intervals:

<2005
between 2005–2010
between 2011–2015
between 2016–2020
>2020 - Partitions are created yearly

Soln:-



```
ALTER TABLE rental
PARTITION BY RANGE(YEAR(rental_date))
(
PARTITION rental_less_than_2005 VALUES LESS THAN (2005),
PARTITION rental_between_2005_2010 VALUES LESS THAN (2011),
PARTITION rental_between_2011_2015 VALUES LESS THAN (2016),
PARTITION rental_between_2016_2020 VALUES LESS THAN (2021),
PARTITION rental_greater_than_2020 VALUES LESS THAN MAXVALUE
);
```

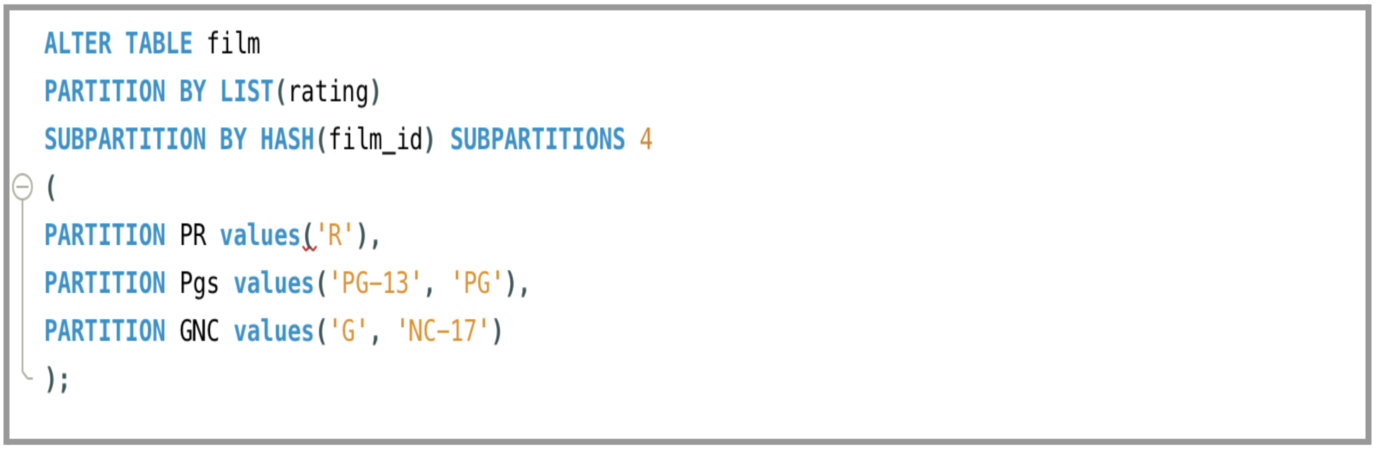
Approach:-

- Here In this particular question first I have to drop the foreign keys as partitioning was not supporting the foreign key constraints in MYSQL as it can change the underlying structure of the table
- Next I partitioned the rental table by rental_date with the specified range of years.
- First Partition “rental_less_than_2005” will contain those values where year is less than 2005

- Second Partition “rental_between_2005_2010” will contain those values where year is between 2005 and 2010
- Third Partition “rental_between_2011_2015” will contain those values where year is between 2011 and 2015
- Fourth Partition “rental_between_2016_2020” will contain those values where year is between 2016 and 2020
- And everything after 2020 will be stored in the partition “rental_greater_than_2020”

Q10. Modify the table "film" to be partitioned using PARTITION command based on ‘rating’ from below list. Further apply hash sub-partitioning based on ‘film_id’ into 4 sub-partitions.

partition_1 - "R"
partition_2 - "PG-13", "PG"
partition_3 - "G", "NC-17"



```
ALTER TABLE film
PARTITION BY LIST(rating)
SUBPARTITION BY HASH(film_id) SUBPARTITIONS 4
(
PARTITION PR values('R'),
PARTITION Pgs values('PG-13', 'PG'),
PARTITION GNC values('G', 'NC-17')
);
```

Approach:-

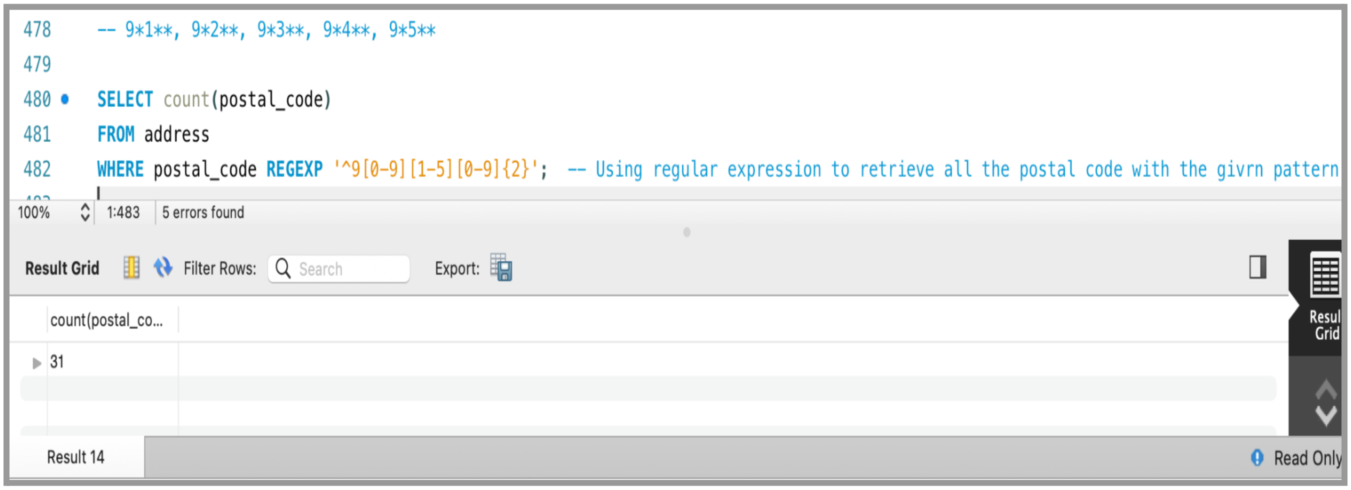
- Here first I have used PARTITION BY LIST based on rating
- Then I have used SUBPARTITION BY HASH based on film_id

Q11. Write a query to count the total number of addresses from the “address” table where the ‘postal_code’ is of the below formats. Use regular expression.

9*1**, 9*2**, 9*3**, 9*4**, 9*5**

Soln:-

Query screenshot with Output:-



```
-- 9*1**, 9*2**, 9*3**, 9*4**, 9*5**

SELECT count(postal_code)
FROM address
WHERE postal_code REGEXP '^9[0-9][1-5][0-9]{2}';  -- Using regular expression to retrieve all the postal code with the givrn pattern
```

```
100%   1:483   5 errors found
```

```
count(postal_co...
31
```

Total Rows Fetched: 1

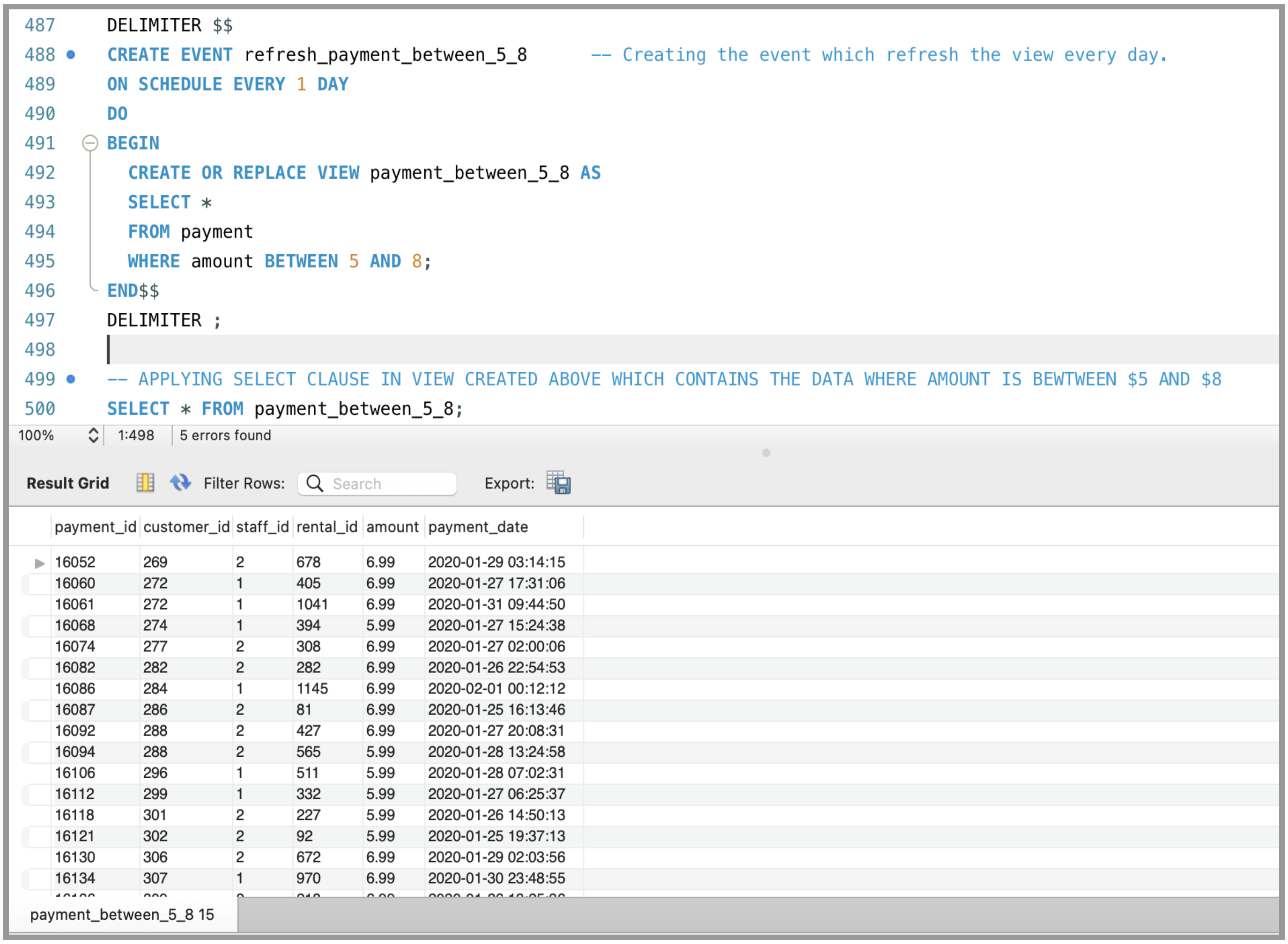
Approach:-

- Here I have used the address table to extract postal_code of the specific pattern by using a regular expression in the where clause to filter out the data
- Next I used the Aggregate function COUNT to count the number of postal code following the specific pattern

Q12. Write a query to create a materialized view from the “payment” table where ‘amount’ is between(inclusive) $5 to $8. The view should manually refresh on demand. Also write a query to manually refresh the created materialized view.

Soln:-

Query screenshot with Output:-

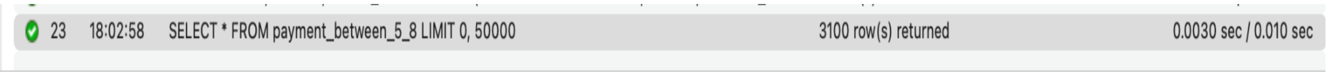


```sql
DELIMITER $$
CREATE EVENT refresh_payment_between_5_8      -- Creating the event which refresh the view every day.
ON SCHEDULE EVERY 1 DAY
DO
BEGIN
  CREATE OR REPLACE VIEW payment_between_5_8 AS
  SELECT *
  FROM payment
  WHERE amount BETWEEN 5 AND 8;
END$$
DELIMITER ;

-- APPLYING SELECT CLAUSE IN VIEW CREATED ABOVE WHICH CONTAINS THE DATA WHERE AMOUNT IS BEWTWEEN $5 AND $8
SELECT * FROM payment_between_5_8;
```

payment_id	customer_id	staff_id	rental_id	amount	payment_date
16052	269	2	678	6.99	2020-01-29 03:14:15
16060	272	1	405	6.99	2020-01-27 17:31:06
16061	272	1	1041	6.99	2020-01-31 09:44:50
16068	274	1	394	5.99	2020-01-27 15:24:38
16074	277	2	308	6.99	2020-01-27 02:00:06
16082	282	2	282	6.99	2020-01-26 22:54:53
16086	284	1	1145	6.99	2020-02-01 00:12:12
16087	286	2	81	6.99	2020-01-25 16:13:46
16092	288	2	427	6.99	2020-01-27 20:08:31
16094	288	2	565	5.99	2020-01-28 13:24:58
16106	296	1	511	5.99	2020-01-28 07:02:31
16112	299	1	332	5.99	2020-01-27 06:25:37
16118	301	2	227	5.99	2020-01-26 14:50:13
16121	302	2	92	5.99	2020-01-25 19:37:13
16130	306	2	672	6.99	2020-01-29 02:03:56
16134	307	1	970	6.99	2020-01-30 23:48:55

Total Rows Fetched: 3100



```
23    18:02:58    SELECT * FROM payment_between_5_8 LIMIT 0, 50000    3100 row(s) returned    0.0030 sec / 0.010 sec
```

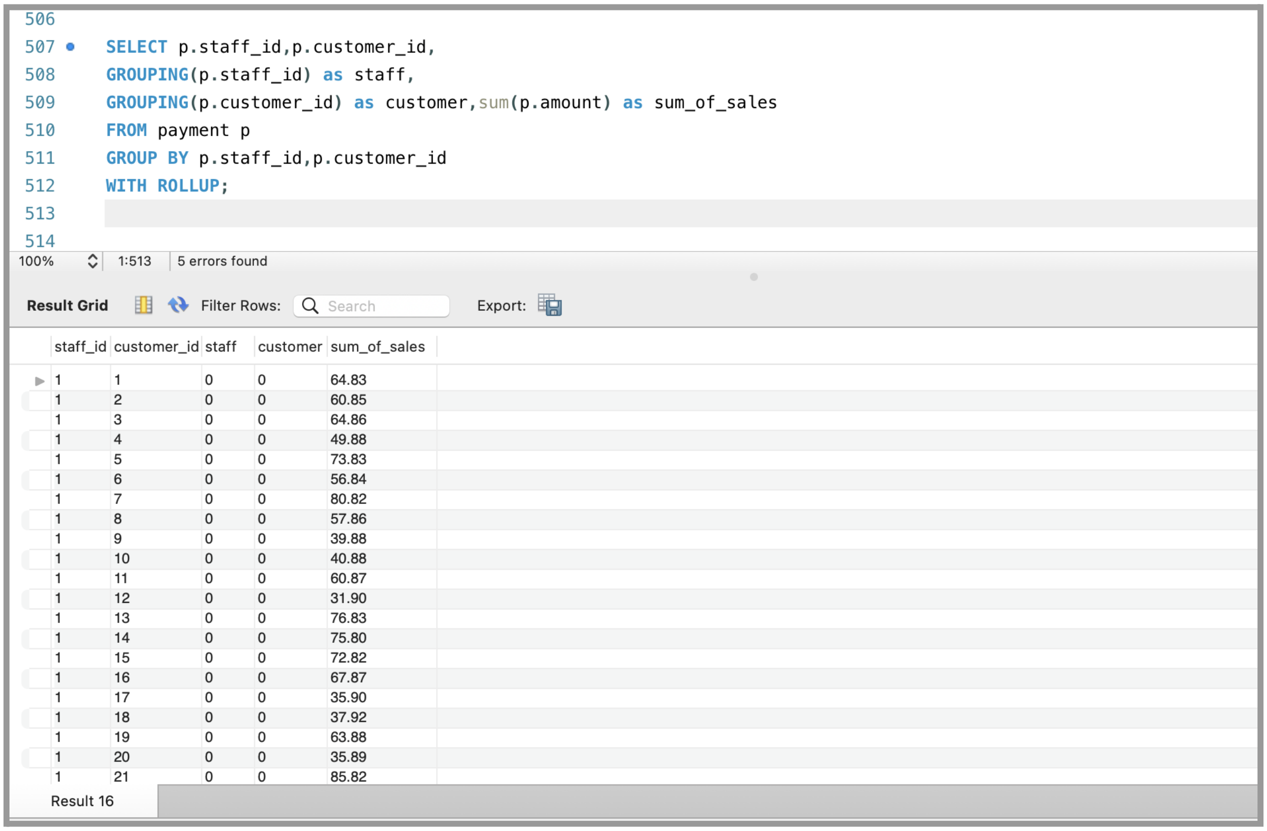
Approach:-

- As materialized view was not supported by MYSQL so i have created the event name “refresh_payment_between_5_8” which is scheduled to refresh in every one day and inside that I have created the normal view which stores all the data of the payment table where amount is between $5 and $ 8

Q13. Write a query to list down the total sales of each staff with each customer from the ‘payment’ table. In the same result, list down the total sales of each staff

i.e. sum of sales from all customers for a particular staff. Use the ROLLUP command. Also use GROUPING command to indicate null values.
Soln:-

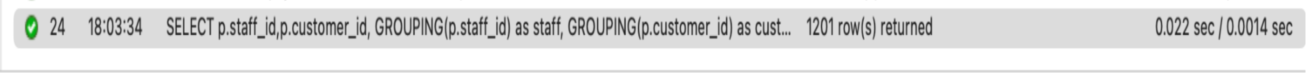
Query screenshot with Output:-



```
SELECT p.staff_id,p.customer_id,
GROUPING(p.staff_id) as staff,
GROUPING(p.customer_id) as customer,sum(p.amount) as sum_of_sales
FROM payment p
GROUP BY p.staff_id,p.customer_id
WITH ROLLUP;
```

staff_id	customer_id	staff	customer	sum_of_sales
1	1	0	0	64.83
1	2	0	0	60.85
1	3	0	0	64.86
1	4	0	0	49.88
1	5	0	0	73.83
1	6	0	0	56.84
1	7	0	0	80.82
1	8	0	0	57.86
1	9	0	0	39.88
1	10	0	0	40.88
1	11	0	0	60.87
1	12	0	0	31.90
1	13	0	0	76.83
1	14	0	0	75.80
1	15	0	0	72.82
1	16	0	0	67.87
1	17	0	0	35.90
1	18	0	0	37.92
1	19	0	0	63.88
1	20	0	0	35.89
1	21	0	0	85.82

Total Rows Fetched: 1201



```
24    18:03:34    SELECT p.staff_id,p.customer_id, GROUPING(p.staff_id) as staff, GROUPING(p.customer_id) as cust...    1201 row(s) returned    0.022 sec / 0.0014 sec
```

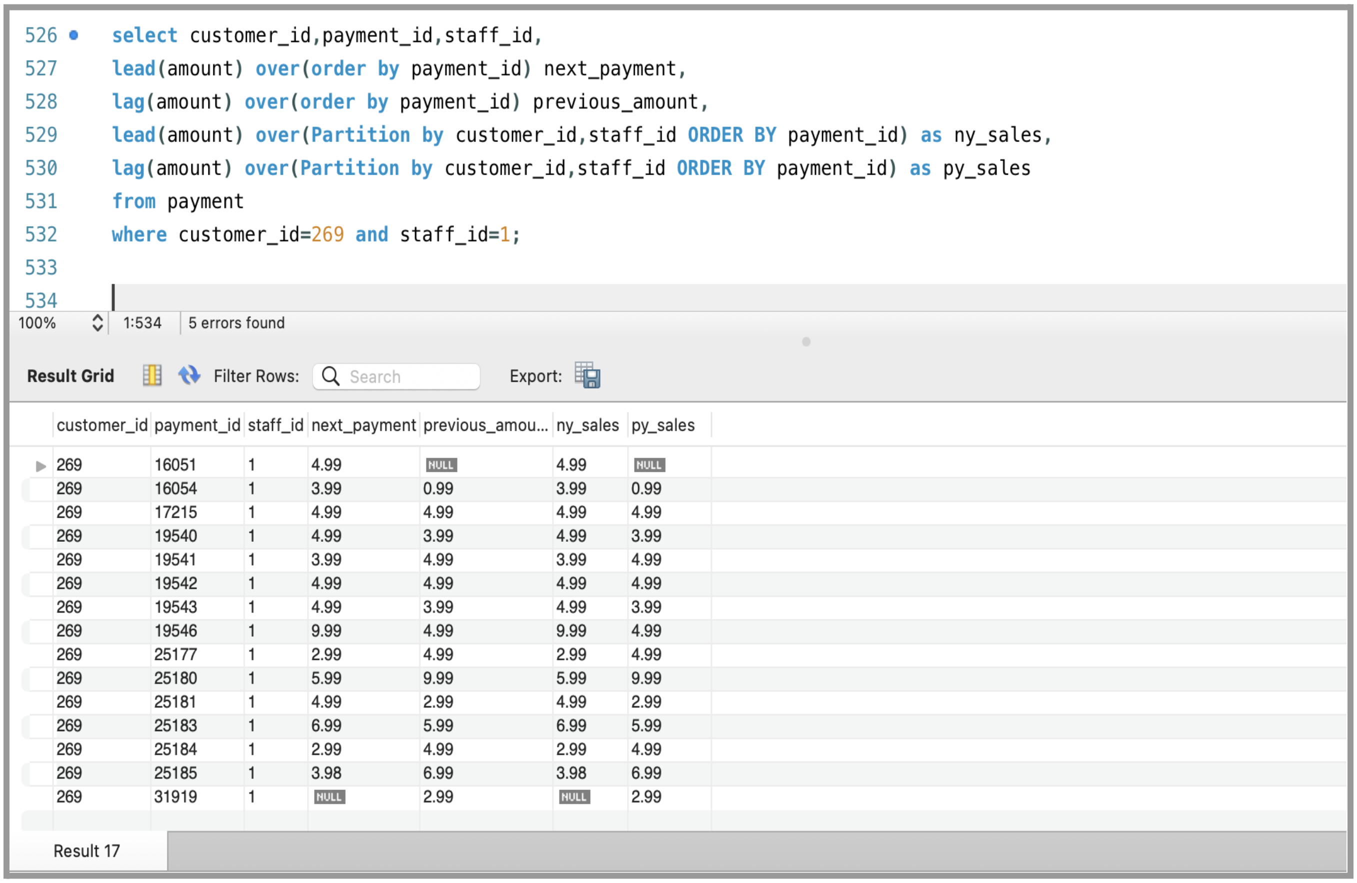
Approach:-

- Here we have been asked to list down all the total sales of each staff with each customer. Also we need list down the total sales of each staff i.e. sum of sales from all customers for a particular staff.
- So for that I have used the payment table to extract staff_id,customer_id and applied grouping on both customer_id and staff_id to handle null values and clearly indicate the ROLLUP sum for the particular column

Q.14 Write a single query to display the customer_id, staff_id, payment_id, amount, amount on immediately previous payment_id, amount on immediately next payment_id ny_sales for the payments from customer_id ‘269’ to staff_id ‘1’.

Soln:-

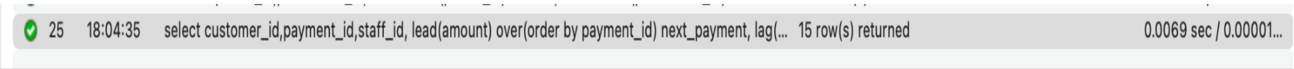
Query screenshot with Output:-



```sql
select customer_id,payment_id,staff_id,
lead(amount) over(order by payment_id) next_payment,
lag(amount) over(order by payment_id) previous_amount,
lead(amount) over(Partition by customer_id,staff_id ORDER BY payment_id) as ny_sales,
lag(amount) over(Partition by customer_id,staff_id ORDER BY payment_id) as py_sales
from payment
where customer_id=269 and staff_id=1;
```

customer_id	payment_id	staff_id	next_payment	previous_amou...	ny_sales	py_sales
269	16051	1	4.99	NULL	4.99	NULL
269	16054	1	3.99	0.99	3.99	0.99
269	17215	1	4.99	4.99	4.99	4.99
269	19540	1	4.99	3.99	4.99	3.99
269	19541	1	3.99	4.99	3.99	4.99
269	19542	1	4.99	4.99	4.99	4.99
269	19543	1	4.99	3.99	4.99	3.99
269	19546	1	9.99	4.99	9.99	4.99
269	25177	1	2.99	4.99	2.99	4.99
269	25180	1	5.99	9.99	5.99	9.99
269	25181	1	4.99	2.99	4.99	2.99
269	25183	1	6.99	5.99	6.99	5.99
269	25184	1	2.99	4.99	2.99	4.99
269	25185	1	3.98	6.99	3.98	6.99
269	31919	1	NULL	2.99	NULL	2.99

Total Rows Fetched: 15



```
25    18:04:35    select customer_id,payment_id,staff_id, lead(amount) over(order by payment_id) next_payment, lag(...    15 row(s) returned    0.0069 sec / 0.00001...
```

Approach:-

- Here we have been asked to display the customer_id, staff_id, payment_id, amount, amount on immediately previous payment_id, amount on immediately next payment_id
- For this we have used the payment table and used the window function LEAD to extract the next payment and LAG to extract the previous payment order by payment_id

- Next we need to find the ny_sales - the amount of the next payment made by the same customer to the same staff member and py sales - the amount of the previous payment made by the same customer to the same staff member.For this as well i have used lead and lag function partitioned by customer_id and staff_id
- Used where clause to filter out the data where customer_id=269 and staff_id=1