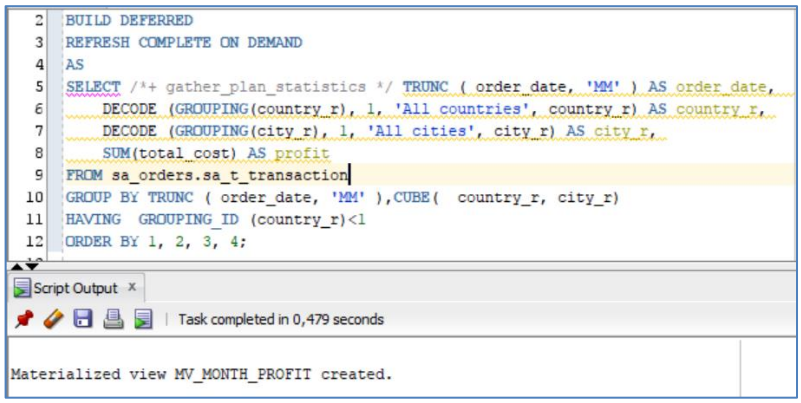
Lab report #7

Sadovskaya Veronika

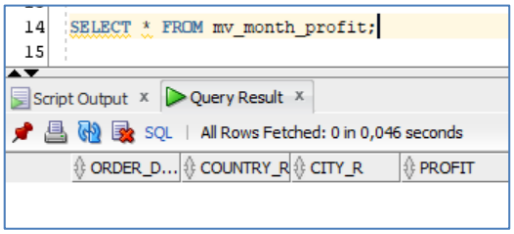
GitHub: https://github.com/sdveronika/DataMola22

Task 1 - Create Materialized Views - ON DEMAND

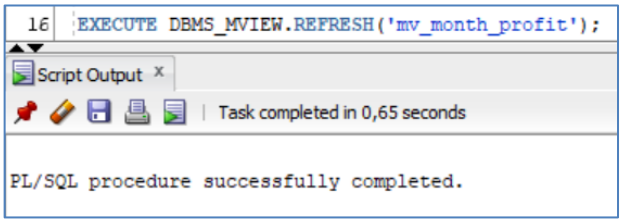
Create materialized view:



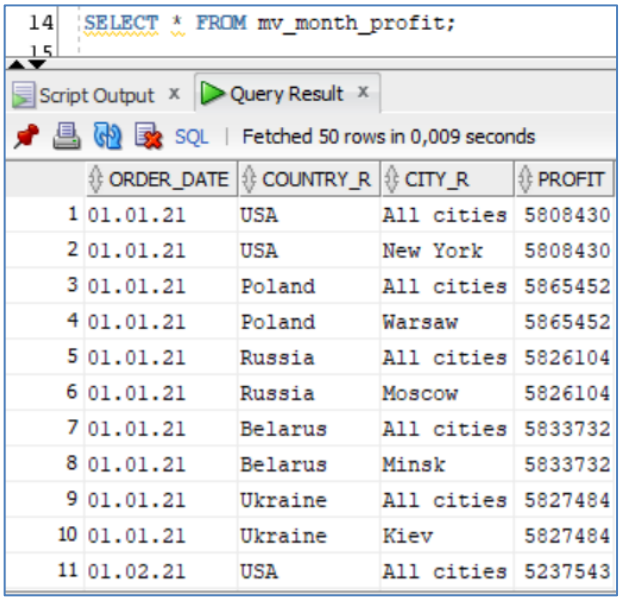
After that let's select data from our view. As we can see the view remains empty:



Now let's update our view using the function DBMS_MVIEW.REFRESH:



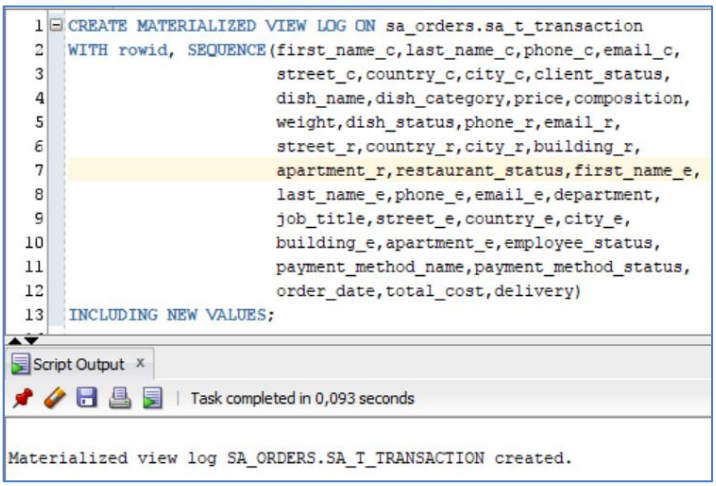
If we now execute select with our view, we will see that it is full:



	ORDER_DATE	COUNTRY_R	CITY_R	PROFIT
1	01.01.21	USA	All cities	5808430
2	01.01.21	USA	New York	5808430
3	01.01.21	Poland	All cities	5865452
4	01.01.21	Poland	Warsaw	5865452
5	01.01.21	Russia	All cities	5826104
6	01.01.21	Russia	Moscow	5826104
7	01.01.21	Belarus	All cities	5833732
8	01.01.21	Belarus	Minsk	5833732
9	01.01.21	Ukraine	All cities	5827484
10	01.01.21	Ukraine	Kiev	5827484
11	01.02.21	USA	All cities	5237543

Task 2 - Create Materialized Views - ON COMMIT

Create materialized view log on our transactions table:

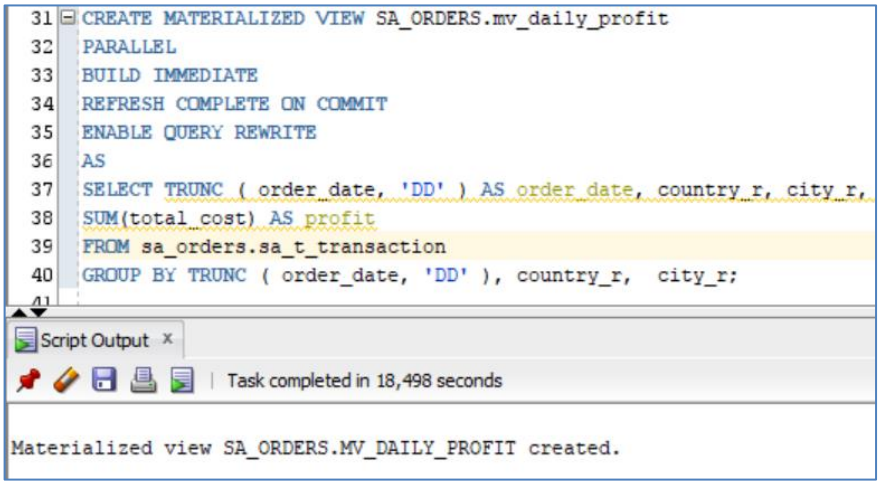


```
CREATE MATERIALIZED VIEW LOG ON sa_orders.sa_t_transaction
WITH rowid, SEQUENCE(first_name_c,last_name_c,phone_c,email_c,
                     street_c,country_c,city_c,client_status,
                     dish_name,dish_category,price,composition,
                     weight,dish_status,phone_r,email_r,
                     street_r,country_r,city_r,building_r,
                     apartment_r,restaurant_status,first_name_e,
                     last_name_e,phone_e,email_e,department,
                     job_title,street_e,country_e,city_e,
                     building_e,apartment_e,employee_status,
                     payment_method_name,payment_method_status,
                     order_date,total_cost,delivery)
INCLUDING NEW VALUES;
```

Script Output | Task completed in 0,093 seconds

```
Materialized view log SA_ORDERS.SA_T_TRANSACTION created.
```

Create materialized view:

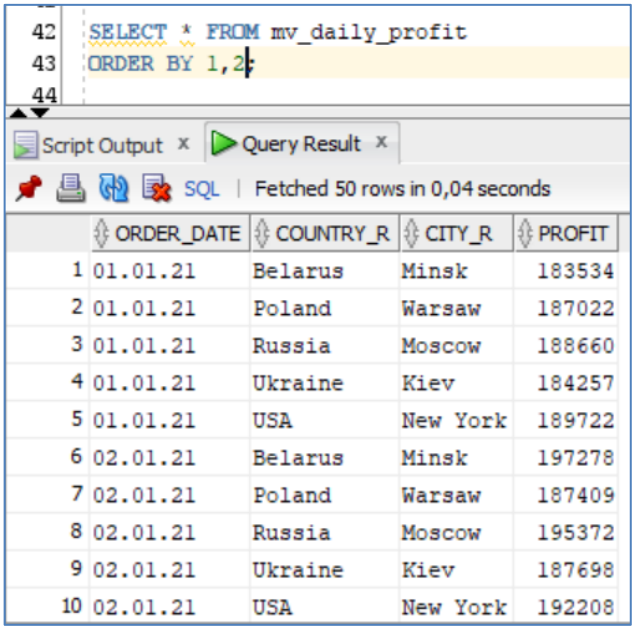


```
CREATE MATERIALIZED VIEW SA_ORDERS.mv_daily_profit
PARALLEL
BUILD IMMEDIATE
REFRESH COMPLETE ON COMMIT
ENABLE QUERY REWRITE
AS
SELECT TRUNC ( order_date, 'DD' ) AS order_date, country_r, city_r,
SUM(total_cost) AS profit
FROM sa_orders.sa_t_transaction
GROUP BY TRUNC ( order_date, 'DD' ), country_r,  city_r;
```

Script Output | Task completed in 18,498 seconds

```
Materialized view SA_ORDERS.MV_DAILY_PROFIT created.
```

After that let's select data from our view. As we can see the view remains is full:



```
SELECT * FROM mv_daily_profit
ORDER BY 1,2;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0,04 seconds

	ORDER_DATE	COUNTRY_R	CITY_R	PROFIT
1	01.01.21	Belarus	Minsk	183534
2	01.01.21	Poland	Warsaw	187022
3	01.01.21	Russia	Moscow	188660
4	01.01.21	Ukraine	Kiev	184257
5	01.01.21	USA	New York	189722
6	02.01.21	Belarus	Minsk	197278
7	02.01.21	Poland	Warsaw	187409
8	02.01.21	Russia	Moscow	195372
9	02.01.21	Ukraine	Kiev	187698
10	02.01.21	USA	New York	192208

Now let's change the data in our resource table (increase total_cost by 2 times) and commit changes:

```
45  UPDATE sa_orders.sa_t_transaction
46  SET total_cost=total_cost*2
47  WHERE country_r='Russia';
```

Script Output | Query Result | Task completed in 22,893 seconds

```
91 250 rows updated.
```

```
42  SELECT * FROM mv_daily_profit
43  ORDER BY 1,2;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0,04 seconds

	ORDER_DATE	COUNTRY_R	CITY_R	PROFIT
1	01.01.21	Belarus	Minsk	183534
2	01.01.21	Poland	Warsaw	187022
3	01.01.21	Russia	Moscow	188660
4	01.01.21	Ukraine	Kiev	184257
5	01.01.21	USA	New York	189722
6	02.01.21	Belarus	Minsk	197278
7	02.01.21	Poland	Warsaw	187409
8	02.01.21	Russia	Moscow	195372
9	02.01.21	Ukraine	Kiev	187698
10	02.01.21	USA	New York	192208

```
49  COMMIT;
```

Script Output | Query Result | Task comple

```
Commit complete.
```

```
43  SELECT * FROM mv_daily_profit
44  ORDER BY 1,2;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0,03 seconds

	ORDER_DATE	COUNTRY_R	CITY_R	PROFIT
1	01.01.21	Belarus	Minsk	183534
2	01.01.21	Poland	Warsaw	187022
3	01.01.21	Russia	Moscow	377320
4	01.01.21	Ukraine	Kiev	184257
5	01.01.21	USA	New York	189722
6	02.01.21	Belarus	Minsk	197278
7	02.01.21	Poland	Warsaw	187409
8	02.01.21	Russia	Moscow	390744
9	02.01.21	Ukraine	Kiev	187698
10	02.01.21	USA	New York	192208
11	03.01.21	Belarus	Minsk	187675
12	03.01.21	Poland	Warsaw	204149

Task 3 - Create Materialized Views - Refreshing at definitive Time moment

Let's create a materialized view that will be updated every 5 minutes, based on a monthly report from 5 labs 2 tasks:

```
CREATE MATERIALIZED VIEW sa_orders.monthly_dish_name_per_count
BUILD IMMEDIATE
REFRESH COMPLETE
NEXT SYSDATE + 5/1440
AS
WITH CTE_FN AS
( SELECT dish_name,
TRUNC(order_date, 'YYYY') AS year,
TO_CHAR(order_date, 'MM') AS month,
'MONTH' period,
COUNT(*) AS dish_count
FROM sa_orders.sa_t_transaction
GROUP BY dish_name, TRUNC(order_date, 'YYYY'),TO_CHAR(order_date, 'MM')
ORDER BY dish_name, year, month)
SELECT dish_name, year, month, period, dish_count
FROM CTE_FN
```

Script Output | Task completed in 0,667 seconds

```
Materialized view SA_ORDERS.MONTHLY_DISH_NAME_PER_COUNT created.
```

```
SELECT * FROM sa_orders.monthly_dish_name_per_count;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0,008 seconds

	DISH_NAME	YEAR	MONTH	PERIOD	DISH_COUNT
1	greek salad	01.01.21	01	MONTH	7750
2	greek salad	01.01.21	02	MONTH	7000
3	greek salad	01.01.21	03	MONTH	7750
4	greek salad	01.01.21	04	MONTH	7500
5	greek salad	01.01.21	05	MONTH	7750
6	greek salad	01.01.21	06	MONTH	7500
7	greek salad	01.01.21	07	MONTH	7750
8	greek salad	01.01.21	08	MONTH	7750
9	greek salad	01.01.21	09	MONTH	7500
10	greek salad	01.01.21	10	MONTH	7750
11	greek salad	01.01.21	11	MONTH	7500
12	greek salad	01.01.21	12	MONTH	7750
13	greek salad	01.01.21	(null)	YEAR	91250
14	greek salad	(null)	(null)	ALL	91250
15	pizza	01.01.21	01	MONTH	7750