Lab report #10

Sadovskaya Veronika

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

Task 1 - Transformation Description

Transforming Data Using PL/SQL

In a data warehouse environment, you can use procedural languages such as PL/SQL to implement complex transformations in the Oracle Database. Whereas CTAS operates on entire tables and emphasizes parallelism, PL/SQL provides a row-based approached and can accommodate very sophisticated transformation rules. For example, a PL/SQL procedure could open multiple cursors and read data from multiple source tables, combine this data using complex business rules, and finally insert the transformed data into one or more target table. It would be difficult or impossible to express the same sequence of operations using standard SQL statements.

Using a procedural language, a specific transformation (or number of transformation steps) within a complex ETL processing can be encapsulated, reading data from an intermediate staging area and generating a new table object as output. A previously generated transformation input table and a subsequent transformation will consume the table generated by this specific transformation. Alternatively, these encapsulated transformation steps within the complete ETL process can be integrated seamlessly, thus streaming sets of rows between each other without the necessity of intermediate staging. You can use table functions to implement such behavior.

Task 2 - Loading to SAL Layer Data

Let's create a view that contains information about the duration of work of all employees in the current position:

```
22  CREATE OR REPLACE VIEW sal_cl.employee_dim_actual_position AS
23  SELECT first_name|| ' ' || last_name AS employee_name,
24         phone,
25         email,
26         department,
27         job_title AS position,
28         start_date,
29         (TO_DATE(SYSDATE, 'DD/MM/YYYY')-TO_DATE(start_date,'DD/MM/YYYY')) AS num_daya_at_position
30  FROM dw_data.employee_dimension
31  WHERE is_active='Y';
```

Script Output

Task completed in 0,062 seconds

```
View SAL_CL.EMPLOYEE_DIM_ACTUAL_POSITION created.
```

Select from this view:

```
33  SELECT * FROM sal_cl.employee_dim_actual_position;
```

Script Output | Query Result

All Rows Fetched: 25 in 0,033 seconds

	EMPLOYEE_NAME	PHONE	EMAIL	DEPARTMENT	POSITION	START_DATE	NUM_DAYA_AT_POSITION
1	Adriana Karnitskaya	375295542493	AdrianaKarnitskaya@mail.ru	department_name_1	director	17.08.22	1
2	Alyssa Malysheva	375296172078	AlyssaMalysheva@mail.ru	department_name_4	employee	17.08.22	1
3	Rita Astafyeva	375294007451	RitaAstafyeva@mail.ru	department_name_2	manager	17.08.22	1
4	Clara Zaykova	375297467234	ClaraZaykova@mail.ru	department_name_2	manager	17.08.22	1
5	Eva Moshko	375297667494	EvaMoshko@mail.ru	department_name_4	employee	17.08.22	1
6	Ivan Semin	375293676621	IvanSemin@mail.ru	department_name_4	employee	17.08.22	1
7	Ian Astafyev	375294996103	IanAstafyev@mail.ru	department_name_3	employee	17.08.22	1
8	Ada Alymova	375292270876	AdaAlymova@mail.ru	department_name_2	manager	17.08.22	1
9	Maxim Mayorov	375298589471	MaximMayorov@mail.ru	department_name_2	manager	17.08.22	1
10	Veronika Sadovskaya	375291959498	VeronikaSadovskaya@mail.ru	department_name_3	employee	17.08.22	1
11	Pavel Moshko	375295241317	PavelMoshko@mail.ru	department_name_4	employee	17.08.22	1

Let's create a view that contains information about the amount of work of each employee per month (number of orders and total profit):

```
35 CREATE OR REPLACE VIEW sal_cl.employee_dim_work AS
36 SELECT DISTINCT TRUNC(date_id, 'MM') AS month,
37        first_name|| ' ' || last_name AS employee_name,
38        job_title AS position,
39        COUNT(order_id) AS orders_count_employee,
40        SUM(total_cost) AS employee_profit
41 FROM dw_data.order_fact ord
42   RIGHT JOIN dw_data.employee_dimension emp
43   ON (ord.employee_id=emp.employee_id)
44 GROUP BY TRUNC(date_id, 'MM'), first_name|| ' ' || last_name, job_title
45 ORDER BY 1,2,3,4,5;
```

Script Output | Task completed in 0,045 seconds

```
View SAL_CL.EMPLOYEE_DIM_WORK created.
```

Select from this view:

```
47 SELECT * FROM sal_cl.employee_dim_work;
48
```

Script Output | Query Result | SQL | Fetched 50 rows in 0,236 seconds

	MONTH	EMPLOYEE_NAME	POSITION	ORDERS_COUNT_EMPLOYEE	EMPLOYEE_PROFIT
1	01.01.21	Ada Alymova	manager	7750	5793895
2	01.01.21	Adriana Karnitskaya	director	7750	5907859
3	01.01.21	Miron Buynova	employee	7750	5810239
4	01.01.21	Miron Parfenov	employee	7750	5783234
5	01.01.21	Nikita Malyshev	employee	7750	5802457
6	01.01.21	Pavel Moshko	employee	7750	5814850
7	01.02.21	Ada Alymova	manager	7000	5272705
8	01.02.21	Adriana Karnitskaya	director	7000	5281627
9	01.02.21	Miron Buynova	employee	7000	5241723
10	01.02.21	Miron Parfenov	employee	7000	5286688
11	01.02.21	Nikita Malyshev	employee	7000	5201214
12	01.02.21	Pavel Moshko	employee	7000	5247193
13	01.03.21	Ada Alymova	manager	7750	5822997
14	01.03.21	Adriana Karnitskaya	director	7750	5811832
15	01.03.21	Miron Buynova	employee	7750	5796356
16	01.03.21	Miron Parfenov	employee	7750	5832271
17	01.03.21	Nikita Malyshev	employee	7750	5801947

Let's create a view that contains information about the profit for the month for each restaurant, the number of dishes sold and the total number of orders:

```
49 CREATE OR REPLACE VIEW sal_cl.restaurant_profit AS
50 SELECT DISTINCT TRUNC(date_id, 'MM') AS month,
51        country,
52        city,
53        SUM (total_cost) AS total_profit,
54        SUM (dish_amount) AS total_dish_amount,
55        COUNT (order_id) AS total_order_count
56 FROM dw_data.order_fact ord
57   RIGHT JOIN dw_data.dish_dimension dsh
58   ON (ord.dish_id=dsh.dish_id)
59   RIGHT JOIN dw_data.restaurant_dimension rst
60   ON (rst.restaurant_id=ord.restaurant_id)
61 GROUP BY TRUNC(date_id, 'MM'),country,city
62 ORDER BY 1,4,5,6;
```

Script Output | Task completed in 0,05 seconds

```
View SAL_CL.RESTAURANT_PROFIT created.
```

Select from this view:

```
64  SELECT * FROM sal_cl.restaurant_profit;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0,253 seconds

	MONTH	COUNTRY	CITY	TOTAL_PROFIT	TOTAL_DISH_AMOUNT	TOTAL_ORDER_COUNT
1	01.01.21	Ukraine	Kiev	6949272	200707	9300
2	01.01.21	Russia	Moscow	6963031	200022	9300
3	01.01.21	USA	New York	6973329	199000	9300
4	01.01.21	Belarus	Minsk	6992781	201567	9300
5	01.01.21	Poland	Warsaw	7034121	201540	9300
6	01.02.21	Russia	Moscow	6231573	179280	8400
7	01.02.21	USA	New York	6259544	179442	8400
8	01.02.21	Ukraine	Kiev	6324705	181011	8400
9	01.02.21	Poland	Warsaw	6352629	181906	8400
10	01.02.21	Belarus	Minsk	6362699	183313	8400
11	01.03.21	Belarus	Minsk	6942576	197324	9300
12	01.03.21	Ukraine	Kiev	6960043	199358	9300
13	01.03.21	Russia	Moscow	6971340	198941	9300
14	01.03.21	USA	New York	6974337	200391	9300
15	01.03.21	Poland	Warsaw	6991878	201533	9300

Let's create a view that contains information about the popularity of each dish in each restaurant for a month:

```
80  SELECT * FROM sal_cl.top_dishes;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0,278 seconds

	MONTH	COUNTRY	CITY	DISH_NAME	TOTAL_DISH_AMOUNT
1	01.01.21	Belarus	Minsk	soup	93578
2	01.01.21	Belarus	Minsk	pizza	54888
3	01.01.21	Belarus	Minsk	chebupelli	21783
4	01.01.21	Belarus	Minsk	greek salad	16322
5	01.01.21	Belarus	Minsk	pasta	14996
6	01.01.21	Poland	Warsaw	soup	92888
7	01.01.21	Poland	Warsaw	pizza	54838
8	01.01.21	Poland	Warsaw	chebupelli	22181
9	01.01.21	Poland	Warsaw	greek salad	16287
10	01.01.21	Poland	Warsaw	pasta	15346
11	01.01.21	Russia	Moscow	soup	91575

```
66  CREATE OR REPLACE VIEW sal_cl.top_dishes AS
67  SELECT DISTINCT TRUNC(date_id, 'MM') AS month,
68         country,
69         city,
70         dish_name,
71         SUM(dish_amount) AS total_dish_amount
72  FROM dw_data.order_fact ord
73    RIGHT JOIN dw_data.dish_dimension dsh
74    ON (ord.dish_id=dsh.dish_id)
75    RIGHT JOIN dw_data.restaurant_dimension rst
76    ON (rst.restaurant_id=ord.restaurant_id)
77  GROUP BY TRUNC(date_id, 'MM'),country,city, dish_name
78  ORDER BY 1,2,3,5 DESC;
```

Script Output | Task completed in 0,064 seconds

```
View SAL_CL.TOP_DISHES created.
```

Select from this view:

Let's create a view that contains information about the attendance of each restaurant for a month:

```
82  CREATE OR REPLACE VIEW sal_cl.restaurant_visits AS
83  SELECT DISTINCT TRUNC(date_id, 'MM') AS month,
84         country,
85         city,
86         COUNT(order_id) AS total_visits_count
87  FROM dw_data.order_fact ord
88    RIGHT JOIN dw_data.restaurant_dimension rst
89    ON (rst.restaurant_id=ord.restaurant_id)
90  WHERE delivery='N'
91  GROUP BY TRUNC(date_id, 'MM'),country,city
92  ORDER BY 1,4 DESC;
```

Script Output | Task completed in 0,055 seconds

```
View SAL_CL.RESTAURANT_VISITS created.
```

Select from this view:

```
94  SELECT * FROM sal_cl.restaurant_visits;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0,271 seconds

	MONTH	COUNTRY	CITY	TOTAL_VISITS_COUNT
1	01.01.21	Poland	Warsaw	6161
2	01.01.21	Ukraine	Kiev	6141
3	01.01.21	Belarus	Minsk	6137
4	01.01.21	USA	New York	6127
5	01.01.21	Russia	Moscow	6105
6	01.02.21	USA	New York	5549
7	01.02.21	Belarus	Minsk	5536
8	01.02.21	Poland	Warsaw	5509
9	01.02.21	Ukraine	Kiev	5487
10	01.02.21	Russia	Moscow	5437
11	01.03.21	Poland	Warsaw	6162

Let's create a view that stores all information about clients:

```
33  CREATE OR REPLACE VIEW sal_cl.w_client_dimension
34  AS SELECT * FROM dw_data.client_dimension;
```

Script Output | Task completed in 0,037 seconds

```
Table SAL_CL.ORDER_FACT created.


View SAL_CL.W_DISH_DIMENSION created.


View SAL_CL.W_CLIENT_DIMENSION created.
```

Select from this view:

```
SELECT * FROM sal_cl.w_client_dimension;
```

Query Result — All Rows Fetched: 5 in 0,03 seconds

CLIENT_ID	FIRST_NAME	LAST_NAME	PHONE	EMAIL	STREET	COUNTRY	CITY	STATUS
1	Adriana	Karnitskaya	375294217925	AdrianaKarnitskaya@mail.ru	Yesenin	Belarus	Minsk	Y
2	Ian	Astafyev	375295321165	IanAstafyev@mail.ru	Champs Elysees	France	Paris	Y
3	Ada	Alymova	375292376362	AdaAlymova@mail.ru	Nikolskaya	Russia	Moscow	Y
4	Pavel	Moshko	375291330060	PavelMoshko@mail.ru	Antoine Dansaert	Belgium	Brussels	Y
5	Alice	Lysenko	375294487084	AliceLysenko@mail.ru	Fifth Avenue	USA	New York	Y

Let's create a view that stores all information about dishes:

```
CREATE OR REPLACE VIEW sal_cl.w_dish_dimension
AS SELECT * FROM dw_data.dish_dimension;
```

Script Output — Task completed in 0,05 seconds

```
Table SAL_CL.ORDER_FACT created.


View SAL_CL.W_DISH_DIMENSION created.
```

Select from this view:

```
SELECT * FROM sal_cl.w_dish_dimension;
```

Query Result — All Rows Fetched: 5 in 0,027 seconds

DISH_ID	DISH_NAME	DISH_CATEGORY	PRICE	COMPOSITION	WEIGHT	STATUS
1	soup	hot	15	soup ingredients	915	Y
2	pizza	hot	25	pizza ingredients	486	Y
3	pasta	hot	87	pasta ingredients	959	Y
4	chebupelli	hot	61	chebupelli ingredients	512	Y
5	greek salad	appetizer	82	greek salad ingredients	797	Y

Let's create a view that stores all information about restaurants:

```
CREATE OR REPLACE VIEW sal_cl.w_restaurant_dimension
AS SELECT * FROM dw_data.restaurant_dimension;
```

Script Output — Task completed in 0,045 seconds

```
View SAL_CL.W_RESTAURANT_DIMENSION created.
```

Select from this view:

```
SELECT * FROM sal_cl.w_restaurant_dimension;
```

Query Result — All Rows Fetched: 5 in 0,029 seconds

RESTAURANT_ID	PHONE	EMAIL	ADDRESS	COUNTRY	CITY	BUILDING	APARTMENT	STATUS
1	375295841669	5@mail.ru	Gurchevskaya	Poland	Warsaw	46	230	Y
2	375292356572	3@mail.ru	Fifth Avenue	USA	New York	30	189	Y
3	375299013465	2@mail.ru	Nikolskaya	Russia	Moscow	32	165	Y
4	375295176493	4@mail.ru	Vladimirskaya	Ukraine	Kiev	36	185	Y
5	375294308850	1@mail.ru	Yesenin	Belarus	Minsk	81	132	Y

Let's create a view that stores all information about employees:

```
CREATE OR REPLACE VIEW sal_cl.w_employee_dimension
AS SELECT * FROM dw_data.employee_dimension;
```

Script Output — Task completed in 0,053 seconds

```
View SAL_CL.W_EMPLOYEE_DIMENSION created.
```

Select from this view:

```
SELECT * FROM sal_cl.w_employee_dimension;
```

Query Result — All Rows Fetched: 25 in 0,023 seconds

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	PHONE	EMAIL	DEPARTMENT	JOB_TITLE	ADDRESS	COUNTRY	CI
1	Adriana	Karnitskaya	375295542493	AdrianaKarnitskaya@mail.ru	department_name_1	director	Yesenin	Belarus	Mins
2	Alyssa	Malysheva	375296172078	AlyssaMalysheva@mail.ru	department_name_4	employee	Vladimirskaya	Ukraine	Kiev
3	Rita	Astafyeva	375294007451	RitaAstafyeva@mail.ru	department_name_2	manager	K Kaiser-Friedrich	Germany	Berl
4	Clara	Zaykova	375297467234	ClaraZaykova@mail.ru	department_name_2	manager	Abby	England	Lonc
5	Eva	Moshko	375297667494	EvaMoshko@mail.ru	department_name_4	employee	Aloyas	Latvia	Riga
6	Ivan	Semin	375293676621	IvanSemin@mail.ru	department_name_4	employee	Via del Corso	Belgium	Brus
7	Ian	Astafyev	375294996103	IanAstafyev@mail.ru	department_name_3	employee	Champs Elysees	France	Pari
8	Ada	Alymova	375292270876	AdaAlymova@mail.ru	department_name_2	manager	Nikolskaya	Russia	Mosc
9	Maxim	Mayorov	375298589471	MaximMayorov@mail.ru	department_name_2	manager	D Dunkri	England	Lonc
10	Veronika	Sadovskaya	375291959498	VeronikaSadovskaya@mail.ru	department_name_3	employee	D Dunkri	Estonia	Tall
11	Pavel	Moshko	375295241317	PavelMoshko@mail.ru	department_name_4	employee	Antoine Dansaert	Belgium	Brus
12	Alexandra	Etkina	375293183125	AlexandraEtkina@mail.ru	department_name_1	director	Graben	Austria	Vier
13	Nikita	Malyshev	375292197332	NikitaMalyshev@mail.ru	department_name_4	employee	G Galve	Latvia	Riga

Let's create a view that stores all information about payment methods:

```
CREATE OR REPLACE VIEW sal_cl.w_payment_method_dimension
AS SELECT * FROM dw_data.payment_method_dimension;
```

Script Output — Task completed in 0,042 seconds

```
View SAL_CL.W_PAYMENT_METHOD_DIMENSION created.
```

Select from this view:

```
SELECT * FROM sal_cl.w_payment_method_dimension;
```

Query Result — All Rows Fetched: 2 in 0,027 seconds

PAYMENT_METHOD_ID	PAYMENT_METHOD_NAME	STATUS
1	bank card	Y
2	cash	Y

Let's create a view that stores all information about gen periods:

```
CREATE OR REPLACE VIEW sal_cl.w_dim_gen_period
AS SELECT * FROM dw_data.dim_gen_period;
```

Script Output — Task completed in 0,045 seconds

```
View SAL_CL.W_DIM_GEN_PERIOD created.
```

Select from this view:

```
SELECT * FROM sal_cl.w_dim_gen_period;
```

Query Result — All Rows Fetched: 6 in 0,024 seconds

PERIOD_ID	VALID_FROM	VALID_TO	PROMOTION_NAME	PROMOTION_PERCENT	DECRIPTION
1	01.04.21	30.04.21	promotion_name_4	5	decription_4
2	01.02.21	28.02.21	promotion_name_2	15	decription_2
3	01.06.21	31.12.21	promotion_name_6	20	decription_6
4	01.03.21	31.03.21	promotion_name_3	10	decription_3
5	01.05.21	31.05.21	promotion_name_5	20	decription_5
6	01.01.21	31.01.21	promotion_name_1	25	decription_1

Let's create a table that stores all information about orders:

```
CREATE TABLE sal_cl.order_fact(
order_id NUMBER,
client_id NUMBER NOT NULL,
employee_id NUMBER NOT NULL,
restaurant_id NUMBER NOT NULL,
date_id DATE NOT NULL,
period_id NUMBER NOT NULL,
payment_method_id NUMBER NOT NULL,
dish_id NUMBER NOT NULL,
dish_amount INT NOT NULL,
total_cost DECIMAL (11,2) NOT NULL,
delivery CHAR(1) NOT NULL CHECK (delivery IN ('N','Y')))
PARTITION BY RANGE (date_id)
--subpartition by hash(client_id) subpartitions 4
(
PARTITION quarter_1 VALUES LESS THAN(to_date('01.04.2021','DD.MM.YYYY')),
PARTITION quarter_2 VALUES LESS THAN(to_date('01.07.2021','DD.MM.YYYY')),
PARTITION quarter_3 VALUES LESS THAN(to_date('01.10.2021','DD.MM.YYYY')),
PARTITION quarter_4 VALUES LESS THAN(to_date('01.01.2022','DD.MM.YYYY'))
);
```

Script Output | Task completed in 0,066 seconds

```
Table SAL_CL.ORDER_FACT created.
```

Let’s create package and procedure to load data to this table:

```
CREATE OR REPLACE PACKAGE pkg_etl_sal_level
AS
   PROCEDURE load_sal_order_fact;
END pkg_etl_sal_level;
```

Script Output | Task completed in 0,069 seconds

```
Package PKG_ETL_SAL_LEVEL compiled
```

```
  PROCEDURE load_sal_order_fact
  AS
  BEGIN
    MERGE INTO sal_cl.order_fact a
    USING (SELECT * FROM dw_data.order_fact) b
    ON (a.order_id=b.order_id)
    WHEN MATCHED THEN
                UPDATE SET a.client_id = b.client_id
    WHEN NOT MATCHED THEN
                INSERT (a.order_id,
                        a.client_id,
                        a.employee_id,
                        a.restaurant_id,
```

Script Output | Task completed in 0,211 seconds

```
Grant succeeded.


Session altered.


Package Body PKG_ETL_SAL_LEVEL compiled
```

```
42  EXECUTE pkg_etl_sal_level.load_sal_order_fact;
```

Script Output | Task completed in 0,631 seconds

```
PL/SQL procedure successfully completed.
```

Select from this table:

```
23  SELECT * FROM sal_cl.order_fact;
24
```

Query Result | SQL | Fetched 50 rows in 0,032 seconds

	ORDER_ID	CLIENT_ID	EMPLOYEE_ID	RESTAURANT_ID	DATE_ID	PERIOD_ID	PAYMENT_METHOD_ID	DISH_ID	DISH_AMOUNT	TOTAL_COST	DELIVERY
1	306223	5	11	2	31.03.21	4	1	4	23	1438	N
2	306224	5	21	4	31.03.21	4	1	5	6	498	N
3	306286	5	1	3	28.01.21	6	1	2	4	100	N
4	306287	5	8	2	28.01.21	6	1	1	96	1447	N
5	306288	5	13	4	28.01.21	6	1	5	2	179	N
6	306289	5	13	1	28.01.21	6	1	3	12	1113	N
7	306290	5	1	3	30.01.21	6	2	5	4	374	N
8	306291	5	13	5	31.01.21	6	1	4	23	1419	N
9	306292	5	14	4	30.01.21	6	2	4	5	344	N
10	306293	5	21	1	30.01.21	6	2	5	0	35	N
11	306294	5	13	4	31.01.21	6	1	5	0	35	N
12	306295	5	11	5	29.01.21	6	2	5	10	845	N
13	306296	5	8	3	29.01.21	6	2	2	52	1313	N