Lab report #5

Sadovskaya Veronika

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

Task 1 - Create Packages for Reload Dimension from SA_*

1) Use DBMS_SQL.TO_REFCURSOR Function

Truncate table dw_data.payment_method_dimension that was created in Lab 4 unit 2:

```
22  truncate table dw_data.payment_method_dimension;
```

Script Output | Task completed in 0,041 seconds

```
Table DW_DATA.PAYMENT_METHOD_DIMENSION truncated.
```

Create new procedure for moving data from cleansing layer to DW layer using the example of payment_method_dimension:

```
21    END load_CLEAN_PAYMENT_METHODS_DW;
22
23    PROCEDURE load_CLEAN_PAYMENT_METHODS_DW_with_to_refcursor_func
24    AS
25      BEGIN
26      DECLARE
27        cursor_id NUMBER (25);
28        cur_count NUMBER (38);
29        quary_cur VARCHAR2(2000);
30        TYPE ref_crsr IS REF CURSOR;
31        ref_cursor ref_crsr;
32        TYPE type_rec IS RECORD
33        (  payment_method_name  VARCHAR2 ( 100 )
34         , status char(1)
35         , payment_method_id  INT
36        );
37        one_record type_rec;
38
39        BEGIN
40        quary_cur:= 'SELECT payment_method_name, status, payment_method_id FROM
```

Script Output | Query Result | Task completed in 0,589 seconds

```
Package Body PKG_ETL_PAYMENT_METHODS_DW compiled
```

Execute new procedure:

```
1  BEGIN
2  --    pkg_etl_clients_dw.load_CLEAN_CLIENTS_DW;
3  --    pkg_etl_restaurants_dw.load_CLEAN_RESTAURANTS_DW;
4  --    pkg_etl_employees_dw.load_CLEAN_EMPLOYEES_DW;
5  --    pkg_etl_dishes_dw.load_CLEAN_DISHES_DW;
6  --    pkg_etl_payment_methods_dw.load_CLEAN_PAYMENT_METHODS_DW;
7  --    pkg_etl_gen_periods_dw.load_CLEAN_GEN_PERIODS_DW;
8  --    pkg_etl_transactions_dw.load_CLEAN_TRANSACTIONS_DW;
9      pkg_etl_payment_methods_dw.load_CLEAN_PAYMENT_METHODS_DW_with_to_refcursor_func;
10 END;
```

Script Output | Task completed in 0,053 seconds

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

Select data from table dw_data.payment_method_dimension that was filled using the new procedure:

```
20  SELECT * FROM dw_data.payment_method_dimension;
```

Script Output | Query Result | SQL | All Rows Fetched: 2 in 0,03 seconds

	PAYMENT_METHOD_ID	PAYMENT_METHOD_NAME	STATUS
1	1	bank card	Y
2	2	cash	Y

2) Use DBMS_SQL.TO_CURSOR_NUMBER Function

Truncate table dw_data.payment_method_dimension that was created in Lab 4 unit 2:

```
22  truncate table dw_data.payment_method_dimension;
```

Script Output | Task completed in 0,041 seconds

```
Table DW_DATA.PAYMENT_METHOD_DIMENSION truncated.
```

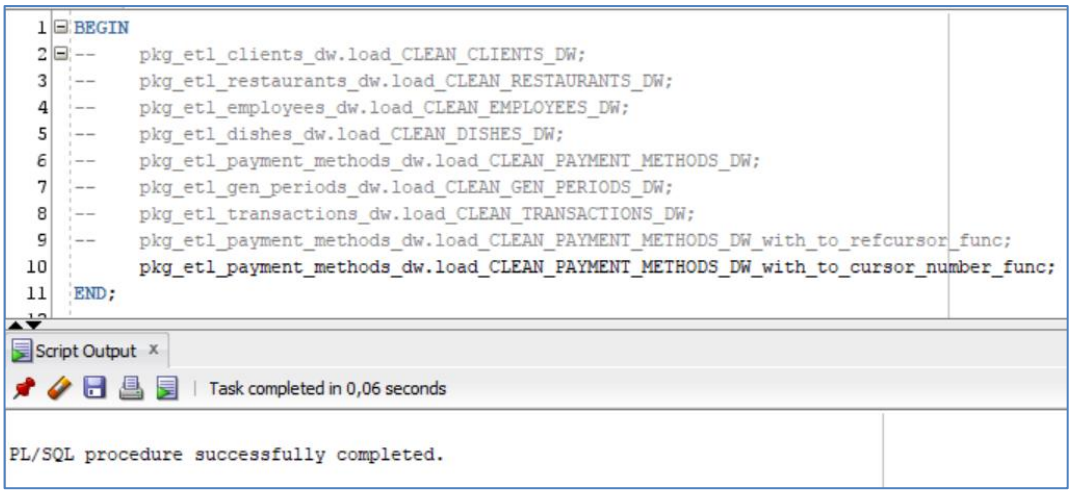
Create new procedure for moving data from cleansing layer to DW layer using the example of payment_method_dimension:

```
73   PROCEDURE load_CLEAN_PAYMENT_METHODS_DW_with_to_cursor_number_func
74   AS
75  BEGIN
76    DECLARE
77     TYPE curtype IS REF CURSOR;
78     src_cur   curtype;
79     curid     NUMBER;
80     desctab   DBMS_SQL.DESC_TAB;
81     colcnt    NUMBER;
82     payment_method_name VARCHAR(20);
83     status CHAR(1);
84     payment_method_id NUMBER;
85     sql_stmt VARCHAR2(2000);
86
87     BEGIN
88     sql_stmt:= 'SELECT payment_method_name, status, payment_method_id FROM
89                        (SELECT source.payment_method_name AS payment_method_name,
90                         source.payment_method_status AS status,
```

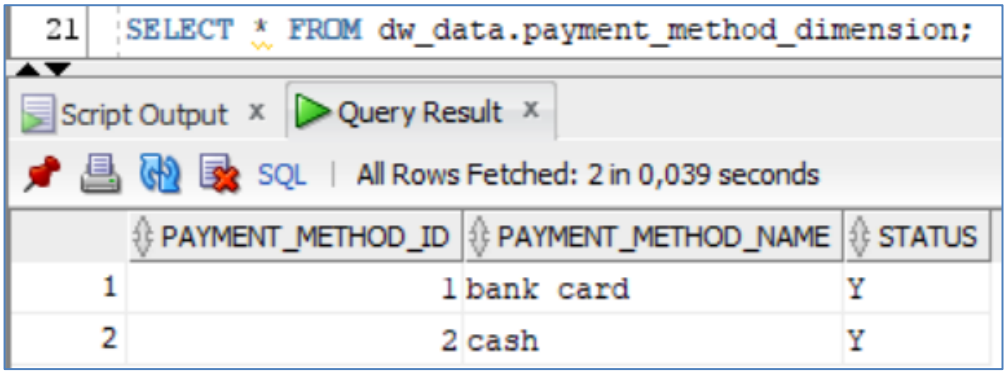
Script Output | Query Result | Task completed in 0,775 seconds

```
Package Body PKG_ETL_PAYMENT_METHODS_DW compiled
```

Execute new procedure:

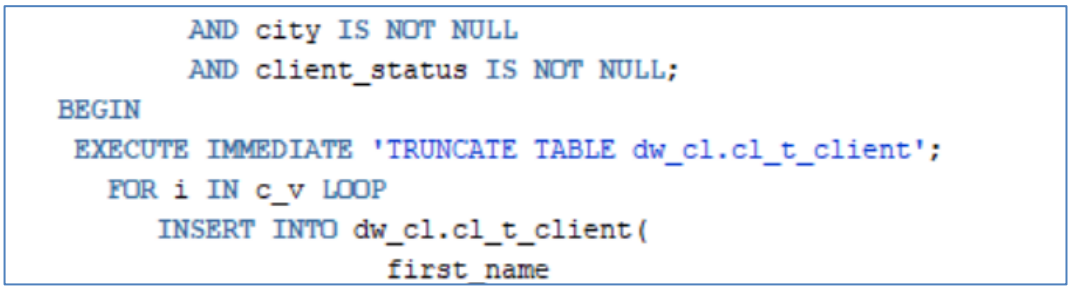


Select data from table dw_data.payment_method_dimension that was filled using the new procedure:



3) Use Execute Immediate with Bind Parameters

Execute Immediate was used back in lab 4 when creating a layer, when we needed to first clear the table before populating it. Example (DW_CL/clients/ pkg_etl_clients_cl_imp):



Task 2 - CREATE Monthly Reports Layouts

Using the model, the total number of dishes sold for each item is calculated for the year and for the entire time the restaurant has been operating.

```
10  SELECT dish_name, year, month, period, dish_count
11  FROM CTE_FN
12  MODEL
13  PARTITION BY (dish_name)
14  DIMENSION BY (year, month, period)
15  MEASURES(dish_count)
16  RULES (
17      dish_count[FOR year IN (SELECT DISTINCT year FROM CTE_FN),
18      NULL, 'YEAR']=SUM(dish_count)[cv(year), ANY, 'MONTH'],
19      dish_count [NULL, NULL, 'ALL']=SUM(dish_count)[ANY, NULL,'YEAR']
20      )
21  ORDER BY dish_name, year, month;
```

Query Result

SQL | All Rows Fetched: 70 in 0,212 seconds

	DISH_NAME	YEAR	MONTH	PERIOD	DISH_COUNT
62	soup	01.01.21	06	MONTH	7500
63	soup	01.01.21	07	MONTH	7750
64	soup	01.01.21	08	MONTH	7750
65	soup	01.01.21	09	MONTH	7500
66	soup	01.01.21	10	MONTH	7750
67	soup	01.01.21	11	MONTH	7500
68	soup	01.01.21	12	MONTH	7750
69	soup	01.01.21	(null)	YEAR	91250
70	soup	(null)	(null)	ALL	91250