# Report

Laboratory Work 4

Dmitry Ladutsko

August 13, 2022

### 1. Prerequisites

#### 1.1. Passwords Index

Password Group	Login Name	Password
Operation System	root	"rootadmin"
	oracle	"oracleadmin"
Oracle System	sys	"sysadmin"
	system	"sysadmin"
Oracle Users	All DB users	"%PWD%"

#### 1.2. Folder Paths Index

Path Group	Path Description	Path
Operation System	Oracle RDBMS – BIN	/oracle/app/oracle
	Oracle Inventory	/oracle/app/oraInventory
	Oracle Database Storage	/oracle/oradata
	Oracle Install Directory	/oracle/install
Oracle	ORACLE_BASE	/oracle/app/oracle
	ORACLE_HOME	\$ORACLE_BASE/product/11.2
FTP	ftp Incoming Folder	/ftp/incoming

# 2. Business analyses tasks – Reports

# 2.1. Task 01: Create Packages for Reload Dimension from SA \*

<u>The Main Task</u> is to independent packages to reload dimension according your DWH solution concept which was developed on Module 6. Introduction to DWH.

### **Required points:**

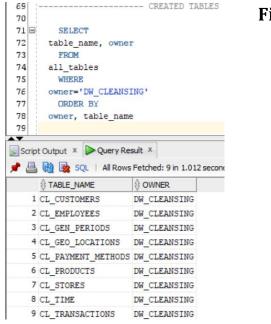
- Create all required Dim objects on DW Layer
- Grant all required Privileges to DW\_CL (Cleansing Layer)
- Create Packages to reload Dim data (one package = one dimension.)
- Example future SAL.DIM\_GEO\_SCD will store all procedure on pkg\_etl\_dim\_geo\_dw. But this package will store all small sub dims T\_COUNTRIES, T\_REGIONS etc. )
  - Use Explicit Cursor (One package)
  - Use Explicit Cursor and FORALL Bulk Insertion (One package)

- Use Variable Cursor and FORALL Bulk Insertion (One package)
- Use Merge (One packages)

#### **Task Results:**

Create required objects:

- Put objects script to Git.
- Prepare Document with Screenshot of Data on Dimensions
- Test data for consistent
- Test Procedure for Repeatable execution (Nothing should change)



### Firstly, I created tables on Cleansing level

```
SELECT

table_name, owner

FROM

all_tables

WHERE

owner='DW_CLEANSING'

ORDER BY

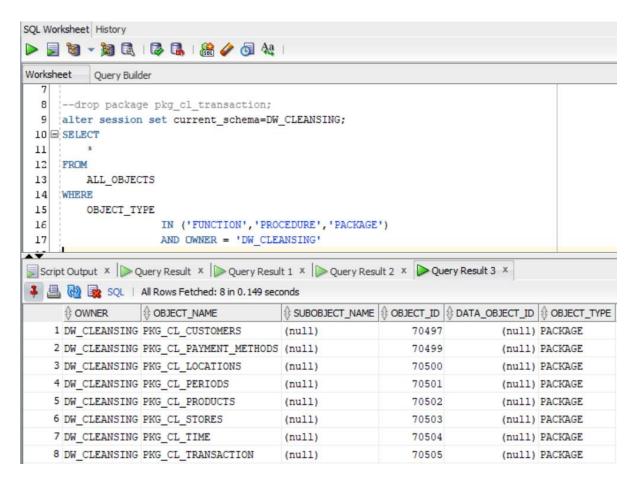
owner, table_name
```

- CL\_CUSTOMERS
- CL\_EMPLOYEES
- CL\_GEN\_PERIODS
- CL\_GEO\_LOCATIONS
- CL\_PAYMENT\_METHODS
- CL\_PRODUCTS
- CL\_STORES
- CL\_TIME
- CL\_TRANSACTIONS

**Note.** All scripts stored in GitHub, exit task folder. But also duplicated in lab4 folder AS CL folder.

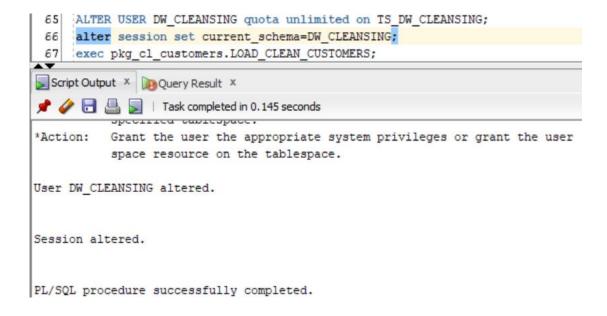
Then I began creating packages, which consist of 2 files:

- \*table\_name\*\_define.sql
- \*table\_name\*\_body.sql

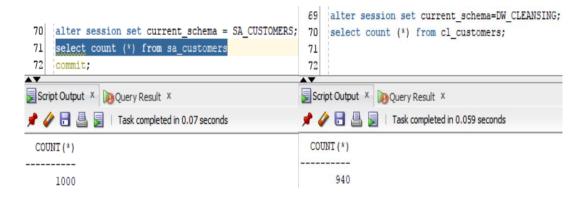


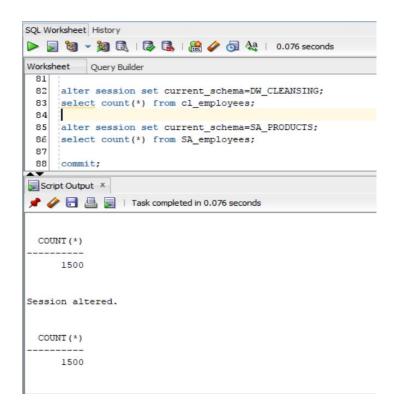
**Note.** Later I will show the **Data Flow Diagram** to explain how I store files / sub – folders / folders

Now let's try to create package bodies. And of course execute them to move data from SA level to CL level



\*After almost an hour I finally understood that it is a great idea to grant my user more space resource :)





As you can see, if we **count** rows from the same tables (but on different layers(different tablespaces)) thee number of rows differs because in \*.SA table were some rows with NULL values.

**Note.** Next I am going to make a body statements for every table (except of some(1-3), which are not needed to be cleansed)

*Note.* However, employees table **was not cleansed** because of they were created using *NOT NULL* **constraint**. It seems to be useless cleansing such

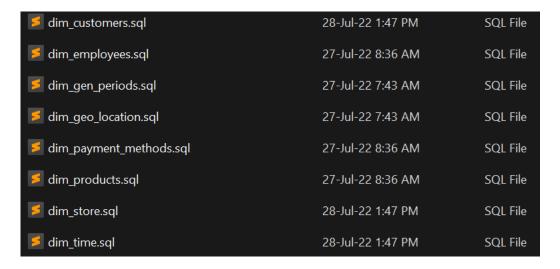
tables but if someone will change this table structure **in future** and try to use *null* values in some rows, then **it will be worthy**.

*Note.* I coded every package and executed their bodies before moving data from sa\_\* tables to cl\_\* tables, and generating transactions table on cleansing layer.

```
SQL Worksheet History
Worksheet
           Query Builder
 66
 67
         DW_CLEANSING.cl_products p
 68
 69
          CROSS JOIN DW_CLEANSING.cl_customers c
 70
 71
             CROSS JOIN DW_CLEANSING.cl_payment_methods pm
 72
                 CROSS JOIN DW_CLEANSING.CL_EMPLOYEES e
 73
 74
                  INNER JOIN DW_CLEANSING.cl_time t ON t.date_id = c.CUSTOMER_SALE_DATE
 75
 76
      WHERE c.CUSTOMER_SALE_DATE > TO DATE ( '01.01.20', 'MM/DD/YY' )
 77
             AND c.CUSTOMER_SALE_DATE < TO DATE ( '01.01.22', 'MM/DD/YY' ) and
 78
 79
 80
              (mod(c.PHONE_CUSTOMER, 2) = 0 ) and
 81
              e.phone EMPLOYEE LIKE '2%' AND
 82
 83
              e.FIRST NAME EMPLOYEE LIKE 'D%'
 84
 85
                                             );
Script Output X Query Result X Query Result 1 X
📌 🧽 🔚 💂 📘 | Task completed in 15.367 seconds
2,436,480 rows inserted.
```

*Note.* Finally, I got  $\sim$ 2.5 m rows inserted into cl.transaction table fulfilled with fully cleansed data (of course with some artificial constraints)!

**Note.** In fact, we need to add only decision – making data. I thought that I could throw away some columns (e.g. customers/employees emails, phones etc.), but then I realised I could use it to, for example, select customers emails to target marketing content. So, anyway if I decide it is not needed, I will alter this tables.



*Note.* We have created dimension tables before. Copy of dimensions creation scripts duplicated in lab4 folder.

```
DROP SEQUENCE SEQ_CUSTOMERS;
CREATE SEQUENCE SEQ_CUSTOMERS
START WITH 1
INCREMENT BY 1
NOCACHE
NOCYCLE;
```

**Note.** I created such sequences to auto increment primary key id's. Then again added packages using following structure :

- \*table\_name\*\_define.sql
- \*table\_name\*\_body.sql

```
Worksheet Query Builder
 14 CREATE OR REPLACE PACKAGE body pkg_dw_customers
 16 PROCEDURE LOAD_DW_CUSTOMERS
 18
          BEGIN
 19 🖃
          MERGE INTO DW DATA.dim customers A
          USING ( SELECT FIRST_NAME_CUSTOMER, LAST_NAME_CUSTOMER, COUNTRY_CUTY_CUSTOMER, ADRESS_CUSTOMER, EMAIL, PHONE_CUSTOMER, AGE, IS_ACTI
 20
                  FROM DW_CLEANSING.cl_customers) B
 21
                  ON (a.PHONE_CUSTOMER = b.PHONE_CUSTOMER)
                     UPDATE SET a.ADRESS_CUSTOMER = b.ADRESS_CUSTOMER
 25
                  WHEN NOT MATCHED THEN
 26
                     INSERT (a.CUSTOMER_ID, a.FIRST_NAME_CUSTOMER, a.LAST_NAME_CUSTOMER, a.COUNTRY_CITY_CUSTOMER, a.ADRESS_CUSTOMER, a.EMAIL
 27
                     VALUES (SEQ_CUSTOMERS.NEXTVAL, b.FIRST_NAME_CUSTOMER, b.LAST_NAME_CUSTOMER, b.COUNTRY_CITY_CUSTOMER, b.ADRESS_CUSTOMER,
 28
          COMMIT:
        END LOAD DW CUSTOMERS:
 29
 30 END pkg_dw_customers;
```



I used **Use Explicit Cursor** in pkg\_cl\_employees\_body.sql

**Merge** in pkg\_dw\_employees\_body.sql

**Explicit Cursor and FOR ALL BLUNK Insertion** in pkg\_dw\_stores\_body.sql