U2M5.LW.Advanced SQL, PL/SQL

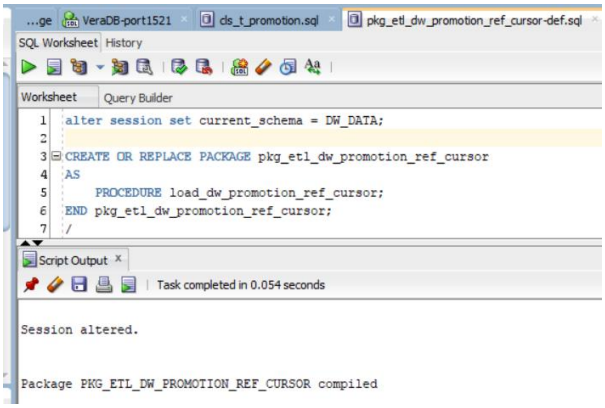
Shkrabatouskaya Vera

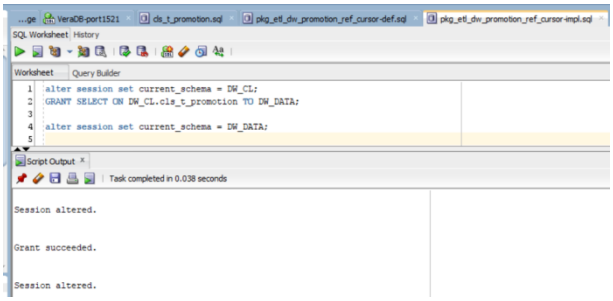
https://github.com/VeraShkrabatouskaya/DataMola_Data-Camping-2022

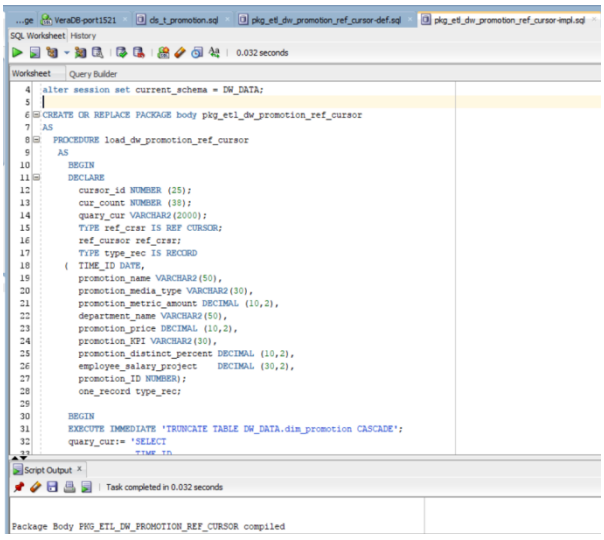
2. Business analyses tasks – Dimensions

2.1. Task 01: Create Packages for Reload Dimension from SA_*

Let's create independent packages to reload dimension from CL-level to DW-level according DWH solution concept using DBMS_SQL.TO_REFCURSOR Function.







```
alter session set current_schema = DW_DATA;
alter user DW_DATA QUOTA UNLIMITED ON TS_DW_DATA_01;

EXEC pkg_etl_dw_promotion_ref_cursor.load_dw_promotion_ref_cursor;
```

Script Output | Task completed in 68.339 seconds

```
Session altered.

User DW_DATA altered.

PL/SQL procedure successfully completed.
```

```
SELECT * FROM DW_DATA.DIM_promotion order by 1;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0.173 seconds

	PROMOTION_ID	TIME_ID	PROMOTION_NAME	PROMOTION_MEDIA_TYPE	PROMOTION_METRIC_AMOUNT	DEPARTMENT_NAME	PROMOTION_PRICE	PROMOTION_KPI
1	1	04-JAN-22	5315 SAMSUNG_Vacuum Cleaners_press	press	92	Media Planning and Buying	353	CPL
2	2	04-JAN-22	5324 SAMSUNG_Vacuum Cleaners_press	press	19	Media Planning and Buying	250	CPA
3	3	04-JAN-22	5326 SAMSUNG_Smart watches_press	press	50	Media Planning and Buying	651	CPA
4	4	04-JAN-22	5327 SAMSUNG_Smart watches_press	press	72	Media Planning and Buying	303	CPA
5	5	04-JAN-22	5332 SAMSUNG_Smart watches_press	press	31	Media Planning and Buying	915	CPL
6	6	04-JAN-22	5350 SAMSUNG_TVs_press	press	13	Media Planning and Buying	587	CPL
7	7	04-JAN-22	5364 SAMSUNG_Smartphones_press	press	52	Media Planning and Buying	720	CPL
8	8	04-JAN-22	5366 SAMSUNG_Smartphones_press	press	22	Media Planning and Buying	490	CPA
9	9	04-JAN-22	5377 SAMSUNG_Headphones_press	press	88	Media Planning and Buying	698	CPL
10	10	04-JAN-22	5395 SAMSUNG_Tablets_press	press	55	Media Planning and Buying	175	CPA

```
SELECT count(*) FROM DW_DATA.DIM_promotion;
```

Script Output | Query Result | SQL | All Rows Fetched: 1 in 0.017 seconds

	COUNT(*)
1	336000

Code:

```
CREATE OR REPLACE PACKAGE body pkg_etl_dw_promotion_ref_cursor
AS
 PROCEDURE load_dw_promotion_ref_cursor
  AS
   BEGIN
   DECLARE
    cursor_id NUMBER (25);
    cur_count NUMBER (38);
    quary_cur VARCHAR2(2000);
    TYPE ref_crsr IS REF CURSOR;
    ref_cursor ref_crsr;
    TYPE type_rec IS RECORD
  (  TIME_ID DATE,
    promotion_name VARCHAR2(50),
    promotion_media_type VARCHAR2(30),
    promotion_metric_amount DECIMAL (10,2),
    department_name VARCHAR2(50),
    promotion_price DECIMAL (10,2),
    promotion_KPI VARCHAR2(30),
    promotion_distinct_percent DECIMAL (10,2),
    employee_salary_project    DECIMAL (30,2),
    promotion_ID NUMBER);
    one_record type_rec;

   BEGIN
   EXECUTE IMMEDIATE 'TRUNCATE TABLE DW_DATA.dim_promotion CASCADE';
   quary_cur:= 'SELECT
            TIME_ID,
            promotion_name,
            promotion_media_type,
            promotion_metric_amount,
            department_name,
```

```
            promotion_price,
            promotion_KPI,
            promotion_distinct_percent,
            employee_salary_project,
            promotion_ID FROM
                (SELECT source.TIME_ID AS TIME_ID,
                 source.promotion_name AS promotion_name,
                 source.promotion_media_type AS  promotion_media_type,
                 source.promotion_metric_amount AS  promotion_metric_amount,
                 source.department_name AS department_name,
                 source.promotion_price AS promotion_price,
                 source.promotion_KPI AS promotion_KPI,
                 source.promotion_distinct_percent AS promotion_distinct_percent,
                 source.employee_salary_project AS employee_salary_project,
                 stage.promotion_ID AS promotion_ID
            FROM DW_CL.cls_t_promotion source
            LEFT JOIN DW_DATA.DIM_promotion stage
            ON (source.promotion_name = stage.promotion_name AND source.TIME_ID = stage.TIME_ID))';

    cursor_id:=DBMS_SQL.open_cursor;

    DBMS_SQL.PARSE(cursor_id, quary_cur, DBMS_SQL.NATIVE);

    cur_count:= DBMS_SQL.EXECUTE(cursor_id);

    ref_cursor:= DBMS_SQL.TO_REFCURSOR(cursor_id);

    LOOP
    FETCH ref_cursor INTO one_record;
    EXIT WHEN ref_cursor%NOTFOUND;
    IF (one_record.promotion_ID IS NULL) THEN
          INSERT INTO DW_DATA.DIM_promotion(  promotion_ID,
                                  TIME_ID,
                                  promotion_name,
                                  promotion_media_type,
                                  promotion_metric_amount,
                                  department_name,
                                  promotion_price,
                                  promotion_KPI,
                                  promotion_distinct_percent,
                                  employee_salary_project)
          VALUES (DW_DATA.SQ_DIM_promotion.NEXTVAL,
                one_record.TIME_ID,
                one_record.promotion_name,
                one_record.promotion_media_type,
                one_record.promotion_metric_amount,
                one_record.department_name,
                one_record.promotion_price,
                one_record.promotion_KPI,
                one_record.promotion_distinct_percent,
                one_record.employee_salary_project
                );
    END IF;
    END LOOP;
   COMMIT;
   END;
  END load_dw_promotion_ref_cursor;
END pkg_etl_dw_promotion_ref_cursor;
```

Let’s create independent packages to reload dimension from CL-level to DW-level according DWH solution concept using DBMS_SQL.TO_CURSOR_NUMBER Function.

```
alter session set current_schema = DW_DATA;

CREATE OR REPLACE PACKAGE pkg_etl_dw_product_cursor_number
AS
    PROCEDURE load_dw_product_cursor_number;
END pkg_etl_dw_product_cursor_number;
/
```

```
Session altered.

Package PKG_ETL_DW_PRODUCT_CURSOR_NUMBER compiled
```

```
alter session set current_schema = DW_CL;
GRANT SELECT ON DW_CL.cls_t_product TO DW_DATA;

alter session set current_schema = DW_DATA;
```

```
Session altered.

Grant succeeded.

Session altered.
```

```
alter session set current_schema = DW_DATA;

CREATE OR REPLACE PACKAGE body pkg_etl_dw_product_cursor_number
AS
 PROCEDURE load_dw_product_cursor_number
   AS
   BEGIN
      DECLARE

        TYPE BIG_CURSOR IS REF CURSOR;
        TYPE T_REC_product IS RECORD
        (
        brand_name  VARCHAR2(50),
        product_name  VARCHAR2(50),
        category_name  VARCHAR2(50),
        subcategory_name  VARCHAR2(50),

        product_name_stage VARCHAR(20),
        product_ID NUMBER );
        TYPE t_product  IS TABLE OF T_REC_product ;
        ALL_INF BIG_CURSOR;
     record_product  t_product ;
     curid NUMBER ( 25 );
 BEGIN
    OPEN ALL_INF FOR
        SELECT source_CL.brand_name AS brand_name_source_CL
                , source_CL.product_name AS product_name_source_CL
                , source_CL.category_name AS category_name_source_CL
                , source_CL.subcategory_name AS subcategory_name_source_CL
```

```
Package Body PKG_ETL_DW_PRODUCT_CURSOR_NUMBER compiled
```

```
alter session set current_schema = DW_DATA;
alter user DW_DATA QUOTA UNLIMITED ON TS_DW_DATA_01;

EXEC pkg_etl_dw_product_cursor_number.load_dw_product_cursor_number;
```

```
Session altered.

User DW_DATA altered.

PL/SQL procedure successfully completed.
```

```
SELECT * FROM DW_DATA.DIM_product order by 1;
```

All Rows Fetched: 16 in 0.003 seconds

	PRODUCT_ID	BRAND_NAME	PRODUCT_NAME	CATEGORY_NAME	SUBCATEGORY_NAME
1	1	SAMSUNG	Microwave ovens	Appliances	Convection Microwave
2	2	SAMSUNG	Smart watches	Computerized wristwatch	Smart watch for kids?
3	3	SAMSUNG	TVs	Appliances	QLED
4	4	SAMSUNG	Cooking ovens	Appliances	Electric cooktop
5	5	SAMSUNG	Ovens	Appliances	Conventional ovens
6	6	SAMSUNG	Vacuum Cleaners	Appliances	Robotic vacuum cleaner
7	7	SAMSUNG	Smartphones	Phones	Android Phones
8	8	VISA	Service	Services	Internet banks
9	9	VISA	Card	Services	Contactless smart card
10	10	SAMSUNG	Refrigerators	Appliances	Fridge
11	11	SAMSUNG	Dishwashers	Appliances	Built-in Dishwasher
12	12	SAMSUNG	Tablets	Tablet computer	Galaxy Tab series?
13	13	SAMSUNG	Headphones	Accessories	Earbuds
14	14	VISA	Sponsorship	Services	Networking sponsor
15	15	SAMSUNG	Washing machines	Appliances	Fully Automatic washing machine
16	16	VISA	Benefit	Services	Retirement Savings

Code:

```
CREATE OR REPLACE PACKAGE body pkg_etl_dw_product_cursor_number
AS
 PROCEDURE load_dw_product_cursor_number
  AS
  BEGIN
    DECLARE

     TYPE BIG_CURSOR IS REF CURSOR;
     TYPE T_REC_product IS RECORD
     (
     brand_name  VARCHAR2(50),
     product_name  VARCHAR2(50),
     category_name  VARCHAR2(50),
     subcategory_name  VARCHAR2(50),

     product_name_stage VARCHAR(20),
     product_ID NUMBER );
     TYPE t_product  IS TABLE OF T_REC_product ;
     ALL_INF BIG_CURSOR;
   record_product  t_product ;
   curid NUMBER ( 25 );
 BEGIN
   OPEN ALL_INF FOR
      SELECT source_CL.brand_name AS brand_name_source_CL
            , source_CL.product_name AS product_name_source_CL
            , source_CL.category_name AS category_name_source_CL
            , source_CL.subcategory_name AS subcategory_name_source_CL

            , stage.product_name AS product_name_stage
            , STAGE.product_ID AS product_ID
        FROM (SELECT DISTINCT * FROM DW_CL.cls_t_product) source_CL
               LEFT JOIN
                 DW_DATA.DIM_product stage
               ON (source_CL.product_name = stage.product_name);


   FETCH ALL_INF
   BULK COLLECT INTO record_product ;


     curid := dbms_sql.to_cursor_number ( ALL_INF );
   dbms_sql.close_cursor ( curid );

   FOR i IN record_product.FIRST .. record_product.LAST LOOP
     IF ( record_product(i).product_ID IS NULL ) THEN
       INSERT INTO dim_product(  product_ID,
                                 brand_name,
                                 product_name,
                                 category_name,
                                 subcategory_name)
           VALUES ( SQ_DIM_product.NEXTVAL
                 , record_product(i).brand_name
                   , record_product(i).product_name
                   , record_product(i).category_name
                   , record_product(i).subcategory_name);

       COMMIT;
     END IF;
   END LOOP;
 END;
  END load_dw_product_cursor_number;
END pkg_etl_dw_product_cursor_number;
```

3. Business analyses tasks – Reports

3.1. Task 02: CREATE Monthly Reports Layouts

Create selects to calculate Salary of the Starcom advertising network employees by month using Module Clause.

```sql
--reports by Month
--------------------------------------------------------------------------------
WITH salary_employees AS (
SELECT TRUNC(TIME_ID, 'YYYY') AS year,
TO_CHAR(TIME_ID, 'MM') AS month,'MONTH' period
,SUM (employee_salary_project) as net_salary_employee_dollar_amount
FROM   DW_CL.cls_t_transaction
GROUP BY TRUNC(TIME_ID, 'YYYY'), TO_CHAR(TIME_ID, 'MM'))
select year,month,period, net_salary_employee_dollar_amount
from salary_employees
    MODEL
      DIMENSION BY ( year, month,period)
      MEASURES ( net_salary_employee_dollar_amount)
       RULES(
    net_salary_employee_dollar_amount[FOR year IN (SELECT DISTINCT year FROM salary_employees),
    NULL, 'YEAR']=SUM(net_salary_employee_dollar_amount)[cv(year), ANY, 'MONTH'],
    net_salary_employee_dollar_amount [NULL, NULL, 'ALL']=SUM(net_salary_employee_dollar_amount)[ANY, NULL,'YEAR'])
;
```

All Rows Fetched: 9 in 0.21 seconds

	YEAR	MONTH	PERIOD	NET_SALARY_EMPLOYEE_DOLLAR_AMOUNT
1	01-JAN-22	01	MONTH	14286455
2	01-JAN-22	02	MONTH	12919417
3	01-JAN-22	03	MONTH	14324052
4	01-JAN-22	04	MONTH	13823913
5	01-JAN-22	05	MONTH	14313747
6	01-JAN-22	06	MONTH	13847563
7	01-JAN-22	07	MONTH	8826010
8	01-JAN-22	(null)	YEAR	92341157
9	(null)	(null)	ALL	92341157

```sql
WITH salary_employees AS (
SELECT TRUNC(TIME_ID, 'YYYY') AS year,
TRUNC (TIME_ID, 'MM') AS month,'MONTH' period
,SUM (ROUND(employee_salary_project*(1+ agency_VAT_percent/100),2)) as gross_salary_employee_dollar_amount
,SUM (employee_salary_project) as net_salary_employee_dollar_amount
FROM   DW_CL.cls_t_transaction
GROUP BY TRUNC(TIME_ID, 'YYYY'), TRUNC (TIME_ID, 'MM'))

select year, month, period, gross_salary_employee_dollar_amount, net_salary_employee_dollar_amount
from salary_employees
    MODEL
      DIMENSION BY ( year, month, period)
      MEASURES ( gross_salary_employee_dollar_amount, net_salary_employee_dollar_amount)
       RULES(

    gross_salary_employee_dollar_amount[FOR year IN (SELECT DISTINCT year FROM salary_employees),
    NULL, 'YEAR']=SUM(gross_salary_employee_dollar_amount)[cv(year), ANY, 'MONTH'],
    gross_salary_employee_dollar_amount [NULL, NULL,'ALL']=SUM(gross_salary_employee_dollar_amount)[ANY, NULL,'YEAR'],

    net_salary_employee_dollar_amount[FOR year IN (SELECT DISTINCT year FROM salary_employees),
```

All Rows Fetched: 9 in 0.258 seconds

	YEAR	MONTH	PERIOD	GROSS_SALARY_EMPLOYEE_DOLLAR_AMOUNT	NET_SALARY_EMPLOYEE_DOLLAR_AMOUNT
1	01-JAN-22	01-JAN-22	MONTH	16740345.73	14286455
2	01-JAN-22	01-FEB-22	MONTH	15139839.59	12919417
3	01-JAN-22	01-MAR-22	MONTH	16782887.55	14324052
4	01-JAN-22	01-APR-22	MONTH	16197508.94	13823913
5	01-JAN-22	01-MAY-22	MONTH	16768528.37	14313747
6	01-JAN-22	01-JUN-22	MONTH	16224818.16	13847563
7	01-JAN-22	01-JUL-22	MONTH	10340195.31	8826010
8	01-JAN-22	(null)	YEAR	108194123.65	92341157
9	(null)	(null)	ALL	108194123.65	92341157

Create selects to calculate Revenues, Costs and Profits of the Starcom advertising network by month using Module Clause.

Lab6_Task3.sql | Welcome Page | VeraDB | Lab5_Task2_Business analyses tasks.sql | Lab 2_Tasks 03-05_Reports.sql | t_FCT_business.sql

SQL Worksheet | History

Worksheet | Query Builder

```
--Revenues, costs, profits of the Starcom advertising network by month using Module Clause
-----------------------------------------------------------------------------
WITH budget_agencies AS (
SELECT TRUNC(TIME_ID, 'YYYY') AS year,
TRUNC (TIME_ID, 'MM') AS month,'MONTH' period
,       SUM (ROUND(((promotion_metric_amount*promotion_price*(1-promotion_distinct_percent/100))*agency_fee_percent/100)*(1+agency_VAT_percent/100),2)) as gross_re
,       SUM (ROUND(((promotion_metric_amount*promotion_price*(1-promotion_distinct_percent/100))*agency_fee_percent/100),2)) as net_revenue_dollar_amount
,       SUM (ROUND(employee_salary_project*(1+ agency_VAT_percent/100),2)) as gross_cost_dollar_amount
,       SUM (employee_salary_project) as net_cost_dollar_amount
,       SUM (ROUND((((promotion_metric_amount*promotion_price*(1-promotion_distinct_percent/100))*agency_fee_percent/100)*(1+agency_VAT_percent/100))-(employee_sal
,       SUM (ROUND(((((promotion_metric_amount*promotion_price*(1-promotion_distinct_percent/100))*agency_fee_percent/100))-employee_salary_project),2)) as net_pro
FROM    DW_CL.cls_t_transaction
GROUP BY TRUNC(TIME_ID, 'YYYY'), TRUNC (TIME_ID, 'MM'))

select year, month, period, gross_revenue_dollar_amount, net_revenue_dollar_amount, gross_cost_dollar_amount, net_cost_dollar_amount, gross_profit_dollar_amount,
from budget_agencies
    MODEL
      DIMENSION BY ( year, month, period)
      MEASURES ( gross_revenue_dollar_amount, net_revenue_dollar_amount, gross_cost_dollar_amount, net_cost_dollar_amount, gross_profit_dollar_amount, net_profit_
```

Query Result | Query Result 1 | Query Result 2

SQL | All Rows Fetched: 9 in 0.513 seconds

	YEAR	MONTH	PERIOD	GROSS_REVENUE_DOLLAR_AMOUNT	NET_REVENUE_DOLLAR_AMOUNT	GROSS_COST_DOLLAR_AMOUNT	NET_COST_DOLLAR_AMOUNT	GROSS_PROFIT_DOLLAR_AMOUNT	NET
1	01-JAN-22	01-JAN-22	MONTH	70046770.67	59628652.72	16740345.73	14286455	53306426.07	
2	01-JAN-22	01-FEB-22	MONTH	63104735.04	53720403.44	15139839.59	12919417	47964896.56	
3	01-JAN-22	01-MAR-22	MONTH	69942057.43	59541532.54	16782887.55	14324052	53159171.46	
4	01-JAN-22	01-APR-22	MONTH	67700034.25	57620043.59	16197508.94	13823913	51502526.26	
5	01-JAN-22	01-MAY-22	MONTH	70040549.3	59622260.05	16768528.37	14313747	53272022.79	
6	01-JAN-22	01-JUN-22	MONTH	66666509.45	56736334.8	16224818.16	13847563	50441692.44	
7	01-JAN-22	01-JUL-22	MONTH	43176592.47	36750041.89	10340195.31	8826010	32836398.19	
8	01-JAN-22	(null)	YEAR	450677248.61	383619269.03	108194123.65	92341157	342483133.77	
9	(null)	(null)	ALL	450677248.61	383619269.03	108194123.65	92341157	342483133.77	