EISTI

École Internationale des Sciences du Traitement de l'Information

www.eisti.fr

PlSQL/Bussiness intelignece

Gustavo Fleury Soares

Induraj P. ramamurthy

Professor:

Rachid Chelouah

Cergy / FR

December, 2019

Table of Contents

1 Introduction

The Motive of this Work is to:

- Combine the knowledge obtained from PLSQL class and BI class towards a real project;
- Combining the PLSQL with Business intelligence to transform OLTP database to OLAP database (star schema).

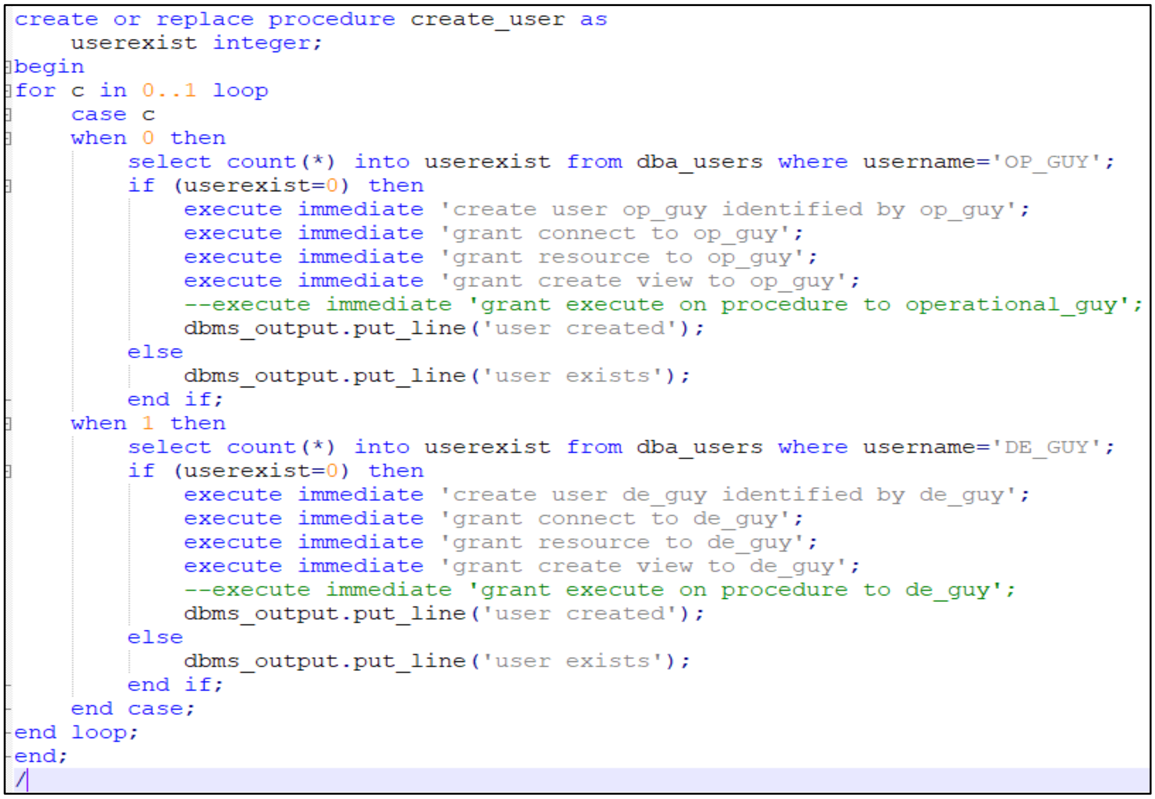
To realize the project, we used

- Procedures;
- Triggers;
- Sequence;
- Surrogate key;
- Use the Random package to Populate the database;
- Create OLAP cube and Link the OLAP database to Visualization Tools Qlik Sense.

2 Sample Codes of using PLSQL

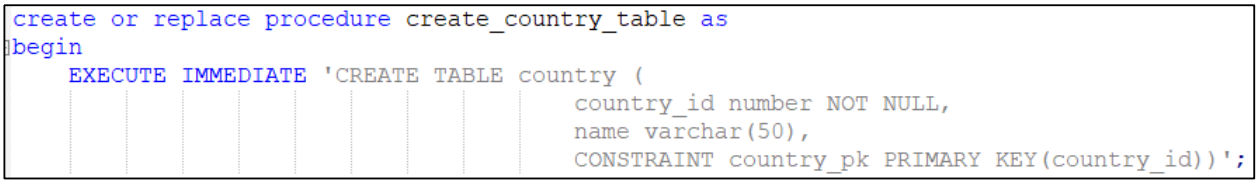
An example of the procedure used,

2.1 TO CREATE USERS



```
create or replace procedure create_user as
    userexist integer;
begin
for c in 0..1 loop
    case c
    when 0 then
        select count(*) into userexist from dba_users where username='OP_GUY';
        if (userexist=0) then
            execute immediate 'create user op_guy identified by op_guy';
            execute immediate 'grant connect to op_guy';
            execute immediate 'grant resource to op_guy';
            execute immediate 'grant create view to op_guy';
            --execute immediate 'grant execute on procedure to operational_guy';
            dbms_output.put_line('user created');
        else
            dbms_output.put_line('user exists');
        end if;
    when 1 then
        select count(*) into userexist from dba_users where username='DE_GUY';
        if (userexist=0) then
            execute immediate 'create user de_guy identified by de_guy';
            execute immediate 'grant connect to de_guy';
            execute immediate 'grant resource to de_guy';
            execute immediate 'grant create view to de_guy';
            --execute immediate 'grant execute on procedure to de_guy';
            dbms_output.put_line('user created');
        else
            dbms_output.put_line('user exists');
        end if;
    end case;
end loop;
end;
/
```

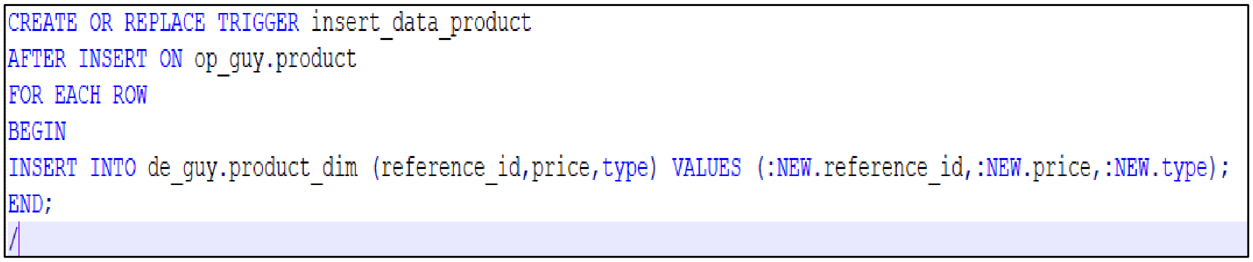
2.2 TO CREATE TABLES



```
create or replace procedure create_country_table as
begin
    EXECUTE IMMEDIATE 'CREATE TABLE country (
                                    country_id number NOT NULL,
                                    name varchar(50),
                                    CONSTRAINT country_pk PRIMARY KEY(country_id))';
```

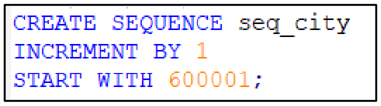
An example of Trigger used to update OLAP dimensions on inserting data into OLTP tables

2.3 TO UPDATE THE OLAP DIMENSIONS



```
CREATE OR REPLACE TRIGGER insert_data_product
AFTER INSERT ON op_guy.product
FOR EACH ROW
BEGIN
INSERT INTO de_guy.product_dim (reference_id,price,type) VALUES (:NEW.reference_id,:NEW.price,:NEW.type);
END;
/
```

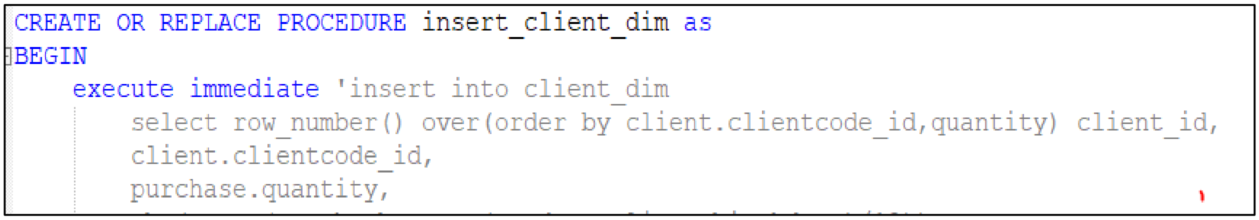
2.4 USE OF SEQUENCE



```
CREATE SEQUENCE seq_city
INCREMENT BY 1
START WITH 600001;
```

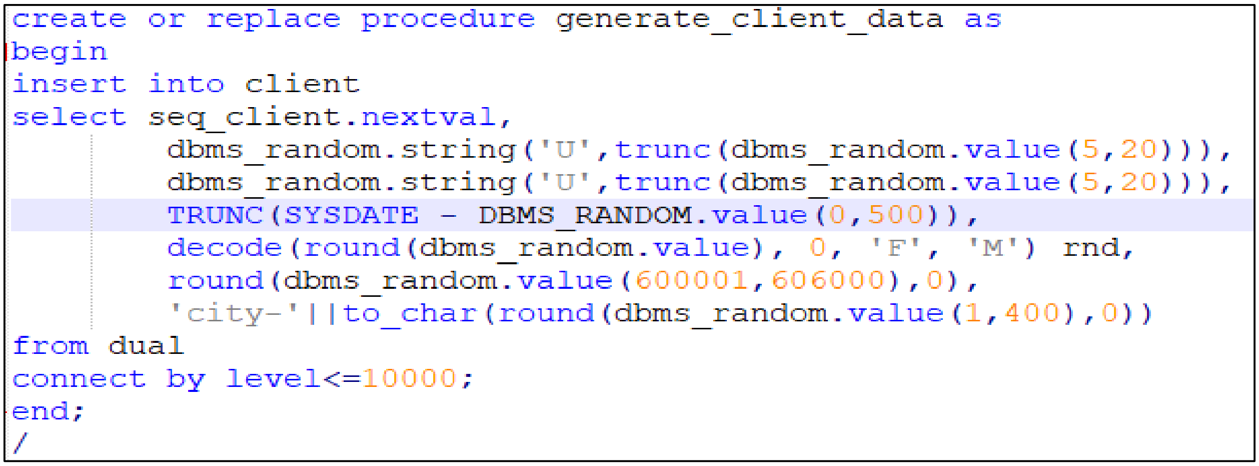
2.5 USE OF SURROGATE KEY IN OLAP

The surrogate key here is realized by using the **Analytic function** Row_number( )



```
CREATE OR REPLACE PROCEDURE insert_client_dim as
BEGIN
    execute immediate 'insert into client_dim
        select row_number() over(order by client.clientcode_id,quantity) client_id,
        client.clientcode_id,
        purchase.quantity,
```

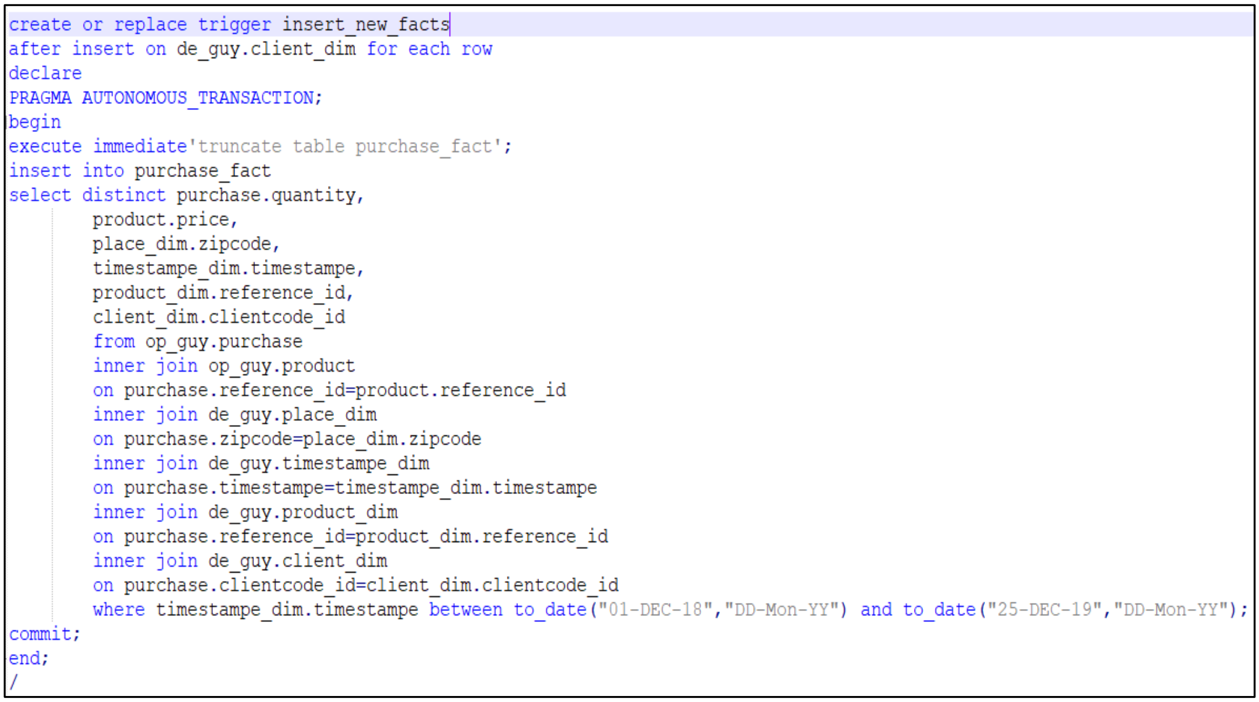
2.6 USE OF RANDOM PACKAGE TO POPULATE DATABASE



```
create or replace procedure generate_client_data as
begin
insert into client
select seq_client.nextval,
        dbms_random.string('U',trunc(dbms_random.value(5,20))),
        dbms_random.string('U',trunc(dbms_random.value(5,20))),
        TRUNC(SYSDATE - DBMS_RANDOM.value(0,500)),
        decode(round(dbms_random.value), 0, 'F', 'M') rnd,
        round(dbms_random.value(600001,606000),0),
        'city-'||to_char(round(dbms_random.value(1,400),0))
from dual
connect by level<=10000;
end;
/
```

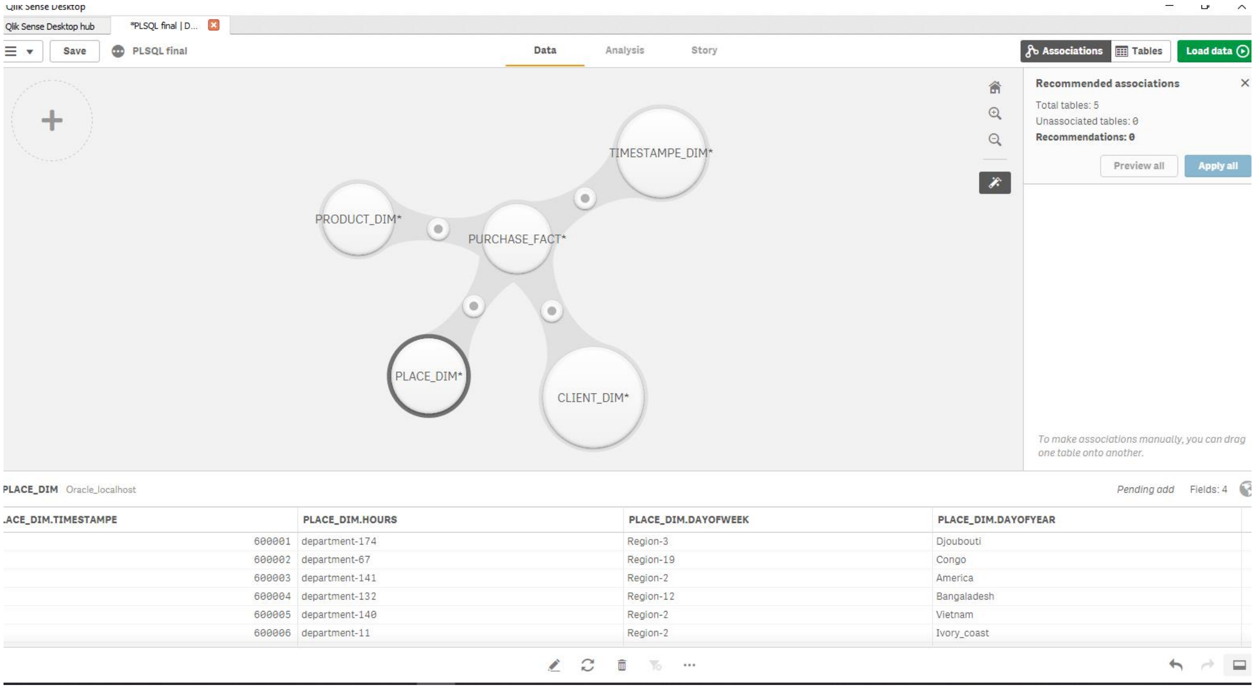
2.7 INIATIALLY THE FACT TABLE IS LOADED WITH ALL DATA And TO OVERWRITE THE FACT TABLE WITH NEW QUERIES

We used Combination of Truncate and insert with Triggers



```
create or replace trigger insert_new_facts
after insert on de_guy.client_dim for each row
declare
PRAGMA AUTONOMOUS_TRANSACTION;
begin
execute immediate'truncate table purchase_fact';
insert into purchase_fact
select distinct purchase.quantity,
        product.price,
        place_dim.zipcode,
        timestampe_dim.timestampe,
        product_dim.reference_id,
        client_dim.clientcode_id
        from op_guy.purchase
        inner join op_guy.product
        on purchase.reference_id=product.reference_id
        inner join de_guy.place_dim
        on purchase.zipcode=place_dim.zipcode
        inner join de_guy.timestampe_dim
        on purchase.timestampe=timestampe_dim.timestampe
        inner join de_guy.product_dim
        on purchase.reference_id=product_dim.reference_id
        inner join de_guy.client_dim
        on purchase.clientcode_id=client_dim.clientcode_id
        where timestampe_dim.timestampe between to_date("01-DEC-18","DD-Mon-YY") and to_date("25-DEC-19","DD-Mon-YY");
commit;
end;
/
```

3 Visualizing OLAP star schema in Qlik Sense



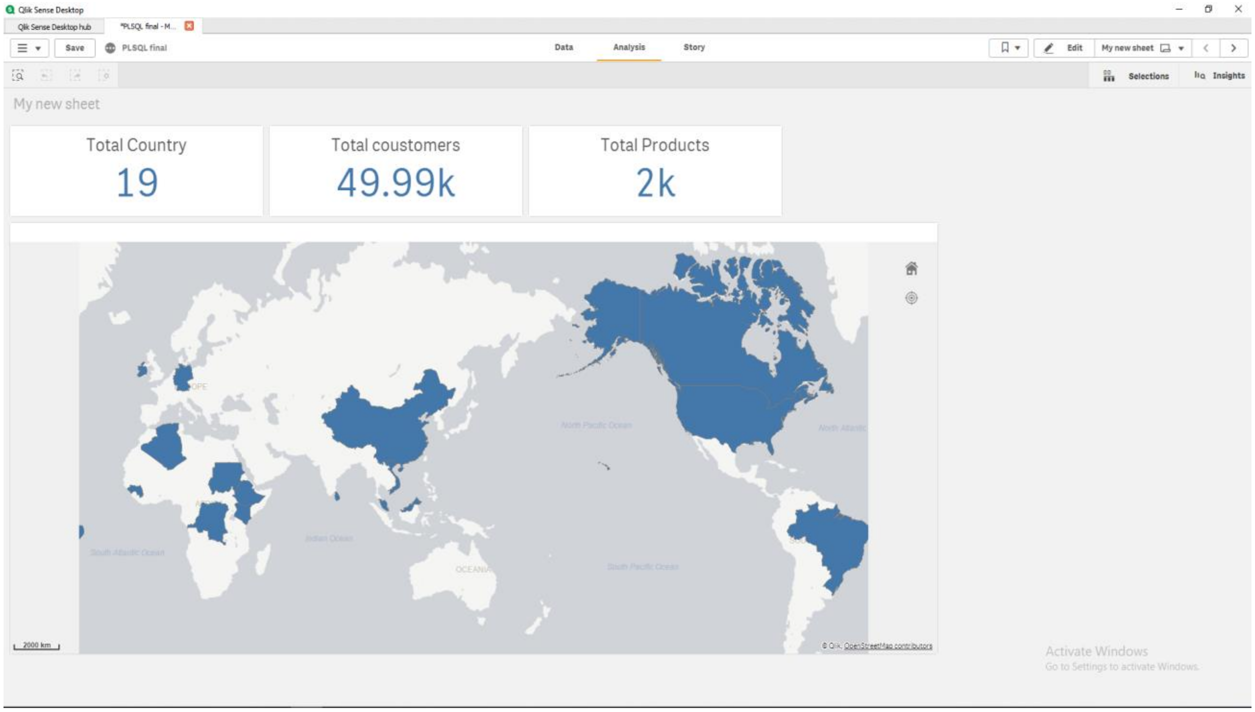
Recommended associations

Total tables: 5
Unassociated tables: 0
Recommendations: 0

To make associations manually, you can drag one table onto another.

PLACE_DIM Oracle_localhost — *Pending add* Fields: 4

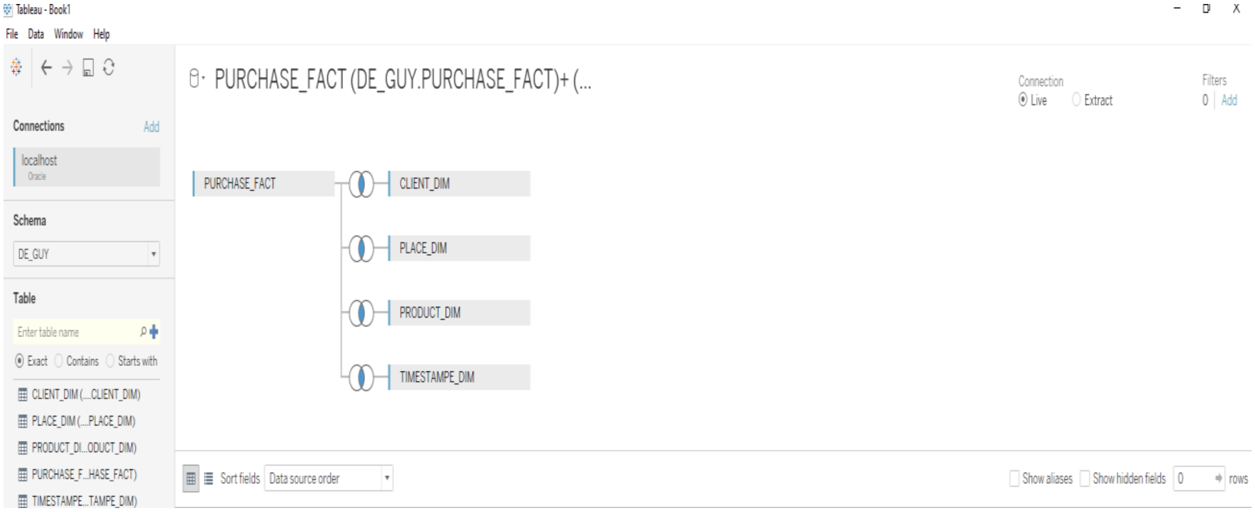
.ACE_DIM.TIMESTAMPE	PLACE_DIM.HOURS	PLACE_DIM.DAYOFWEEK	PLACE_DIM.DAYOFYEAR
600001	department-174	Region-3	Djoubouti
600002	department-67	Region-19	Congo
600003	department-141	Region-2	America
600004	department-132	Region-12	Bangaladesh
600005	department-140	Region-2	Vietnam
600006	department-11	Region-2	Ivory_coast



My new sheet

Total Country	Total coustomers	Total Products
19	49.99k	2k

4 Visualizing OLAP star schema in Tableau



PURCHASE_FACT (DE_GUY.PURCHASE_FACT)+ (...

Connection: ● Live ○ Extract

Filters: 0 Add

Connections: Add

localhost
Oracle

Schema: DE_GUY

Table

Enter table name

● Exact ○ Contains ○ Starts with

- CLIENT_DIM (...CLIENT_DIM)
- PLACE_DIM (...PLACE_DIM)
- PRODUCT_DI...ODUCT_DIM)
- PURCHASE_F...HASE_FACT)
- TIMESTAMPE...TAMPE_DIM)

PURCHASE_FACT — CLIENT_DIM

PURCHASE_FACT — PLACE_DIM

PURCHASE_FACT — PRODUCT_DIM

PURCHASE_FACT — TIMESTAMPE_DIM

Sort fields: Data source order

□ Show aliases □ Show hidden fields 0 rows