|  |
| --- |
|  |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REVISION HISTORY | | | | | |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.0 | Initial status | [Kiryl Bucha](mailto:Kiryl_Bucha@epam.com) | 12-JAN-2012 |  |  |
| 2.0 | Updated in accordance with renewed content | [Elias Nema](mailto:Elias_Nema@epam.com) | 20-JAN-2014 |  |  |
| 3.0 | Report | Aksana Kuratnik | 06-DEC-2017 |  |  |

Contents

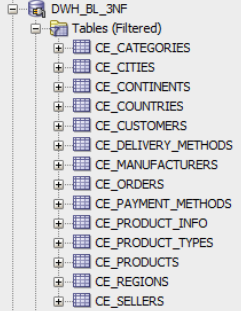
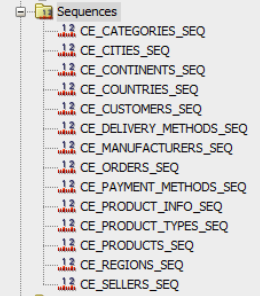
[1. Create Packages for Reload Dimension from SA\_\* 3](#_Toc383391238)

[2. Task Results 3](#_Toc383391239)

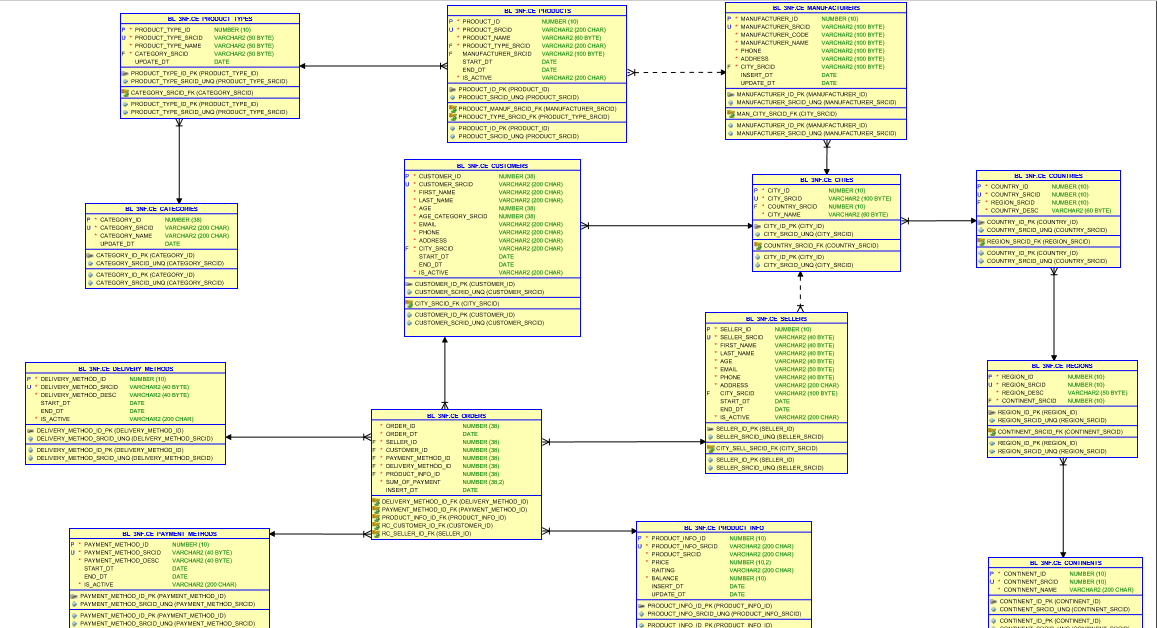
# Create packages to load dimensions from source to staging (3NF) layer

* Create all required dim objects on ST Layer (T\_...).

On the 3NF layer necessary tables and sequences were created.

3NF Scheme:



* Grant all required privileges to ST\_CL (Cleansing Layer).

All privileges on tables and sequences from 3NF-layer were granted to BL\_CL-layer.

GRANT INSERT ON BL\_3NF.CE\_CONTINENTS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_CONTINENTS TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_CONTINENTS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_CONTINENTS\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_REGIONS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_REGIONS TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_REGIONS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_REGIONS\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_COUNTRIES TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_COUNTRIES TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_COUNTRIES TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_COUNTRIES\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_CITIES TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_CITIES TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_CITIES TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_CITIES\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_PRODUCT\_TYPES TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_PRODUCT\_TYPES TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_PRODUCT\_TYPES TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_PRODUCT\_TYPES\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_PRODUCT\_INFO TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_PRODUCT\_INFO TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_PRODUCT\_INFO TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_PRODUCT\_INFO\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_CUSTOMERS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_CUSTOMERS TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_CUSTOMERS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_CUSTOMERS\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_MANUFACTURERS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_MANUFACTURERS TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_MANUFACTURERS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_MANUFACTURERS\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_SELLERS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_SELLERS TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_SELLERS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_SELLERS\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_PAYMENT\_METHODS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_PAYMENT\_METHODS TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_PAYMENT\_METHODS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_PAYMENT\_METHODS\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_DELIVERY\_METHODS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_DELIVERY\_METHODS TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_DELIVERY\_METHODS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_DELIVERY\_METHODS\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_ORDERS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_ORDERS TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_ORDERS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_ORDERS\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_PRODUCTS TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_PRODUCTS TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_PRODUCTSTO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_PRODUCTS\_SEQ TO BL\_CL\_1ST;

GRANT INSERT ON BL\_3NF.CE\_CATEGORIES TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_CATEGORIES TO BL\_CL\_1ST;

GRANT UPDATE ON BL\_3NF.CE\_CATEGORIES TO BL\_CL\_1ST;

GRANT SELECT ON BL\_3NF.CE\_CATEGORIES\_SEQ TO BL\_CL\_1ST;

* Create packages to load dim data (one package = one dimension) on ST\_CL.

**CE\_CONTINENTS:**

--PKG\_INSERT\_CONTINENTS.

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_continents

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_continents;

END pkg\_etl\_insert\_continents;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_continents

AS

PROCEDURE merge\_ce\_continents

IS

BEGIN

MERGE INTO bl\_3nf.ce\_continents t USING

( SELECT continent\_id,

continent\_name

FROM cls\_continents

MINUS

SELECT continent\_srcid AS continent\_id,

continent\_name

FROM bl\_3nf.ce\_continents

) c ON ( c.continent\_id = t.continent\_srcid

AND c.continent\_name = t.continent\_name)

WHEN NOT matched THEN

INSERT

(

continent\_id,

continent\_srcid,

continent\_name

)

VALUES

(

bl\_3nf.ce\_continents\_seq.NEXTVAL,

c.continent\_id,

c.continent\_name

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

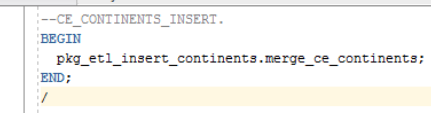
RAISE;

END merge\_ce\_continents;

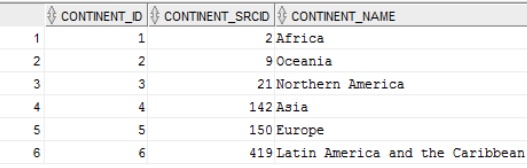
END pkg\_etl\_insert\_continents;

/

Implementation of the pkg\_etl\_insert\_continents:



The result of insert:



**CE\_REGIONS:**

--PKG\_INSERT\_REGIONS.

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_regions

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_regions;

END pkg\_etl\_insert\_regions;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_regions

AS

PROCEDURE merge\_ce\_regions

IS

BEGIN

MERGE INTO bl\_3nf.ce\_regions t USING

( SELECT continent\_id,

region\_id,

region\_desc

FROM cls\_regions

MINUS

SELECT continent\_srcid,

region\_srcid AS region\_id,

region\_desc

FROM bl\_3nf.ce\_regions

) c ON ( c.region\_id = t.region\_srcid

AND c.continent\_id = t.continent\_srcid

AND c.region\_desc = t.region\_desc)

WHEN NOT matched THEN

INSERT

(

region\_ID ,

region\_srcid ,

continent\_srcid,

region\_desc

)

VALUES

(

bl\_3nf.ce\_regions\_seq.NEXTVAL,

c.region\_id,

c.continent\_id,

c.region\_desc

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

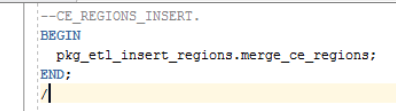
RAISE;

END merge\_ce\_regions;

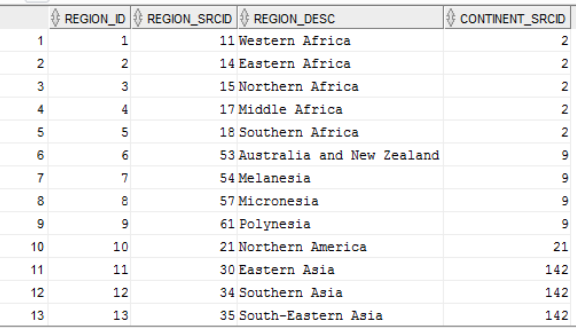
END pkg\_etl\_insert\_regions;

/

Implementation of the pkg\_etl\_insert\_regions:



The result of insert:



**CE\_COUNTRIES:**

--PKG\_INSERT\_COUNTRIES.

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_countries

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_countries;

END pkg\_etl\_insert\_countries;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_countries

AS

PROCEDURE merge\_ce\_countries

IS

BEGIN

MERGE INTO bl\_3nf.ce\_countries t USING

( SELECT region\_id,

country\_id,

country\_desc

FROM cls\_countries

MINUS

SELECT region\_srcid,

country\_srcid AS country\_id,

country\_desc

FROM bl\_3nf.ce\_countries

) c ON ( c.country\_id = t.country\_srcid

AND c.region\_id = t.region\_srcid

AND c.country\_desc = t.country\_desc)

WHEN NOT matched THEN

INSERT

(

country\_ID ,

country\_SRCID ,

region\_srcid,

country\_desc

)

VALUES

(

bl\_3nf.ce\_countries\_seq.NEXTVAL,

c.country\_id,

c.region\_id,

c.country\_desc

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

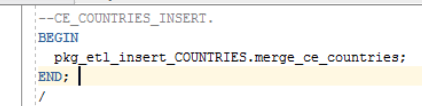
RAISE;

END merge\_ce\_countries;

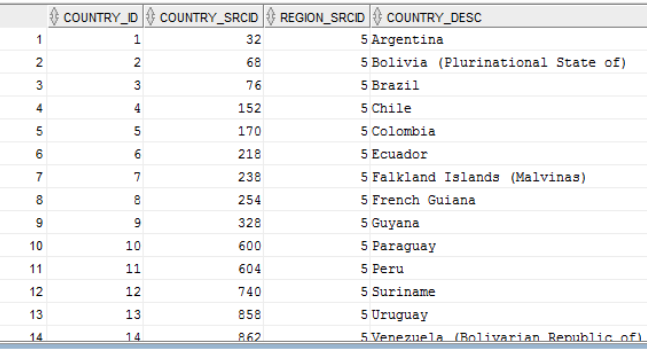
END pkg\_etl\_insert\_countries;

/

Implementation of the pkg\_etl\_insert\_countries:



The result of the insert:



**CE\_CITIES:**

--PKG\_INSERT\_CITIES.

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_cities

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_cities;

END pkg\_etl\_insert\_cities;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_cities

AS

PROCEDURE merge\_ce\_cities

IS

BEGIN

MERGE INTO bl\_3nf.ce\_cities t USING

( SELECT country\_id,

city\_id,

city\_name

FROM cls\_cities

MINUS

SELECT country\_srcid,

city\_srcid AS city\_id,

city\_name

FROM bl\_3nf.ce\_cities

) c ON ( c.city\_id = t.city\_srcid)

WHEN NOT matched THEN

INSERT

(

city\_id,

city\_srcid,

country\_srcid,

city\_name

)

VALUES

(

bl\_3nf.ce\_cities\_seq.NEXTVAL,

c.city\_id,

c.country\_id,

c.city\_name

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

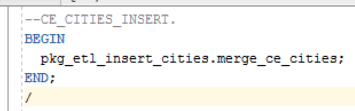
RAISE;

END merge\_ce\_cities;

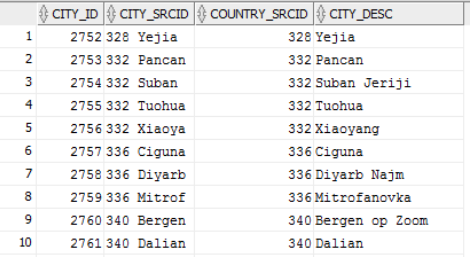
END pkg\_etl\_insert\_cities;

/

Implementation of the pkg\_etl\_insert\_cities:



The result of insert:



**CE\_SELLERS:**

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_sellers

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_sellers;

END pkg\_etl\_insert\_sellers;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_sellers

AS

PROCEDURE merge\_ce\_sellers

IS

BEGIN

MERGE INTO bl\_3nf.ce\_sellers t USING

( SELECT seller\_id,

first\_name,

last\_name,

age,

email,

phone,

address,

city\_id,

start\_dt,

end\_dt,

is\_active

FROM cls\_sellers

MINUS

SELECT

seller\_srcid AS seller\_id,

first\_name,

last\_name,

age,

email,

phone,

address,

city\_srcid AS city\_id,

start\_dt,

end\_dt,

is\_active

FROM bl\_3nf.ce\_sellers

) c ON ( c.seller\_id = t.seller\_srcid

AND c.first\_name = t.first\_name

AND c.last\_name = t.last\_name

AND t.age = c.age

AND c.email = t.email

AND c.phone = t.phone

)

WHEN MATCHED THEN

UPDATE SET

t.end\_dt = c.end\_dt,

t.is\_active = c.is\_active

WHEN NOT matched THEN

INSERT

(

seller\_id,

seller\_srcid,

first\_name,

last\_name,

age,

email,

phone,

address,

city\_srcid,

start\_dt,

end\_dt,

is\_active

)

VALUES

(

bl\_3nf.ce\_sellers\_seq.NEXTVAL,

c.seller\_id,

c.first\_name,

c.last\_name,

c.age,

c.email,

c.phone,

c.address,

c.city\_id,

c.start\_dt,

c.end\_dt,

c.is\_active

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

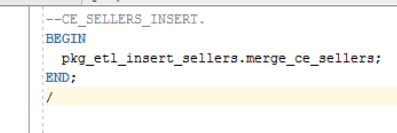
RAISE;

END merge\_ce\_sellers;

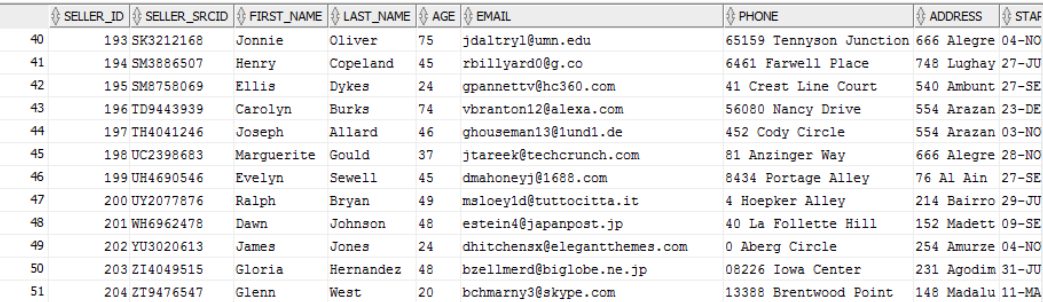
END pkg\_etl\_insert\_sellers;

/

Implementation of the pkg\_etl\_insert\_sellers:



The result of insert:



**CE\_CUSTOMERS:**

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_customers

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_customers;

END pkg\_etl\_insert\_customers;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_customers

AS

PROCEDURE merge\_ce\_customers

IS

BEGIN

MERGE INTO bl\_3nf.ce\_customers t USING

( SELECT customer\_id,

first\_name,

last\_name,

age,

email,

phone,

address,

city\_id,

start\_dt,

end\_dt,

is\_active

FROM cls\_customers

MINUS

SELECT customer\_srcid AS customer\_id,

first\_name,

last\_name,

age,

email,

phone,

address,

city\_srcid AS city\_id,

start\_dt,

end\_dt,

is\_active

FROM bl\_3nf.ce\_customers

) c ON ( c.customer\_id = t.customer\_srcid

AND c.first\_name = t.first\_name

AND c.last\_name = t.last\_name

AND c.age = t.age

AND c.email = t.email

AND c.phone = t.phone

AND c.address = t.address

AND t.city\_srcid = c.city\_id

AND c.start\_dt = t.start\_dt)

WHEN MATCHED THEN

UPDATE SET

t.end\_dt = c.end\_dt,

t.is\_active = c.is\_active

WHEN NOT matched THEN

INSERT

(

customer\_id,

customer\_srcid,

first\_name,

last\_name,

age,

email,

phone,

address,

city\_srcid,

start\_dt,

end\_dt,

is\_active

)

VALUES

(

bl\_3nf.ce\_customers\_seq.NEXTVAL,

c.customer\_id,

c.first\_name,

c.last\_name,

c.age,

c.email,

c.phone,

c.address,

c.city\_id,

c.start\_dt,

c.end\_dt,

'TRUE'

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

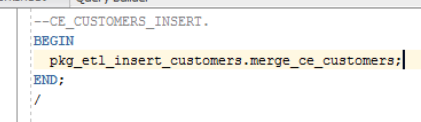
RAISE;

END merge\_ce\_customers;

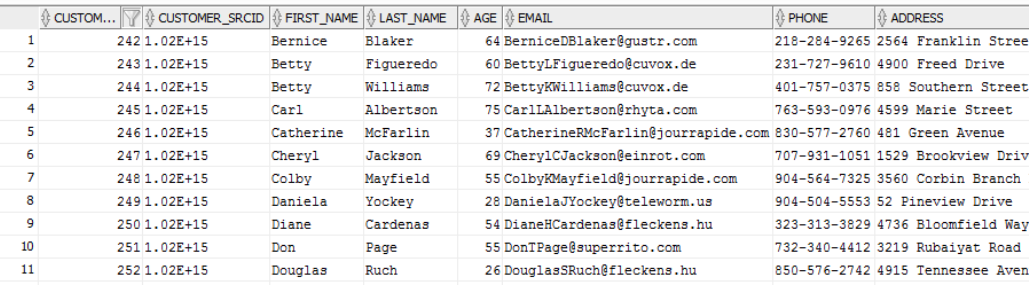
END pkg\_etl\_insert\_customers;

/

Implementation of the pkg\_etl\_insert\_customers:



The result of insert:



**CE\_CATEGORIES:**

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_categories

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_categories;

END pkg\_etl\_insert\_categories;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_categories

AS

PROCEDURE merge\_ce\_categories

IS

BEGIN

MERGE INTO bl\_3nf.ce\_categories t USING

( SELECT category\_id,

category\_name,

start\_dt AS update\_dt

FROM cls\_categories

MINUS

SELECT category\_srcid AS category\_id,

category\_name,

update\_dt

FROM bl\_3nf.ce\_categories

) c ON (c.category\_name = t.category\_name

AND c.category\_id = t.category\_srcid

)

WHEN MATCHED THEN

UPDATE SET

t.update\_dt = SYSDATE

WHEN NOT matched THEN

INSERT

(

category\_id,

category\_srcid,

category\_name,

update\_dt

)

VALUES

(

bl\_3nf.ce\_categories\_seq.NEXTVAL,

c.category\_id,

c.category\_name,

SYSDATE

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

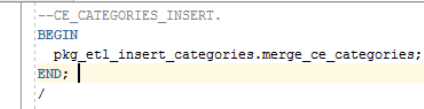
RAISE;

END merge\_ce\_categories;

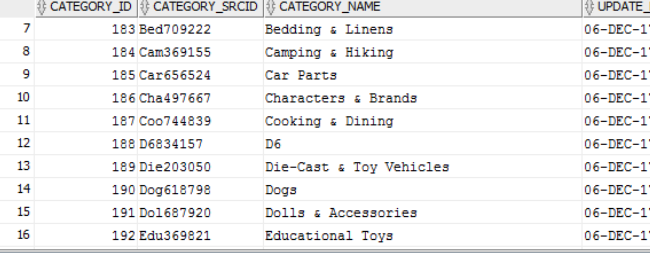
END pkg\_etl\_insert\_categories;

/

Implementation of the pkg\_etl\_insert\_categories:



The result of insert:



**CE\_PRODUCT\_TYPES:**

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_product\_types

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_product\_types;

END pkg\_etl\_insert\_product\_types;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_product\_types

AS

PROCEDURE merge\_ce\_product\_types

IS

BEGIN

MERGE INTO bl\_3nf.ce\_product\_types t USING

( SELECT product\_type\_id,

product\_type\_name,

category\_id,

start\_dt AS update\_dt

FROM cls\_product\_types

MINUS

SELECT product\_type\_srcid AS product\_type\_id,

product\_type\_name AS product\_type\_name,

category\_srcid AS category\_id,

update\_dt

FROM bl\_3nf.ce\_product\_types

) c ON ( c.product\_type\_name = t.product\_type\_name

AND t.product\_type\_srcid = c.product\_type\_id

AND t.category\_srcid = c.category\_id

)

WHEN MATCHED THEN

UPDATE SET

t.update\_dt = SYSDATE

WHEN NOT matched THEN

INSERT

(

product\_type\_id,

product\_type\_srcid,

product\_type\_name,

category\_srcid,

update\_dt

)

VALUES

(

bl\_3nf.ce\_product\_types\_seq.NEXTVAL,

c.product\_type\_id,

c.product\_type\_name,

c.category\_id,

SYSDATE

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

RAISE;

END merge\_ce\_product\_types;

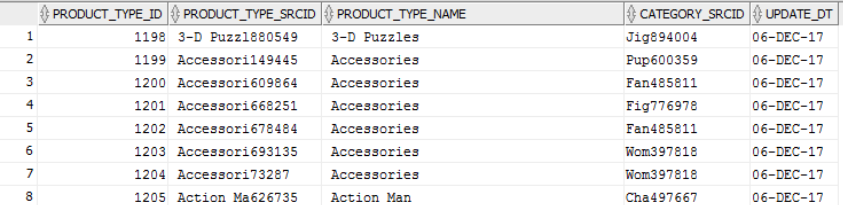
END pkg\_etl\_insert\_product\_types;

/

Implementation of the pkg\_etl\_insert\_product\_types:



The result of insert:



**CE\_MANUFACTURERS:**

--PKG\_INSERT\_MANUFACTURERS

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_manufacturers

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_manufacturers;

END pkg\_etl\_insert\_manufacturers;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_manufacturers

AS

PROCEDURE merge\_ce\_manufacturers

IS

BEGIN

MERGE INTO bl\_3nf.ce\_manufacturers t USING

( SELECT manufacturer\_id,

manufacturer\_code,

manufacturer\_name,

phone,

address,

city AS city\_srcid,

insert\_dt,

update\_dt

FROM cls\_manufacturers

MINUS

SELECT manufacturer\_srcid AS manufacturer\_id,

manufacturer\_code,

manufacturer\_name,

phone,

address,

city\_srcid,

insert\_dt,

update\_dt

FROM bl\_3nf.ce\_manufacturers

) c ON ( c.manufacturer\_id = t.manufacturer\_srcid

AND t.manufacturer\_name = c.manufacturer\_name

AND t.manufacturer\_code = c.manufacturer\_code

AND t.phone = c.phone

AND t.address = c.address

AND t.city\_srcid = c.city\_srcid

AND c.insert\_dt = t.insert\_dt )

WHEN matched THEN

UPDATE SET

t.update\_dt = c.update\_dt

WHEN NOT matched THEN

INSERT

(

manufacturer\_id,

manufacturer\_srcid,

manufacturer\_code,

manufacturer\_name,

phone,

address,

city\_srcid,

insert\_dt,

update\_dt

)

VALUES

(

bl\_3nf.ce\_manufacturers\_seq.NEXTVAL,

c.manufacturer\_id,

c.manufacturer\_code,

c.manufacturer\_name,

c.phone,

c.address,

c.city\_srcid,

c.insert\_dt,

SYSDATE

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

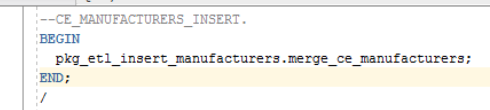
RAISE;

END merge\_ce\_manufacturers;

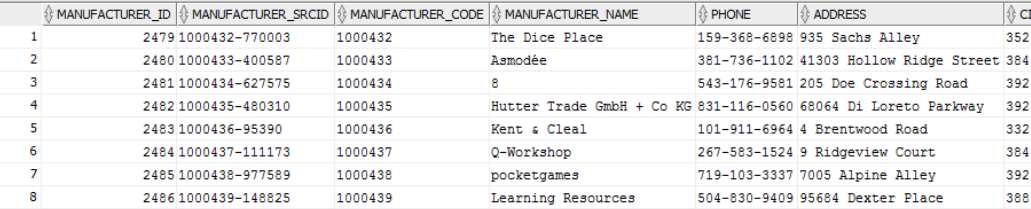
END pkg\_etl\_insert\_manufacturers;

/

Implementation of the pkg\_etl\_insert\_manufacturers:



The result of insert:



**CE\_PRODUCTS:**

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_products

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_products;

END pkg\_etl\_insert\_products;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_products

AS

PROCEDURE merge\_ce\_products

IS

BEGIN

MERGE INTO bl\_3nf.ce\_products t USING

( SELECT product\_id,

product\_name,

product\_type\_id,

start\_dt,

end\_dt,

is\_active

FROM cls\_products

MINUS

SELECT product\_srcid AS product\_id,

product\_name,

product\_type\_srcid AS product\_type\_id,

start\_dt,

end\_dt,

is\_active

FROM bl\_3nf.ce\_products

) c ON ( t.product\_srcid = c.product\_id

AND c.product\_name = t.product\_name

AND t.product\_type\_srcid = c.product\_type\_id

AND c.start\_dt = t.start\_dt

)

WHEN MATCHED THEN

UPDATE SET

t.end\_dt = c.end\_dt,

t.is\_active = c.is\_active

WHEN NOT matched THEN

INSERT

(

product\_id,

product\_srcid,

product\_name,

product\_type\_srcid,

start\_dt,

end\_dt,

is\_active

)

VALUES

(

bl\_3nf.ce\_products\_seq.NEXTVAL,

c.product\_id,

c.product\_name,

c.product\_type\_id,

c.start\_dt,

c.end\_dt,

c.is\_active

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

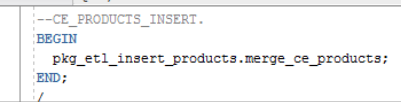
RAISE;

END merge\_ce\_products;

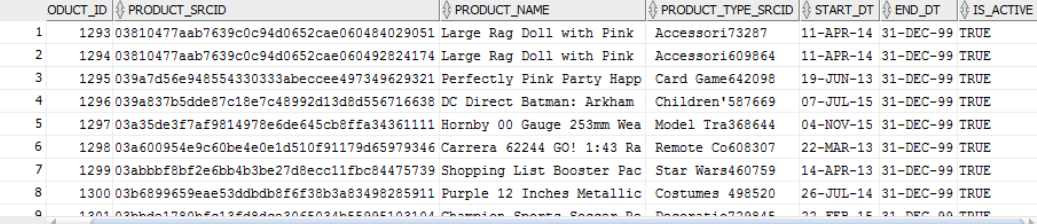
END pkg\_etl\_insert\_products;

/

Implementation of the pkg\_etl\_insert\_manufacturers:



The result of insert:



**CE\_PRODUCT\_INFO:**

* **Use Explicit Cursor (One or more procedures).**

Loading data from BL\_CL Layer by means of Explicit Cursor was made on the table that contains information about payment methods.

PROCEDURE insert\_table\_payment\_methods

IS

CURSOR my\_cursor

IS

SELECT payment\_method\_id,

payment\_method\_name AS payment\_method,

start\_dt,

end\_dt,

is\_active

FROM wrk\_payment\_methods

WHERE payment\_method\_name IS NOT NULL

AND start\_dt IS NOT NULL;

pm\_my\_cursor cls\_payment\_methods%rowtype;

BEGIN

EXECUTE IMMEDIATE 'TRUNCATE TABLE cls\_payment\_methods';

OPEN my\_cursor;

LOOP

FETCH my\_cursor INTO pm\_my\_cursor;

IF my\_cursor%found THEN

INSERT

INTO cls\_payment\_methods

(

payment\_method\_id,

payment\_method,

start\_dt,

end\_dt,

is\_active

)

VALUES

(

pm\_emp\_cursor.payment\_method\_id,

pm\_emp\_cursor.payment\_method,

pm\_emp\_cursor.start\_dt,

pm\_emp\_cursor.end\_dt,

pm\_emp\_cursor.is\_active

) ;

END IF;

EXIT

WHEN my\_cursor%notfound;

END LOOP;

COMMIT;

CLOSE my\_cursor;

COMMIT;

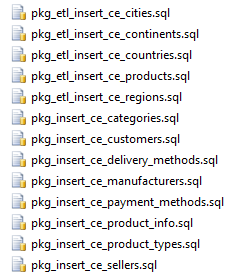
EXCEPTION

WHEN OTHERS THEN

RAISE;

END insert\_cls\_payment\_methods;

Packages:



* **Use Implicit Cursor (One or more procedures).**

Loading data from BL\_CL Layer by means of Explicit Cursor was made on the table that contains information about payment methods. The examples of Implicit Cursors are INSERT, UPDATE or SELECT.

PROCEDURE insert\_table\_payment\_methods

IS

BEGIN

EXECUTE IMMEDIATE ('TRUNCATE TABLE cls\_payment\_methods');

INSERT INTO cls\_payment\_methods (

payment\_method\_id,

payment\_method,

start\_dt,

end\_dt,

is\_active

)

SELECT SUBSTR(payment\_method,1,5) || '-' || TRUNC(dbms\_random.value(1,1000000)) AS payment\_method\_id,

payment\_method\_name AS payment\_method,

start\_dt,

end\_dt,

is\_active

FROM wrk\_payment\_methods;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

RAISE;

END insert\_table\_payment\_methods;

* **Use Merge (One or more procedures).**

Loading data about regions from BL\_CL Layer to BL\_3NF Layer by means of MERGE:

--PKG\_INSERT\_REGIONS.

CREATE OR REPLACE PACKAGE pkg\_etl\_insert\_regions

AUTHID CURRENT\_USER

AS

PROCEDURE merge\_ce\_regions;

END pkg\_etl\_insert\_regions;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_insert\_regions

AS

PROCEDURE merge\_ce\_regions

IS

BEGIN

MERGE INTO bl\_3nf.ce\_regions t USING

( SELECT continent\_id,

region\_id,

region\_desc

FROM cls\_regions

MINUS

SELECT continent\_srcid,

region\_srcid AS region\_id,

region\_desc

FROM bl\_3nf.ce\_regions

) c ON ( c.region\_id = t.region\_srcid

AND c.continent\_id = t.continent\_srcid

AND c.region\_desc = t.region\_desc)

WHEN NOT matched THEN

INSERT

(

region\_ID ,

region\_srcid ,

continent\_srcid,

region\_desc

)

VALUES

(

bl\_3nf.ce\_regions\_seq.NEXTVAL,

c.region\_id,

c.continent\_id,

c.region\_desc

) ;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

RAISE;

END merge\_ce\_regions;

END pkg\_etl\_insert\_regions;