|  |
| --- |
|  |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REVISION HISTORY | | | | | |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.0 | Initial status | Hanna Klimovich | 23-NOV-2017 |  |  |

Contents

[1. CREATE Storage Objects 3](#_Toc499130086)

[2. Generate Test Data in Storage Layers 3](#_Toc499130087)

[2.1. Analyse 3](#_Toc499130088)

[2.2. Create users and grants 3](#_Toc499130089)

[2.3. External tables 4](#_Toc499130090)

[2.4. Cleansing tables 5](#_Toc499130091)

[2.5. Loading into cleansing tables 6](#_Toc499130092)

[2.6. 3NF 7](#_Toc499130093)

[2.7. 3NF Schema 9](#_Toc499130094)

[3. SQL\*Plus 9](#_Toc499130095)

[3.1. Show execution plans in SQL\*Plans 10](#_Toc499130096)

[3.2. Set timing on 10](#_Toc499130097)

[3.3. Run script 11](#_Toc499130098)

[3.4. Save data to file 11](#_Toc499130099)

# CREATE Storage Objects

# Generate Test Data in Storage Layers

## Analyse

File iso\_3166.tab has an information: country\_id, country name, country code.

File iso\_3166\_geo\_un\_contries.tab has an information: country\_id, country\_name, region\_id, region\_name.

File iso\_3166\_geo\_un.tab has an information: earth, continent, regions

Finally, our schema will looks like World<-Continents<-Regions<-Countries

## Create users and grants

CREATE USER sa\_src

IDENTIFIED BY "123"

DEFAULT TABLESPACE tbs\_pdb\_test;

GRANT CONNECT TO sa\_src;

GRANT RESOURCE TO sa\_src;

--GRANT all privileges to sa\_src;

CREATE USER bl\_cl

IDENTIFIED BY "123"

DEFAULT TABLESPACE tbs\_pdb\_test;

GRANT CONNECT TO bl\_cl;

GRANT RESOURCE TO bl\_cl;

--GRANT all privileges to bl\_cl;

CREATE USER bl\_3nf

IDENTIFIED BY "123"

DEFAULT TABLESPACE tbs\_pdb\_test;

GRANT CONNECT TO bl\_3nf;

GRANT RESOURCE TO bl\_3nf;

-- GRANT all privileges to bl\_3nf;

CREATE DIRECTORY ext\_tables

as '/media/sf\_my';

Package for grants:

CREATE OR REPLACE PACKAGE grants\_mgmt

AUTHID CURRENT\_USER

AS

PROCEDURE GRANT\_BLAT (GRANT\_NAME IN VARCHAR2,

SCHEMA\_NAME IN VARCHAR2,

OBJECT\_NAME IN VARCHAR2,

USER\_NAME IN VARCHAR2);

PROCEDURE GRANT\_BLAT (GRANT\_NAME IN VARCHAR2,

USER\_NAME IN VARCHAR2,

WAO IN BOOLEAN := FALSE);

END grants\_mgmt;

/

CREATE OR REPLACE PACKAGE BODY grants\_mgmt AS

PROCEDURE GRANT\_BLAT (GRANT\_NAME IN VARCHAR2,

SCHEMA\_NAME IN VARCHAR2,

OBJECT\_NAME IN VARCHAR2,

USER\_NAME IN VARCHAR2 )

IS

BEGIN

EXECUTE IMMEDIATE ('GRANT ' || GRANT\_NAME || ' ON ' || SCHEMA\_NAME || '.' || OBJECT\_NAME || ' TO ' || USER\_NAME);

END GRANT\_BLAT;

PROCEDURE GRANT\_BLAT (GRANT\_NAME VARCHAR2, USER\_NAME VARCHAR2, WAO IN BOOLEAN := FALSE) IS

BEGIN

IF WAO THEN

EXECUTE IMMEDIATE ('GRANT ' || GRANT\_NAME || ' TO ' || USER\_NAME || ' WITH ADMIN OPTION');

END IF;

EXECUTE IMMEDIATE ('GRANT ' || GRANT\_NAME || ' TO ' || USER\_NAME);

END;

END grants\_mgmt;

/

EXEC grants\_mgmt.grant\_blat('CREATE ANY TABLE', 'SA\_SRC');

EXEC grants\_mgmt.grant\_blat('SELECT ANY TABLE', 'SA\_SRC');

EXEC grants\_mgmt.grant\_blat('READ,WRITE ON DIRECTORY ext\_tables', 'SA\_SRC');

EXEC grants\_mgmt.grant\_blat('CREATE ANY TABLE', 'BL\_CL');

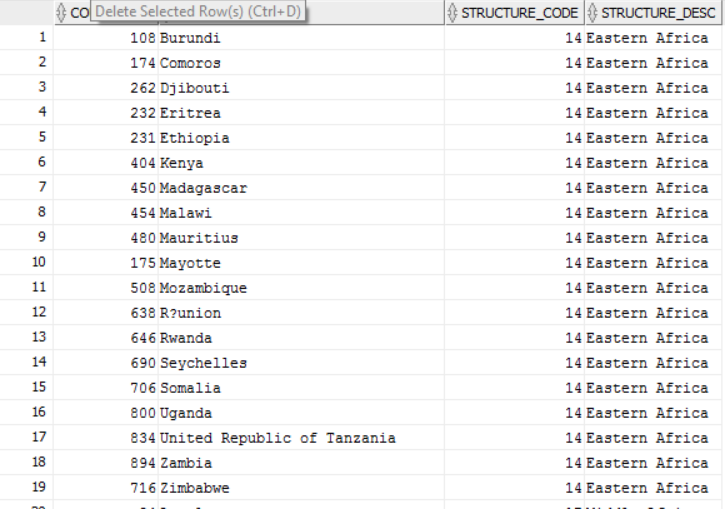
EXEC grants\_mgmt.grant\_blat('SELECT ANY TABLE', 'BL\_CL');

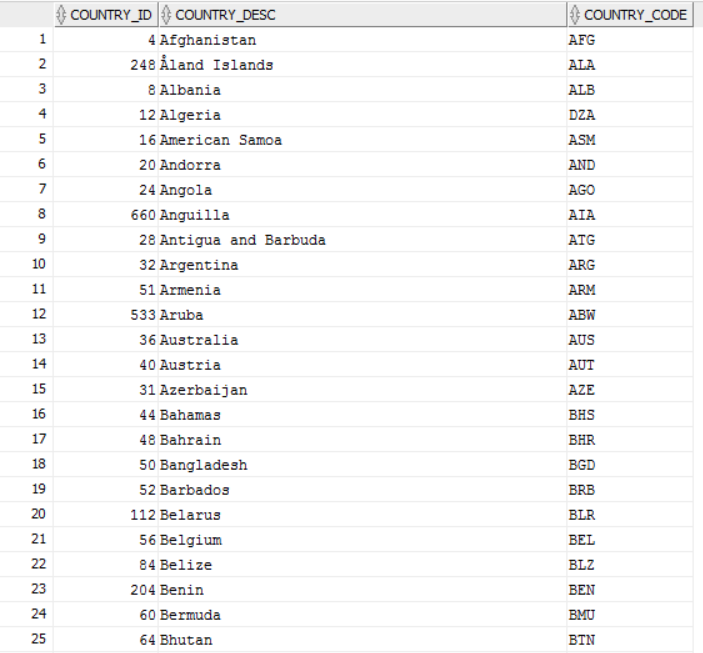
EXEC grants\_mgmt.grant\_blat('READ,WRITE ON DIRECTORY ext\_tables', 'BL\_CL');

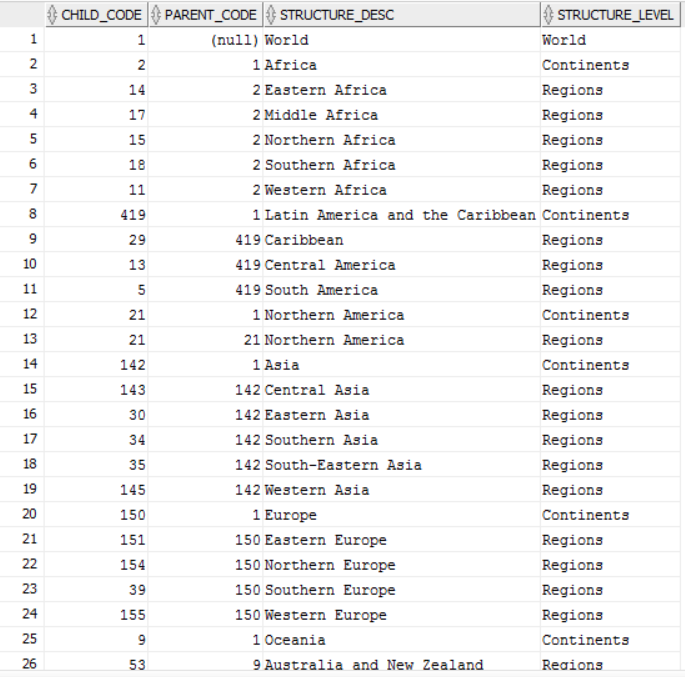
EXEC grants\_mgmt.grant\_blat('CREATE ANY TABLE', 'BL\_3NF');

EXEC grants\_mgmt.grant\_blat('SELECT ANY TABLE', 'BL\_3NF');

## External tables







## Cleansing tables

CREATE TABLE cl\_world

(

world\_id NUMBER(10),

world\_desc VARCHAR2 ( 200)

);

CREATE TABLE cl\_continents

(

continent\_id NUMBER(10),

continent\_desc VARCHAR2 ( 200 ),

global\_id NUMBER(10)

);

CREATE TABLE cl\_regions

(

region\_id NUMBER(10),

region\_desc VARCHAR2 ( 200 ),

continent\_id NUMBER(10)

);

CREATE TABLE cl\_countries(

country\_id NUMBER(10),

country\_desc VARCHAR2 ( 200 ),

country\_code VARCHAR2 ( 200 ),

region\_id NUMBER(10)

);

## Loading into cleansing tables

CREATE OR REPLACE PACKAGE pkg\_etl\_geo

AS

PROCEDURE load\_worlds;

PROCEDURE load\_continents;

PROCEDURE load\_regions;

PROCEDURE load\_countries;

END pkg\_etl\_geo;

/

CREATE OR REPLACE PACKAGE body pkg\_etl\_geo as

PROCEDURE load\_worlds

IS

BEGIN

EXECUTE immediate 'truncate table cl\_world';

INSERT INTO cl\_world

(world\_id,

world\_desc)

SELECT child\_code,

structure\_desc

FROM sa\_src.ext\_geo\_structure\_iso3166

WHERE structure\_level='World';

COMMIT;

EXCEPTION

WHEN OTHERS THEN

RAISE;

END load\_worlds;

PROCEDURE load\_continents

IS

BEGIN

EXECUTE immediate 'truncate table cl\_continents';

INSERT INTO cl\_continents

(continent\_id,

continent\_desc,

global\_id)

SELECT child\_code,

structure\_desc,

parent\_code

FROM sa\_src.ext\_geo\_structure\_iso3166

WHERE structure\_level='Continents';

COMMIT;

EXCEPTION

WHEN OTHERS THEN

RAISE;

END load\_continents;

PROCEDURE load\_regions

IS

BEGIN

EXECUTE immediate 'truncate table cl\_regions';

INSERT INTO cl\_regions

(region\_id,

region\_desc,

continent\_id)

SELECT child\_code,

structure\_desc,

parent\_code

FROM sa\_src.ext\_geo\_structure\_iso3166

WHERE structure\_level='Regions';

COMMIT;

EXCEPTION

WHEN OTHERS THEN

RAISE;

END load\_regions;

PROCEDURE load\_countries

IS

BEGIN

EXECUTE immediate 'truncate table cl\_countries';

INSERT INTO cl\_countries

(country\_id,

country\_desc,

country\_code,

region\_id)

SELECT country.country\_id,

country.country\_desc,

country.country\_code,

region.STRUCTURE\_CODE

FROM SA\_SRC.EXT\_GEO\_COUNTRIES\_ISO3166 country

JOIN SA\_SRC.EXT\_CNTR2STRUCTURE\_ISO3166 region

ON country.COUNTRY\_ID=region.COUNTRY\_ID;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

RAISE;

END load\_countries;

END pkg\_etl\_geo;

/

EXEC PKG\_ETL\_GEO.load\_worlds;

EXEC PKG\_ETL\_GEO.load\_continents;

EXEC PKG\_ETL\_GEO.load\_regions;

EXEC PKG\_ETL\_GEO.load\_countries/

## 3NF

REATE TABLE ce\_worlds

(

world\_id NUMBER(10,0) PRIMARY KEY ,

world\_desc VARCHAR2 ( 200 CHAR ) NOT NULL

);

CREATE TABLE ce\_regions

(

region\_id NUMBER(10,0) PRIMARY KEY,

region\_desc VARCHAR2 ( 200 CHAR ) NOT NULL,

continent\_id NUMBER(10,0) NOT NULL

);

CREATE TABLE ce\_countries

(

country\_id NUMBER(10,0) PRIMARY KEY NOT NULL,

country\_desc VARCHAR2 ( 200 CHAR ) NOT NULL,

country\_code VARCHAR2 ( 3 ),

region\_id NUMBER(10,0)NOT NULL);

CREATE TABLE ce\_continents

(

continent\_id NUMBER(10,0) PRIMARY KEY ,

continent\_desc VARCHAR2 ( 200 CHAR ) NOT NULL,

world\_id NUMBER(10,0) NOT NULL);

INSERT INTO ce\_worlds

(world\_id,

world\_desc)

SELECT world\_id,

world\_desc

FROM bl\_cl.cl\_world;

INSERT INTO ce\_continents

(continent\_id,

continent\_desc,

world\_id)

SELECT continent\_id,

continent\_desc,

world\_id

FROM bl\_cl.cl\_continents;

INSERT INTO ce\_regions

(region\_id,

region\_desc,

continent\_id)

SELECT region\_id,

region\_desc,

continent\_id

FROM bl\_cl.cl\_regions;

INSERT INTO ce\_countries

(country\_id,

country\_desc,

country\_code,

region\_id)

SELECT country\_id,

country\_desc,

country\_code,

region\_id

FROM bl\_cl.cl\_countries;

Alter table ce\_regions ADD CONSTRAINT FK\_continent\_id FOREIGN KEY (continent\_id)

REFERENCES ce\_continents(continent\_id);

Alter table ce\_countries add CONSTRAINT FK\_country\_id FOREIGN KEY (region\_id)

REFERENCES ce\_regions(region\_id);

Alter table ce\_continents add CONSTRAINT FK\_world\_id FOREIGN KEY (world\_id)

REFERENCES ce\_worlds (world\_id) ;

INSERT INTO ce\_worlds

(world\_id,

world\_desc)

SELECT world\_id,

world\_desc

FROM bl\_cl.cl\_world;

INSERT INTO ce\_continents

(continent\_id,

continent\_desc,

world\_id)

SELECT continent\_id,

continent\_desc,

global\_id

FROM bl\_cl.cl\_continents;

INSERT INTO ce\_regions

(region\_id,

region\_desc,

continent\_id)

SELECT region\_id,

region\_desc,

continent\_id

FROM bl\_cl.cl\_regions;

INSERT INTO ce\_countries

(country\_id,

country\_desc,

country\_code,

region\_id)

SELECT country\_id,

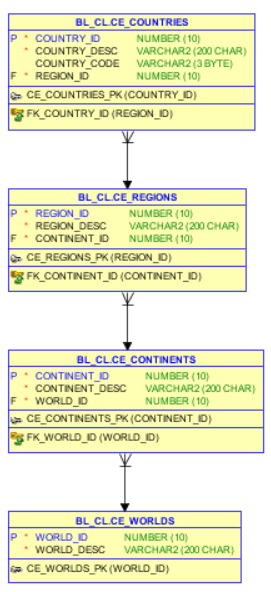
country\_desc,

country\_code,

region\_id

FROM bl\_cl.cl\_countries;

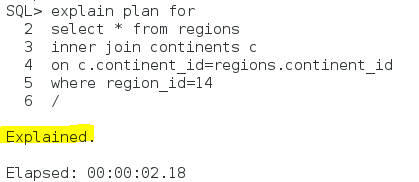
## 3NF Schema

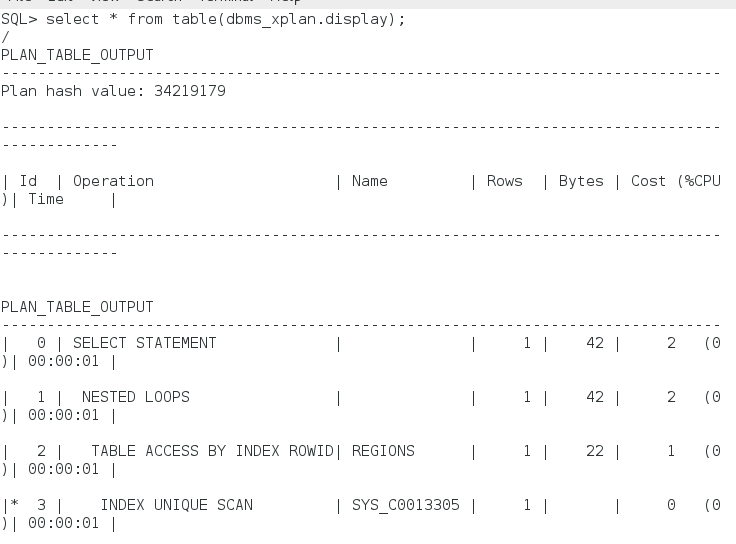


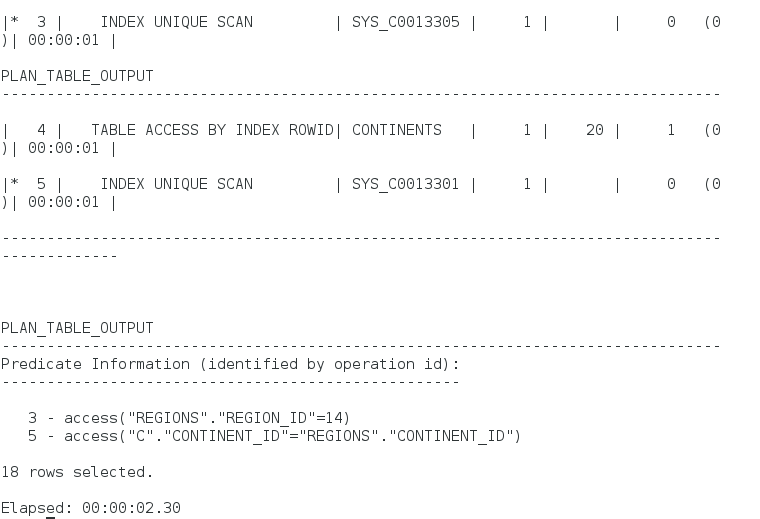
# SQL\*Plus

Connect to a database via SQL\*Plus client and do next steps:

## Show execution plans in SQL\*Plans



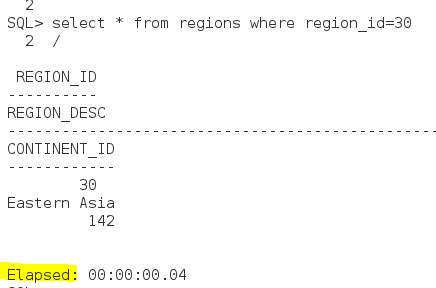




## Set timing on



## Run script



## Save data to file

