|  |
| --- |
|  |
| **Labwork 3** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REVISION HISTORY | | | | | |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
|  | Report on the task | Vitaliya Adamchuk | 22-NOV-2017 |  |  |

Contents

[1. Create Geo Hierarchy 3](#_Toc384725836)

[2. Dimension Hierarchy Analysis 5](#_Toc384725837)

# Geo Hierarchy

This task is based on the data from the previous one.

CREATE VIEW GEO\_DATA AS

SELECT CONTINENT\_ID AS ID,

CONTINENT\_DESC AS NAME,

NULL AS PRT,

'CONTINENT' AS LVL

FROM CE\_CONTINENTS

UNION ALL

SELECT REGION\_ID AS ID,

REGION\_DESC AS NAME,

CONTINENT\_ID AS PRT,

'REGION' AS LVL

FROM CE\_REGIONS

UNION ALL

SELECT COUNTRY\_ID AS ID,

COUNTRY\_DESC AS NAME,

REGION\_ID AS PRT,

'COUNTRY' AS LVL

FROM CE\_COUNTRIES;

SELECT g.ID ,

g.NAME ,

ltrim ( SYS\_CONNECT\_BY\_PATH ( NAME, '==>' ), '==>' ) AS PATH ,

level lvl ,

CONNECT\_BY\_ISLEAF ,

(

CASE

WHEN level = **1**

THEN NAME

WHEN level = **2**

THEN ' '

|| NAME

WHEN level = **3**

THEN ' '

|| NAME

END) AS HIERARCHY ,

(

CASE

WHEN level = **1**

THEN 'Branch'

WHEN level = **2**

THEN 'Root'

WHEN level = **3**

THEN 'Leaf'

END) AS Entity ,

(SELECT COUNT(\*) FROM GEO\_DATA START WITH PRT = g.ID CONNECT BY PRIOR ID = PRT

) AS Children

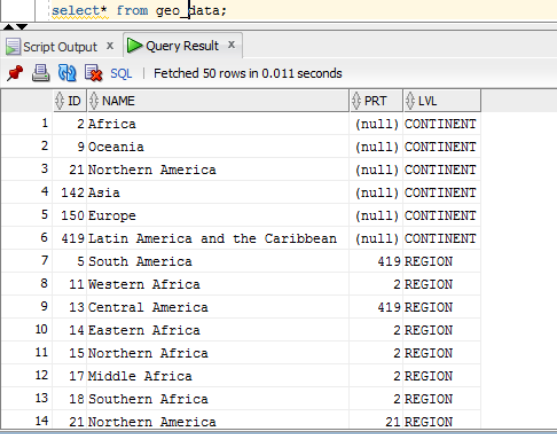
FROM GEO\_DATA g

START WITH PRT IS NULL

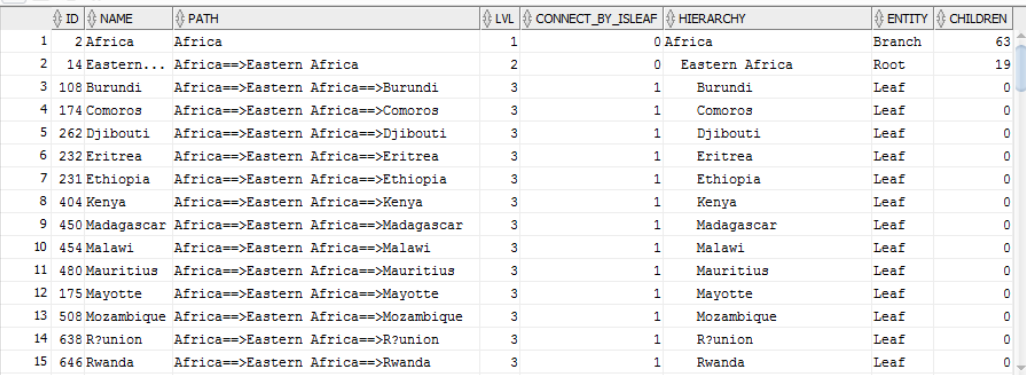
CONNECT BY prior ID = PRT

ORDER SIBLINGS BY **2** ;

**Before executing statment**



**After executing statment**



# Dimension Hierarchy Analysis

In this part recursive WITH clause is used instead of CONNECT BY:

WITH gd (id,

name,

prt,

lvl)

AS

(SELECT id,

name,

prt,

**0** lvl

FROM geo\_data

WHERE prt IS NULL

UNION ALL

SELECT gd\_1.id,

gd\_1.name,

gd.id,

gd.lvl + **1**

FROM geo\_data gd\_1, gd

WHERE gd\_1.prt = gd.id)

SEARCH DEPTH FIRST by id ASC SET ord

CYCLE id SET is\_cycle TO **1** DEFAULT **0**

SELECT id

, name

, prt as parent\_is

, lvl as level\_is

, (CASE WHEN lvl = **0** THEN NAME

WHEN lvl = **1** THEN ' ' || NAME

WHEN lvl = **2** THEN ' ' || NAME END) as HIERARCHY

, (CASE WHEN lvl = **0** THEN 'Branch'

WHEN lvl = **1** THEN 'Root'

WHEN lvl = **2** THEN 'Leaf' END) as Entity

FROM gd T

ORDER BY ord;

**After executing statment**

