

|  |
| --- |
| Хранилище данных |
| Географическая иерархия |

Contents

[1 дАННЫЕ 3](#_Toc498983252)

[1.1 оПИСАНИЕ ДАННЫХ 3](#_Toc498983253)

[1.1.1 Создание таблиц измерений 3](#_Toc498983254)

[1.1.2 Ограничения для таблиц измерений 5](#_Toc498983255)

[1.1.3 Описание фактовой таблицы 6](#_Toc498983256)

[1.1.4 Схема хранилища 6](#_Toc498983257)

[1.2 Генерация данных 7](#_Toc498983258)

[1.2.1 Event\_date 7](#_Toc498983259)

[1.2.2 Product\_id 7](#_Toc498983260)

[1.2.3 Customer\_id 7](#_Toc498983261)

[1.2.4 Store\_id 8](#_Toc498983262)

[1.2.5 Amount 8](#_Toc498983263)

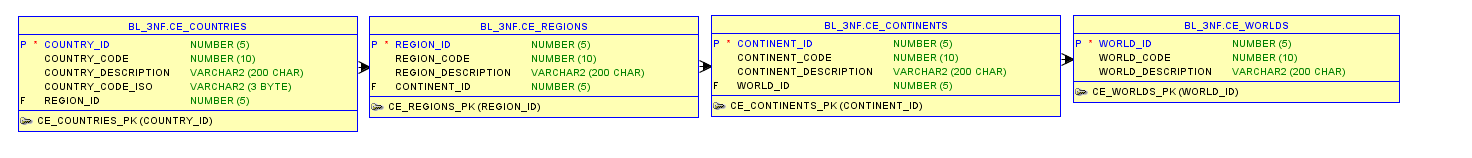
[1.2.6 Фактовая таблица 8](#_Toc498983264)

# Иерархия

## Данные

Данные для выполнения данного задания были взяты на основе первого задания по программе ETL.

Ниже приведена схема модели, которая использована для данного задания



## Создание представления

Создадим представление, используя данные из первого задания. Ниже приведен скрипт для создания представления и его заполнения данными:

CREATE VIEW dim\_location AS

SELECT to\_number (Null) as child\_id,

'World' as Level\_name,

world\_code as parent\_id,

world\_description as Name

FROM ce\_worlds

UNION ALL

SELECT w.world\_id as child\_id,

'Continent' as Level\_name,

c.continent\_code as parent\_id,

continent\_description as Name

FROM ce\_continents c JOIN ce\_worlds w ON c.world\_id=w.world\_id

UNION ALL

SELECT c.continent\_code as child\_id,

'Region' as Level\_name,

r.region\_code as parent\_id,

r.region\_description as Name

FROM ce\_regions r JOIN ce\_continents c ON r.continent\_id=c.continent\_id

UNION ALL

SELECT r.region\_code as child\_id,

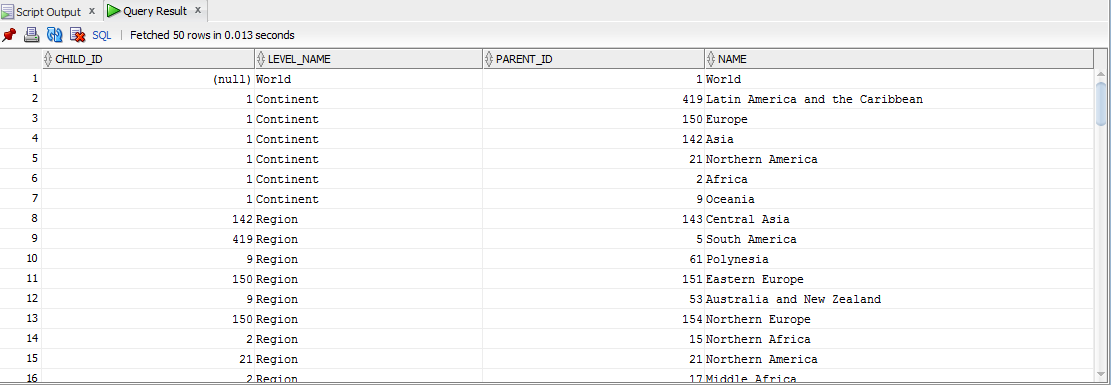
'Country' as Level\_name,

c.country\_code as parent\_id,

c.country\_description as Name

FROM CE\_COUNTRIES c JOIN ce\_regions r ON c.region\_id=r.region\_id;

Результат выполнения скрипта:



# использование connect\_by

Напишем запрос, который позволит:

* + - посмотреть полную иерархию для кадой строки,
    - укажет где находиться branch, root, leaf,
    - количество leaf для branch или root.

Представленный ниже скрипт позволяет нам удовлетворить заданные выше условия:

SELECT Parent\_id,

Name,

ltrim ( SYS\_CONNECT\_BY\_PATH ( Name, '==>' ), '==>' ) as PATH,

CASE

WHEN level = 4 AND connect\_by\_isleaf = 1 THEN 'Leaf'

WHEN level = 3 AND connect\_by\_isleaf = 0 THEN 'Branch'

WHEN level = 2 AND connect\_by\_isleaf = 0 THEN 'Branch'

WHEN level = 1 AND connect\_by\_isleaf = 0 THEN 'Root'

END as Definition,

(SELECT COUNT(\*) FROM dim\_location dl WHERE dim.parent\_id=child\_id) AS amount\_of\_children

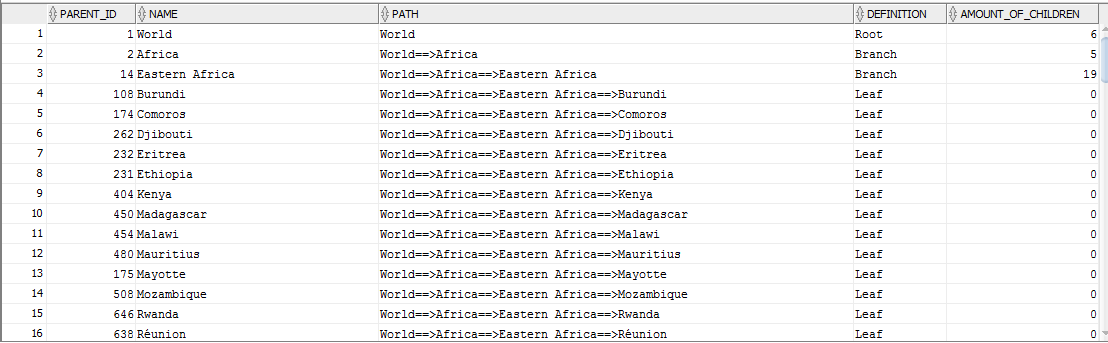
FROM dim\_location dim

START WITH child\_id IS NULL

CONNECT BY prior parent\_id= child\_id

ORDER SIBLINGS by 2;

Результат выполнения:



# Использование With\_as

Написать любой запрос, использующий конструкцию with\_as. Допустим мы хотим посмотреть только основные ветки нашего предыдущего запроса, без листьев (т.е. мир, континенты, регионы) и количество строк в каждой ветке. Следущющий запрос проилистрирует это:

WITH with\_query AS

(SELECT Parent\_id,

Name,

ltrim ( SYS\_CONNECT\_BY\_PATH ( Name, '==>' ), '==>' ) as PATH,

CASE

WHEN level = 4 AND connect\_by\_isleaf = 1 THEN 'Leaf'

WHEN level = 3 AND connect\_by\_isleaf = 0 THEN 'Branch'

WHEN level = 2 AND connect\_by\_isleaf = 0 THEN 'Branch'

WHEN level = 1 AND connect\_by\_isleaf = 0 THEN 'Root'

END as Definition,

(SELECT COUNT(\*) FROM dim\_location dl WHERE dim.parent\_id=child\_id) AS amount\_of\_children

FROM dim\_location dim

START WITH child\_id IS NULL

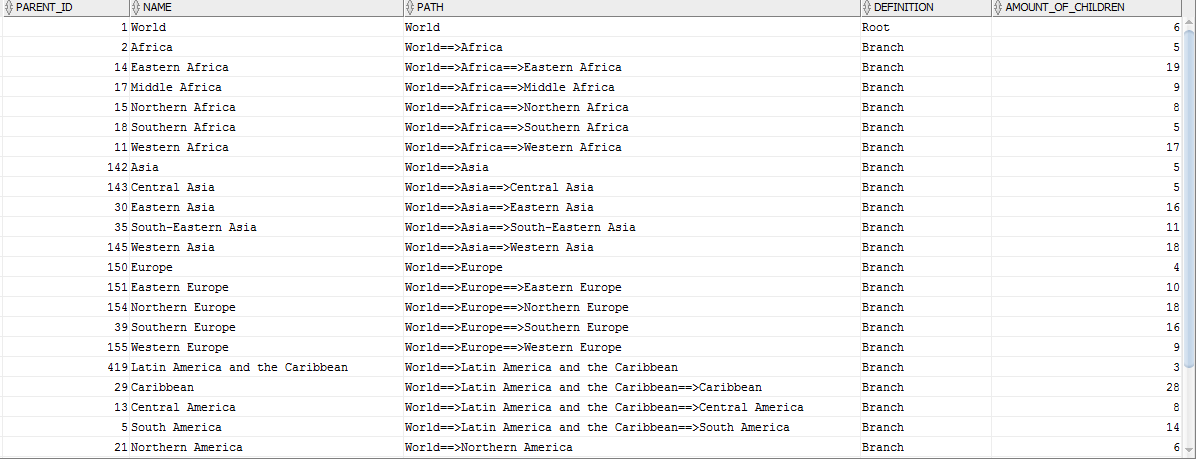
CONNECT BY prior parent\_id= child\_id

ORDER SIBLINGS by 2)

SELECT \* FROM with\_query

WHERE amount\_of\_children!=0;

Результат выполнения запроса:



| REVISION HISTORY | | | | | |
| --- | --- | --- | --- | --- | --- |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.0 | Начальный статус | Алина Макарец | 21-НОЯ-2017 |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |