

|  |
| --- |
| Хранилище данных |
| Генерация фактовой таблицы |

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)

# дАННЫЕ

## оПИСАНИЕ ДАННЫХ

При выполнении задания я использовала данные из следующих таблиц:

SELECT \* FROM oe.customers;

SELECT \* FROM oe.products;

SELECT \* FROM oe.categories\_tab;

SELECT \* FROM sh.countries;

+ моя таблица измерений с датами и данными для этой таблицы

На основе данных из таблиц, я создала 4 измерения и 1 фактовую таблицу.

### Создание таблиц измерений

#### Dim\_customers

Таблица dim\_customers была создана с помощью данных из схемы OE таблицы customers. Для создания таблицы нам понадобилась последовательность, для присвоения покупателю уникального идентификатора. Ниже приведен скрипт для создания последовательности:

CREATE SEQUENCE seq\_customers

INCREMENT BY 1

START WITH 1

MINVALUE 1

NOCYCLE;

Ниже приведен скрипт для создания и наполнения таблицы данными:

CREATE TABLE dim\_customers AS

SELECT seq\_customers.nextval as customer\_id,

cust\_first\_name as first\_name,

cust\_last\_name as last\_name,

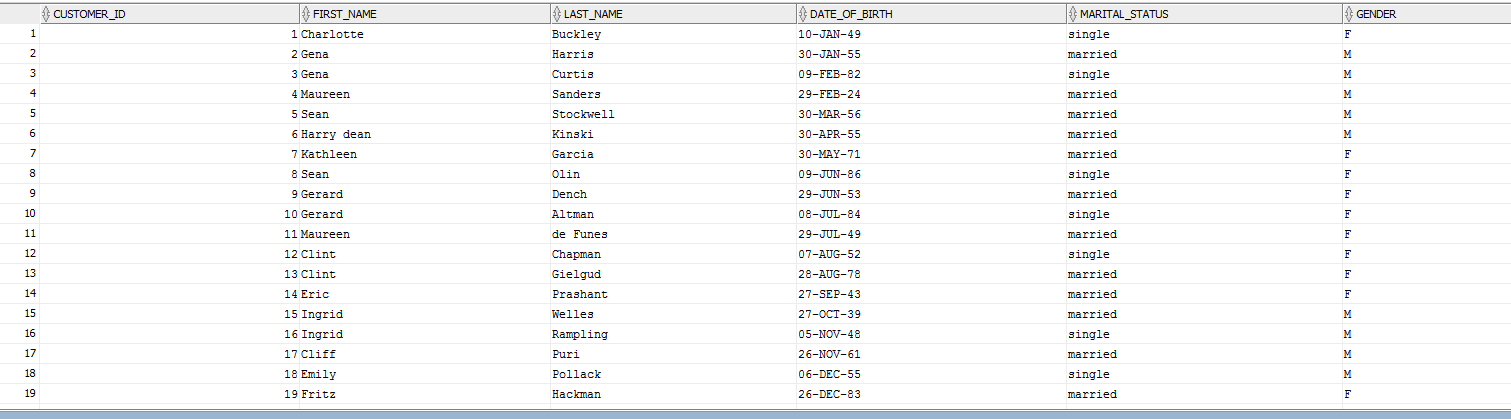
date\_of\_birth,

marital\_status,

gender

FROM oe.customers;

Результат выполнения скрипта приведен на рисунке:



#### Dim\_products

Таблица dim\_products была создана с помощью данных из схемы OE таблиц categories\_tab и products путем присоединения продуктов к категориям. Для создания таблицы нам понадобилась последовательность, для присвоения продукту уникального идентификатора. Ниже приведен скрипт для создания последовательности:

CREATE SEQUENCE seq\_products

INCREMENT BY 1

START WITH 1

MINVALUE 1

NOCYCLE;

Более того в нашем измерении указано наименование продукта и его категория (это не тоже самое,что и подкатегория). Ниже представлен скрипт для создания и наполнения таблицы данными:

CREATE TABLE dim\_products AS

SELECT seq\_products.nextval as product\_id,

p.product\_name,

nvl(p.list\_price,0) as list\_price,

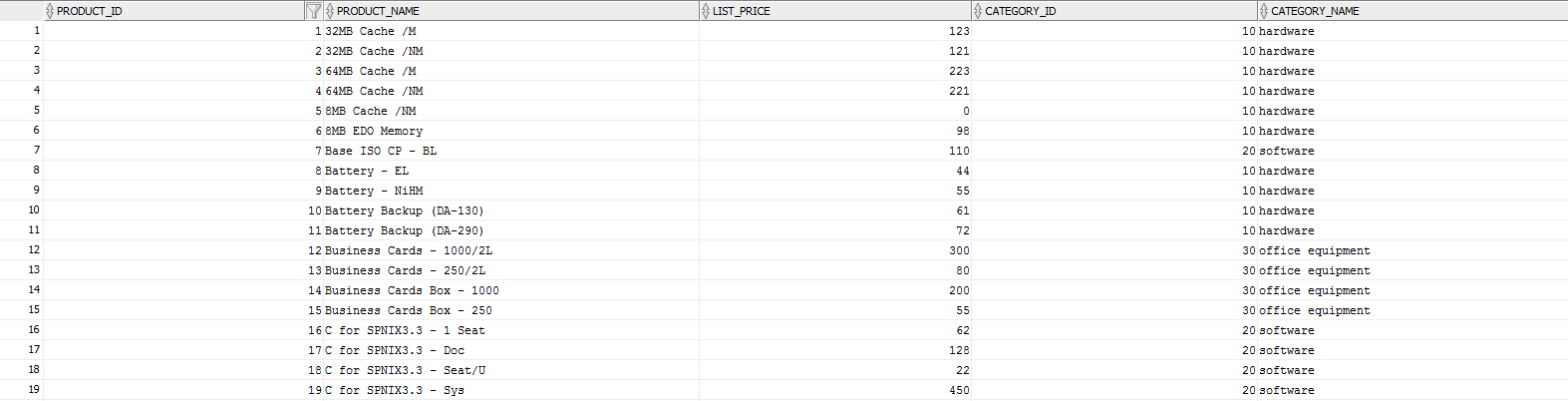
cat.category\_id,

cat.category\_name

FROM oe.products p JOIN oe.categories\_tab c ON p.category\_id=c.category\_id

JOIN oe.categories\_tab cat ON cat.category\_id = c.parent\_category\_id;

Результат выполнения скрипта приведен на рисунке:



#### DIM\_TIME\_DAY

В данном задании, я использовала таблицу измерение DIM\_TIME\_DAY, которая в последующем будет использоваться мною для построения моего хранилища данных, поэтому здесь вы сможете видеть названия на русском языке, прошу это не учитывать, т.к. главными колнками в данной таблице будут являться колонки с датами. Ниже приведены скрипты для создания таблицы и ее наполнения:

CREATE TABLE DIM\_TIME\_DAY (

Date\_id DATE NOT NULL,

Day\_of\_week NUMBER(2) NOT NULL,

Day\_name\_of\_week VARCHAR2(25) NOT NULL,

Day\_of\_month NUMBER(8) NOT NULL,

Day\_of\_year NUMBER(8) NOT NULL,

Week\_of\_month NUMBER(8) NOT NULL,

Week\_of\_year NUMBER(8) NOT NULL,

Month\_number NUMBER(8) NOT NULL,

Month\_name VARCHAR2(20) NOT NULL,

Quarter NUMBER(2) NOT NULL,

First\_day\_of\_month DATE NOT NULL,

Last\_day\_of\_month DATE NOT NULL,

Year NUMBER(8) NOT NULL,

Year\_Quater VARCHAR2(25) NOT NULL,

CONSTRAINT PK\_DATE\_ID

PRIMARY KEY (Date\_id)

);

INSERT INTO DIM\_TIME\_DAY

SELECT TO\_DATE(SYSDATE+rownum-365\*7,'DD-MM-YYYY') as Date\_id,

to\_number(to\_char(SYSDATE+rownum-365\*7-1, 'D')) as Day\_of\_week,

to\_char(SYSDATE+rownum-365\*7,'Day','NLS\_DATE\_LANGUAGE = RUSSIAN') AS Day\_name\_of\_week,

to\_number(to\_char(extract(day from SYSDATE+rownum-365\*7)))as Day\_of\_month,

to\_number(to\_char(SYSDATE+rownum-365\*7, 'DDD')) AS Day\_of\_year,

to\_number(to\_char(SYSDATE+rownum-365\*7, 'W')) AS Week\_of\_month,

to\_number(to\_char(SYSDATE+rownum-365\*7, 'IW')) AS Week\_of\_year,

to\_number(to\_char(extract(month from SYSDATE+rownum-365\*7))) as Month\_number,

to\_char(SYSDATE+rownum-365\*7, 'MONTH', 'NLS\_DATE\_LANGUAGE=Russian') AS Month\_name,

to\_number(to\_char(SYSDATE+rownum-365\*7, 'Q')) AS Quarter,

to\_date(last\_day(SYSDATE+rownum-365\*7)+1,'DD-MM-YYYY') AS First\_day\_of\_month,

to\_date(last\_day(SYSDATE+rownum-365\*7),'DD-MM-YYYY') AS Last\_day\_of\_month,

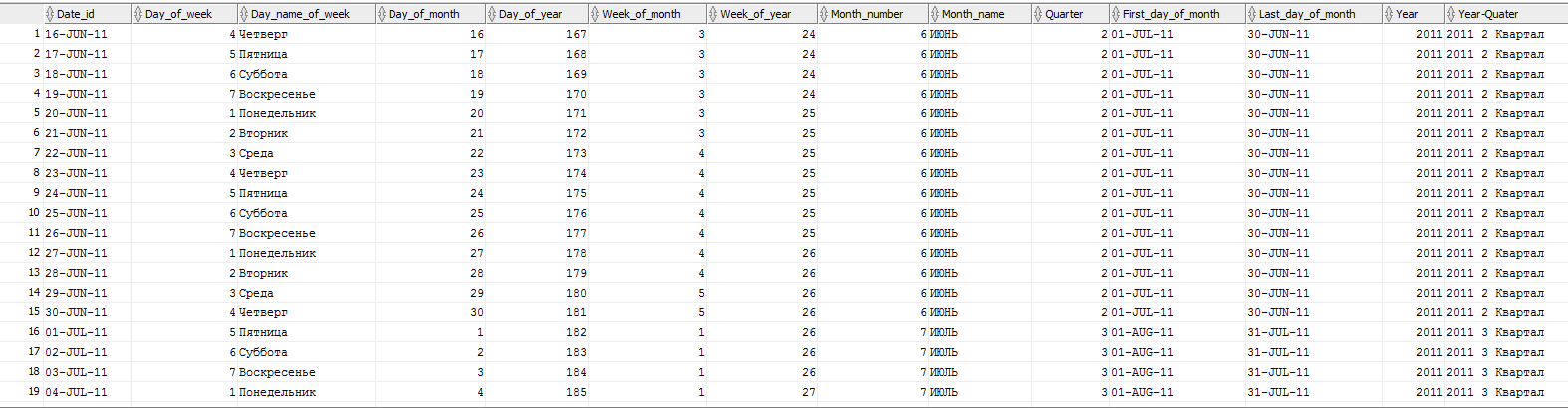
to\_number(to\_char(extract(year from SYSDATE+rownum-365\*7))) as Year,

to\_char(extract(year from SYSDATE+rownum-365\*7) || ' '|| to\_char(SYSDATE+rownum-365\*7, 'Q') || ' ' ||'???????') as "Year-Quater"

FROM dual

CONNECT BY rownum <=365\*7;

Результат выполнения скрипта приведен на рисунке:



#### Dim\_stores

Для создания следующей таблицы нам понадобиться последовательность, для присвоения магазину уникального идентификатора. Ниже приведен скрипт для создания последовательности:

CREATE SEQUENCE seq\_stores

INCREMENT BY 1

START WITH 1

MINVALUE 1

NOCYCLE;

В данном измерении будем считать, что только один магазин расположен в одной странеДанные были использованы из схемы SH таблицы countries. . Ниже приведен скрипт для создания таблицы и ее наполнения.

CREATE TABLE dim\_stores AS

SELECT seq\_stores.nextval as store\_id,

country\_id,

country\_name,

country\_subregion,

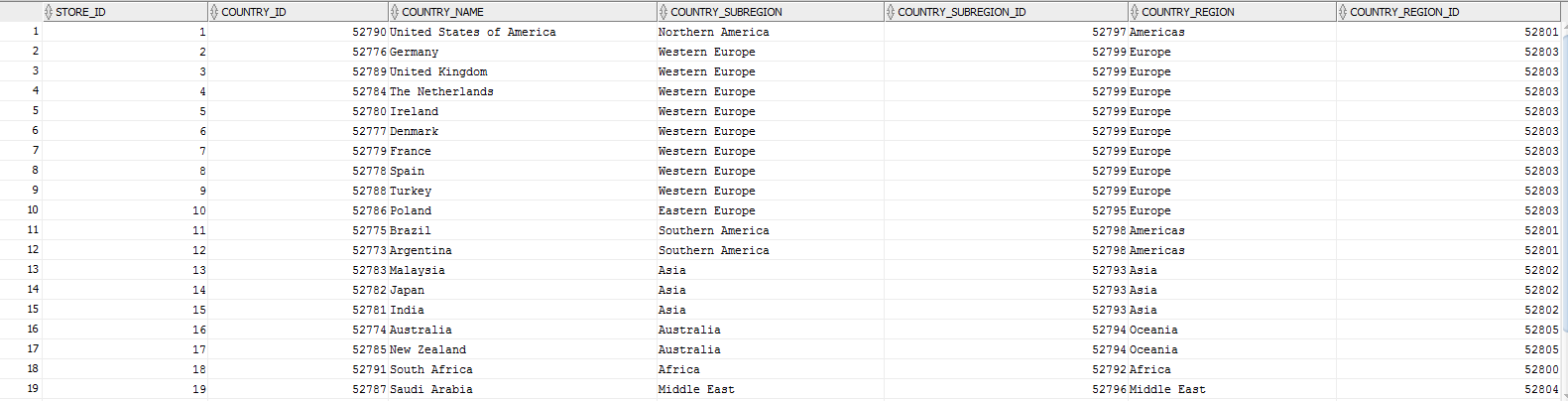
country\_subregion\_id,

country\_region,

country\_region\_id

FROM sh.countries;

Результат выполнения скрипта приведен на рисунке:



### Ограничения для таблиц измерений

Наложим ограничения на наши таблицы измерений путем изменения таблиц для добавления первичных ключей. Ниже приведен скрипт для обновления структур таблиц:

ALTER TABLE dim\_customers

ADD CONSTRAINT PK\_customer\_id PRIMARY KEY (customer\_id);

ALTER TABLE dim\_products

ADD CONSTRAINT PK\_product\_id PRIMARY KEY (product\_id);

ALTER TABLE dim\_stores

ADD CONSTRAINT PK\_store\_id PRIMARY KEY (store\_id);

### Описание фактовой таблицы

Зерном нашей фактовой таблицы будет являться продажа определенному покупателю в определенном магазине в определенную дату. Также предположим, что один покупатель не может купить за один раз более 5 единиц одного товара.

Фактовая таблица будет содержать в себе 4 внешних ключа и 2 меры: Event\_date из Dim\_time\_day, Product\_id из Dim\_products, Customer\_id из Dim\_customers, Store\_id из Dim\_stores, Amount и Total\_price.

Создадим таблицу, в которую будем вставлять данные. Ниже приведен скрипт для создания таблицы:

CREATE TABLE fct\_table

(Event\_date DATE NOT NULL,

Product\_id NUMBER NOT NULL,

Customer\_id NUMBER NOT NULL,

Store\_id NUMBER NOT NULL,

Total\_price NUMBER NOT NULL,

Amount NUMBER NOT NULL,

CONSTRAINT FK\_Event\_date FOREIGN KEY (Event\_date)

REFERENCES Dim\_time\_day (Date\_id),

CONSTRAINT FK\_Product\_id FOREIGN KEY (Product\_id)

REFERENCES Dim\_Products (Product\_id),

CONSTRAINT FK\_Customer\_id FOREIGN KEY (Customer\_id)

REFERENCES Dim\_Customers (Customer\_id),

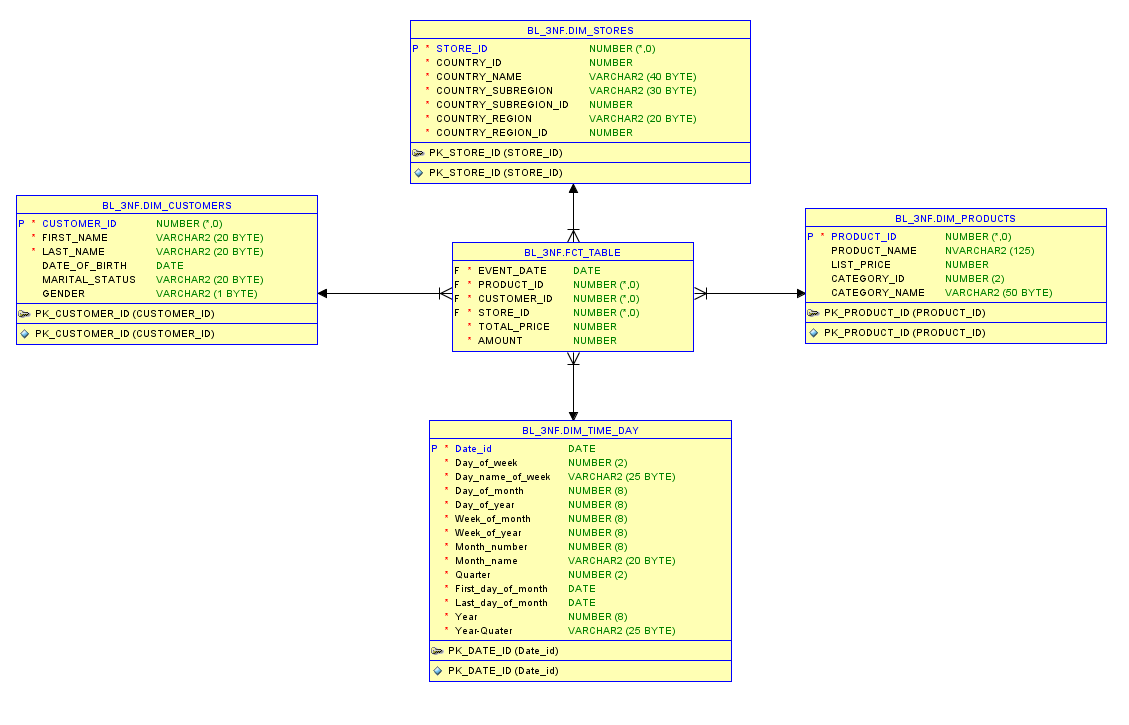
CONSTRAINT FK\_Store\_id FOREIGN KEY (Store\_id)

REFERENCES Dim\_Stores (Store\_id)

);

### Схема хранилища

В итоге мы имеем следующую схему хранилища:



## Генерация данных

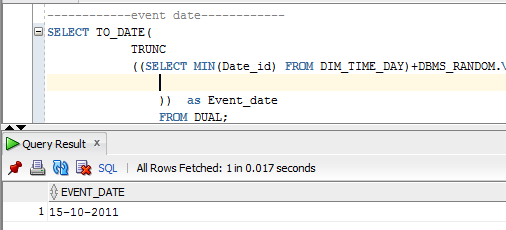
Для генерации данных будем использовать функция dbms\_random.

### Event\_date

Найдем char значение для минимальной и максимальной даты и в данном промежутке будем генерировать значения дат. Приведу пример.

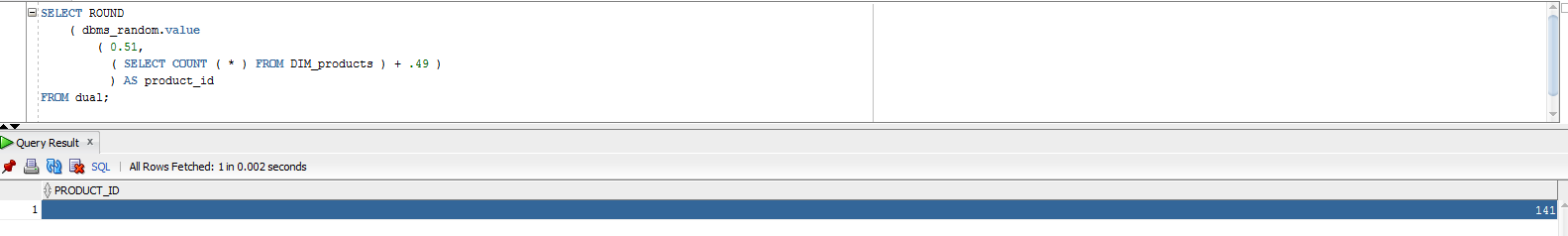


Значения 2455524 – представление даты в char формате. Получается, мы можем генерировать даты в данном промежутке, но сперва вычтем полученные числа, таким образом мы найдем промежуток, в которм мы сгенерируем наши данные следующим образом:



### Product\_id

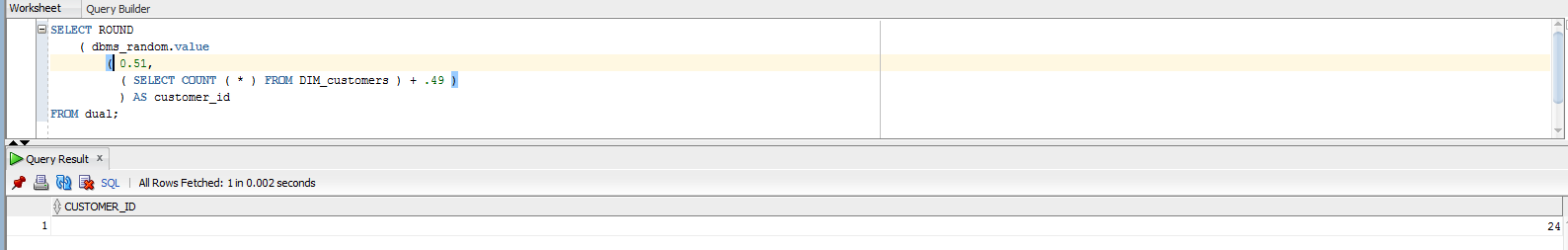
Значения для продуктов будут генерироваться следующим образом:



Максимальное число продуктов 288, в нашем случае не будет сгенерировано число больше 288.

### Customer\_id

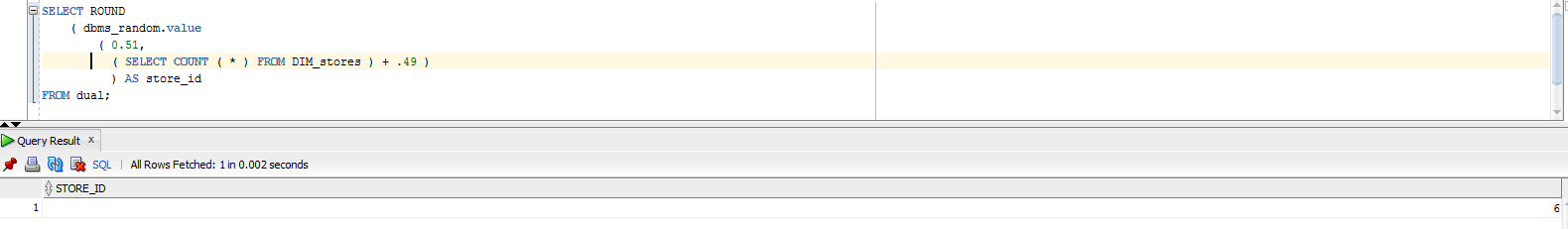
Значения для покупателей будут генерироваться следующим образом:



Максимальное число покупателей 319, в нашем случае не будет сгенерировано число больше 319.

### Store\_id

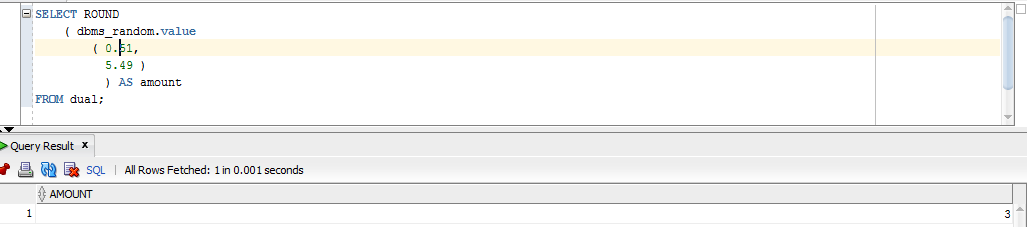
Значения для магазинов будут генерироваться следующим образом:



Максимальное число магазинов 23, в нашем случае не будет сгенерировано число больше 23.

### Amount

В описании факта было упомянуто правило, что покупатель не может совершить покупку более 5 количества определенного продукта. В данном скрипте это отбражено. Ниже приведен скрин, подтверждающий это:



### Фактовая таблица

Ниже приведен скрипт, который может нам сгенерировать данные для нашей фактовой таблицы:

BEGIN

FOR i IN 1..50

LOOP

dbms\_random.seed

(

i \* 5

)

;

INSERT INTO fct\_table

SELECT fct.Event\_date,

fct.product\_id,

fct.customer\_id,

fct.store\_id,

fct.amount\*prd.list\_price AS total\_price,

fct.amount

FROM

(SELECT TO\_DATE( ------------event date------------

TRUNC

(

(SELECT MIN(Date\_id) FROM DIM\_TIME\_DAY)+DBMS\_RANDOM.VALUE(1,2554)

)) as Event\_date,

ROUND ------------product\_id------------

(DBMS\_RANDOM.VALUE(

( SELECT MIN(PRODUCT\_ID) FROM dim\_products),

(SELECT MAX(PRODUCT\_ID) FROM dim\_products)

)) AS product\_id,

ROUND ------------customer\_id------------

(DBMS\_RANDOM.VALUE

( (SELECT MIN(CUSTOMER\_ID) FROM dim\_customers),

(SELECT MAX(CUSTOMER\_ID) FROM dim\_customers)

)) AS customer\_id,

ROUND ------------store\_id------------

(DBMS\_RANDOM.VALUE

( (SELECT MIN(STORE\_ID) FROM dim\_stores),

(SELECT MAX(STORE\_ID) FROM dim\_stores)

)) AS store\_id,

ROUND ------------amount------------

(DBMS\_RANDOM.VALUE

(1,5)

) AS amount

FROM DUAL

CONNECT BY level <= 100000

) fct,

(SELECT product\_id,

list\_price

FROM dim\_products

) prd,

(SELECT customer\_id

FROM dim\_customers) cust,

(SELECT store\_id

FROM dim\_stores) str,

(SELECT date\_id

FROM dim\_time\_day) dtd

WHERE fct.product\_id=prd.product\_id

AND fct.customer\_id=cust.customer\_id

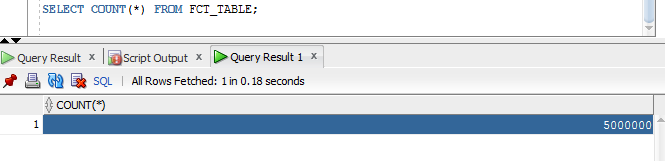
AND fct.store\_id=str.store\_id

AND to\_date(fct.event\_date,'dd-mm-yyyy')=to\_date(dtd.date\_id,'dd-mm-yyyy');

END LOOP;

END;

Результат генерации данных:



# Adhoc отчеты

## Daily report

### CUBE EXTENSION

SELECT p.category\_name,

fct.event\_date,

TO\_CHAR(SUM(fct.total\_price),'9,999,999,999') as Sales

FROM dim\_products p,

fct\_table fct

WHERE p.product\_id = fct.product\_id

AND extract(YEAR FROM fct.event\_date)=2017

GROUP BY CUBE(p.category\_name,

fct.event\_date)

ORDER BY p.category\_name NULLS LAST;

Результат выполнения:



### ROLLUP WITH GROUPING

Скрипт к выполнению:

SELECT DECODE(GROUPING( s.country\_region),1,'All countries', s.country\_region) as region,

DECODE(GROUPING( p.category\_name),1,'All categories', p.category\_name) as categories,

fct.event\_date,

TO\_CHAR(SUM(fct.total\_price),'9,999,999,999') as Sales

FROM dim\_stores s,

fct\_table fct,

dim\_products p

WHERE s.store\_id = fct.store\_id

AND fct.product\_id=p.product\_id

GROUP BY ROLLUP(s.country\_region,

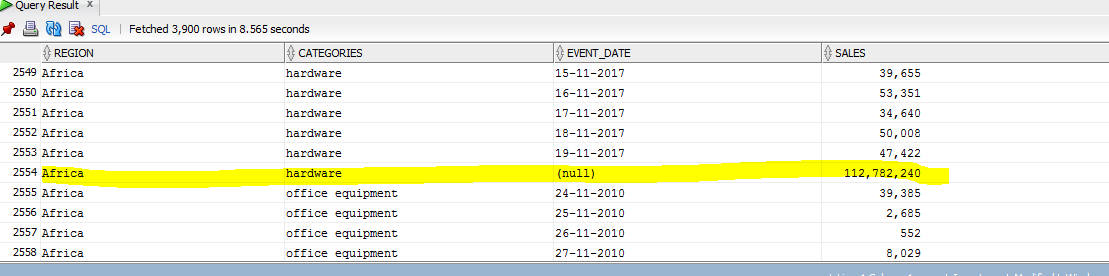
p.category\_name,

fct.event\_date)

ORDER BY s.country\_region,

p.category\_name NULLS LAST;

Результат выполнения:



### GROUPING\_ID

SELECT p.category\_name,

s.country\_region,

fct.event\_date,

TO\_CHAR(SUM(fct.total\_price),'9,999,999,999') as Sales,

GROUPING\_ID (p.category\_name, s.country\_region) as grouping\_id

FROM dim\_stores s,

fct\_table fct,

dim\_products p

WHERE s.store\_id = fct.store\_id

AND p.product\_id=fct.product\_id

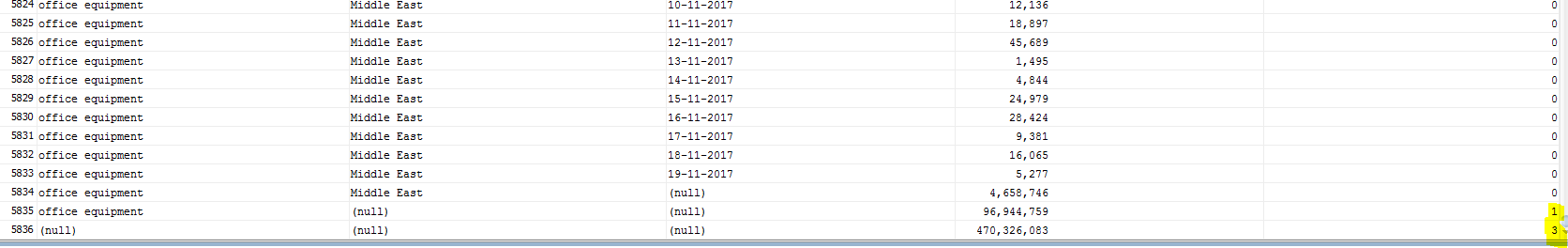
AND extract(year from fct.event\_date)=2017

GROUP BY ROLLUP(p.category\_name,

s.country\_region,

fct.event\_date);

Результат выполнения:



## ROLLUP BY TIME

SELECT DECODE(GROUPING\_ID(dtd.year, dtd.year\_quater, event\_date), 7, 'GRAND TOTAL FOR ' || dtd.year, ' ') AS year,

DECODE(GROUPING\_ID(dtd.year, dtd.year\_quater, event\_date), 3, 'GRAND TOTAL FOR ' || dtd.year\_quater, ' ') AS quarter,

DECODE(GROUPING(event\_date), 1, ' ', event\_date) AS day,

TO\_CHAR(SUM(fct.total\_price), '9,999,999,999') as sales

FROM dim\_products p,

Fct\_table fct,

dim\_time\_day dtd

WHERE p.product\_id = fct.product\_id

AND dtd.date\_id = fct.event\_date

GROUP BY ROLLUP(

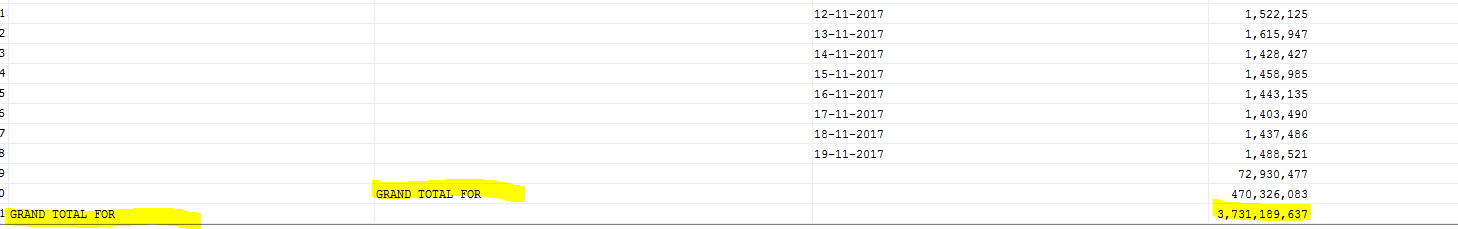
dtd.year,

dtd.year\_quater,

event\_date

);

Результат выполнения:



| REVISION HISTORY | | | | | |
| --- | --- | --- | --- | --- | --- |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.0 | Начальный статус | Алина Макарец | 20-НОЯ-2017 |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |