|  |
| --- |
|  |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 |  |  |

Contents

[1. Data 3](#_Toc384328672)

[2. Advanced Grouping tasks – Reports 3](#_Toc384328673)

[2.1. Create Test AdHoc SQL - Daily Report (CUBE) 3](#_Toc384328674)

[2.2. Create Test AdHoc SQL – ROLLUP by Time 3](#_Toc384328675)

# Data

The Main Task is to generate test data on Storage layers objects that was created on task 01.

Task Results:

* Scripts put on Git
* Storage Layers Objects test data select screenshots

Fact table generation:

INSERT

INTO cls\_orders

(

order\_id,

order\_dt,

seller\_id,

customer\_id,

payment\_method\_id,

delivery\_method\_id,

product\_info\_id,

sum\_of\_payment,

insert\_dt

)

SELECT order\_id,

order\_dt,

seller\_id,

customer\_id,

payment\_method\_id,

delivery\_method\_id,

product\_info\_id,

sum\_of\_payment,

SYSDATE AS insert\_dt

FROM

(SELECT TRUNC(dbms\_random.value(100000000000, 9999999999999)) AS order\_id ,

TRUNC ( (sysdate +4) - dbms\_random.value ( 1, 1000 ) ) AS order\_dt,

ROUND ( dbms\_random.value (

(SELECT MIN ( seller\_id)

FROM bl\_3nf.ce\_sellers

WHERE UPPER(SUBSTR(TRIM(is\_active),1,4))='TRUE'

),

(SELECT MAX ( seller\_id)

FROM bl\_3nf.ce\_sellers

WHERE UPPER(SUBSTR(TRIM(is\_active),1,4))='TRUE'

))) AS seller\_id,

ROUND ( dbms\_random.value (

(SELECT MIN ( customer\_id )

FROM bl\_3nf.ce\_customers

WHERE UPPER(SUBSTR(TRIM(is\_active),1,4))='TRUE'

),

(SELECT MAX ( customer\_id )

FROM bl\_3nf.ce\_customers

WHERE UPPER(SUBSTR(TRIM(is\_active),1,4))='TRUE'

) ) ) AS customer\_id,

ROUND ( dbms\_random.value (

(SELECT MIN ( payment\_method\_id )

FROM bl\_3nf.ce\_payment\_methods

WHERE UPPER(SUBSTR(TRIM(is\_active),1,4))='TRUE'

),

(SELECT MAX ( payment\_method\_id )

FROM bl\_3nf.ce\_payment\_methods

WHERE UPPER(SUBSTR(TRIM(is\_active),1,4))='TRUE'

) ) ) AS payment\_method\_id,

ROUND ( dbms\_random.value (

(SELECT MIN ( delivery\_method\_id )

FROM bl\_3nf.ce\_delivery\_methods

WHERE UPPER(SUBSTR(TRIM(is\_active),1,4))='TRUE'

),

(SELECT MAX ( delivery\_method\_id )

FROM bl\_3nf.ce\_delivery\_methods

WHERE UPPER(SUBSTR(TRIM(is\_active),1,4))='TRUE'

) ) ) AS delivery\_method\_id,

ROUND ( dbms\_random.value (

(SELECT MIN ( product\_info\_id )

FROM bl\_3nf.ce\_product\_info a

INNER JOIN bl\_3nf.ce\_products b

ON a.product\_srcid = b.product\_srcid

WHERE UPPER(SUBSTR(TRIM(is\_active),1,4))='TRUE'

),

(SELECT MAX ( product\_info\_id )

FROM bl\_3nf.ce\_product\_info a

INNER JOIN bl\_3nf.ce\_products b

ON a.product\_srcid = b.product\_srcid

WHERE UPPER(SUBSTR(TRIM(is\_active),1,4))='TRUE'

) ) ) AS product\_info\_id,

ROUND ( dbms\_random.value( 100, 99999), 2) AS sum\_of\_payment

FROM

(SELECT \* FROM dual CONNECT BY level <1000000

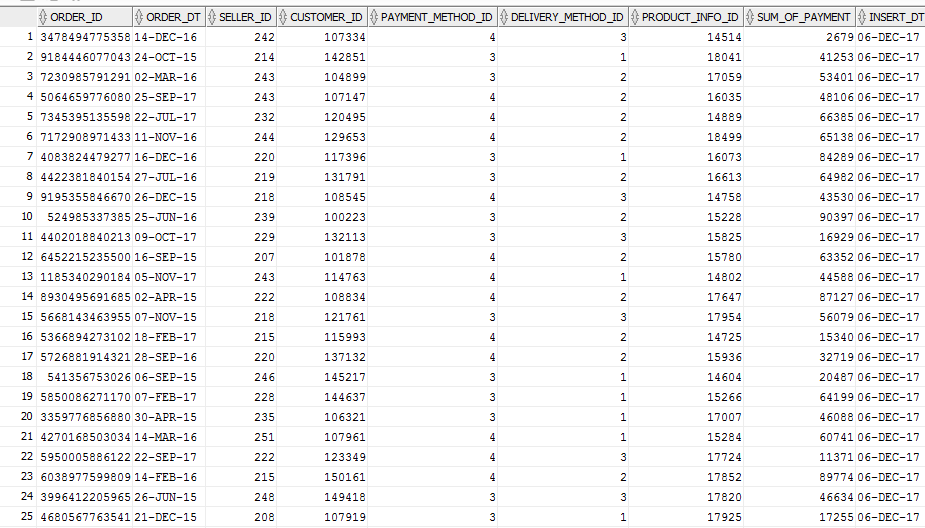
)

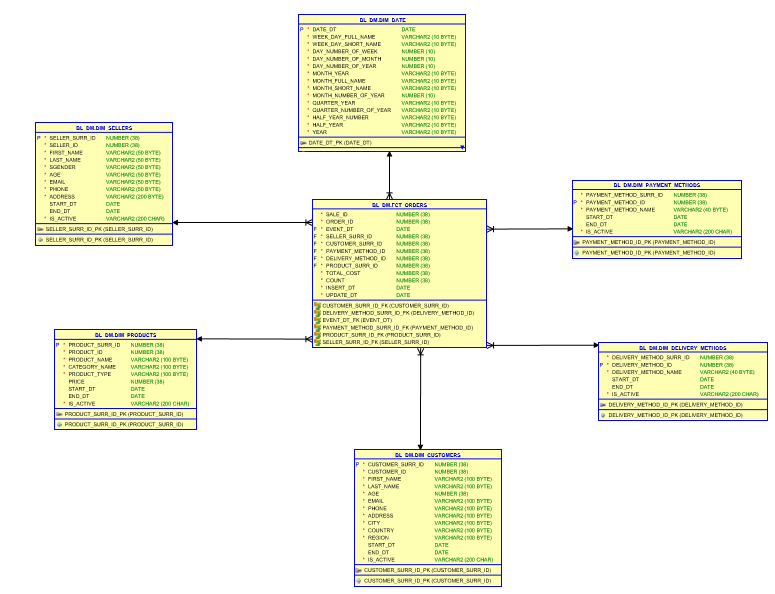
);

Fact table contains 999 999 rows:



Fact\_table:



Star Scheme: 

# Advanced Grouping tasks – Reports

## Create Test AdHoc SQL - Daily Report (CUBE)

Requirements:

* Use CUBE Extension, ROLLUP, Grouping(), Grouping\_ID functions

CUBE:

SELECT concat(p.first\_name,' '

||p.last\_name) AS seller,

extract(YEAR FROM fct.order\_dt) AS YEAR ,

TO\_CHAR(SUM(fct.sum\_of\_payment),'9,999,999,999') AS Sales

FROM bl\_3nf.ce\_sellers p,

cls\_orders fct

WHERE p.seller\_id = fct.seller\_id

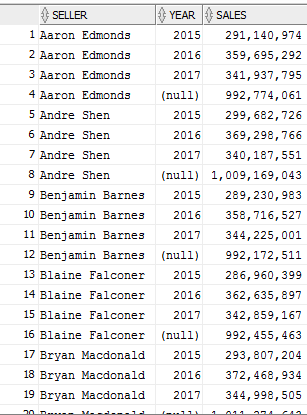
GROUP BY CUBE(concat(p.first\_name,' '

||p.last\_name), extract(YEAR FROM fct.order\_dt))

ORDER BY concat(p.first\_name,' '

||p.last\_name) NULLS LAST;

This analysis provides us with opportunity to analyze sales per sellers.



Task Results:

* Create document that will compare script results and report layouts
* Put scripts on Git

## Create Test AdHoc SQL – ROLLUP by Time

SELECT

CASE

WHEN GROUPING\_id(d.year,d.quarter\_of\_year,d.month\_of\_year,order\_dt)=1

THEN 'Total by month number '

||d.month\_of\_year

ELSE TO\_CHAR(order\_dt)

END AS order\_dt,

CASE

WHEN GROUPING\_id(d.year,d.quarter\_of\_year,d.month\_of\_year,order\_dt)=3

THEN 'Total by quarter number '

||d.quarter\_of\_year

ELSE TO\_CHAR(month\_of\_year)

END AS month\_of\_year,

CASE

WHEN GROUPING\_id(d.year,d.quarter\_of\_year,d.month\_of\_year,order\_dt)=7

THEN 'Total by year '

||d.year

ELSE TO\_CHAR(quarter\_of\_year)

END AS quarter\_of\_year,

CASE

WHEN GROUPING\_id(d.year,d.quarter\_of\_year,d.month\_of\_year,order\_dt)=15

THEN 'GRAND TOTAL '

||d.year

ELSE TO\_CHAR(YEAR)

END AS YEAR,

SUM(sum\_of\_payment),

grouping\_id(d.year,d.quarter\_of\_year,d.month\_of\_year,order\_dt) AS gr

FROM cls\_orders f

LEFT JOIN bl\_dm.dim\_date d

ON f.order\_dt=to\_date(d.date\_id)

GROUP BY rollup(d.year,d.quarter\_of\_year,d.month\_of\_year,order\_dt)

ORDER BY gr DESC;

This analysis provides us with opportunity to analyze sales per date criteria.

