Table of Contents

[1. Using a single subquery 1](#_Toc363654872)

[2. Using multiple subqueries 3](#_Toc363654873)

[2.1. Nesting subqueries 4](#_Toc363654874)

A table expression determines a virtual table. We commonly see table expressions in the From clause of a query. We started with the simplest table expression- a single base table. (Remember a base table is a persistent table; we create the base table with a Create Table statement.).

Select \*

From zoo;

In unit 4 we added Inner Joins to connect two or more base tables to create a virtual table. That join is a table expression.

Select an\_name, cl\_name\_last

From vt\_animals an

Join vt\_clients cl on an.cl\_id = cl.cl\_id;

In this unit we added other table expressions using outer joins to create a virtual table. Those joins also define table expressions.

Now we are going to discuss a technique that uses a subquery as a table expression. This is sometimes called an inline view.

1. Using a single subquery

Suppose you have a fairly complex query dealing with customer orders that you need to run only for a particular query. You would like to break the query down into smaller, more manageable chunks that you could test separately. One solution is to create a subquery that handles part of the query and then use that query in the From clause of the main query.

In unit 04 we had a query that did not work. We could not refer to the column alias ClientName in the Select clause because that alias was defined in the same Select clause.

Select concat(cl\_name\_last , ' ', cl\_name\_first) as ClientName

, concat(ClientName, ' lives in ', cl\_state )

From vt\_clients;

1. Using a subquery in the From clause

Select concat(ClientName, ' lives in ', cl\_state )

From (

Select concat(cl\_name\_last , ' ', cl\_name\_first) as ClientName

, cl\_state

From vt\_clients

) tbl

;

+---------------------------------------------+

| concat(ClientName, ' lives in ', cl\_state ) |

+---------------------------------------------+

| Carter James lives in AR |

| Harris Eddie lives in AR |

| Dalrymple Jack lives in ND |

| Hawkins Coleman lives in OH |

| Monk Theo lives in NY |

| Montgomery Wes lives in OH |

| NULL |

| NULL |

| Biederbecke Sue lives in IL |

| Biederbecke Sam lives in CA |

| Drake Donald lives in MO |

| Brubeck Dave lives in MA |

| Boston Edger lives in MA |

| Turrentine Stanley lives in CA |

+---------------------------------------------+

14 rows in set (0.00 sec)

The subquery is shown here. It is a Select that exposes the cl\_state and an expression named Client Name

Select concat(cl\_name\_last , ' ', cl\_name\_first) as ClientName

, cl\_state

From vt\_clients

The subquery is enclosed in parentheses, given a table alias, and placed in the From clause of the main query. The main query can use the exposed columns from the subquery. That allows us to use the calculated column by referencing its alias.

1. This is a more complex subquery that assembles the data for the orders and exposes three columns which are used in the main query.

Select ord\_id

, ord\_date

, itemTotal

From (

Select

OH.ord\_id

, OH.ord\_date

, OD.quoted\_price \* quantity\_ordered as itemTotal

From a\_oe.order\_headers OH

Join a\_oe.order\_details OD on OH.ord\_id = OD.ord\_id

Join a\_prd.products PR on OD.prod\_id = PR.prod\_id

where quoted\_price > 0 and quantity\_ordered > 0

)rpt\_base

where ord\_date < '2012-11-01'

;

+--------+---------------------+-----------+

| ord\_id | ord\_date | itemTotal |

+--------+---------------------+-----------+

| 105 | 2012-10-01 00:00:00 | 300.00 |

| 105 | 2012-10-01 00:00:00 | 155.40 |

| 105 | 2012-10-01 00:00:00 | 750.00 |

| 106 | 2012-10-01 00:00:00 | 255.95 |

| 107 | 2012-10-02 00:00:00 | 49.99 |

| 108 | 2012-10-02 00:00:00 | 22.50 |

| 109 | 2012-10-12 00:00:00 | 149.99 |

| 110 | 2012-10-12 00:00:00 | 149.99 |

| 110 | 2012-10-12 00:00:00 | 149.99 |

| 400 | 2012-10-15 00:00:00 | 115.00 |

| 400 | 2012-10-15 00:00:00 | 75.00 |

| 400 | 2012-10-15 00:00:00 | 225.00 |

1. Using multiple subqueries
2. This uses two subqueries and joins them. Each subquery has a name. The subqueries produce virtual tables and we are just joining the two virtual tables.

Select t\_cust.cust\_id

, cust\_name

, prod\_id

, ext\_price

From (

Select

cust\_id

, substring(cust\_name\_first + ' ' + cust\_name\_last,1, 20) as cust\_name

From a\_oe.customers

where cust\_name\_first = 'William'

) t\_cust

Join (

Select

OH.ord\_id

, ord\_date

, cust\_id

, prod\_id

, quoted\_price \* quantity\_ordered as ext\_price

From a\_oe.order\_headers OH

join a\_oe.order\_details OD on OH.ord\_id = OD.ord\_id

) t\_ord on t\_cust.cust\_id = t\_ord.cust\_id

;

+---------+------------------+---------+-----------+

| cust\_id | cust\_name | prod\_id | ext\_price |

+---------+------------------+---------+-----------+

| 404950 | William Morris | 1090 | 149.99 |

| 404950 | William Morris | 1130 | 149.99 |

| 401890 | William Northrep | 1110 | 99.98 |

| 402100 | William Morise | 1130 | 625.00 |

| 402100 | William Morise | 1000 | 200.00 |

| 402100 | William Morise | 1120 | 1900.00 |

| 402100 | William Morise | 1080 | 25.00 |

| 402100 | William Morise | 1100 | 180.00 |

| 402100 | William Morise | 1150 | 19.96 |

| 402100 | William Morise | 1141 | 300.00 |

| 402100 | William Morise | 1030 | 27.00 |

| 404950 | William Morris | 1071 | 15.00 |

| 404950 | William Morris | 1071 | 50.00 |

| 404950 | William Morris | 1072 | 24.25 |

| 401890 | William Northrep | 1020 | 64.75 |

| 401890 | William Northrep | 1110 | 49.99 |

| 404950 | William Morris | 1040 | 300.00 |

| 404950 | William Morris | 1050 | 225.00 |

| 404950 | William Morris | 1080 | 45.00 |

| 404950 | William Morris | 1110 | 49.99 |

| 404950 | William Morris | 1152 | 55.25 |

+---------+------------------+---------+-----------+

21 rows in set (0.00 sec)

I can also join the subquery to a base table

1. Joining the common table expression to a base table

Select cust\_id

, cust\_name\_last

, prod\_id

, ext\_price

From a\_oe.customers

Join (Select OH.ord\_id

, ord\_date

, cust\_id

, prod\_id

, quoted\_price \* quantity\_ordered as ext\_price

From a\_oe.order\_headers OH

join a\_oe.order\_details OD on OH.ord\_id = OD.ord\_id

) t\_ord using (cust\_id)

;

+---------+----------------+---------+-----------+

| cust\_id | cust\_name\_last | prod\_id | ext\_price |

+---------+----------------+---------+-----------+

| 403000 | Williams | 1030 | 300.00 |

| 403000 | Williams | 1020 | 155.40 |

| 403000 | Williams | 1010 | 750.00 |

| 401250 | Morse | 1060 | 255.95 |

| 403050 | Hamilton | 1110 | 49.99 |

| 403000 | Williams | 1080 | 22.50 |

| 403000 | Williams | 1130 | 149.99 |

| 404950 | Morris | 1090 | 149.99 |

Nesting subqueries

1. This nests two subqueries in the From clause. As it stands it is simply a complex way to get customers with the first name William, but it does show nested subqueries

Select cust\_name

From (

Select concat(cust\_name\_first , ' ' , cust\_name\_last) as cust\_name

From (

Select cust\_id, cust\_name\_first, cust\_name\_last

From a\_oe.customers

Where cust\_name\_first = 'William'

) tbl1

) tbl2

;

+------------------+

| cust\_name |

+------------------+

| William Northrep |

| William Morise |

| William Morris |

| William Morris |

| William Max |

+------------------+

5 rows in set (0.00 sec)