Table of Contents

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_2014;
```

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;
```

Demo 01: 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 a_vets.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                    |
| Davis Donald lives in NM                    |
| Boston Edger lives in MA                    |
| Turrentine Stanley lives in CA              |
+---------------------------------------------+
15 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.

Demo 02: 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
   where quoted_price > 0 and quantity_ordered > 0
  )rpt_base
where ord_date < '2013-11-01'
order by ord_date, ord_id
;
+--------+---------------------+-----------+
| ord_id | ord_date            | itemTotal |
+--------+---------------------+-----------+
|    227 | 2013-08-01 00:00:00 |    212.50 |
|    227 | 2013-08-01 00:00:00 |    227.97 |
|    223 | 2013-08-05 00:00:00 |    148.99 |
|    223 | 2013-08-05 00:00:00 |    362.50 |
|    223 | 2013-08-05 00:00:00 |    227.97 |
|    224 | 2013-08-07 00:00:00 |   1459.90 |
|    224 | 2013-08-07 00:00:00 |    150.00 |
|    218 | 2013-08-08 00:00:00 |     79.75 |
|    218 | 2013-08-08 00:00:00 |   2500.00 |
|    218 | 2013-08-08 00:00:00 |    227.97 |
|    218 | 2013-08-08 00:00:00 |     14.50 |
|    225 | 2013-08-09 00:00:00 |     62.25 |
|    605 | 2013-09-05 00:00:00 |    300.00 |
|    605 | 2013-09-05 00:00:00 |   1355.40 |
|    605 | 2013-09-05 00:00:00 |    625.00 |
|    605 | 2013-09-05 00:00:00 |    300.00 |
|    605 | 2013-09-05 00:00:00 |    125.00 |
. . .
```

In the query above, I cannot display an attribute such as quantity_ordered. That attribute is not exposed by the subquery; it is not in the Select list of the subquery.

```
ERROR 1054 (42S22): Unknown column 'quantity_ordered' in 'field list'
```

2. Using multiple subqueries

Demo 03: 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
   , concat(cust_name_first , ' ' ,cust_name_last) as cust_name
   From a_oe.customers
   where cust_name_first = 'William'
       ) t_cust
Join (
   Select
     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
Order by t_cust.cust_id, prod_id;
+---------+-----------------+---------+-----------+
| cust_id | cust_name       | prod_id | ext_price |
+---------+-----------------+---------+-----------+
|  401890 | William Northrep |    1020 |     64.75 |
|  401890 | William Northrep |    1110 |     99.98 |
|  401890 | William Northrep |    1110 |     49.99 |
|  402100 | William Morise   |    1000 |    200.00 |
|  402100 | William Morise   |    1030 |     27.00 |
|  402100 | William Morise   |    1080 |     25.00 |
|  402100 | William Morise   |    1100 |    180.00 |
|  402100 | William Morise   |    1120 |   1900.00 |
|  402100 | William Morise   |    1130 |    625.00 |
|  402100 | William Morise   |    1141 |    300.00 |
|  402100 | William Morise   |    1150 |     19.96 |
|  404950 | William Morris   |    1040 |    300.00 |
|  404950 | William Morris   |    1050 |    225.00 |
|  404950 | William Morris   |    1060 |    511.90 |
|  404950 | William Morris   |    1071 |     50.00 |
|  404950 | William Morris   |    1071 |     15.00 |
|  404950 | William Morris   |    1071 |     50.00 |
|  404950 | William Morris   |    1072 |     48.50 |
|  404950 | William Morris   |    1072 |     24.25 |
|  404950 | William Morris   |    1080 |     45.00 |
|  404950 | William Morris   |    1090 |    149.99 |
|  404950 | William Morris   |    1110 |     49.99 |
|  404950 | William Morris   |    1130 |    149.99 |
|  404950 | William Morris   |    1152 |     55.25 |
+---------+-----------------+---------+-----------+
24 rows in set (0.00 sec)
```

Since we are joining the two virtual table on the cust_id values, each subquery needs to expose that column. The first subquery contributes the cust_name and the second subquery contributes the prod_id and the ext_price.

Demo 04: 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 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 |
|  404950 | Morris         |    1130 |    149.99 |
|  403000 | Williams       |    1150 |    249.50 |
|  403000 | Williams       |    1141 |     75.00 |
|  401890 | Northrep       |    1110 |     99.98 |
|  401250 | Morse          |    1080 |     22.50 |
|  402100 | Morise         |    1130 |    625.00 |
|  402100 | Morise         |    1000 |    200.00 |
```

2.1. Nesting subqueries

Demo 05: 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      |
+------------------+
```