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.).

Then we added Inner Joins and Outer 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 vets.vt_animals an
join vets.vt_clients cl on an.cl_id = cl.cl_id;
```

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. Since we are using this subquery as a table expression, the subquery can have multiple columns and multiple rows. We also will provide a table alias to have a name for the subquery table expression.

This query does not work since you cannot use the column alias as a column name in the same Select clause,

```
select concat(cl_name_last , ' ', cl_name_first) as ClientName
, concat(ClientName, ' lives in ', cl_state )
from vets.vt_clients;
```

Demo 01: Using a subquery in the From clause. The subquery exposes the alias ClientName which we can then use in the Where clause of the main query. The subquery table alias is ClientNames

```
select concat(ClientName, ' lives in ', cl_state )
from (
   select concat(cl_name_last , ', ', cl_name_first) as ClientName
   , cl_state
   from vets.vt_clients
) ClientNames
;
+-------------------------------------------+
| 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                                      |
. . .
```

The subquery is shown here. It is a Select that exposes the cl_state and an expression named ClientName

```
   select concat(cl_name_last , ' ', cl_name_first) as ClientName
   , cl_state
   from vets.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.

We could also focus on dealing with that concat expression with a null first name in the subquery.

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. The subquery is a table expression and is given the table alias rpt_base.

```
select order_id
, order_date
, itemTotal
from (
   select
     OH.order_id
   , OH.order_date
   , OD.quoted_price * quantity_ordered as itemTotal
   from orderEntry.orderHeaders OH
   join orderEntry.orderDetails OD on OH.order_id = OD.order_id
   where quoted_price > 0 and quantity_ordered > 0
  )rpt_base
where order_date < '2015-07-01'
order by order_date, order_id
;
+----------+---------------------+-----------+
| order_id | order_date          | itemTotal |
+----------+---------------------+-----------+
|      522 | 2015-04-05 00:00:00 |     45.00 |
|      540 | 2015-06-02 00:00:00 |     45.00 |
|      540 | 2015-06-02 00:00:00 |     49.99 |
|      540 | 2015-06-02 00:00:00 |     55.25 |
|      301 | 2015-06-04 00:00:00 |    205.00 |
|      302 | 2015-06-04 00:00:00 |    120.00 |
|      302 | 2015-06-04 00:00:00 |    349.95 |
|      306 | 2015-06-04 00:00:00 |    500.00 |
|      306 | 2015-06-04 00:00:00 |    500.00 |
|      307 | 2015-06-04 00:00:00 |   2250.00 |
|      307 | 2015-06-04 00:00:00 |   2250.00 |
|      390 | 2015-06-04 00:00:00 |   1400.00 |
|      395 | 2015-06-04 00:00:00 |   2925.00 |
|      312 | 2015-06-07 00:00:00 |   3000.00 |
|      312 | 2015-06-07 00:00:00 |   2500.00 |
|      312 | 2015-06-07 00:00:00 |   1405.00 |
|      312 | 2015-06-07 00:00:00 |   2500.00 |
|      313 | 2015-06-07 00:00:00 |    125.00 |
|      303 | 2015-06-10 00:00:00 |    125.00 |
|      324 | 2015-06-11 00:00:00 |    599.00 |
|      378 | 2015-06-14 00:00:00 |   2250.00 |
|      378 | 2015-06-14 00:00:00 |   2250.00 |
+----------+---------------------+-----------+
22 rows in set (0.00 sec)
```

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.

2. Using multiple subqueries

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 in the same way we have done other inner joins.
Since we are joining the two virtual table on the customer_id values, each subquery needs to expose that column. The first subquery contributes the customer_name and the second subquery contributes the prod_id and the ext_price.

Demo 03:

```
select t_cust.customer_id
, customer_name
, prod_id
, ext_price
from (
   select
     customer_id
   , concat(customer_name_first , ' ' ,customer_name_last) as customer_name
   from customer.customers
   where customer_name_first = 'William'
        ) t_cust

join (
   select
     customer_id
   , prod_id
   , quoted_price * quantity_ordered as ext_price
   from orderEntry.orderHeaders OH
   join orderEntry.orderDetails OD on OH.order_id = OD.order_id
      ) t_ord  on t_cust.customer_id = t_ord.customer_id

order by t_cust.customer_id, prod_id;
+-------------+-----------------+---------+-----------+
| customer_id | customer_name   | prod_id | ext_price |
+-------------+-----------------+---------+-----------+
|      401890 | William Northrep |    1020 |     64.75 |
|      401890 | William Northrep |    1110 |     49.99 |
|      401890 | William Northrep |    1110 |     99.98 |
|      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 |     50.00 |
|      404950 | William Morris   |    1071 |     15.00 |
|      404950 | William Morris   |    1072 |     24.25 |
|      404950 | William Morris   |    1072 |     48.50 |
|      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)
```

The From clause here is

```
From (subQuery1) t_cust
Join  (subQuery2) t_ord on t_cust.cust_id = t_ord.cust_id
```

The demo has the ON syntax for the join. We could also code the join with the syntax

```
Select . . .
From (subQuery1)  t_cust
Join   (subQuery2)  t_ord using(cust_id)
```

Demo 04: Joining a subquery table expression to a base table

```
/*  demo 04  */
select customer_id
, customer_name_last
, prod_id
, ext_price
from customer.customers
join (select customer_id
       , prod_id
       , quoted_price * quantity_ordered as ext_price
       from orderEntry.orderHeaders OH
       join orderEntry.orderDetails OD on OH.order_id = OD.order_id
       )  t_ord using (customer_id)
;
+-------------+--------------------+---------+-----------+
| customer_id | customer_name_last | prod_id | ext_price |
+-------------+--------------------+---------+-----------+
|      400300 | McGold             |    1120 |   2250.00 |
|      400300 | McGold             |    1125 |   2250.00 |
|      401250 | Morse              |    1060 |    255.95 |
|      401250 | Morse              |    1080 |     22.50 |
|      401250 | Morse              |    1070 |    225.00 |
|      401250 | Morse              |    1100 |    205.00 |
|      401250 | Morse              |    2984 |     79.35 |
|      401250 | Morse              |    2014 |     77.95 |
|      401890 | Northrep           |    1110 |     99.98 |
|      401890 | Northrep           |    1020 |     64.75 |
|      401890 | Northrep           |    1110 |     49.99 |
|      402100 | Morise             |    1130 |    625.00 |
|      402100 | Morise             |    1000 |    200.00 |
|      402100 | Morise             |    1120 |   1900.00 |
|      402100 | Morise             |    1080 |     25.00 |
|      402100 | Morise             |    1100 |    180.00 |
|      402100 | Morise             |    1150 |     19.96 |
|      402100 | Morise             |    1141 |    300.00 |
|      402100 | Morise             |    1030 |     27.00 |
```

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 customer_name
from (
   select concat(customer_name_first , ' ' , customer_name_last)
         as customer_name
   from (
      select customer_id, customer_name_first, customer_name_last
      from customer.customers
      where customer_name_first = 'William'
      ) tblWilliam
   ) tblConcatName
;
+------------------+
| cust_name        |
+------------------+
| William Northrep |
| William Morise   |
| William Morris   |
| William Morris   |
| William Max      |
+------------------+
```