SeQueL 5 – Queries – Talk to the ORACLE!

by Clark Anderson

For quite a while, I wondered just how different SQL for ORACLE would be from SQL for MS Access. I will show some of the similarities and differences. To provide examples I will revisit the bagel shop of my earlier articles. Here are two tables I used:

BagelsBaked Table

BakedID	BagelID	Name	Quantity
1	1	Plain	360
2	4	Sesame Seed	240
3	7	Poppy Seed	240
4	2	Egg	240

BagelsOrdered Table

OrderID	BagelID	Name	Quantity
1	1	Plain	240
2	2	Egg	120
3	5	Whole Wheat	120
4	3	Everything	60

The MS Access SQL, shown below, brings data from two tables together.

```
SELECT BagelsBaked.Name, BagelsBaked.Quantity,
BagelsOrdered.Quantity, BagelsOrdered.Name
FROM BagelsBaked INNER JOIN BagelsOrdered ON
BagelsBaked.Name = BagelsOrdered.Name;
```

Here is an equivalent query for ORACLE:

```
SELECT BagelsBaked.Name, BagelsBaked.Quantity,
BagelsOrdered.Quantity, BagelsOrdered.Name
FROM BagelsBaked, BagelsOrdered WHERE
BagelsBaked.Name = BagelsOrdered.Name;
```

Notice that the SELECT clause is the same, but the FROM clause merely lists the table names. Also note that the information in the MS Access JOIN clause is converted to a WHERE clause for ORACLE.

BagelsBaked		BagelsOrdered	
Name	Quantity	Quantity	Name
Egg	240	120	Egg
Plain	360	240	Plain

We can adjust the SQL to use the AS keyword providing different, alias, names for the columns:

```
SELECT BagelsBaked.Name AS BakBagels,
BagelsBaked.Quantity AS BakedQty,
BagelsOrdered.Quantity AS OrderedQty,
BagelsOrdered.Name AS OrdBagels
FROM BagelsBaked INNER JOIN BagelsOrdered ON
BagelsBaked.Name = BagelsOrdered.Name
ORDER BY BagelsBaked.Name
```

Here is an equivalent query for ORACLE:

```
SELECT BagelsBaked.Name BakBagels,
BagelsBaked.Quantity BakedQty,
BagelsOrdered.Quantity OrderedQty,
BagelsOrdered.Name OrdBagels
FROM BagelsBaked, BagelsOrdered WHERE
BagelsBaked.Name = BagelsOrdered.Name
ORDER BY BagelsBaked.Name
```

ORACLE SQL achieves the same result by placing the alias name immediately after the original name with just a space instead of the AS keyword.

BagelsBaked		BagelsOrdered	
BakBagels	BakedQty	OrderedQty	OrdBagels
Egg	240	120	Egg
Plain	360	240	Plain

ORACLE can take this alias syntax a step further with the table names listed in the FROM clause:

```
SELECT BB.Name BakBagels, BB.Quantity BakedQty,
BO.Quantity OrderedQty, BO.Name OrdBagels
FROM BagelsBaked BB, BagelsOrdered BO WHERE
BB.Name = BO.Name;
ORDER BY BakBagels
```

The BB alias, defined in the FROM clause, for the BagelsBaked table can be used every where else in the query! Of course the same is true for the alias defined for the BagelsOrdered table, BO. I find this feature very handy. I have seen some very long table and column names in ORACLE databases. Also, please note that the ORDER BY clause can refer to the column's alias name, BakBagels.

Now I will recall an example from another article. Here is part of a reference table in a LookUp database:

```
SELECT StateCode, StateName, Country
FROM [D:\SCRATCH\\LookUp.mdb].LUStates
WHERE StateCode IN
('CO','CT','WA','KS','ON','FL','AB');
```

LUStates Table

StateCode	StateName	Country
CO	Colorado	USA
CT	Connecticut	USA
FL	Florida	USA
KS	Kansas	USA
WA	Washington	USA
AB	Alberta	Canada
ON	Ontario	Canada

And another table:

CoffeeHouses Table

Shop	City	State
Moe's	Bloomfield	CT
Real Coffee	Seattle	WA
Cafe Luna	Longmont	CO
Expresso Roma	Boulder	CO
Central Café	Lyons	KS
Aggie's Diner	Ottawa	ON

I can JOIN these tables from their separate databases:

```
SELECT CoffeeHouses.Shop, CoffeeHouses.City,
LUStates.StateName AS State
FROM CoffeeHouses INNER JOIN
[D:\SCRATCH\LookUp.mdb].LUStates ON
CoffeeHouses.State = LUStates.StateCode;
```

Shop	City	State
Moe's	Bloomfield	Connecticut
Real Coffee	Seattle	Washington
Cafe Luna	Longmont	Colorado
Expresso Roma	Boulder	Colorado
Central Café	Lyons	Kansas
Aggie's Diner	Ottawa	Ontario

Whereas MS Access often has separate database files, ORACLE and, I have been told, many other “true SQL” databases tend to combine these separate databases into one and refer to each as separate and independent SCHEMA. To convert this query into ORACLE, I will assume two SCHEMAs, COFFEE and LOOKUP.

```
SELECT CoffeeHouses.Shop, CoffeeHouses.City,
LUStates.StateName State
FROM COFFEE.CoffeeHouses, LOOKUP.LUStates
WHERE CoffeeHouses.State = LUStates.StateCode;
```

Again, the FROM clause completely identifies the tables involved in this INNER JOIN. The syntax is SCHEMA name, dot, table name. I have learned, that it is a very good idea, to always specify the SCHEMA name for each table in the FROM clause.

The same use of alias names still applies:

```
SELECT CH.Shop, CH.City, LUS.StateName State
FROM COFFEE.CoffeeHouses CH, LOOKUP.LUStates
LUS WHERE CH.State = LUS.StateCode;
```

I have provided aliases for each of the tables, CH and LUS, as well as the last column, State.

One of my friends, an Oracle DBA, has pointed out that there are other formats available on Oracle SQL to indicate aliases:

```
StateName AS State
StateName AS “State Name”
```

The AS keyword is optional and double quotes are used, instead of square brackets, to identify a name that contains spaces or other special characters. These variations on the alias syntax indicate that in the SQL used by ORACLE, “there are many ways to skin a cat”.

There is certainly a lot more to learn about ORACLE’s “Standard SQL”, but this is a pretty good start for using this variation of SQL.

I have delivered a lot of software product using MS Access SQL, but it is only this year that I have worked with the SQL used by ORACLE.

I have recently learned the syntax for outer joins, but I want to save that for another article. I will also get more experience with the use of wild cards in ORACLE’s Standard SQL and present that in a future article. I am happy to share my discoveries!

It is even more encouraging to discover that that what I am learning in my ORACLE SQL experience has the even broader use.

If someone says “I know you are skilled in MS Access SQL, but we have this task involving ORACLE / SQL Server / InterBase / Informix / DB2…” I hope this article will help you respond, “I can do that!”