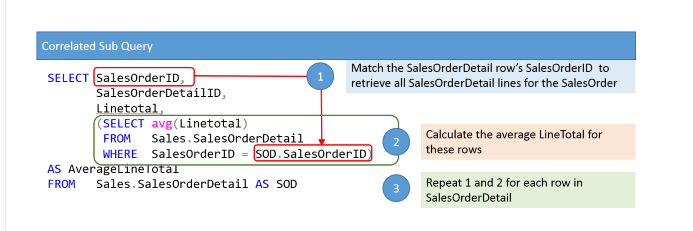
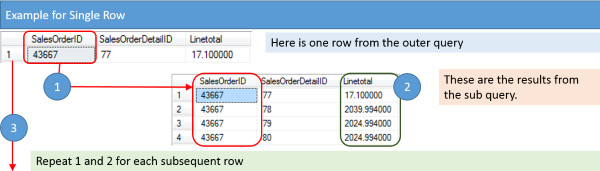
* A subquery that depends on the outer query.
* It means that the [WHERE clause](http://www.zentut.com/sql-tutorial/sql-where/) of the correlated subquery uses the data of the outer query.
* The main difference between a correlated subquery and a non-correlated subquery is that you cannot execute a correlated subquery alone like a non-correlated subquery.
* A correlated subquery is also known as repeating subquery or synchronized subquery.



To further elaborate on the diagram.  The SELECT statement consists of two portions, the outer query and the subquery.  The outer query is used to retrieve all SalesOrderDetail lines.  The subquery is used to find and summarize sales order details lines for a specific SalesOrderID.

[](http://www.essentialsql.com/wp-content/uploads/2015/01/Correlated-Subquery-one-Record.png)

If I was to verbalize the steps we are going to take, I would summarize them as:

1. Get the SalesOrderID.
2. Return the Average LineTotal from All SalesOrderDetail items where the SalesOrderID matches.
3. Continue on to the next SalesOrderID in the outer query and repeat steps 1 and 2.

The query you can run in the AdventureWork2012 database is:

SELECT SalesOrderID,

SalesOrderDetailID,

LineTotal,

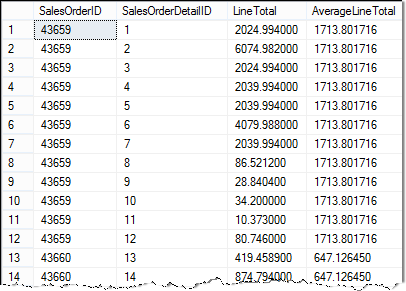
(SELECT AVG(LineTotal)

FROM   Sales.SalesOrderDetail

WHERE  SalesOrderID = SOD.SalesOrderID)

AS AverageLineTotal

FROM   Sales.SalesOrderDetail SOD

[](http://www.essentialsql.com/wp-content/uploads/2015/01/Correlated-Subquery-Results.png)

There are a couple of items to point out.

1. Used column aliases to help make the query results easier to read.
2. Used a table alias, SOD, for the outer query. This makes it possible to use the outer query’s values in the subquery.  Otherwise, the query isn’t correlated!
3. Using the table aliases make it unambiguous which columns are from each table

**Breaking down the Correlated Subquery**

**Let’s now try to break this down using SQL.**

To start let’s assume we’re going to just get our example for SalesOrderDetailID **20**.  The corresponding SalesOrderID is **43661**.

To get the average LineTotal for this item is easy

SELECT AVG(LineTotal)

FROM   Sales.SalesOrderDetail

WHERE  SalesOrderID = **43661**

This returns the value 2181.765240.

Now that we have the average we can plug it in to our query

SELECT SalesOrderID,

SalesOrderDetailID,

LineTotal,

2181.765240 AS AverageLineTotal

FROM   Sales.SalesOrderDetail

WHERE  SalesOrderDetailID = **20**

Using subqueries this becomes

SELECT SalesOrderID,

SalesOrderDetailID,

LineTotal,

(SELECT AVG(LineTotal)

FROM Sales.SalesOrderDetail

WHERE SalesOrderID = **43661**) AS AverageLineTotal

FROM   Sales.SalesOrderDetail

WHERE  SalesOrderDetailID = **20**

Final query is:

SELECT SalesOrderID,

SalesOrderDetailID,

LineTotal,

(SELECT AVG(LineTotal)

FROM Sales.SalesOrderDetail

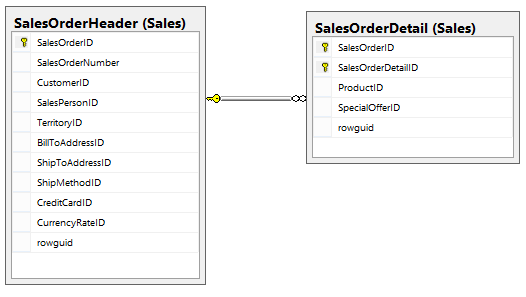
WHERE  SalesOrderID = SOD.SalesOrderID) AS AverageLineTotal

FROM   Sales.SalesOrderDetail AS SOD

**Correlated Subquery with a Different Table**

A Correlated subquery, or for that matter any subquery, can use a different table than the outer query.  This can come in handy when you’re working with a “parent” table, such as SalesOrderHeader, and you want to include in result a summary of child rows, such as those from SalesOrderDetail.

Let’s return the OrderDate, TotalDue, and number of sales order detail lines.  To do this we can use the following diagram to gain our bearings:

[](http://www.essentialsql.com/wp-content/uploads/2015/01/Correlated-Subquery-Datamodel.png)

To do this we’ll include a correlated subquery in our SELECT statement to return the COUNT of SalesOrderDetail lines.  We’ll ensure we are counting the correct SalesOrderDetail item by filtering on the outer query’s SalesOrderID.

Here is the final SELECT statement:

SELECT SalesOrderID,

OrderDate,

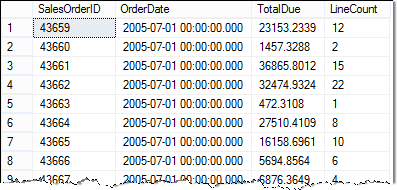
TotalDue,

(SELECT COUNT(SalesOrderDetailID)

FROM Sales.SalesOrderDetail

WHERE SalesOrderID = SO.SalesOrderID) as LineCount

FROM   Sales.SalesOrderHeader SO

[](http://www.essentialsql.com/wp-content/uploads/2015/01/Correlated-Subquery-Different-Table-Example-Results.png)

Some things to notice with this example are:

* The subquery is selecting data from a different table than the outer query.
* I used table and column aliases to make it easier to read the SQL and results.
* Be sure to double check your where clause! If you forget to include the table name or aliases in the subquery WHERE clause, the query won’t be correlated.