**What is the difference between correlated subqueries and uncorrelated subqueries?**

Let’s start out with an example of what an uncorrelated subquery looks like, and then we can compare that with a correlated subquery. Here is an example of an uncorrelated subquery:

**Example of an Uncorrelated Subquery**

Here is an example of some SQL that represents an uncorrelated subquery:

select Salesperson.Name from Salesperson

where Salesperson.ID NOT IN(

select Orders.salesperson\_id from Orders, Customer

where Orders.cust\_id = Customer.ID

and Customer.Name = 'Samsonic')

If the SQL above looks scary to you, don’t worry – it’s still easy to understand for our purposes here. The subquery portion of the SQL above begins after the “NOT IN” statement. The reason that the query above is an uncorrelated subquery is that the subquery can be run ***independently*** of the outer query. Basically, the subquery has no relationship with the outer query.

Now, a correlated subquery has the opposite property – the subquery can **not** be run independently of the outer query. You can take a look at this example of a correlated subquery below and easily see the difference yourself:

## Example of a correlated subquery

SELECT \*

FROM Employee Emp1

WHERE (1) = (

SELECT COUNT(DISTINCT(Emp2.Salary))

FROM Employee Emp2

WHERE Emp2.Salary > Emp1.Salary)

What you will notice in the correlated subquery above is that the inner subquery uses Emp1.Salary, but the alias Emp1 is created in the outer query. This is why it is called a correlated subquery, because the subquery references a value in it’s WHERE clause (in this case, it uses a column belonging to Emp1) that is used in the outer query.

**How does a correlated query work exactly?**

|  |
| --- |
|  |

It’s important to understand the order of operations in a correlated subquery. First, a row is processed in the outer query. Then, for that particular row the subquery is executed – so for each row processed by the outer query, the subquery will also be processed. In our example of a correlated subquery above, every time a row is processed for Emp1, the subquery will also choose that row’s value for Emp1.Salary and run. And then the outer query will move on to the next row, and the subquery will execute for that row’s value of Emp1.Salary. This will continue until the “WHERE (1) = (… )” condition is satisfied.

You can also read this particular SQL problem to get more detailed information on how the correlated query above works: [SQL – Find nth highest salary](http://www.programmerinterview.com/index.php/database-sql/find-nth-highest-salary-sql/)