Instructor:
- Mr. Nazeef Ul Haq and Mr. Nauman Babar

Learning Objectives:

- Understanding of Subqueries.

Helping Material:

1. Subquery:
A subquery is a query that is nested inside a SELECT, INSERT, UPDATE, or DELETE statement, or inside another subquery. A subquery can be used anywhere an expression is allowed.

A subquery nested in the outer SELECT statement has the following components:
- A regular SELECT query including the regular select list components.
- A regular FROM clause including one or more table or view names.
- An optional WHERE clause.
- An optional GROUP BY clause.
- An optional HAVING clause.

The SELECT query of a subquery is always enclosed in parentheses. It cannot include a COMPUTE or FOR BROWSE clause, and may only include an ORDER BY clause when a TOP clause is also specified.

A subquery can be nested inside the WHERE or HAVING clause of an outer SELECT, INSERT, UPDATE, or DELETE statement, or inside another subquery. Up to 32 levels of nesting is possible, although the limit varies based on available memory and the complexity of other expressions in the query. Individual queries may not support nesting up to 32 levels. A subquery can appear anywhere an expression can be used, if it returns a single value.

If a table appears only in a subquery and not in the outer query, then columns from that table cannot be included in the output (the select list of the outer query).

Statements that include a subquery usually take one of these formats:
- WHERE expression [NOT] IN (subquery)
- WHERE expression comparison_operator [ANY | ALL] (subquery)
- WHERE [NOT] EXISTS (subquery)

In some Transact-SQL statements, the subquery can be evaluated as if it were an independent query. Conceptually, the subquery results are substituted into the outer query (although this is not necessarily how SQL Server actually processes Transact-SQL statements with subqueries). There are three basic types of subqueries. Those that:

- Operate on lists introduced with IN, or those that a comparison operator modified by ANY or ALL.
- Are introduced with an unmodified comparison operator and must return a single value.
- Are existence tests introduced with EXISTS.

2. **Subquery Rules**

A subquery is subject to the following restrictions:
- The select list of a subquery introduced with a comparison operator can include only one expression or column name (except that EXISTS and IN operate on SELECT * or a list, respectively).
- If the WHERE clause of an outer query includes a column name, it must be join-compatible with the column in the subquery select list.
- The **ntext, text,** and **image** data types cannot be used in the select list of subqueries.
- Because they must return a single value, subqueries introduced by an unmodified comparison operator (one not followed by the keyword ANY or ALL) cannot include GROUP BY and HAVING clauses.
- The DISTINCT keyword cannot be used with subqueries that include GROUP BY.
- The COMPUTE and INTO clauses cannot be specified.
- ORDER BY can only be specified when TOP is also specified.
- A view created by using a subquery cannot be updated.
- The select list of a subquery introduced with EXISTS, by convention, has an asterisk (*) instead of a single column name. The rules for a subquery introduced with EXISTS are the same as those for a standard select list, because a subquery introduced with EXISTS creates an existence test and returns TRUE or FALSE, instead of data

3. **Subquery Syntax**

```
SELECT column1 , column2 , column3 , ….
FROM table1
WHERE columnN = (SELECT columnM
                 FROM table2
                 WHERE condition)
```

Lab Tasks + Home Tasks:
- Return customers and their orders, including customers who placed no orders (CustomerID, OrderID, OrderDate)
- Report only those customer IDs who never placed any order. (CustomerID, OrderID, OrderDate)
- Report those customers who placed orders on July,1997. (CustomerID, OrderID, OrderDate)
- Report the total orders of each customer. (customerID, totalorders)
- Write a query to generate a five copies of each employee. (EmployeeID, FirstName, LastName)
- List all the products whose price is more than average price.
- Find the second highest price of product.
- Write a query that returns a row for each employee and day in the range 04-07-1996 through 04-08-1997. (EmployeeID, Date)
- Return US customers, and for each customer return the total number of orders and total quantities. (CustomerID, Totalorders, totalquantity)
- Write a query that returns all customers in the output, but matches them with their respective orders only if they were placed on July 04,1997. (CustomerID, CompanyName, OrderID, Orderdate)

- Are there any employees who are older than their managers?
- List that names of those employees and their ages. (EmployeeName, Age, Manager Age)
- List the names of products which were ordered on 8th August 1997. (ProductName, OrderDate)
- List the addresses, cities, countries of all orders which were serviced by Anne and were shipped late. (Address, City, Country)
- List all countries to which beverages have been shipped. (Country)

HackerRank Tasks:
- https://www.hackerrank.com/challenges/the-pads/problem?isFullScreen=true
- https://www.hackerrank.com/challenges/the-company/problem?isFullScreen=true
- https://www.hackerrank.com/challenges/weather-observation-station-18/problem?isFullScreen=true
- https://www.hackerrank.com/challenges/weather-observation-station-19/problem?isFullScreen=true
- https://www.hackerrank.com/challenges/the-report/problem?isFullScreen=true
- https://www.hackerrank.com/challenges/full-score/problem?isFullScreen=true
- https://www.hackerrank.com/challenges/challenges/problem?isFullScreen=true

What to Submit:

You are required to submit the following files.

- 2022-CS-X.sql (You will replace the value of "X" with you roll number to keep name of file)
- Pdf report.