|  |
| --- |
| **National University of Computer and Emerging Sciences** |
| **Lab Manual 4**  “Nested Queries” |
| **Database Systems Lab** |
| **Spring 2023** |



Department of Computer Science FAST-NU, Lahore, Pakistan



1. Contents
2. [Objective 2](#_bookmark0)
3. [Pre-requisites 2](#_bookmark1)
4. [Nested Queries 3](#_bookmark2)
   * 1. [A subquery (inner query) is a SQL select query nested inside a another select query (outer query) 3](#_bookmark3)
     2. [A subquery can be nested inside: 3](#_bookmark4)
     3. [There are two types of subqueries 3](#_bookmark5)
     4. [Scalar Vs Non-scalar 3](#_bookmark6)

[Non-Correlated Query 4](#_bookmark7)

* + 1. [Non-Correlated Subqueries in SELECT clause 4](#_bookmark8)
    2. [Non-Correlated Subqueries in From Clause 4](#_bookmark9)
    3. [Non-Correlated Subqueries in Where Clause 5](#_bookmark10)

[Correlated queries 5](#_bookmark11)

* + 1. [Correlated Subquery in Select Clause 5](#_bookmark12)
    2. [Correlated Subquery in Where Clause 6](#_bookmark13)
    3. [Correlated Subquery in Having Clause 6](#_bookmark14)

Aggregate Functions

# Objective

* + The purpose of this manual is to get started with nested queries. This lab will cover all the topics we have covered before. Starting from simple Select-From-Where, Joins, Order by, Aggregate functions & Group by, all of these will be used in combination with the nested queries.

# Pre-requisites

* + Lab manual 2 & 3 which includes:
    - Select-From-Where clause
    - Joins and all its types Task Distribution

|  |  |
| --- | --- |
| Total Time | 170 Minutes |
| Nested Queries | 30 Minutes |
| Exercise | 120 Minutes |
| Evaluation | Last 20 Minutes |



# Nested Queries

For this in-lab manual, use the **InLab5TryThisSchema.sql** script to create database and practice the queries given below.

### A subquery (inner query) is a SQL select query nested inside a another select query (outer query)

A subquery may occur in:

* SELECT clause of outer query
* FROM clause of outer query
* WHERE clause of outer query (most commonly used)

### A subquery can be nested inside:

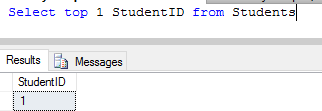
* SELECT statement
* INSERT statement
* UPDATE statement
* DELETE statement
* Another subquery.

### There are two types of subqueries

* Correlated subqueries: where we use some attribute of outer query in inner query, result of inner query will then change according to the attribute of outer query.
* Non-correlated subqueries: where no attribute of outer query is used in inner query, in this case inner query always return same value.

### Scalar Vs Non-scalar

A select query can return a scalar value or a table. Scalar value means one column and one row Example: result of the following query is scalar



A select query can also return non-scalar value, with more than one column and/or more than one row Example:

Select StudentID from Students

Will give non-scalar result.

If you are writing a sub query in Select Clause, the inner query should be Scalar

If you are writiing a subquey in From Clause, inner query can be scalar or non-Scalar

If you are writing a subquery in Where Clause, inner query can be scalar or non-Scalar depending on conditon.



## Non-Correlated Query:

### Non-Correlated Subqueries in SELECT clause

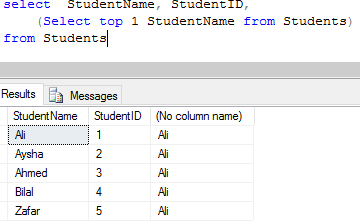
SELECT <List of columns of T>

(select ColumnName from <TableName>) FROM <tablename> AS T

WHERE <condition>

\*\*inner query should be scalar

TRY IT: Non-correlated nested query in Select is not very useful



### Non-Correlated Subqueries in From Clause

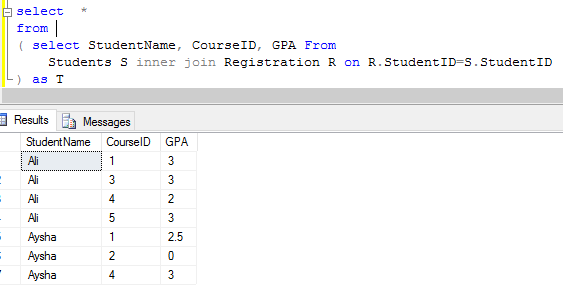
SELECT <List of columns of T ( result of inner query)>

FROM (select ColumnName from <TableName>) as T WHERE <condition>

\*\*inner query can be scalar of non-scalar

\*\*\*always give alias to inner query in from clause

TRY THIS



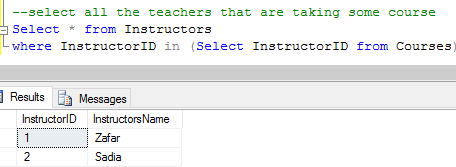


### Non-Correlated Subqueries in Where Clause

SELECT <List of columns of T > FROM TableName as T

WHERE <condition> (select ColumnName from <TableName>)

TRY THIS

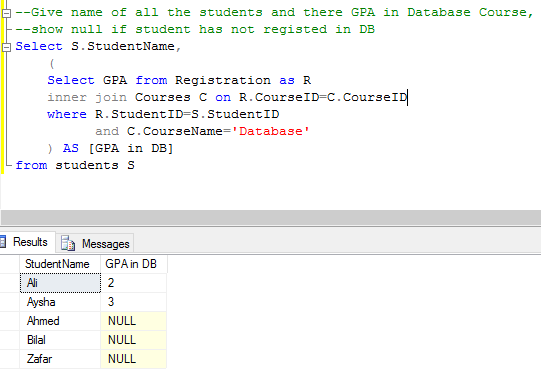


## Correlated queries

When inner query is correlated with outer query, then the inner query is executed for each row of outer query.

### Correlated Subquery in Select Clause

TRY THIS

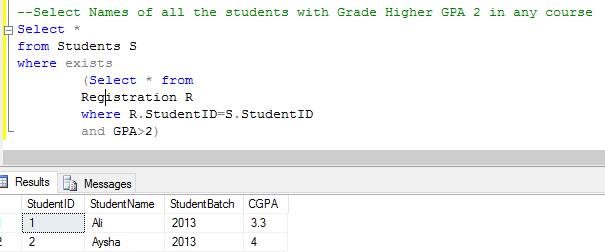


This inner query will get the grade of each row of outer query.



### Correlated Subquery in Where Clause

TRY THIS

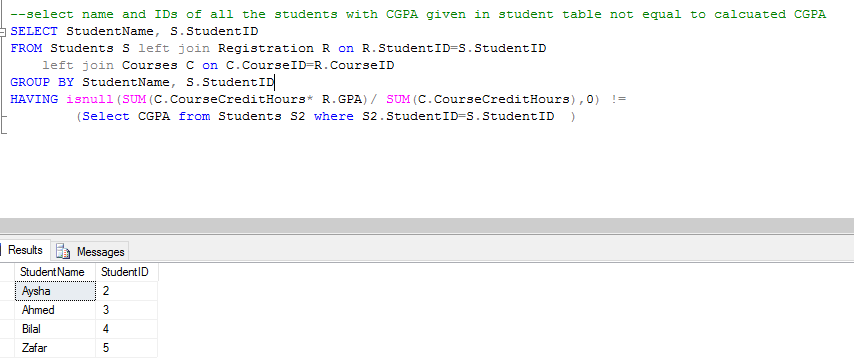


\*\* WHAT DOES THE EXIST CLAUSE DO?

### Correlated Subquery in Having Clause

You can also use subquery in having clause (correlated on non-correlated)

TRY THIS



Modify the query given above to, Show name, IDs, Calcuated CGPA and CGPA given in Student table of all the students with CGPA given in student table lesser to calcuated CGPA



# Aggregation-Grouping

Aggregation allows you to apply calculation on values of column, and it will return a scalar value. Adding the GROUP BY Clause allows you to aggregate on groups of data, a scalar value will be returned for each group of data.

Some examples of Aggregate functions are given below.

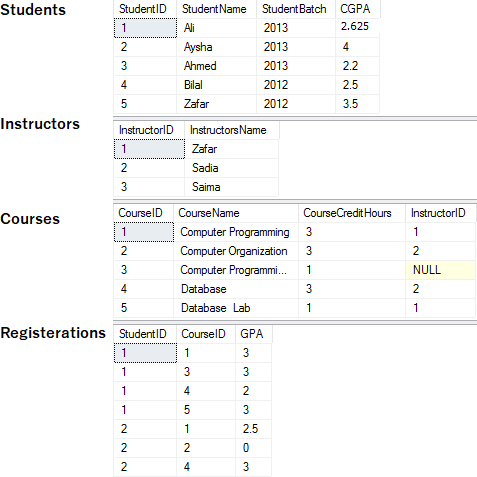
|  |  |  |
| --- | --- | --- |
| Aggregation Function Key work | How it works | No of Column Function can work on |
| AVG() | Returns the average of the values in a group. Null values are ignored. | Single column |
| COUNT() | Returns the number of items in a group. This function always returns  an int data type value | Single Column or List of Columns or \* |
| MAX() | Returns the maximum value in the  expression. | Single column |
| MIN() | Returns the minimum value in the  expression. | Single column |
| SUM() | Returns the sum of all the values in the expression. SUM can be used on  numeric columns only and it ignores all the NULL values. | Single column |

*Figure 1 Aggregation Functions*

Following is the syntax of Aggregation without grouping.

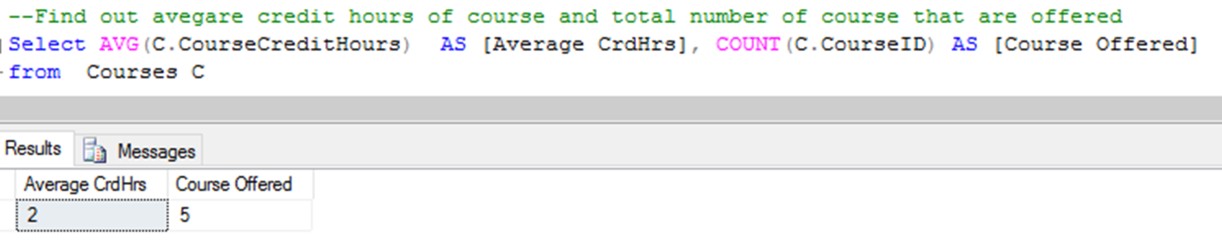
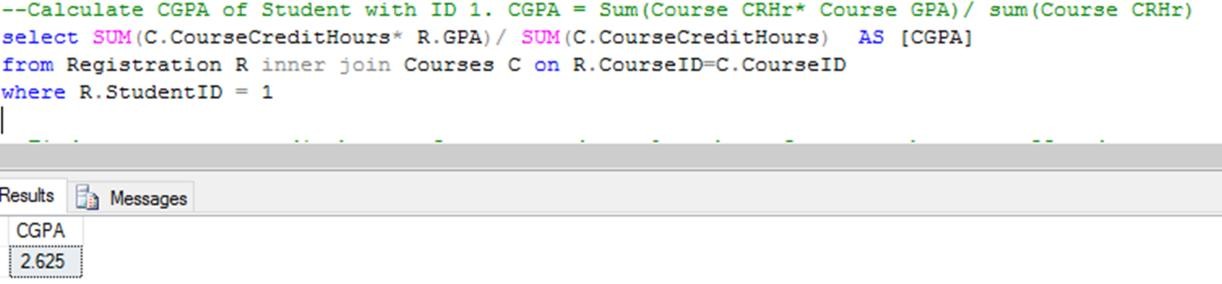
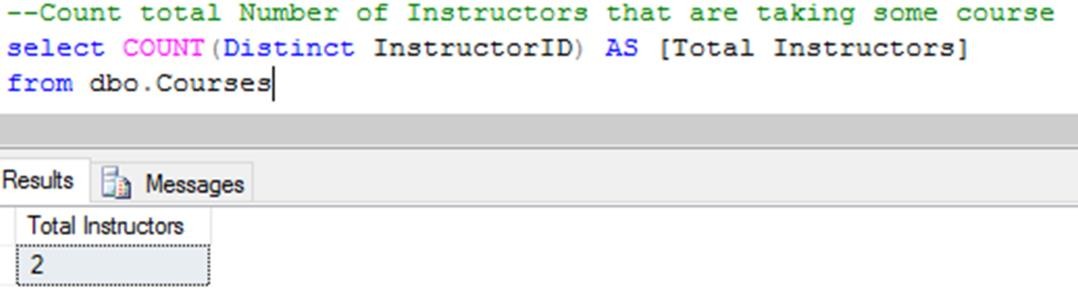
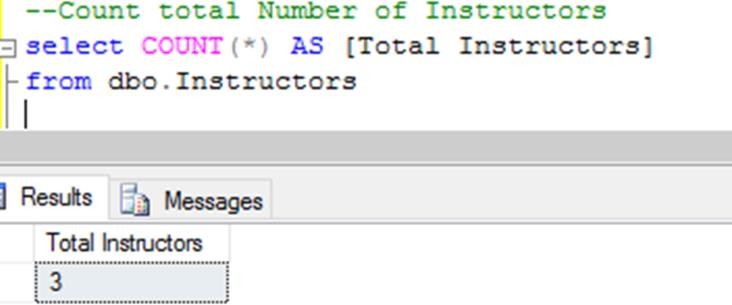
Select <AggregationFunction>(COLUMNs/Column) AS <AliasName> From <TableName>

Use the script (Lab4TryManual.sql Figure 1) to create database to try the following queries.



*Figure 2 University Database*





TRY THIS (Aggregation with Grouping)

\*\*NOTE THE DISTINCT KEY WORD. WHAT DOES IT DO? YOU CAN USE AGGREGATION AND JOING TOGETHER

USE MORE THAN ONE AGGREGATION FUNCTION IN SAME SELECT



**Grouping:**

Syntax:

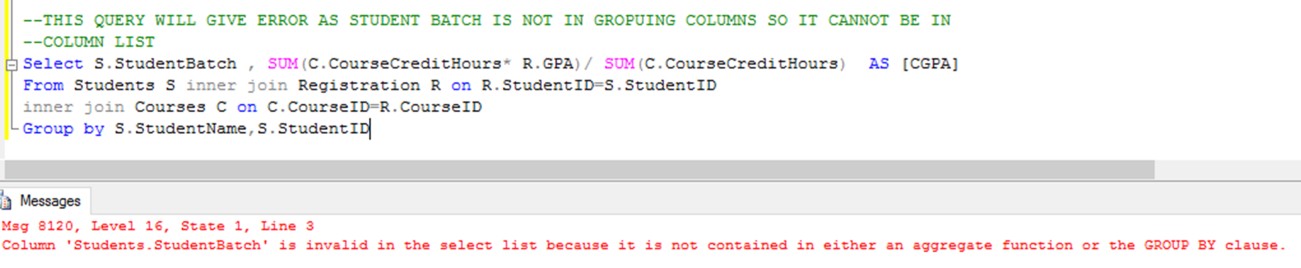
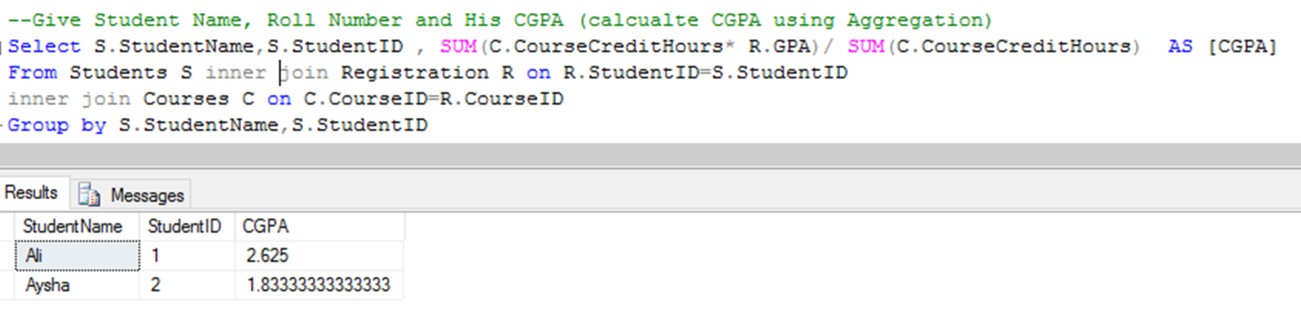
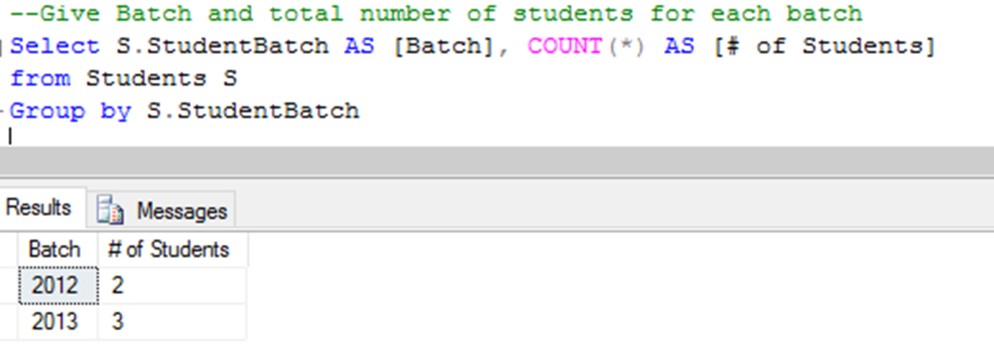
Select T.ColumnX, T.ColumnY Aggreation Function(Column/Columns) AS [Alias] from TableName T

Group by T.ColumnX, T.ColumnY -–comma seperated list of all the column of which

--groping is to be done

NOTE: ONLY THE COLUMNS THAT ARE USED IN GROUPING CAN BE USED IN SELECT CLAUSE

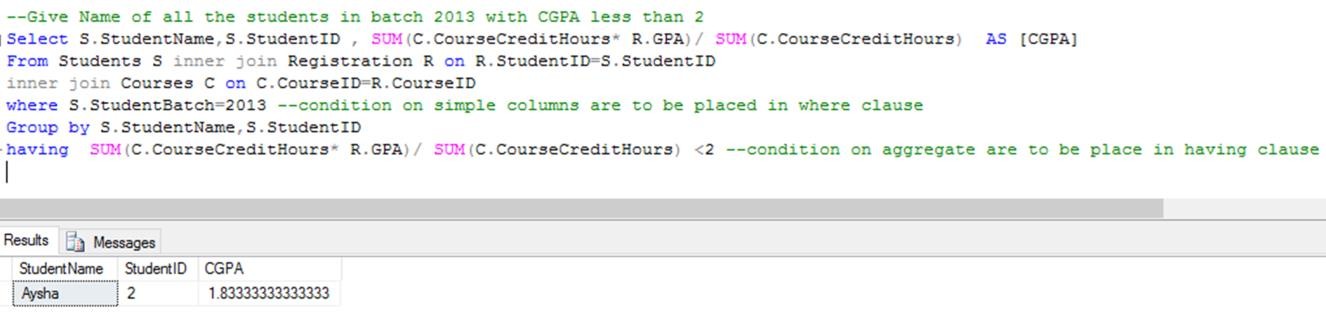
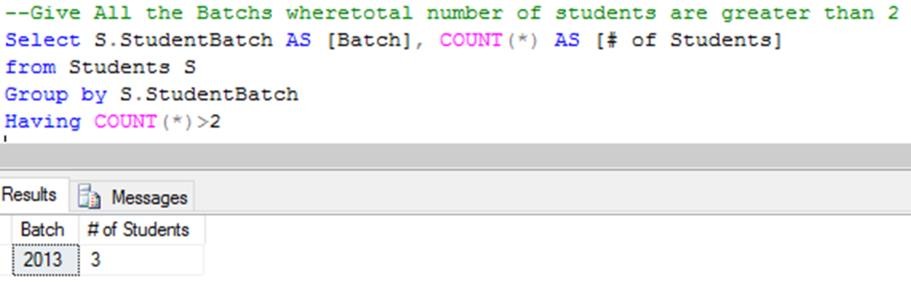
TRY THIS (Aggregate with grouping)





**Having Clause**

Having Clause allows us to filter the data based on the result of aggregation function, it’s the same as where clause except that we cannot use aggregate functions in where clause and we cannot use simple columns having clause.



Try this (aggregate group having)

NOTE: THE ORDER OF EACH CLAUSE IS TO BE MAINTAINED AS FOLLOW

1. SELECT (COMPULSORY)
2. FROM (COMPULSORY)
3. WHERE
4. GROUP
5. HAVING