|  |
| --- |
| **National University of Computer and Emerging Sciences** |
| **Lab Manual 5**  “Data Retrieval Select-from-where, Joins, Order by, Aggregate functions, Group by” |
|  |
| **Database Systems Lab** |
| **Spring 2021** |

Department of Computer Science

FAST-NU, Lahore, Pakistan

Contents

[2. Objective 2](#_Toc1409808)

[3. Pre-requisites 2](#_Toc1409809)

[4. SELECT-FROM-WHERE 3](#_Toc1409810)

[Most Basic Select: 3](#_Toc1409811)

[Retrieving certain Columns from Select 4](#_Toc1409812)

[Retrieving certain Rows from Select- WHERE CLAUSE 4](#_Toc1409813)

[Renaming Resulting Column 5](#_Toc1409814)

[5. Order by Clause 5](#_Toc1409815)

[TOP Clause 6](#_Toc1409816)

[6. Join Operation 6](#_Toc1409817)

[Inner Join: 6](#_Toc1409818)

[Left/Right/Full Outer Join 7](#_Toc1409819)

[Cross Join 9](#_Toc1409820)

[Joining More than two tables 9](#_Toc1409821)

[7. Aggregation-Grouping 10](#_Toc1409822)

[Grouping 12](#_Toc1409823)

[Having Clause 13](#_Toc1409824)

[8. Set operations 14](#_Toc1409825)

# Objective

* The purpose of this manual is to get stared with data retrieval queries, starting from Simple Select-From-Where, going towards Join operation, covering Order by clause and Aggregate functions, Group by.

# Pre-requisites

* Lab 3,4 manual, on how to get started with MS-SQL server
* How Select from Where clause work
* How Joining and all its type work
* How Order by clause works
* Aggregate functions, Group by

Task Distribution

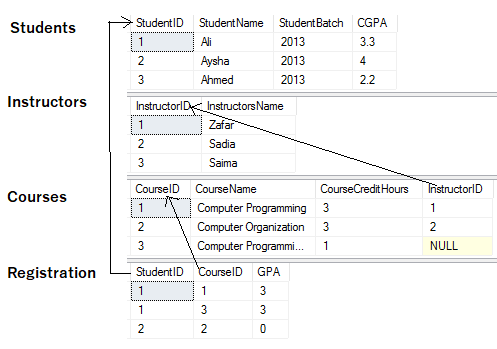
|  |  |
| --- | --- |
| Total Time | 170 Minutes |
| Select from where | 15 Minutes |
| Order by | 15 Minutes |
| Joining | 15 Minutes |
| Group by | 15 Minutes |
| Exercise | 90 Minutes |
| Evaluation | Last 20 Minutes |

# SELECT-FROM-WHERE

Select from where is equivalent to projection and selection in Relational Algebra, it will give output in form of a table.

The most basic select statement includes Select and from clause, and it will retrieve all columns and rows from the table.

We will use the following schema and database for the examples. Script to create this schema is given in Lab5Manual.sql file



## Most Basic Select:

SELECT \*

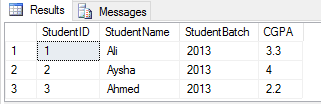
FROM <tableName>

\* after select means that all columns will be retrieved

Try this



Results



## Retrieving certain Columns from Select

To retrieve only certain columns, give a comma separated list of those columns after Select keyword

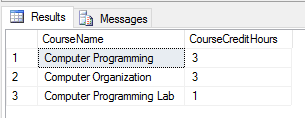
SELECT ColumnX, ColumnY, ColumnZ

FROM <tableName>

Try this



Results



## Retrieving certain Rows from Select- WHERE CLAUSE

Like Selection in RA, rows are filter in SQL using WHERE clause, rows that fulfill where clause conditions will be projected in result. Where clause can put condition on original columns of tables mentioned on from clause, or derived columns.

SELECT \*

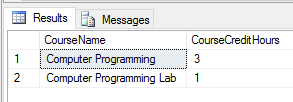
FROM <tableName>

where <conditions>

Try this



Results



## Renaming Resulting Column

You can rename a column in result by using AS keyword also called Alias. The scope of this renaming is only to that select query, this is useful in joining where more than one table have same column names.

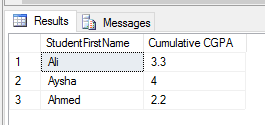
SELECT ColumnX as X , ColumnY as Y, ColumnZ

FROM <tableName> as Table1

Try this



Results



# Order by Clause

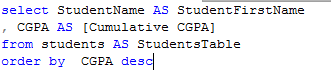
Order by clause is used to arrange the rows in ascending or descending order of one or more columns

SELECT ColumnX as X , ColumnY as Y, ColumnZ

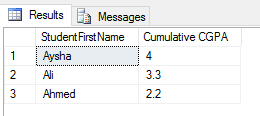
FROM <tableName> as Table1

ORDER BY ColumnX asc/desc, ColumnZ asc/desc

Try this



Results



## TOP Clause

Top n clause will give you first n rows from result instead of all the rows.

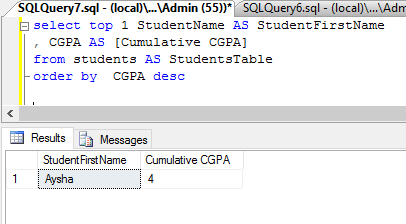
SELECT TOP <n> \*

FROM <tableName>

where <conditions>

Order by <column Name> asc/desc

Try this



# Join Operation

We will use the following tables in examples

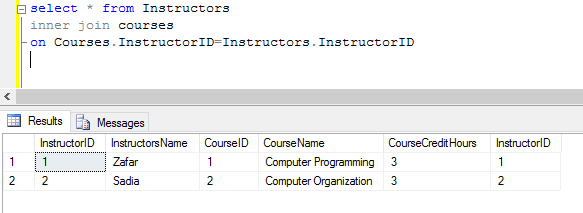
## Inner Join:

Returns only those rows that match in both tables.

SELECT \*

FROM <table1> inner join <table2>

ON <Joining Condition>



## Left/Right/Full Outer Join

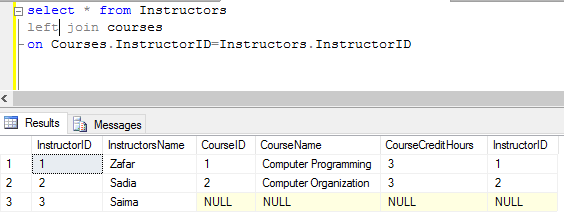
Left Join: Returns all the rows of Left table with corresponding row or null row of right table

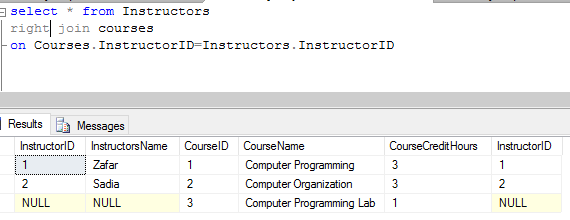
Right Join: Returns all the rows of Right table with corresponding row or null row of Left table

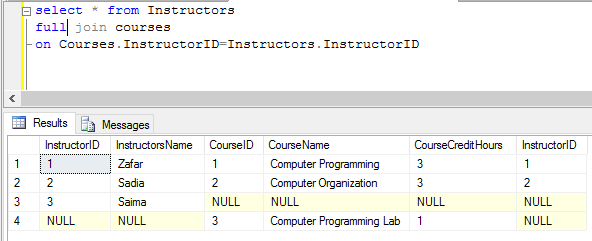
Full Join: Union of Left and Right Outer join

SELECT \* FROM <table1> Left/Right/Full join <table2> ON <Joining Condition>

Try these





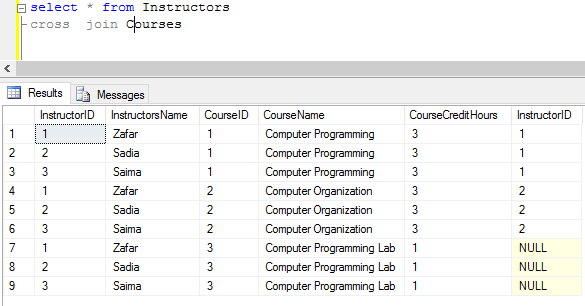


## Cross Join

It’s a cross product of two tables, no ON condition is required here

SELECT \* FROM <table1> cross Join <table2>

Try this



## Joining More than two tables

SELECT \*

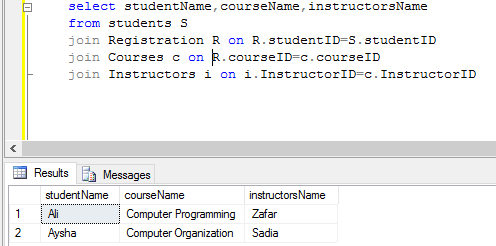
FROM <table1>

Left/Right/Full/Inner join <table2> ON <Joining Condition>

Left/Right/Full/Inner join <table3> ON <Joining Condition>

Left/Right/Full/Inner join <table4> ON <Joining Condition>

Try this



# 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 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.

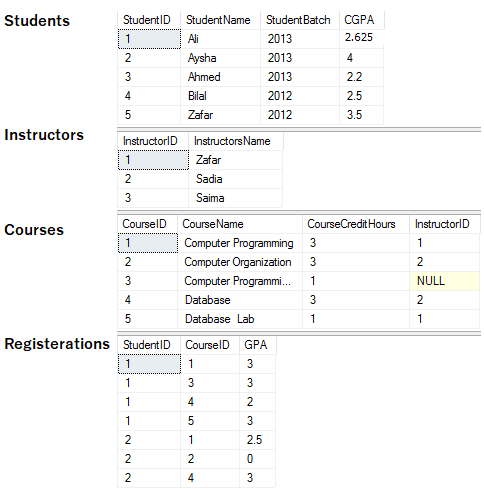
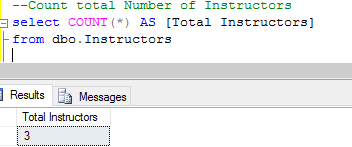
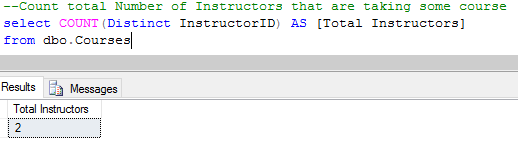


Figure University Database

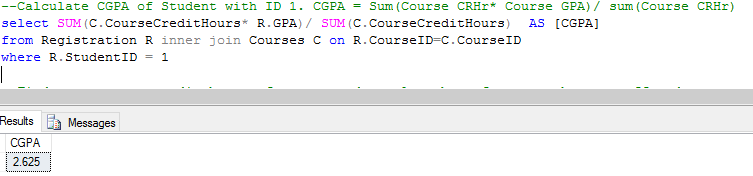
TRY THIS (Aggregation with Grouping)



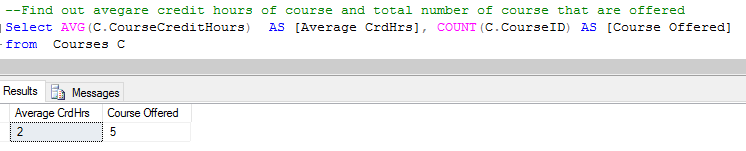


\*\*Note the Distinct key word. What does it do?

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]

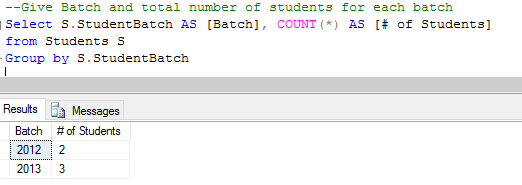
from TableName T

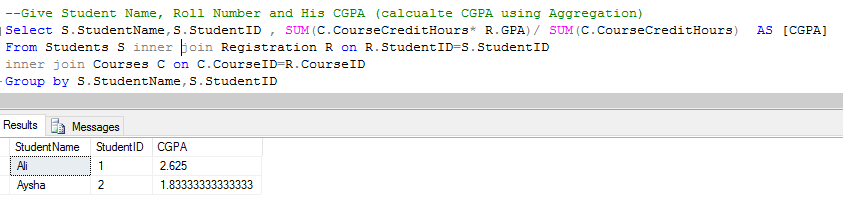
Group by T.ColumnX, T.ColumnY -–comma seperated list of all the column of which

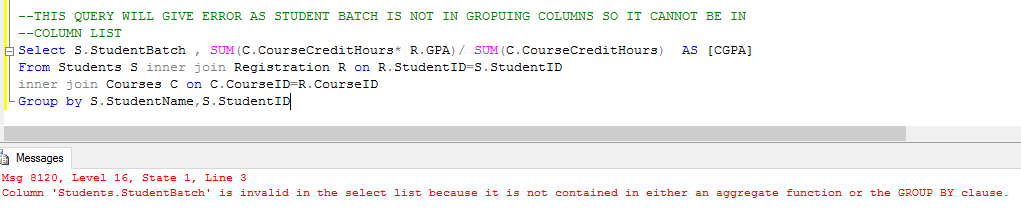
--groping is to be done

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

TRY THIS (Aggregate with grouping)



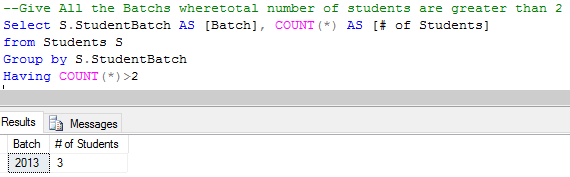


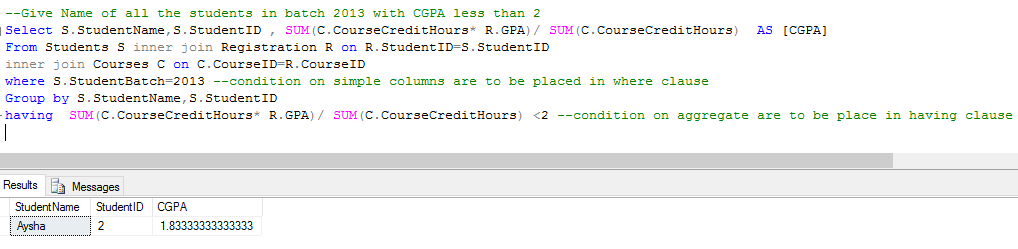


## Having Clause

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

# Set operations

Result of two (or more) select queries can be combined using Set operations such as UNION, INTESECT, EXCEPT.

Syntax

Select ColumnX, ColumnY

From Table1

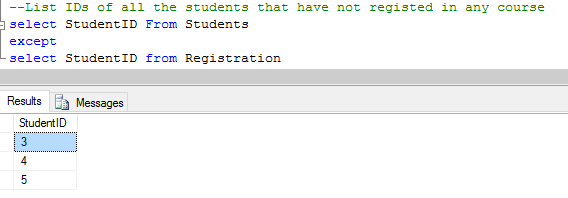
Union/Intersect/Except

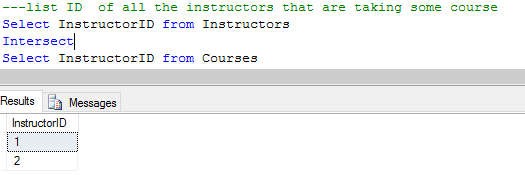
Select ColumnA, ColumnB

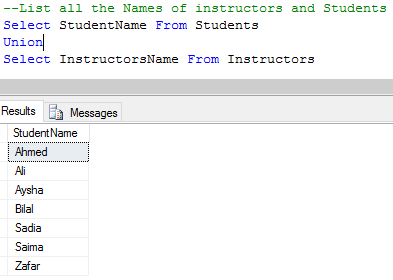
From Table2

NOTE: The output of first select query should have same number and type of column as of second select query.

Try this –Set operations







Try this- error to look out for in set operations

