

DATABASE MANAGEMENT SYSTEM (LAB)

Lab Manual

[Fall 2019]

|  |  |
| --- | --- |
| Student Name: Sabah Iqbal |  |
| Student Id: 14406 |  |

|  |  |
| --- | --- |
| Prepared By: *Dr. Noman Islam* |  |
| Instructor: *Dr. Noman Islam* |  |

**LIST OF EXPERIMENTS**

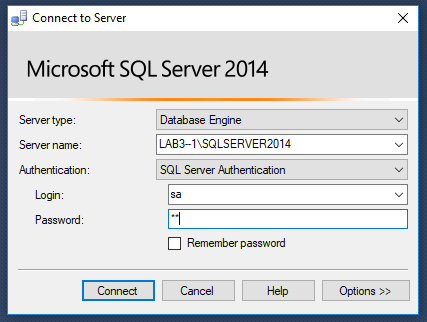
|  |  |  |  |
| --- | --- | --- | --- |
| **S. No** | **Date** | **Experiment** |  |
| **1** | 24/09/19 | To study and explore basic DDL commands in SQL: create, alter, drop |  |
| **2** | 01/10/19 | To study and explore basic SQL commands: select, insert, delete, update |  |
| **3** | 08/10/19 | To study and explore select command in SQL |  |
| **4** | 15/10/19 | To study and implement advanced SQL commands: having and group by |  |
| **5** | 22/10/19 | To study and implement various types of joins in SQL |  |
| **6** | 29/10/19 | To study and implement various types of joins and views |  |
| **7** | 12/11/19 | To study and implement data control language |  |
| **8** | 18/11/19 | To study, understand and implement NoSQL databases using MongoDB |  |
| **9** | 10/12/19 | To study and implement connectivity with MongoDB using Java |  |

**Lab 1: To study and explore basic DDL commands in SQL: create, alter, drop**

Structured Query Language) is a domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS). In this lab, we will study basic data definition language of SQL.

**Lab Tasks:**

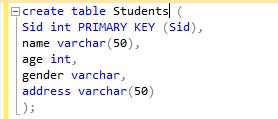
1. Login to the SQL server with the default user id and password.



1. Create a database called School using the CREATE DATABASE command

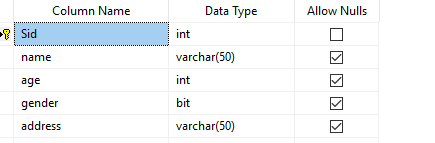


1. Now create a table Student with fields sid (int), name (varchar), age(int), gender(varchar), address(varchar); set sid as the primary key

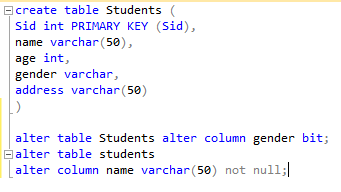


1. Alter the table Student to change the data datatype of gender to bit

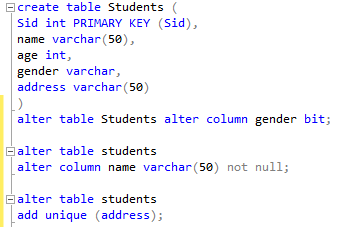




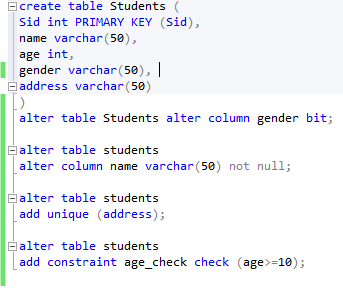
1. Add the constraint to name to be not null

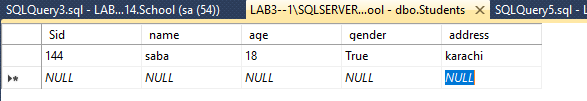


1. Add the constraint address to be unique



1. Add the constraint age to be always greater than or equal to 18

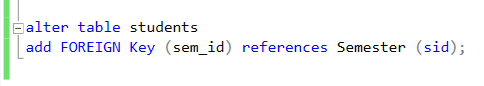


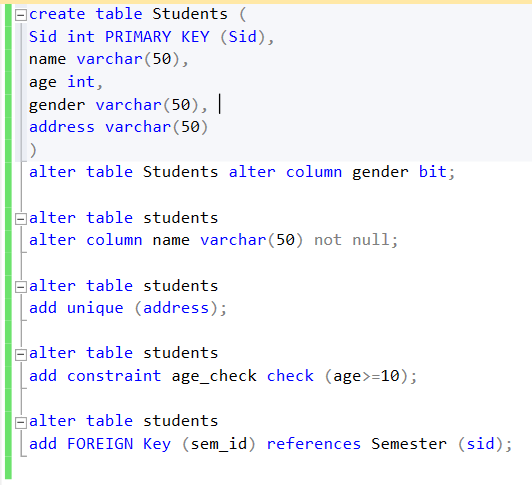


1. Create a table Semester with sid (int) and sdate(date). Set sid to be primary key and make sure it is auto incremented



1. Alter the table Student to add one more field semester with foreign key constraint





1. Drop all the tables
2. Drop the database

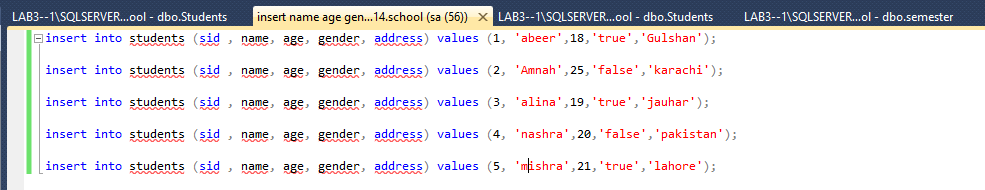
**Lab 2: To study and explore basic SQL commands: select, insert, delete, update**

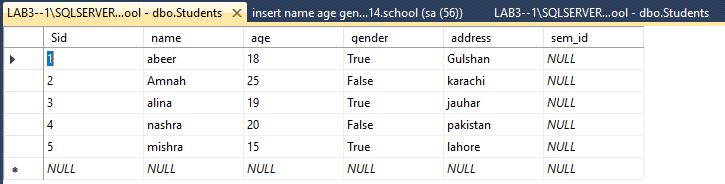
In this lab, we will study basic DML commands used in SQL.

**Lab Tasks:**

1. Assuming you have the tables created in previous lab, insert five rows in the Student table

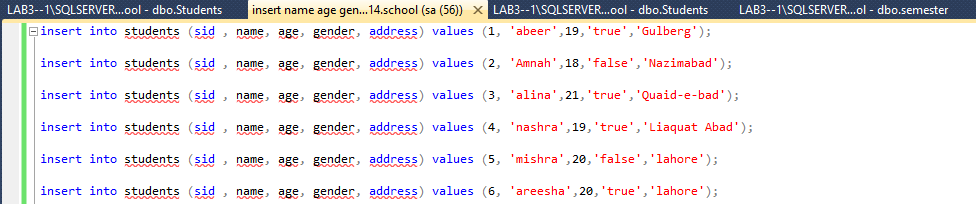
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Name** | **Age** | **Gender** | **Address** |
| 1 | Ali | 19 | True | Gulberg |
| 2 | Zoya | 18 | False | Nazimabad |
| 3 | Rashid | 21 | True | Quaid-e-bad |
| 4 | Ahmed | 19 | True | Liaquat-e-bad |
| 5 | Naima | 20 | False | Lahore |





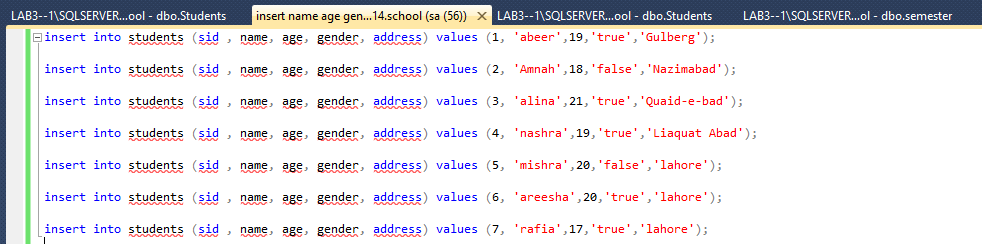
1. Try to insert the following new record. Do you face any issue?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | Asghar | 20 | True | Lahore |

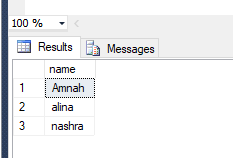


1. Try to insert the following record. Do you face any issue?

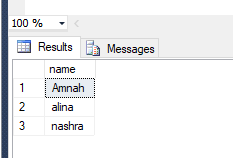
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | Aleem | 17 | True | Lahore |



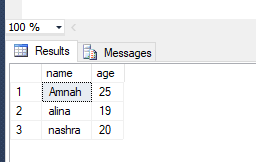
1. Now, select all the students from the Student



1. Select all the female students from the Student



1. Select all the distinct age from Student table



1. Update the address of Zoya to Hostel
2. Find all students whose age is greater than 20
3. Count all the students less than 20 years
4. Find all students whose name starts with ‘A’
5. Select all the students order by age
6. Delete the students who age is less than 19
7. Alter the table Student to add a new field: Status of type varchar
8. Set the status of all the students to active

**Labe 3: To study and explore select command in SQL**

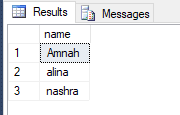
In this lab, we will explore the select SQL command in detail. We will see how we can use select command to see different columns in output, how can we apply aggregate functions etc.

**Lab Tasks:**

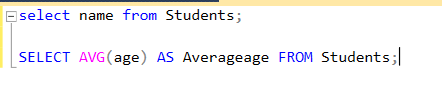
Consider the tables created in previous labs, now perform the following tasks:

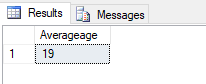
1. Select the names of all the students.





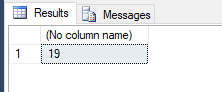
1. Find the average age of the students.





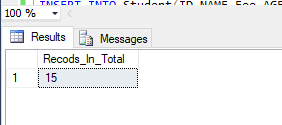
1. Provide the average age an alias of ‘avg\_age’.



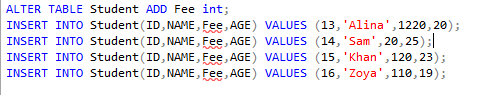


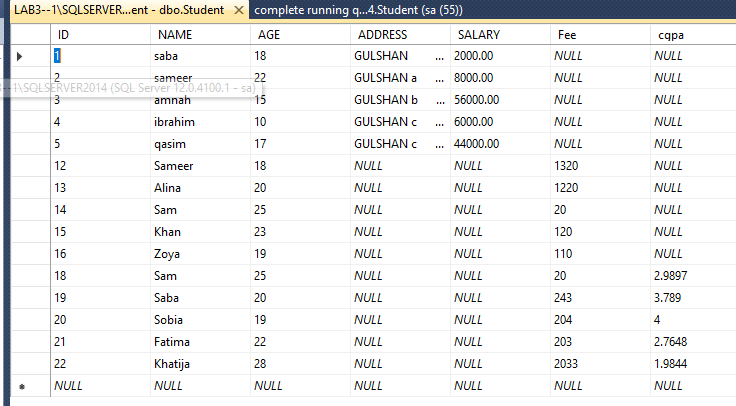
1. Count the number of records in students table. Provide it a suitable alias.





1. Alter the student table to add a fee column of data type int.





1. Find the total sum of fee.
2. Find the maximum age of students.
3. Find all the details of student whose age is maximum.
4. Find the names of student capitalized.
5. Alter the student table to add a cgpa column of appropriate data type.
6. Select all the records of students. The CGPA should be rounded to two decimal places.
7. List all the details of students sorted by name and age in ascending order.
8. Search the students whose name ends with ‘d’.
9. Find all students whose age lies between 20 and 25.
10. Find the name of student whose lives in Liaquat-e-bad.

**Lab 4: To study and implement advanced SQL commands: having and groupby**

Group by is used to group to all the records in a relation together for each and every value of a specific key(s) and then display them for a selected set of fields the relation.

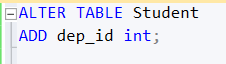
An index is an ordered set of pointers to the data in a table. It is based on the data values in one or more columns of the table. SQL Base stores indexes separately from tables. An index provides two benefits:

* It improves performance because it makes data access faster.
* It ensures uniqueness. A table with a unique index cannot have two rows with the same values in the column or columns that form the index key.

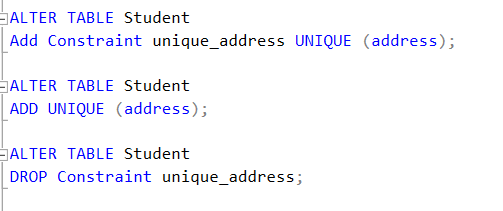
**Lab Tasks:**

Assuming the relations created in previous labs. Perform the following tasks.

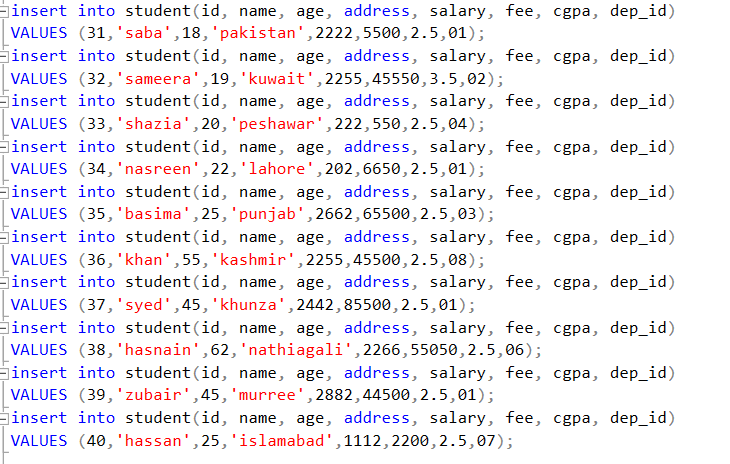
1. Add a field dept id in Student table.



1. Drop the unique constraint on address of the Student table



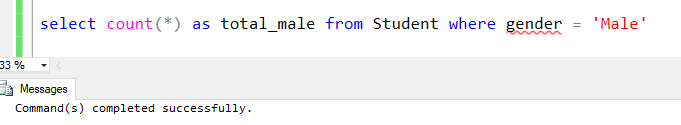
1. Using INSERT clause, enter the information about 10 male students and 10 female students in Student table.

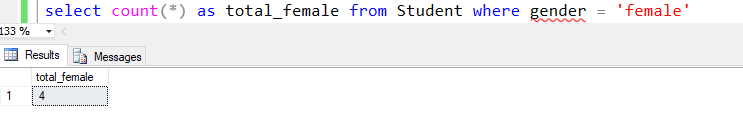


1. Create an index on field address for the Student table.

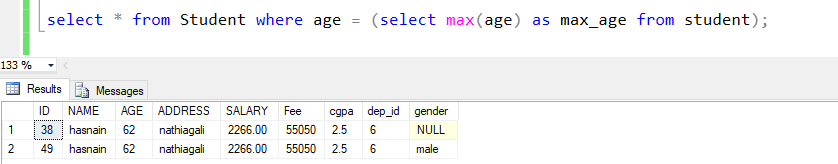


1. Display the number of males and females. Provide an alias for computed fields.

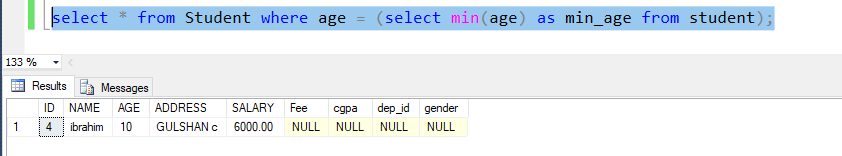




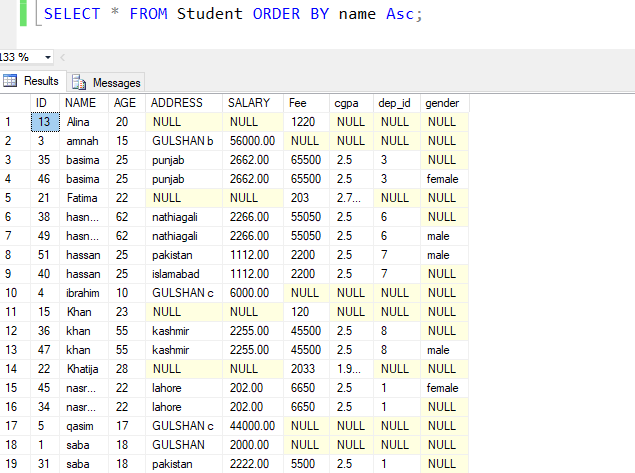
1. Display the average age of males and females. Provide an alias for computed fields.



1. Display the youngest male and female. Provide an alias for computed fields.



1. Display the details of students sorted by Name.



1. Find the number of males and females whose age is greater than 20.
2. For each department, retrieve the department number, the number of students in the department, and their average age.
3. Find the number of male students with dept\_Id=1.
4. Find the number of students group by address.
5. Find the number of students living in Gulshan-e-Iqbal.
6. Find the address where more than two students lives.

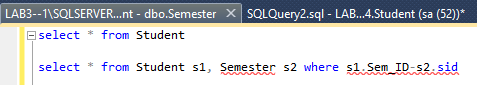
**Lab 5: To study and implement sub-queries, various types of joins in SQL**

The query within another is known as a sub query. The purpose of a join concept is to combine data spread across tables. A join is actually performed by the ‘where’ clause which combines specified rows of tables.

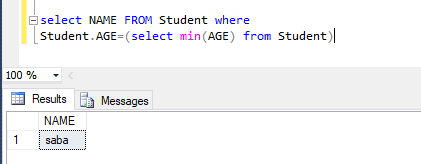
**Lab Tasks:**

Consider the student and semester tables created in previous labs. Now, perform the following tasks:

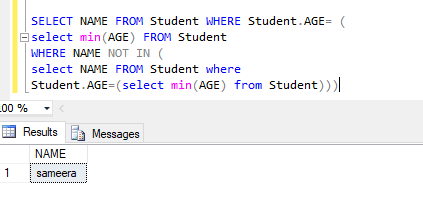
1. Display the list of all the students along with semester details.



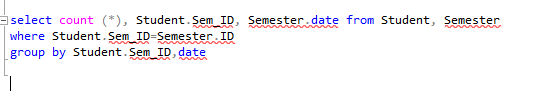
1. Find the youngest student of the class.



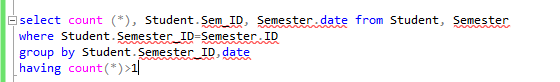
1. Find the second youngest student of the class.



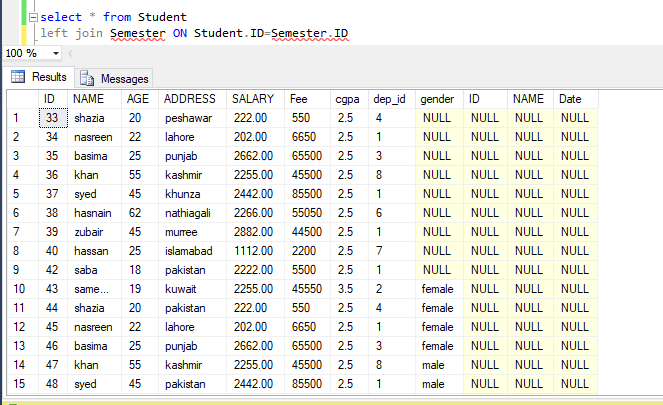
1. Find the top 5 oldest students of class and insert into a new table named junior.
2. Find the number of students in various semesters.



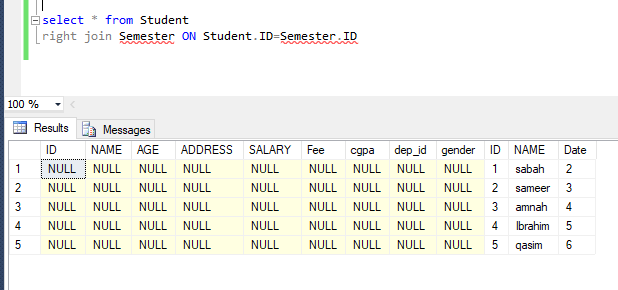
1. Retrieve the details of the semester in which there are more than two students.



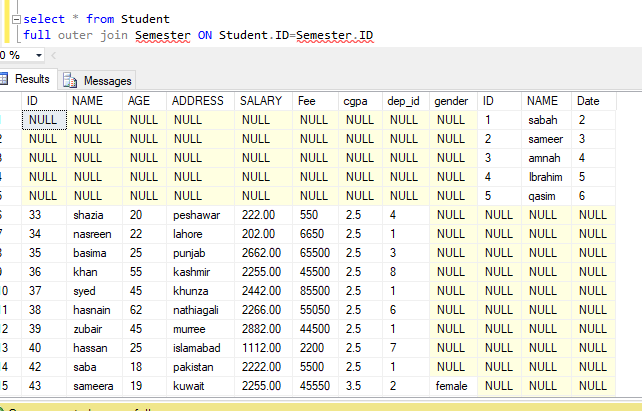
1. Retrieve the details of all the students along with semester. If a student is not enrolled in a semester, the details of the students should be displayed.



1. Retrieve the details of all the semesters along with enrolled students. If a semester has no enrolled students, the details of the semester should be displayed.



1. Retrieve the details of students with semester irrespective of if the students are enrolled in a semester or if a semester has any students or not.



1. Find the names of students who are younger than ‘Saleem’
2. Find the students who are neighbors of ‘Kaleem’.
3. Find the names of all students who are older than average age of students.
4. Find the details of all students who are in semester 1.

**Lab 6: To study and implement data control language**

Data Control Language statements are used to create roles, permissions, and referential integrity as well it is used to control access to database by securing it. DCL Commands are Grant and Revoke.

**Lab Tasks:**

1. Create a user *‘student’* identified by password ‘*abc*’.
2. Create a user *‘administrator’* identified by password ‘*abc*’.
3. Check if the user has been created using select command.
4. Check what permissions are granted to the user *‘student’*.
5. Grant the *‘student’* SELECT privileges on table student, semester.
6. Grant all the privileges to *‘administrator’*.
7. Now, login to the database as ‘*student*’.
8. Select all the records from Student table.
9. Try to insert a new record in Student.
10. Now, login to the database as ‘*administrator*’.
11. Select all the records from table Student.
12. Try to insert a new record in Students.
13. Now, login as user ‘*sa*’.
14. Revoke the select privileges from ‘*student*’.
15. Grant the privilege to select only name, age and gender.
16. Change the password of user *‘student’*.
17. Now login as *‘student’*.
18. Try to select the record from Student table.
19. Login again as ‘*sa*’.
20. Create a role guest.
21. Grant the privilege ‘select’ to role guest.
22. Grant the student role of guest.

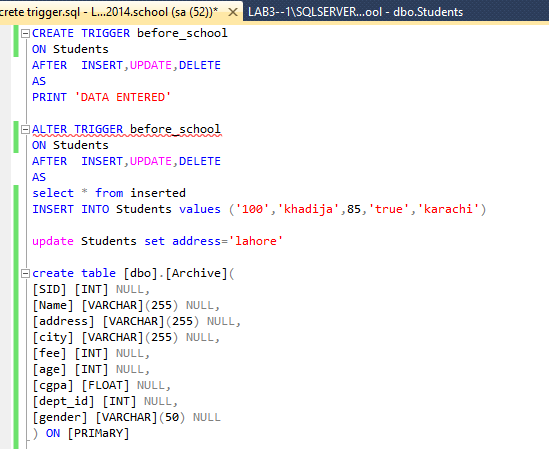
**Lab 7: To study and implement triggers in SQL**

Triggers are nothing but the procedures/functions that involve actions and fired/executed automatically whenever an event occurs such as an insert, delete, or update operation or pressing a button or when mouse button is clicked. In this lab, we will implement triggers in SQL.

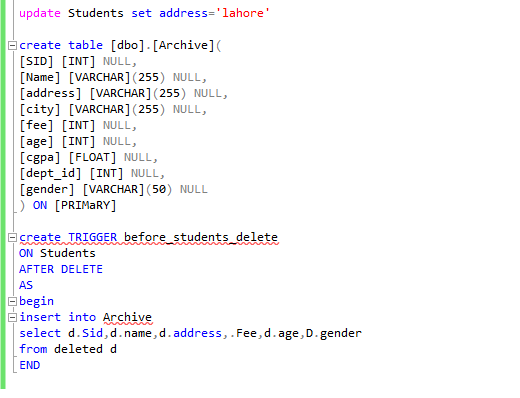
**Lab Tasks:**

Popup DML operations

1. Create a trigger on Student relation, such that before a new record is inserted, updated or deleted, an appropriate console message is displayed.



1. Create a trigger on Semester relation, such that after a new record is inserted, updated or deleted, an appropriate console message is displayed.



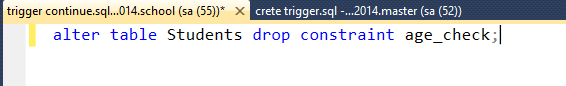
1. Now try inserting some data to Student and Semester relation using SQL.
2. What behavior did you observe?

Maintaining an archive table

1. Create a relation Archive. This relation will to maintain a history of old students.
2. Create a trigger on Students relation such that on delete, the records are inserted to Archive table.
3. Now try removing some data from Student relation?
4. Is the data inserted into Archive relation?

Validating data before insert and update

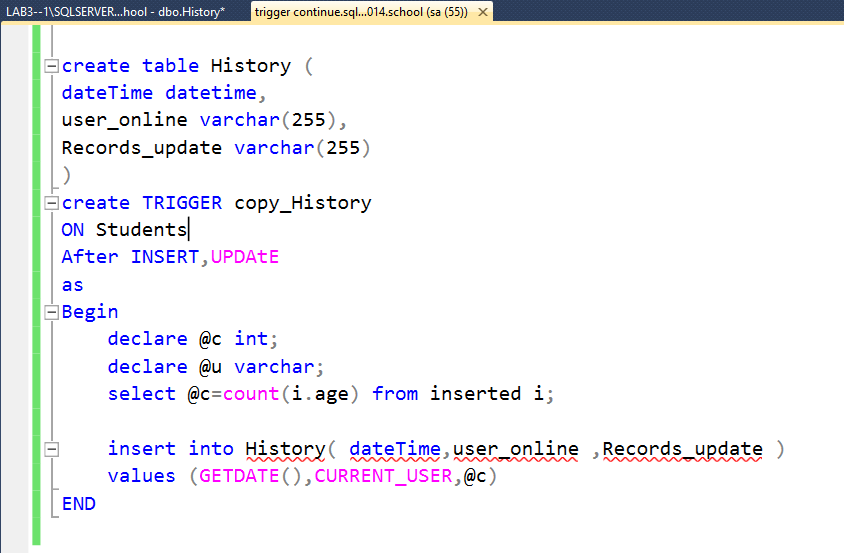
1. Drop the age constraint on Students relation.



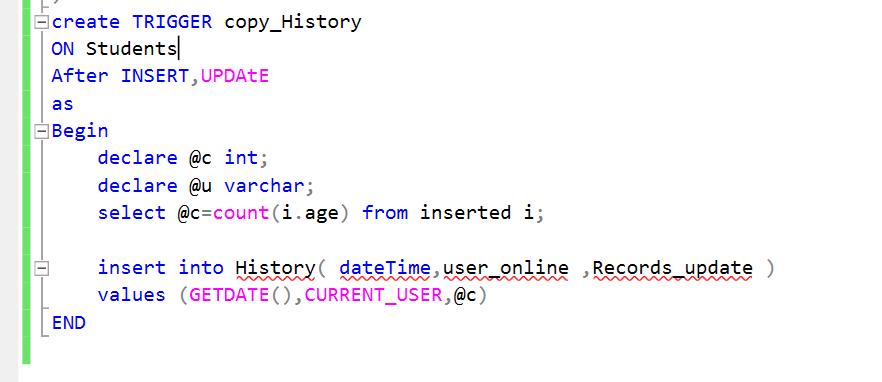
1. Create a trigger that checks if the age provided to Student is valid or not before insertion and updating. If the age is less than 18, raise an error. Also outputs the error to console. **imp**
2. Now try inserting and updating data with some valid and invalid age values.
3. What behavior did you observe?

Maintaining count of Updated records

1. Create a relation History that maintains the date/time, user and number of records updated.



1. Now create a trigger that is executed before Student relation is updated, and inserts the current date/ time, user and number of records updated to History relation.



Removing triggers

1. Now drop all the triggers you have created from database.

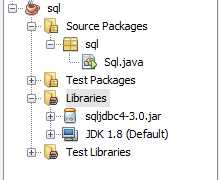
**Lab 8: To study and implement Java Database Connectivity (JDBC)**

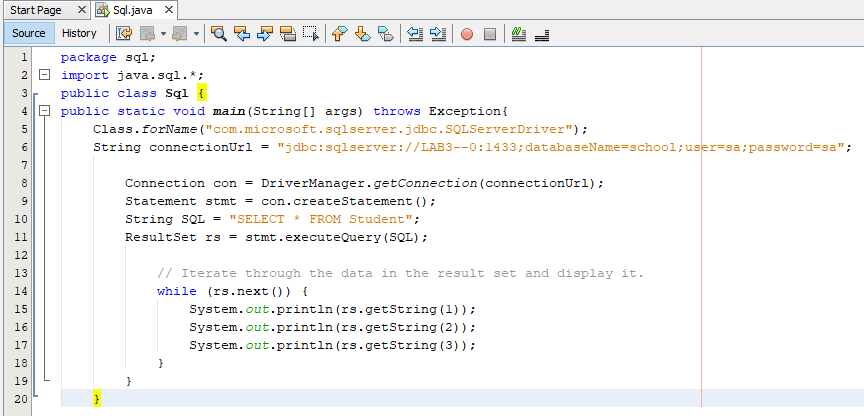
JDBC stands for Java Database Connectivity. JDBC is a Java API to connect and execute the query with the database. It is a part of Java SE (Java Standard Edition). JDBC API uses JDBC drivers to connect with the database. There are four types of JDBC drivers:

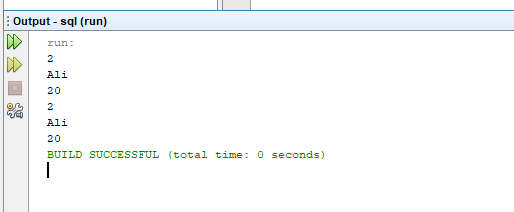
* JDBC-ODBC Bridge Driver,
* Native Driver,
* Network Protocol Driver, and
* Thin Driver

In this lab, we will connect to SQL Server using JDBC and try to execute SQL queries.

**Lab Tasks:**

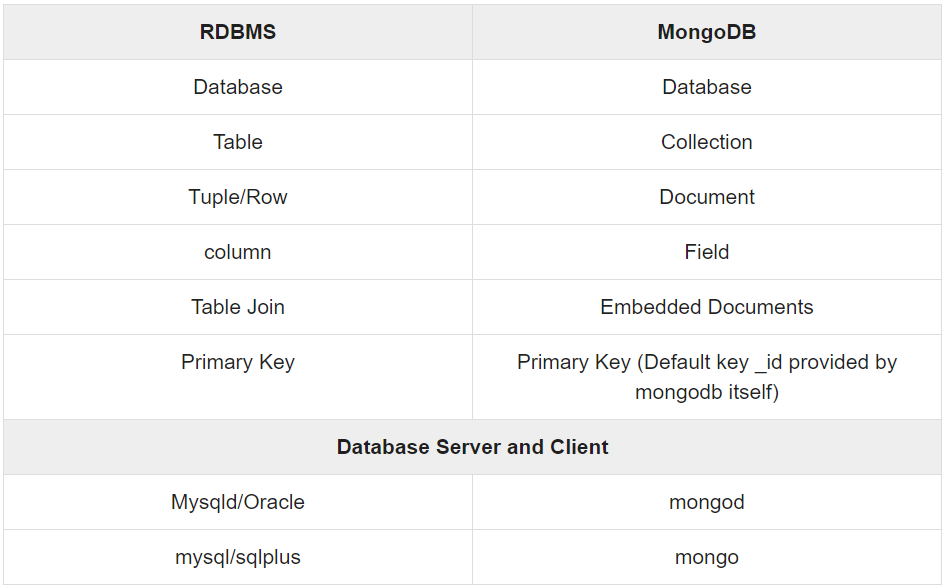
1. Create a database Connection, specifying the appropriate connection string.
2. Create a Statement.
3. Fetch all the records from Students table. Display the record on the console.
4. Insert a record into the Students table with following details: Id=7, name=’Rashid’, age=20, address=’Gulberg’.
5. Delete the student with Id=1 using both Statement and Prepared Statement.
6. Create a stored procedure to insert a student record. Check if the student is above or equal to age 18 before inserting the record.
7. Now, using Callable Statement, call the stored procedure from Java program. Try to insert a student record below 18 and one record of above 18. What did you notice?





**Lab 9: To study, understand and implement NoSQL databases using MongoDB**

MongoDB is an open-source document database and leading NoSQL database. MongoDB is written in C++. MongoDB can be used for creating and deploying a highly scalable and performance-oriented database.



**Lab Tasks:**

1. Start the mongodb server using the following command:

mongod.exe --dbpath "d:\set up\mongodb\data"

1. Start the mongodb client with the following command:

Mongo.exe

1. To switch to a database type: use studentdb.
2. To view your current database, type the command: db.
3. To check list of databases, type show dbs.
4. Create a student collection. with the following details: name, id, age, cgpa, address, using db.createCollection().
5. Insert a student record using command db.student.insert().
6. See the current collections with command: show collections.
7. Now search for a student with the address “Karachi” using db.student.find().
8. Find all the students whose name is Ali or whose age < 20.
9. Find all the students whose name is Asif and whose age < 20.
10. Update the address of ‘Ali’ to Pindi.
11. Remove the student less than 18 years of age.

Stored Procedure:

java

public class JavaApplication15 {

public static void main(String[] args) throws Exception {

Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");

Stringconnectionurl="jdbc:sqlserver://LAB3--0:1433;databaseName=School;user=sa;password=sa";

Connection con=DriverManager.getConnection(connectionurl);

Statement st=con.createStatement();

String query="Select \* from Student";

ResultSet rs=st.executeQuery(query);

// String query="INSERT INTO Student VALUES(10,'rashid1','25',1,'peshawar')";

// st.executeUpdate(query);

while(rs.next()) {

System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getString(3));

}

String query1 = "{ call Stinsert(?,?,?,?) }";

int id=2;

Console c = System.console();

CallableStatement stmt = con.prepareCall(query1);

stmt.setInt(1, 12);

stmt.setString(2,"bibi");

stmt.setInt(3, 34);

stmt.setString(4, "lahroe");

stmt.executeUpdate();

// PreparedStatement stmt=con.prepareStatement("delete from Student where sid=?");

// stmt.setInt(1,4);

//System.out.println(i+" records deleted"); con.close();

}

}

USE [School]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Stinsert] Script Date: 12/3/2019 4:53:03 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Manoj Kalla

-- Create date: 20-Nov-2047

-- Description: To create a new member

-- =============================================

ALTER PROCEDURE [dbo].[Stinsert]

@MemberId int,

@MemberName varchar(50),

@Memberage varchar(25),

@Memberaddress varchar(15)

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

Insert into Student (sid,sname,age,address)

Values (@MemberId,@MemberName,@Memberage, @Memberaddress)

END

Query:

package databaseconnectivity;

import java.sql.\*;

public class DatabaseConnectivity {

public static void main(String[] args) throws Exception

{

Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");

String connectionurl="jdbc:sqlserver://LAB3--0:1433;databaseName=School;user=sa;password=sa";

Connection con=DriverManager.getConnection(connectionurl);

//here sonoo is database name, root is username and password

Statement stmt=con.createStatement();

/\*

stmt.executeUpdate("INSERT INTO Student VALUES (10, 'Saim', '22', 1, 'Pishawar')");

\*/

/\*

ResultSet rs=stmt.executeQuery("select \* from Student");

while(rs.next())

System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getString(3)+ " "+rs.getString(4)+ " "+rs.getString(5));

\*/

/\*

PreparedStatement stmt=con.prepareStatement("delete from Student where sid=?");

stmt.setInt(1,10);

int i=stmt.executeUpdate();

System.out.println(i+" records deleted");

\*/

// String sql = "INSERT INTO Student " +

// "VALUES (4, 'Ahmed', '22', 1, 2, 'Karachi')";

// while(rs.next())

// System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getString(3));

con.close();

}

}

**Lab 10: To study and implement connectivity with MongoDB using Java**

Java is one of the most popular programming languages in the MongoDB Community. For new users, it’s important to provide an overview of how to work with the MongoDB Java driver and how to use MongoDB as a Java developer.

In this lab, we’re going to give you a guide on how to get started with using MongoDB.

**Lab Tasks:**

1. Download the <http://central.maven.org/maven2/org/mongodb/mongo-java-driver/> JAR file of mongo driver and add it to your classpath.
2. Start a local mongo database server.
3. Connect to the local machine.
4. Now, create a MongoClient object in Java.
5. Create a database named ‘hospital’.
6. Insert a doctor record using with the following details: dId, dName, dAge, dSalary, dQualification.
7. List all the current collections.
8. Now search for a doctor with the qualification “FCPS”.
9. Find all the doctors whose name is Ali or whose age < 30.
10. Find all the doctors whose name is Asif and whose age < 30.
11. Update the salary of ‘Dr. Ali’ to PIMS Hospital.
12. Remove all the doctors less than 30 years of age.

CODE:

package iqra.iqra.iqra.iqra;

import com.mongodb.client.MongoClient;

import com.mongodb.client.MongoClients;

import com.mongodb.client.MongoDatabase;

public class App

{

public static void main( String[] args )

{

// System.out.println( "Hello World!" );

// Creating a Mongo client

MongoClient mongoClient = MongoClients.create();

MongoDatabase database = mongoClient.getDatabase("test");

for(String name: database.listCollectionNames()){

System.out.println(name);

}}}