# Database Application Development of Hospital Emergency Department Using Object Oriented Programming

**By: Love Patel & Dharmik Patel**

## Time Estimate

Setting Up the SQL File : 15 min

Testing the Application with Simulation Data : 30 min Developing CRUD Interface : 30 min

# Database Application Development of Hospital Emergency Department Using Object Oriented Programming (100 pts total)

Alfie Medical Hospital has hired you as a software engineer to develop an information system for their Emergency Department to track their patients. This department has 4 visit rooms, 6 additional visit beds, a radiology area, triage, registration, and billing. Your team is assigned to observe the emergency department and create a database of the system and develop a database application of the system. You must create the database and test it before developing a database application. After testing the database, you need to work on a database application by modifying a given database application code template. The following figure shows a simulation model of the hospital emergency department, which we will treat as the "real" system in this assignment.


You will submit two documents for this assignment. Your project export (name your project HospitalCRUD) as HospitalCRUD.zip and your report Assignment2.docx or Assignment2.pdf. Your project export should include your existing database (created by flyway) and all the other necessary source code/libraries to compile and run the project successfully. It is a good idea to test your exported project zip file on another computer before submitting it and see if it is working correctly. In your report, you

must show your work with screenshots and descriptions/captions about those screenshots.

**Ventilator:** Below is a screenshot of a ventilator in the system. Some beds have a ventilator next to it. You need to identify the beds that has ventilator in the database.

By using the AddressBookDAO(fly) project as a template, create a CRUD application for an Emergency Service database (HospitalDb) with two tables.

Patient table

- PatientID (number, PK)
- FirstName (varchar(30), not null)
- LastName (varchar(30), not null)
- BedID (number, foreign key, not null)
- DateTimeBedVisited (datetime, not null)
- RadiologyVisited (boolean)

Bed table

- BedID (number, PK)
- WithVentilator (boolean, not null)

**Part 1:** Setting Up the SQL File (5 pts)

Write a SQL code to create these two tables in a dummy database using NetBeans JavaDB network server (name your database Hospitaldb). Put your screenshots showing that these two tables are created successfully by running your SQL code. **5 points.**

Connection: HospitalDB

```
1   CREATE TABLE PATIENT_TABLE(
2   PATIENTID NUMERIC PRIMARY KEY NOT NULL,
3   FIRST_NAME VARCHAR(30) NOT NULL,
4   LAST_NAME VARCHAR(30) NOT NULL,
5   BED_ID NUMERIC NOT NULL,
6   DATE_TIME_BED_VISITED TIMESTAMP NOT NULL,
7   RADIOLOGY_VISITED BOOLEAN,
8   FOREIGN KEY (BED_ID) REFERENCES BED_TABLE(BED_ID)
9   );
10
```

Output

Java DB Database Process | SQL 2 execution

```
Executed successfully in 0.053 s.
no rows affected.
Line 1, column 1

Execution finished after 0.068 s, no errors occurred.
```

SQL 1 [HospitalDB]

Connection: HospitalDB

```
1   CREATE TABLE BED_TABLE(
2   BED_ID NUMERIC PRIMARY KEY,
3   WITH_VENTILATOR BOOLEAN NOT NULL
4   );
5
```

Output

Java DB Database Process | SQL 1 execution

```
Executed successfully in 0.093 s.
no rows affected.
Line 1, column 1

Execution finished after 0.439 s, no errors occurred.
```

**Part 2:** Testing the Application with Simulation Data (75 pts)

1. Run the Hospital Emergency Department simulation and record 3 patients and their movement. Be sure to record the datetime when the patient visits the bed (check if there is ventilator as well next to bed) and check if they visit the radiology before leaving the hospital in the simulation. **5 points.**

SELECT * FROM HOSPITAL.PA... ×

Max. rows: 100 Fetched Rows: 3 Matching Rows:

| # | PATIENTID | FIRST_NAME | LAST_NAME | BED_ID | DATE_TIME_BED_VISITED | RADIOLOGY_VISITED |
|---|---|---|---|---|---|---|
| 1 | 177 | Robin | Mathew | 2 | 2011-01-03 07:00:00.0 | ☐ |
| 2 | 174 | Dev | Peterson | 1 | 2011-01-03 07:02:11.0 | ☐ |
| 3 | 170 | Dhormamu | Batman | 3 | 2011-01-03 07:05:08.0 | ☐ |

2. Insert the 6 beds from the simulation and identify if they have ventilator or not. Show your

INSERT SQLs and your final Bed table after inserts. Put screenshot(s) below. **2.5 points.**

| # | BED_ID | WITH_VENTILATOR |
|---|---|---|
| 1 | 1 | ☐ |
| 2 | 2 | ☐ |
| 3 | 3 | ☐ |
| 4 | 4 | ☑ |
| 5 | 5 | ☑ |
| 6 | 6 | ☑ |

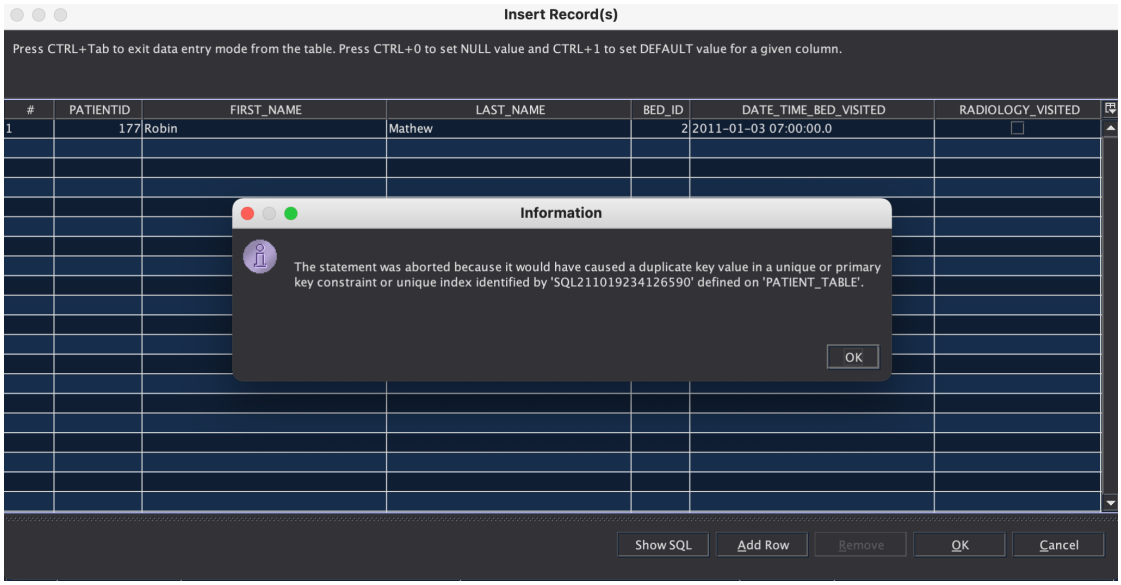
```
INSERT INTO HOSPITAL.BED_TABLE (BED_ID, WITH_VENTILATOR)
        VALUES (4, true)
INSERT INTO HOSPITAL.BED_TABLE (BED_ID, WITH_VENTILATOR)
        VALUES (5, true)
INSERT INTO HOSPITAL.BED_TABLE (BED_ID, WITH_VENTILATOR)
        VALUES (6, true)
```

3. Insert 3 patients from the simulation based on their movement. Show your INSERT SQLs and your final Patient table after inserts. Put screenshot(s) below. **2.5 points.**

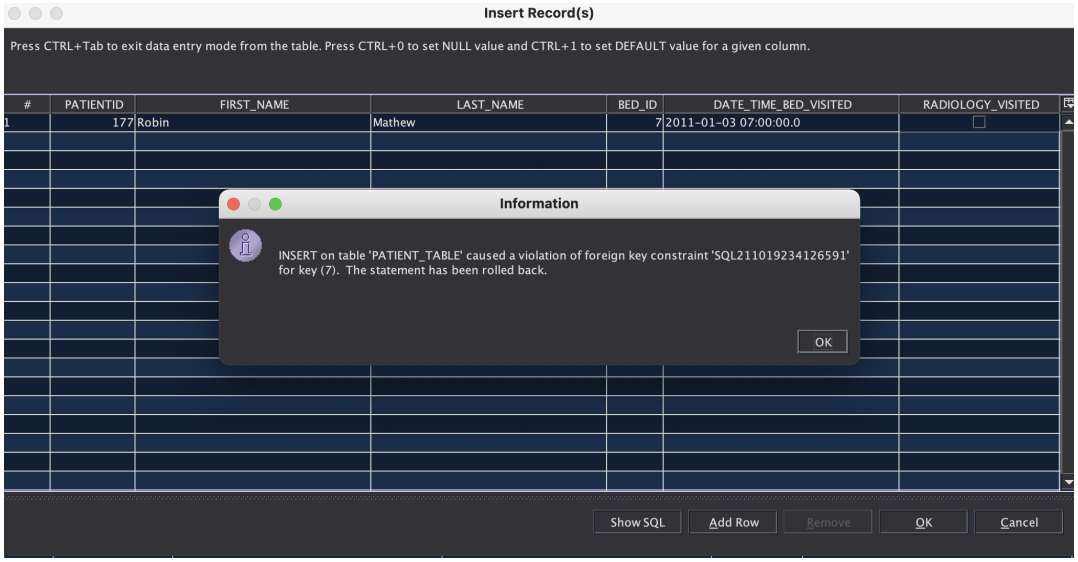
| # | PATIENTID | FIRST_NAME | LAST_NAME | BED_ID | DATE_TIME_BED_VISITED | RADIOLOGY_VISITED |
|---|---|---|---|---|---|---|
| 1 | 177 | Robin | Mathew | 2 | 2011-01-03 07:00:00.0 | ☐ |
| 2 | 174 | Dev | Peterson | 1 | 2011-01-03 07:02:11.0 | ☐ |
| 3 | 170 | Dhormamu | Batman | 3 | 2011-01-03 07:05:08.0 | ☐ |

```
INSERT INTO HOSPITAL.PATIENT_TABLE (PATIENTID, FIRST_NAME, LAST_NAME, BED_ID, DATE_TIME_BED_VISITED, RADIOLOGY_VISITED)
        VALUES (177, 'Robin', 'Mathew', 2, '2011-01-03 07:00:00.0', false)
INSERT INTO HOSPITAL.PATIENT_TABLE (PATIENTID, FIRST_NAME, LAST_NAME, BED_ID, DATE_TIME_BED_VISITED, RADIOLOGY_VISITED)
        VALUES (174, 'Dev', 'Peterson', 1, '2011-01-03 07:02:11.0', false)
INSERT INTO HOSPITAL.PATIENT_TABLE (PATIENTID, FIRST_NAME, LAST_NAME, BED_ID, DATE_TIME_BED_VISITED, RADIOLOGY_VISITED)
        VALUES (170, 'Dhormamu', 'Batman', 3, '2011-01-03 07:05:08.0', false)
```

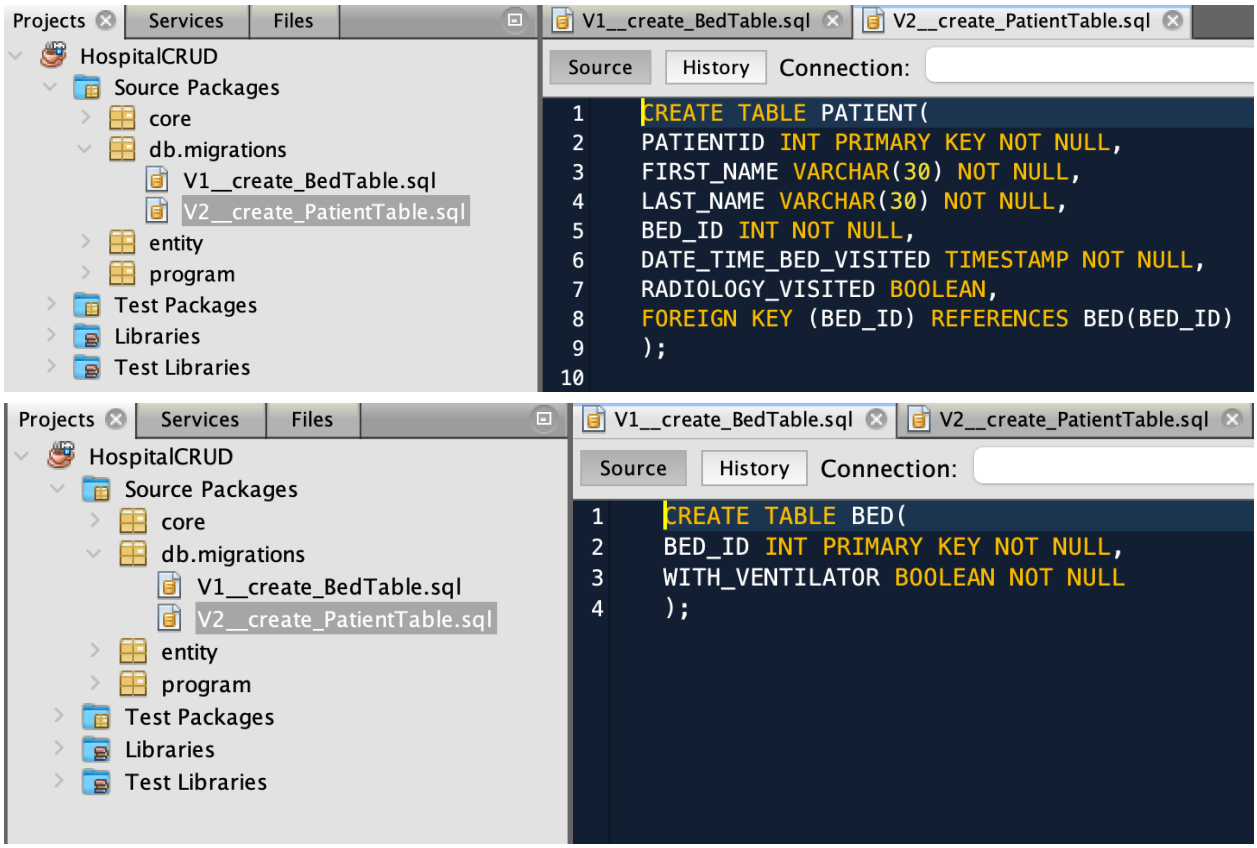
4. Try to add an existing bed and an existing patient into the database. Show primary key violation. Put screenshot(s) of SQL and error below. **2.5 points each. 5 points total.**




5. Try to add a new patient into the database for a bed that does not exist in your database. Show foreign key violation. Put screenshot(s) of SQL and error below. **5 points.**




6. Put those SQL code in a file named as V1_something.sql, V2_something.sql under db.migrations so you can create those tables via FlyWay. Try to give meaningful names to your SQL files. Show your work (put screenshot(s) of contents of the files and where you placed those files in the project). **5 points.**




7. Name your database as hospitaldb so FlyWay will create a databasefolder as hospitaldb. Hint: check DB.java. Put screenshot(s) of your DB.java below. **5 points.**

```java
package core;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import org.flywaydb.core.Flyway;

public class DB {
    private static final String DB_URL =
"jdbc:derby:hospitaldb;create=true;user=hospitaldb;password=hospitaldb";
    private static final String MIGRATION_DIR = "db.migrations";
    private static DB instance = null;
    private final Connection mConnection;

    private DB() throws SQLException
    {
        migrateDb();//Migrate DB first before doing anything
        mConnection = DriverManager.getConnection(DB_URL);
    }

    public static DB getInstance() {
        if (instance == null) {
            try {
                instance = new DB();
            } catch (SQLException ex) {
                System.err.println("Database Error: " + ex.toString());
                System.exit(1);
            }
        }

        return instance;
    }

    public int executeUpdate(String sql) throws SQLException {
        return mConnection.createStatement().executeUpdate(sql);
    }


    public ResultSet executeQuery(String sql) throws SQLException {
        return mConnection.createStatement().executeQuery(sql);
    }


    public PreparedStatement getPreparedStatement(String sql) throws SQLException {
        return mConnection.prepareStatement(sql);
    }


    private void migrateDb() {
        Flyway flyway = new Flyway();
        flyway.setDataSource(DB_URL, null, null);//Set the database
        flyway.setLocations(MIGRATION_DIR);//It will run these SQL scripts under Migration
Directory
        flyway.migrate();//This performs the migration, basically creating tables and
inserting values into tables if necessary
    }
}
```

8. Modify entity classes. Put a screenshot of your Bed.java and Patient.java. **5 points each.**



```java
package entity;

public class Bed {
    private int Bed_ID;
    private boolean With_Ventilator;

    public Bed(int Bed_ID, boolean With_Ventilator) {
        this.With_Ventilator = With_Ventilator;
        this.Bed_ID = Bed_ID;
    }

    public int getBed_ID() {
    return Bed_ID;
}
    public boolean getWith_Ventilator() {
        return With_Ventilator;
    }

}
```



```java
package entity;

public class Patient
{
    private int PatientID;
    private String First_Name;
    private String Last_Name;
    private int Bed_ID;
    private String Date_Time_Bed_Visited;
    private boolean Radiology_Visited;

    public Patient(int PatientID, String First_Name, String Last_Name, int Bed_ID, String
Date_Time_Bed_Visited, Boolean Radiology_Visited){

        this.PatientID = PatientID;
        this.First_Name = First_Name;
        this.Last_Name = Last_Name;
        this.Bed_ID = Bed_ID;
        this.Date_Time_Bed_Visited = Date_Time_Bed_Visited;
        this.Radiology_Visited = Radiology_Visited;
    }

    public int getPatientID() {
        return PatientID;
    }

    public String getFirst_Name() {
        return First_Name;
    }

    public String getLast_Name() {
        return Last_Name;
    }

    public int getBed_ID() {
        return Bed_ID;
    }

    public boolean getRadiology_Visited() {
        return Radiology_Visited;
    }

    public String getDate_Time_Bed_Visited() {
        return Date_Time_Bed_Visited;
    }

    @Override
    public String toString() {
return "PATIENT_TABLE {" + "PATIENT_ID=" + PatientID + ", FIRST_NAME=" + First_Name + ",
LAST_NAME=" + Last_Name + ", BED_ID=" + Bed_ID + ", DATE_TIME_BED_VISITED= " +
Date_Time_Bed_Visited + ", Radiology Visited= " + Radiology_Visited + '}';
    }
}
```

9. Modify entityDAO classes. Put screenshot of your BedDAO.java and PatientDAO.java. **5 points each. 5 points each.** (If you modified DAO.java, put a screenshot of that one too.)

```
Uploaded using RayThis Extension

package entity;
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;
import core.*;
import java.sql.ResultSet;
import java.sql.PreparedStatement;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;

public class PatientDAO implements DAO<Patient>
{
    public PatientDAO() {

    }
    List<Patient> patients;

    @Override
    public Optional<Patient> get(int id) {
        DB db = DB.getInstance();
        ResultSet rs = null;
        try {
            String sql = "SELECT * FROM Patient WHERE PatientID = ?";
            PreparedStatement stmt = db.getPreparedStatement(sql);
            stmt.setInt(1, id);
            rs = stmt.executeQuery();
            Patient patient = null;
            while (rs.next()) {
                patient = new Patient(rs.getInt("PatientID"), rs.getString("First_Name"),
rs.getString("Last_Name"), rs.getInt("Bed_ID"), rs.getString("Date_Time_Bed_Visited"),
rs.getBoolean("Radiology_Visited"));
            }
            return Optional.ofNullable(patient);
        } catch (SQLException ex) {
            System.err.println(ex.toString());
            return null;
        }
    }


    @Override
    public List<Patient> getAll() {
        DB db = DB.getInstance();
        ResultSet rs = null;
        patients = new ArrayList<>();
        try {
            String sql = "SELECT * FROM Patient";
            rs = db.executeQuery(sql);
            Patient patient = null;
            while (rs.next()) {
                patient = new Patient(rs.getInt("PatientID"), rs.getString("First_Name"),
rs.getString("Last_Name"), rs.getInt("Bed_ID"), rs.getString("Date_Time_Bed_Visited"),
rs.getBoolean("Radiology_Visited"));
                patients.add(patient);
            }
            return patients;
        } catch (SQLException ex) {
            System.err.println(ex.toString());
            return null;
        }
    }


    @Override
    public void insert(Patient patient)
    {
        DB db = DB.getInstance();
        try {
            String sql = "INSERT INTO Patient(PatientID, First_Name, Last_Name, Bed_ID,
Date_Time_Bed_Visited, Radiology_Visited) VALUES (?, ?, ?, ?, ?, ?)";
            PreparedStatement stmt = db.getPreparedStatement(sql);
            stmt.setInt(1, patient.getPatientID());
            stmt.setString(2, patient.getFirst_Name());
            stmt.setString(3, patient.getLast_Name());
            stmt.setInt(4, patient.getBed_ID());
            stmt.setString(5, patient.getDate_Time_Bed_Visited());
            stmt.setBoolean(6, patient.getRadiology_Visited());
            int rowInserted = stmt.executeUpdate();
            if (rowInserted > 0) {
                System.out.println("A new patient was inserted successfully!");
            }
        } catch (SQLException ex) {
            System.err.println(ex.toString());
        }
    }


    @Override
    public void update(Patient patient) {
        DB db = DB.getInstance();
        try {
            String sql = "UPDATE Patient SET First_Name=?, Last_Name=?, Bed_ID=?,
Date_Time_Bed_Visited=?, Radiology_Visited=? WHERE PatientID=?";
            PreparedStatement stmt = db.getPreparedStatement(sql);
            stmt.setString(1, patient.getFirst_Name());
            stmt.setString(2, patient.getLast_Name());
            stmt.setInt(3, patient.getBed_ID());
            stmt.setString(4, patient.getDate_Time_Bed_Visited());
            stmt.setBoolean(5, patient.getRadiology_Visited());
            stmt.setInt(6, patient.getPatientID());
            int rowsUpdated = stmt.executeUpdate();
            if (rowsUpdated > 0) {
                System.out.println("An existing patient was updated successfully!");
            }
        } catch (SQLException ex) {
            System.err.println(ex.toString());
        }
    }


    @Override
    public void delete(Patient patient) {
        DB db = DB.getInstance();
        try {
            String sql = "DELETE FROM Patient WHERE PatientID = ?";
            PreparedStatement stmt = db.getPreparedStatement(sql);
            stmt.setInt(1, patient.getPatientID());
            int rowsDeleted = stmt.executeUpdate();
            if (rowsDeleted > 0) {
                System.out.println("A patient was deleted successfully!");
            }
        } catch (SQLException ex) {
            System.err.println(ex.toString());
        }
    }


    @Override
    public List<String> getColumnNames() {
        DB db = DB.getInstance();
        ResultSet rs = null;
        List<String> headers = new ArrayList<>();
        try {
            String sql = "SELECT * FROM Patient WHERE PatientID = -1";//We just need this sql
query to get the column headers
            rs = db.executeQuery(sql);
            ResultSetMetaData rsmd = rs.getMetaData();
            //Get number of columns in the result set
            int numberCols = rsmd.getColumnCount();
            for (int i = 1; i <= numberCols; i++) {
                headers.add(rsmd.getColumnLabel(i));//Add column headers to the list
            }
            return headers;
        } catch (SQLException ex) {
            System.err.println(ex.toString());
            return null;
        }
    }
}
```

```
Uploaded using RayThis Extension

package entity;
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;
import core.*;
import java.sql.ResultSet;
import java.sql.PreparedStatement;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;

public class BedDAO implements DAO<Bed>
{
    public BedDAO() {

    }
    List<Bed> beds;

    @Override
    public Optional<Bed> get(int id) {
        DB db = DB.getInstance();
        ResultSet rs = null;
        try {
            String sql = "SELECT * FROM Bed WHERE Bed_ID = ?";
            PreparedStatement stmt = db.getPreparedStatement(sql);
            stmt.setInt(1, id);
            rs = stmt.executeQuery();
            Bed bed = null;
            while (rs.next()) {
                bed = new Bed(rs.getInt("Bed_ID"), rs.getBoolean("With_Ventilator"));
            }
            return Optional.ofNullable(bed);
        } catch (SQLException ex) {
            System.err.println(ex.toString());
            return null;
        }
    }

    @Override
    public List<Bed> getAll() {
        DB db = DB.getInstance();
        ResultSet rs = null;
        beds = new ArrayList<>();
        try {
            String sql = "SELECT * FROM Bed";
            rs = db.executeQuery(sql);
            Bed bed = null;
            while (rs.next()) {
                bed = new Bed(rs.getInt("Bed_ID"), rs.getBoolean("With_Ventilator"));
                beds.add(bed);
            }
            return beds;
        } catch (SQLException ex) {
            System.err.println(ex.toString());
            return null;
        }
    }

    @Override
    public void insert(Bed bed)
    {
        DB db = DB.getInstance();
        try {
            String sql = "INSERT INTO Bed(Bed_ID, With_Ventilator) VALUES (?, ?)";
            PreparedStatement stmt = db.getPreparedStatement(sql);
            stmt.setInt(1, bed.getBed_ID());
            stmt.setBoolean(2, bed.getWith_Ventilator());
            int rowInserted = stmt.executeUpdate();
            if (rowInserted > 0) {
                System.out.println("A new bed was inserted successfully!");
            }
        } catch (SQLException ex) {
            System.err.println(ex.toString());
        }
    }

    @Override
    public void update(Bed bed) {
        DB db = DB.getInstance();
        try {
            String sql = "UPDATE Bed SET With_Ventilator=? WHERE BED_ID=?";
            PreparedStatement stmt = db.getPreparedStatement(sql);
            stmt.setBoolean(1, bed.getWith_Ventilator());
            stmt.setInt(2, bed.getBed_ID());
            int rowsUpdated = stmt.executeUpdate();
            if (rowsUpdated > 0) {
                System.out.println("An existing bed was updated successfully!");
            }
        } catch (SQLException ex) {
            System.err.println(ex.toString());
        }
    }

    @Override
    public void delete(Bed bed) {
        DB db = DB.getInstance();
        try {
            String sql = "DELETE FROM Bed WHERE Bed_ID = ?";
            PreparedStatement stmt = db.getPreparedStatement(sql);
            stmt.setInt(1, bed.getBed_ID());
            int rowsDeleted = stmt.executeUpdate();
            if (rowsDeleted > 0) {
                System.out.println("A bed was deleted successfully!");
            }
        } catch (SQLException ex) {
            System.err.println(ex.toString());
        }
    }

    @Override
    public List<String> getColumnNames() {
        DB db = DB.getInstance();
        ResultSet rs = null;
        List<String> headers = new ArrayList<>();
        try {
            String sql = "SELECT * FROM Bed WHERE Bed_ID = -1";
            rs = db.executeQuery(sql);
            ResultSetMetaData rsmd = rs.getMetaData();
            //Get number of columns in the result set
            int numberCols = rsmd.getColumnCount();
            for (int i = 1; i <= numberCols; i++) {
                headers.add(rsmd.getColumnLabel(i));
            }
            return headers;
        } catch (SQLException ex) {
            System.err.println(ex.toString());
            return null;
        }
    }
}
```

10. Modify the main method to test your CRUD application for the following tasks. If your application does not run, you will get zero points for this part. Show your work by putting screenshots from your running application. **5 points for each task. 25 points total.**

```
bedDAO = new BedDAO();
Bed bed;
bed = new Bed(1, false);
bedDAO.insert(bed);
bed = new Bed(2, false);
bedDAO.insert(bed);
bed = new Bed(3, false);
bedDAO.insert(bed);
bed = new Bed(4, true);
bedDAO.insert(bed);
bed = new Bed(5, true);
bedDAO.insert(bed);
bed = new Bed(6, false);
bedDAO.insert(bed);
bed= new Bed(6, true);
bedDAO.update(bed);
```

Uploaded using RayThis Extension

```
patientDAO = new PatientDAO();
        Patient patient;
        patient = new Patient(139, "Jahidul", "Robin", 1, "2021-05-23 10:39:23.453", false);
        patientDAO.insert(patient);//Insert John
        patient = new Patient(120, "Steve", "Grace", 2, "2021-06-30 18:12:33.392", false);
        patientDAO.insert(patient);//Insert Steve
        printPatients();
        patient = new Patient(120, "Steve", "Grace", 2, "2021-05-30 08:15:00.392", false);
        patientDAO.update(patient);//Update Steve
        printPatients();
        patient = new Patient(111, "Dev", "Malik", 3, "2021-07-27 11:29:10.233", false);
        patientDAO.insert(patient);//Insert Dev
        patient = new Patient(93, "Mike", "Smith", 4, "2021-07-29 08:48:11.556", true);
        patientDAO.insert(patient);//Insert Mike
        patient = new Patient(102, "Love", "Robinson", 5, "2021-09-19 23:59:59.392", true);
        patientDAO.insert(patient);//Insert Love
        patient = new Patient(102, "Love", "Robinson", -1, "2021-09-19 23:59:59.392", true);
        patientDAO.update(patient);//Update Love
        patient = new Patient(84, "Dhamu", "Patel", 6, "2021-10-16 4:20:19.293", true);
        patientDAO.insert(patient);//Insert Dhamu
        printPatients();
        patientDAO.delete(patient);//Delete Dhamu
        printPatients();
        printBeds();
        patient = getPatient(0);
        System.out.println(patient.getPatientID() + "-" + patient.getFirst_Name() + "-" +
patient.getLast_Name() + "-" + patient.getBed_ID() + "-" + patient.getDate_Time_Bed_Visited()
+ "-" + patient.getRadiology_Visited());
```

```
A new bed was inserted successfully!
A new bed was inserted successfully!
A new bed was inserted successfully!
A new bed was inserted successfully!
A new bed was inserted successfully!
A new bed was inserted successfully!
An existing bed was updated successfully!
A new patient was inserted successfully!
A new patient was inserted successfully!
PATIENTID               FIRST_NAME              LAST_NAME               BED_ID    DATE_TIME_BED_VISITED       RADIOLOGY_VISITED
      139                  Jahidul                 Robin                    1  2021-05-23 10:39:23.453                false
      120                    Steve                 Grace                    2  2021-06-30 18:12:33.392                false
An existing patient was updated successfully!
PATIENTID               FIRST_NAME              LAST_NAME               BED_ID    DATE_TIME_BED_VISITED       RADIOLOGY_VISITED
      139                  Jahidul                 Robin                    1  2021-05-23 10:39:23.453                false
      120                    Steve                 Grace                    2  2021-05-30 08:15:00.392                false
A new patient was inserted successfully!
A new patient was inserted successfully!
A new patient was inserted successfully!
java.sql.SQLIntegrityConstraintViolationException: UPDATE on table 'PATIENT' caused a violation of foreign key constraint 'SQL211021212810830' for key (-1).  The statement has been rolled back.
A new patient was inserted successfully!
PATIENTID               FIRST_NAME              LAST_NAME               BED_ID    DATE_TIME_BED_VISITED       RADIOLOGY_VISITED
      139                  Jahidul                 Robin                    1  2021-05-23 10:39:23.453                false
      120                    Steve                 Grace                    2  2021-05-30 08:15:00.392                false
      111                      Dev                 Malik                    3  2021-07-27 11:29:10.233                false
       93                     Mike                 Smith                    4  2021-07-29 08:48:11.556                 true
      102                     Love               Robinson                   5  2021-09-19 23:59:59.392                 true
       84                   Dhamu                  Patel                    6  2021-10-16 04:20:19.293                 true
A patient was deleted successfully!
PATIENTID               FIRST_NAME              LAST_NAME               BED_ID    DATE_TIME_BED_VISITED       RADIOLOGY_VISITED
      139                  Jahidul                 Robin                    1  2021-05-23 10:39:23.453                false
      120                    Steve                 Grace                    2  2021-05-30 08:15:00.392                false
      111                      Dev                 Malik                    3  2021-07-27 11:29:10.233                false
       93                     Mike                 Smith                    4  2021-07-29 08:48:11.556                 true
      102                     Love               Robinson                   5  2021-09-19 23:59:59.392                 true
   BED_ID          WITH_VENTILATOR
        1                    false
        2                    false
        3                    false
        4                     true
        5                     true
        6                     true
```

**Part 3:** Developing CRUD Interface (20 pts)

Modify the main method (keep the previous code from task 11) to add a user input logic for your CRUD. So, user can perform CRUD operations on two tables. Show your work by putting screenshots from your running application for each task. **2 points for each task (except option 9). 20 points total.**

1. Option 1 Create a bed. Put screenshot(s) below

```
Option 1: Create a Bed
Option 2: Update a Bed
Option 3: Delete a Bed
Option 4: Print Beds
Option 5: Create a Patient
Option 6: Update a Patient
Option 7: Delete a Patient
Option 8: Print Patients
Option 9: Exit Application
Enter your option from 1 to 9
1
Enter Bed ID
7
Enter true or false if ventilator is present
true
A new bed was inserted successfully!
                    BED_ID          WITH_VENTILATOR
                         1                    false
                         2                    false
                         3                    false
                         4                     true
                         5                     true
                         6                     true
                         7                     true
```

2. Option 2 Update an existing bed. Put screenshot(s) below.

```
Option 1: Create a Bed
Option 2: Update a Bed
Option 3: Delete a Bed
Option 4: Print Beds
Option 5: Create a Patient
Option 6: Update a Patient
Option 7: Delete a Patient
Option 8: Print Patients
Option 9: Exit Application
Enter your option from 1 to 9
2
Enter Bed ID
7
Enter true or false if ventilator is present
false
An existing bed was updated successfully!
                    BED_ID          WITH_VENTILATOR
                         1                    false
                         2                    false
                         3                    false
                         4                     true
                         5                     true
                         6                     true
                         7                    false
```

3. Option 3 Delete an existing bed. Put screenshot(s) below.

```
Option 1: Create a Bed
Option 2: Update a Bed
Option 3: Delete a Bed
Option 4: Print Beds
Option 5: Create a Patient
Option 6: Update a Patient
Option 7: Delete a Patient
Option 8: Print Patients
Option 9: Exit Application
Enter your option from 1 to 9
3
Enter Bed ID
7
Enter true or false if ventilator is present
false
A bed was deleted successfully!
                     BED_ID          WITH_VENTILATOR
                          1                    false
                          2                    false
                          3                    false
                          4                     true
                          5                     true
                          6                     true
```

4. Option 4 Print all beds. Put screenshot(s) below.

```
Option 1: Create a Bed
Option 2: Update a Bed
Option 3: Delete a Bed
Option 4: Print Beds
Option 5: Create a Patient
Option 6: Update a Patient
Option 7: Delete a Patient
Option 8: Print Patients
Option 9: Exit Application
Enter your option from 1 to 9
4
```

| BED_ID | WITH_VENTILATOR |
|---|---|
| 1 | false |
| 2 | false |
| 3 | false |
| 4 | true |
| 5 | true |
| 6 | true |

5. Option 5 Create a patient. Put screenshot(s) below.

```
Option 1: Create a Bed
Option 2: Update a Bed
Option 3: Delete a Bed
Option 4: Print Beds
Option 5: Create a Patient
Option 6: Update a Patient
Option 7: Delete a Patient
Option 8: Print Patients
Option 9: Exit Application
Enter your option from 1 to 9
5
Enter Patient ID
463
Enter firstname
Larry
Enter lastname
Skeleton
Enter BedID
7
Enter DateTimeBedVisited
2021-09-19 23:59:59.392
Enter true or false if radiology was visited
false
A new patient was inserted successfully!
```

| PATIENTID | FIRST_NAME | LAST_NAME | BED_ID | DATE_TIME_BED_VISITED | RADIOLOGY_VISITED |
|---|---|---|---|---|---|
| 139 | Jahidul | Robin | 1 | 2021-05-23 10:39:23.453 | false |
| 120 | Steve | Grace | 2 | 2021-05-30 08:15:00.392 | false |
| 111 | Dev | Malik | 3 | 2021-07-27 11:29:10.233 | false |
| 93 | Mike | Smith | 4 | 2021-07-29 08:48:11.556 | true |
| 102 | Love | Robinson | 5 | 2021-09-19 23:59:59.392 | true |
| 463 | Larry | Skeleton | 7 | 2021-09-19 23:59:59.392 | false |

6. Option 6 Update an existing patient. Put screenshot(s) below.

```
Option 1: Create a Bed
Option 2: Update a Bed
Option 3: Delete a Bed
Option 4: Print Beds
Option 5: Create a Patient
Option 6: Update a Patient
Option 7: Delete a Patient
Option 8: Print Patients
Option 9: Exit Application
Enter your option from 1 to 9
6
Enter Patient ID
120
Enter firstname
Steve
Enter lastname
Mathew
Enter BedID
2
Enter DateTimeBedVisited
2021-05-30 08:15:00.392
Enter true or false if radiology was visited
false
An existing patient was updated successfully!
```

| PATIENTID | FIRST_NAME | LAST_NAME | BED_ID | DATE_TIME_BED_VISITED | RADIOLOGY_VISITED |
|---|---|---|---|---|---|
| 139 | Jahidul | Robin | 1 | 2021-05-23 10:39:23.453 | false |
| 120 | Steve | Mathew | 2 | 2021-05-30 08:15:00.392 | false |
| 111 | Dev | Malik | 3 | 2021-07-27 11:29:10.233 | false |
| 93 | Mike | Smith | 4 | 2021-07-29 08:48:11.556 | true |
| 102 | Love | Robinson | 5 | 2021-09-19 23:59:59.392 | true |
| 463 | Larry | Skeleton | 7 | 2021-09-19 23:59:59.392 | false |

7. Option 7 Delete an existing patient. Put screenshot(s) below.

```
Option 1: Create a Bed
Option 2: Update a Bed
Option 3: Delete a Bed
Option 4: Print Beds
Option 5: Create a Patient
Option 6: Update a Patient
Option 7: Delete a Patient
Option 8: Print Patients
Option 9: Exit Application
Enter your option from 1 to 9
7
Enter Patient ID
463
Enter firstname
Larry
Enter lastname
Skeleton
Enter BedID
7
Enter DateTimeBedVisited
2021-09-19 23:59:59.392
Enter true or false if radiology was visited
false
A patient was deleted successfully!
```

| PATIENTID | FIRST_NAME | LAST_NAME | BED_ID | DATE_TIME_BED_VISITED | RADIOLOGY_VISITED |
|---|---|---|---|---|---|
| 139 | Jahidul | Robin | 1 | 2021-05-23 10:39:23.453 | false |
| 120 | Steve | Mathew | 2 | 2021-05-30 08:15:00.392 | false |
| 111 | Dev | Malik | 3 | 2021-07-27 11:29:10.233 | false |
| 93 | Mike | Smith | 4 | 2021-07-29 08:48:11.556 | true |
| 102 | Love | Robinson | 5 | 2021-09-19 23:59:59.392 | true |

8. Option 8 Print all patients. Put screenshot(s) below.

```
Option 1: Create a Bed
Option 2: Update a Bed
Option 3: Delete a Bed
Option 4: Print Beds
Option 5: Create a Patient
Option 6: Update a Patient
Option 7: Delete a Patient
Option 8: Print Patients
Option 9: Exit Application
Enter your option from 1 to 9
8
```

| PATIENTID | FIRST_NAME | LAST_NAME | BED_ID | DATE_TIME_BED_VISITED | RADIOLOGY_VISITED |
|---|---|---|---|---|---|
| 139 | Jahidul | Robin | 1 | 2021-05-23 10:39:23.453 | false |
| 120 | Steve | Mathew | 2 | 2021-05-30 08:15:00.392 | false |
| 111 | Dev | Malik | 3 | 2021-07-27 11:29:10.233 | false |
| 93 | Mike | Smith | 4 | 2021-07-29 08:48:11.556 | true |
| 102 | Love | Robinson | 5 | 2021-09-19 23:59:59.392 | true |

9. Option 9 for Exit Application (4 points). Put screenshot(s) below.

```
Option 1: Create a Bed
Option 2: Update a Bed
Option 3: Delete a Bed
Option 4: Print Beds
Option 5: Create a Patient
Option 6: Update a Patient
Option 7: Delete a Patient
Option 8: Print Patients
Option 9: Exit Application
Enter your option from 1 to 9
9
BUILD SUCCESSFUL (total time: 3 minutes 3 seconds)
```