**Vietnam National University HCM**

**International University**

**School of Computer Science and Engineering**



**REPORT**

**MINI PROJECT**

***Subject: Principle of Database Management***

*Lecturer: Assoc, Prof. Nguyen Thi Thuy Loan*

Members:

Phan Nhật Tân- ITITIU19050

Nguyễn Phước Vĩnh Khang - ITITIU19017

Nguyễn Anh Tuấn - ITITIU19061

Trần Đức Ánh- ITITIU19079

Lê Quang Huy- ITITIU19015

*Semester 2, 2021 - 2022*

**ABSTRACT**

This mini-project aims to provide solutions for the database of applications. By doing this mini-project, we can learn about creating a database, ERD model, Relational model, Normalization….This mini-project includes input/output data, update data, Connect database with java platform,......

**Table of Content**

**Part Page**

1. **Introduction 3**
   1. Overview
   2. General Requirement
      1. Project’s Information 3
      2. Project’s Proposals 4
   3. Participation 4
2. **Database Diagrams 6**
   1. Basic Design 6
   2. ER Diagrams 7
   3. Relational Models 8
3. **Implementation 11**

3.1. SQL implementation 11

3.2. Java implementation 14

3.2.1. Connect Database using JDBC 14

3.2.2. Java for GUI 15

1. **Conclusion 22**
2. **References** 23

**Part 1**

**INTRODUCTION**

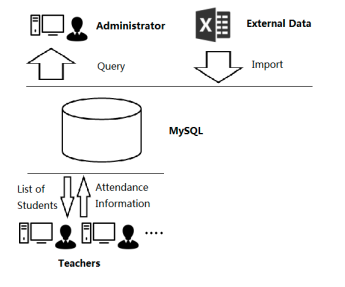
**1.1. Overview**

Data is essential information for every company . A database is an organized collection of structured information, data, or typically stored electronically in a computer system. SQL is a programming language used by nearly all [relational databases](https://www.oracle.com/database/what-is-database/#relational) to query, manipulate, and define data, and to provide access control (1).My Group chose the topic: Attendance Management System which applies to some software used for school, company, …. In particularly, this database is designed for schools

**1.2. General Requirement**

**1.2.1. Introduction**

The absence of lectures in college students has been a big concern since some students pay little attention to their lecturers or professors have problems with their teaching techniques. Most professors prefer to call the roll during class time in order to maintain the attendance rate, and several student attendance systems have been devised to assist them in doing so.The administrators will get a table (an Excel file) containing all the students and courses’ information at the beginning of the term, and import it into database with the help of importing module. After that, if any teacher opens the program installed on the computer in the classroom, it will provide a table containing each student’s name, ID and a checkbox for marking the absence.



**fig 1. Attendance System’s process (Source: Google.com)**

**1.2.3. Project proposals**

we list six proposals in following:

1. Discuss about some realistic constraint (eg. No duplicate students in a class, a student can miss 3 lectures… ): primary key, foreign key, check constraint, and not null for the tables or attributes.
2. Design the database following an E/R approach, then we also double check for the normalization process to modify the collections of tables that are in higher normal forms.
3. Use MS SQL server to create the normalized tables.
4. Populate the database by using SQL server statements.
5. Using the JDBC library to connect the database with the java platform, then create the winform to demo the efficiencies of the database.

**1.3. `Participation**

| Task | Tân | Huy | Tuấn | Khang | Ánh |
| --- | --- | --- | --- | --- | --- |
| Basic Design | **x** |  | **x** |  |  |
| ER Diagram |  | **x** |  | **x** |  |
| Relational Model |  | **x** |  | **x** | **x** |
| Modify the Diagrams | **x** | **x** | **x** | **x** | **x** |
| Implement the database | **x** | **x** |  |  |  |
| Connect database and GUI |  |  | **x** |  | **x** |
| Reports |  |  | **x** | **x** |  |

**Part 2**

**DATABASE DIAGRAMS**

1. **Basic Design**

In General, There are three objects for this database: Student, Teacher, Course.

From these basic objects, we can develop a database with more than three tables which approach the higher normal form such as Boyce- Codd normal forms. In basic, we will list some tables and their attribute:

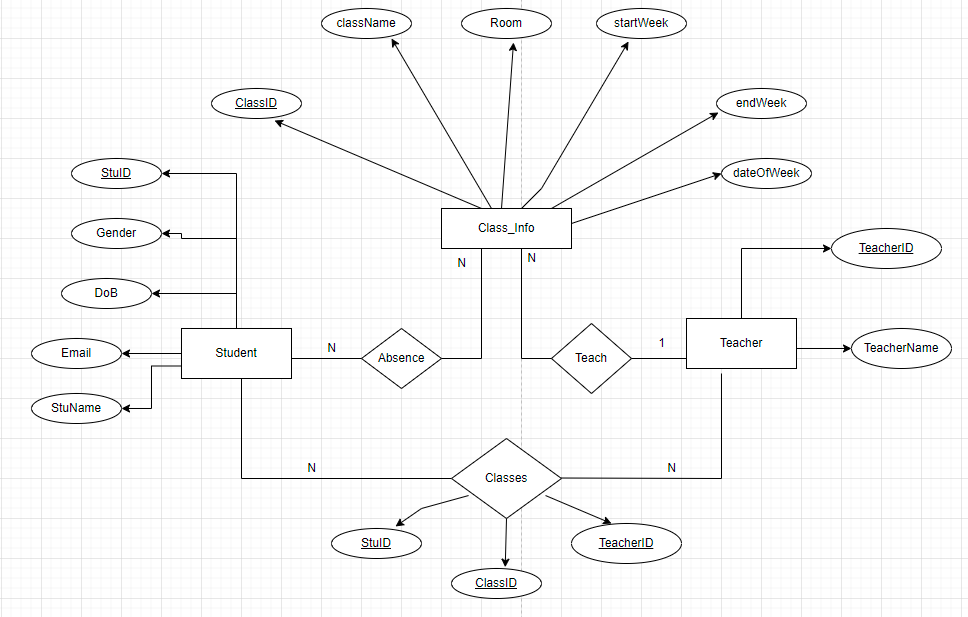
| Table | Columns | Note |
| --- | --- | --- |
| Student | - StudentID- nvarchar(11)  - StuName- nvarchar(50)  - Gender- nvarchar(10)  - DoB- date  - Email- nvarchar(50)  - Department- nvarchar(50)  - classID - nvarchar(11) | Storing the information of the students in the university. |
| Teacher | - TeacherID- nvarchar(11)  - TeacherName- nvarchar(50) | Storing the information of all the lectures. |
| Class | - classID- nvarchar(11)  - className- nvarchar(50)  - TeacherID-nvarchar(11)  - RoomID- nvarchar(15)  - StartWeek- int  - EndWeek- int  - Class\_Begin- int  - Class\_End- int -DayofWeek- nvarchar(20) | Storing detailed information of all the classes. |
| Absence | - classID- nvarchar(11)  - StudentID- nvarchar(11)  - AbsenceDate- date | Storing student record of absence |

**Table 1: Tables and their attributes in basic**

In order to improve performance, some constraints were used to keep spread-out data consistent, any invalid insert or update operation would be rejected.

1. **ER diagram:**

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research(2). From the basic designs, an ER diagram was created.



**fig 2. ER Diagram for attendance Management system**

* The ‘Absence’ relationship used to keep track the student who absences a class. The class can have many absent students and the student can miss many classes. Therefore, this is a many to many relationship.
* The ‘Teach’ relationship is the 1 to many. a teacher can teach many classes and one class has only one lecturer.
* The ‘Classes’ relationship can be used to return the list of students in a class. A teacher can have many students and a student has many teachers. Hence, This is a many to many relationship.

1. **Relational Model**

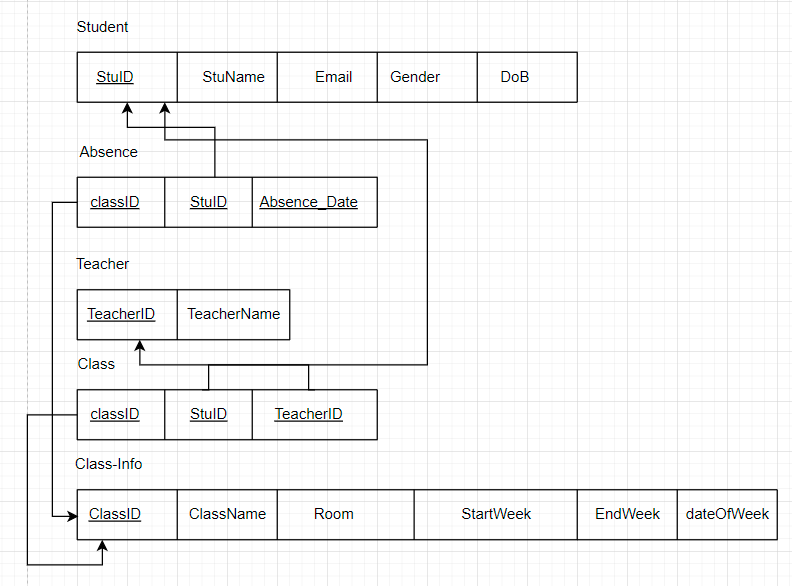
The ER diagram provides us with basic entity-relationship. But, It isn't particularly thorough because it doesn't demonstrate how to establish a relationship between entity sets when building relations or tables (3). Before we implement our database in the SQL server, we create another model that represents all possible relations within the database. In particular, we must convert the ERD to a relational model.

The primary keys of each individual entity of this database is as followed (table - Key):

* Student - studentID
* Teacher - TeacherID
* Class\_Info - classID

According the explanation in the previous part:

* The ‘Absence’ is a ‘many to many’ relationship. Therefore, an Absence entity which takes primary keys of two tables Student and Class\_Info and transfers them to foreign keys is created. In addition, This entity contains the student’s absence date. Because a student can be absent many times, we will set the primary key for absenceDate. Hence, The primary key of this entity is (classID, StudentID, absenceDate).
* The ‘Classes’ is a ‘many to many’ relationship. Hence, we establish a new ‘Class ’ schema and add as a foreign key all of the primary key attributes of two tables Student and Teacher. In the ‘Class’ schema, we add the classID attribute. The primary key of this schema is a set of (classID, StuID,TeacherID). ( Primary key contains classID because a teacher instructs many classes and students can join two or many classes of one lecturer.)
* The ‘Teach’ is a ‘1 to many’ relationship. Therefore, we just establish a new schema (Teacher and Class\_Info) containing the primary keys of the entities that formed it and its own attributes.



**fig 3: Relational Model for attendance management System**

**Note:** The attributes with an underline are the primary key for that table.

From the relational model for this database, our group will design tables and their attributes in detail.

| Table | Columns | Note |
| --- | --- | --- |
| Student | - StuID- nvarchar(11)  - StuName- nvarchar(50)  - Email- nvarchar(50)  - Gender- nvarchar(10)  - DoB- date | Storing the information of the students in the university. |
| Teacher | - TeacherID- nvarchar(11)  - TeacherName- nvarchar(50) | Storing the information of all the lectures. |
| Class | - classID- nvarchar(11)  - StuID- nvarchar(11)  - TeacherID-nvarchar(11) | Storing basic information of lecturers. |
| Class-Info | - classID- nvarchar(11)  - className- nvarchar(50)  - Room- nvarchar(15)  - StartWeek- int  - EndWeek- int  - DateofWeek - nvarchar(20) | Storing detailed information of a class |
| Absence | - classID- nvarchar(11)  - StudentID- nvarchar(11)  - AbsenceDate- date | Storing student record of absence |

**Table 2: Tables and their Attributes in detail.**

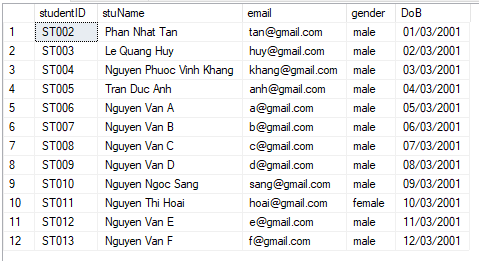
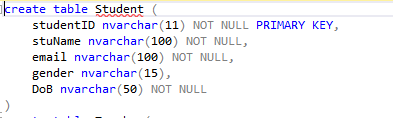
**Part 3**

**Implementation**

1. **SQL Implementation**

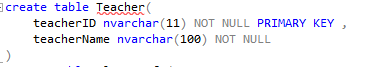
The database for attendance management system has 5 tables:

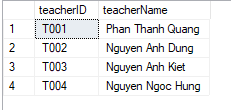
**Student:**

****

**fig 4: SQL code of Student Table.**

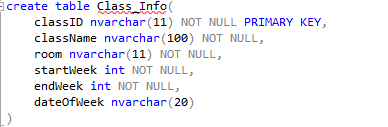
**Teacher:**

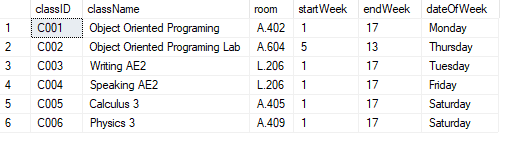
****

****

**fig 5: SQL code of Teacher Table.**

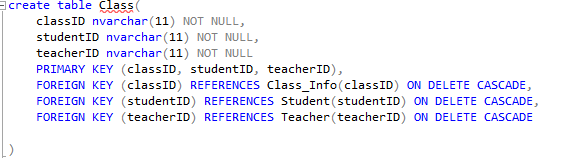
**Class\_Info:**

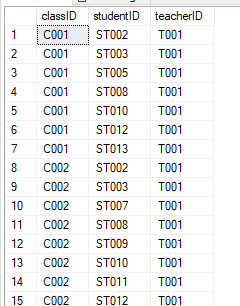
****

****

**fig 6: SQL code of Class\_Info Table.**

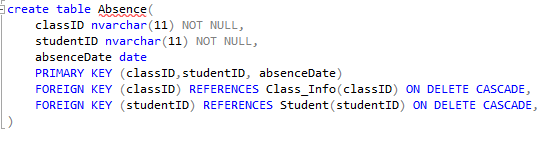
**Class:**

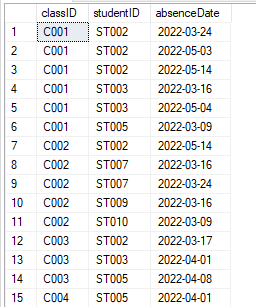
****

****

**fig 7: SQL code of Class Table.**

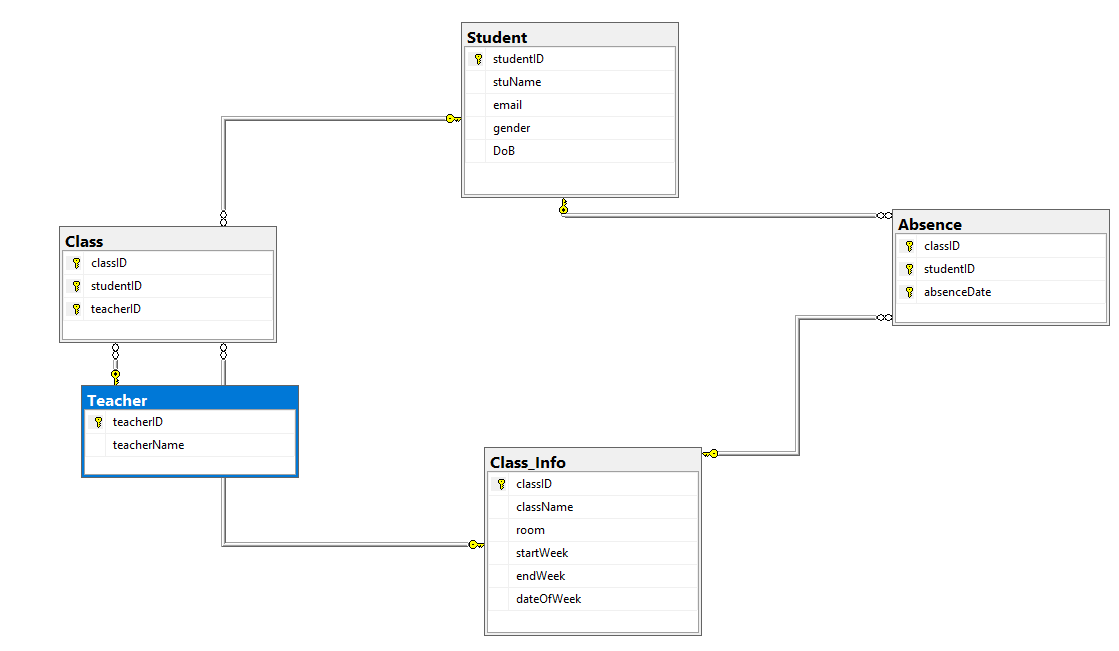
**Absence:**

****

****

**fig 8: SQL code of Class Table.**

**Note:** ‘ON DELETE CASCADE’ : When we create a foreign key using this option, it deletes the referencing rows in the child table when the referenced row is deleted in the parent table which has a primary key (“SQLshack”, Delete cascade and Update Cascade in SQL server foreign key).

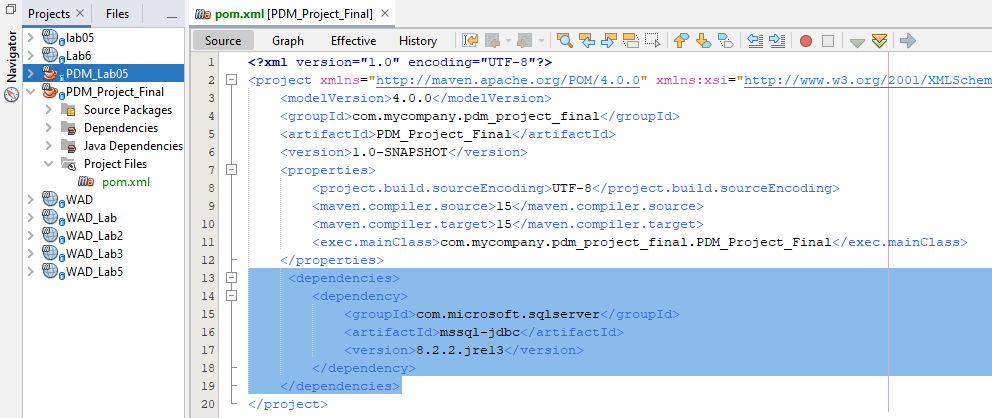
****

**fig 9: Relational Model for attendance management System in SQL- server.**

1. **Java Implementation**

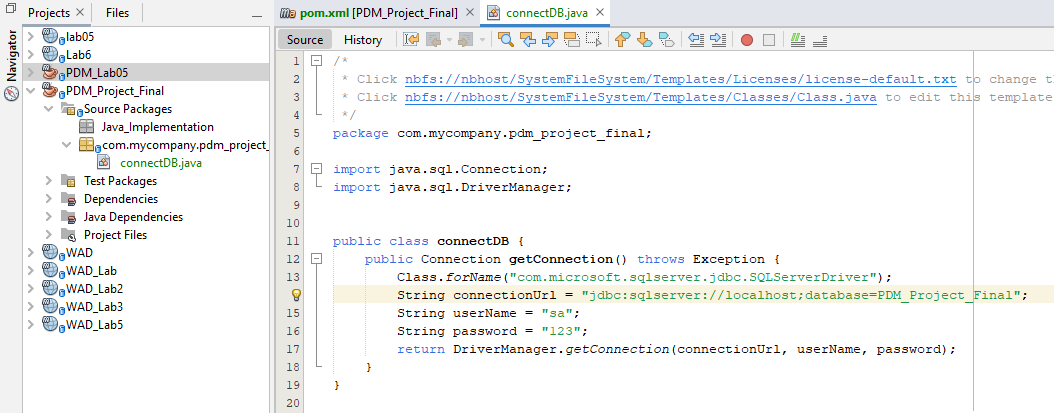
**3.2.1. Connect the database to Java**

We will use the Java programming language in Netbeans Platform for this project. The Version of Netbeans is 13. Depending on the version of Netbeans and type of application, we will connect by using the libraries folder or the Dependencies. In our Netbeans IDE, we cannot figure out the libraries folder (because it is maven project type). Hence, we will use the ‘dependencies’ by modifying the ‘pom.xml’ in the ‘Project Files’ folder.



**fig 10: Adding the dependencies to ‘pom.xml’ file.**

After adding this ‘dependencies’, we will create a class for the connection between this database and our java application.

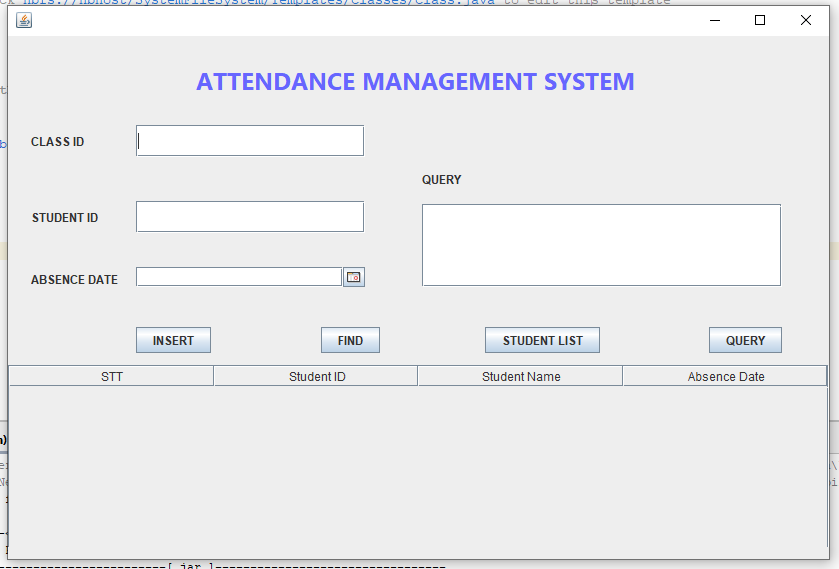


**fig 11:Java Class for the connection database.**

After these steps, we finish the task of connecting the database to our java application.

**3.2.2. Java for GUI**

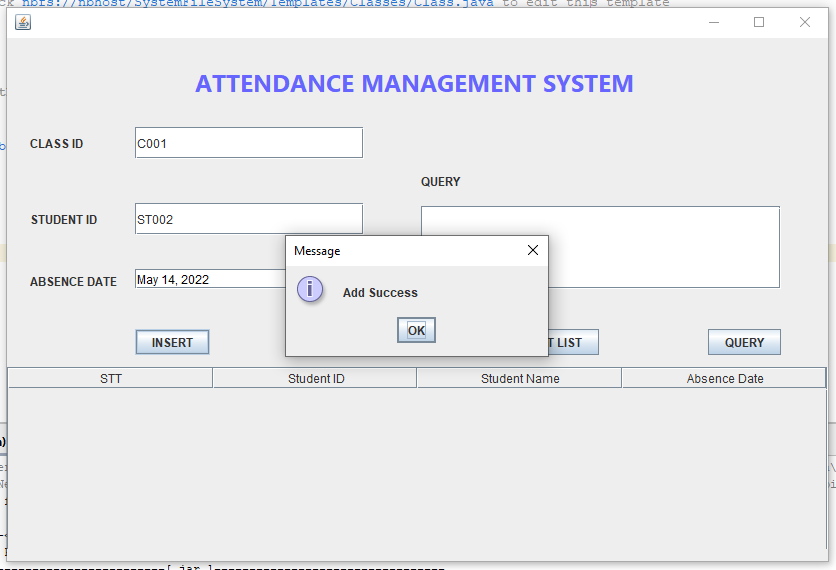
We use the drag-drop function of java.swing to create the form of this project. The Netbeans will generate GUI’s code automatically. We just define some functions for buttons and how the data to be loaded in our GUI.



**fig 12: User Interface for Querying data.**

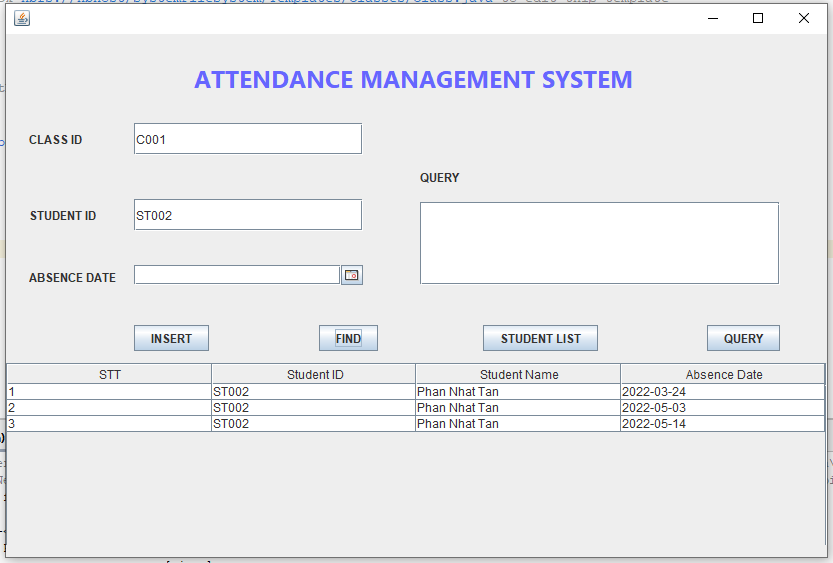
There are four function which help users query data easily:

* INSERT: insert the absence student into the Absence Table. Users will enter the classID, StudentID, Absence date to add a record.



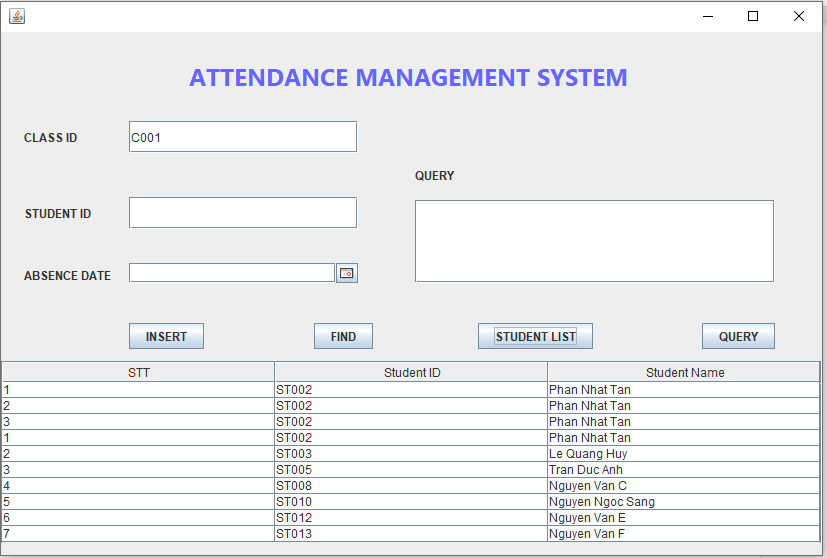
**fig 13: Insert Function.**

* FIND: to return the absence date of a student. Users will input the classID, studentID.



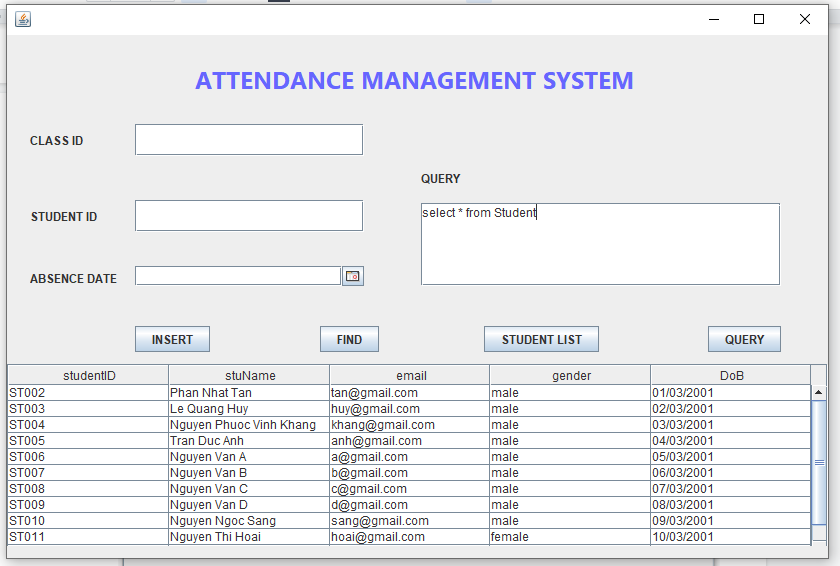
**fig 14: Search Function.**

* Student List: return the student list of a class.



**fig 15: Search Function.**

* Query: In advance , users can input a SQL sentence to get data from the database.

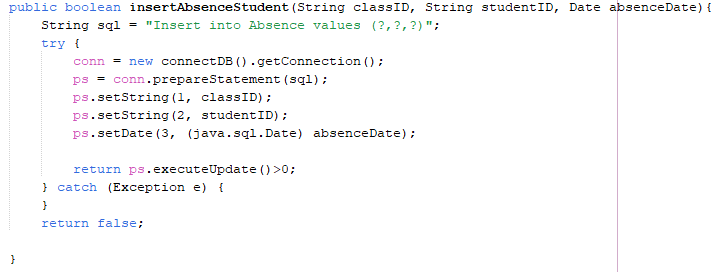


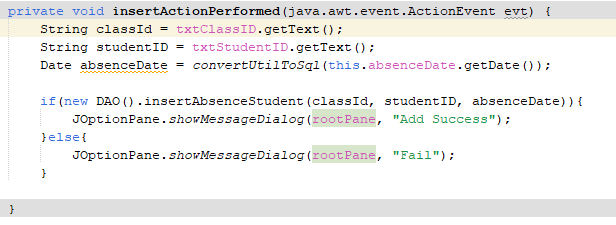
**fig 16: Search Function.**

In this application, we have two main classes:

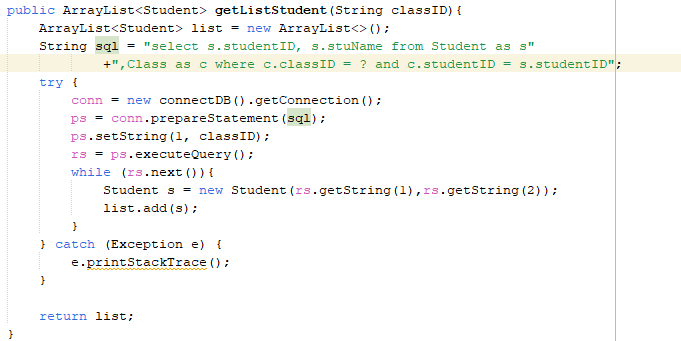
* DAO.java: containing some functions to get data from the database.
* View.java: containing code for GUI and functions to define the method for buttons.

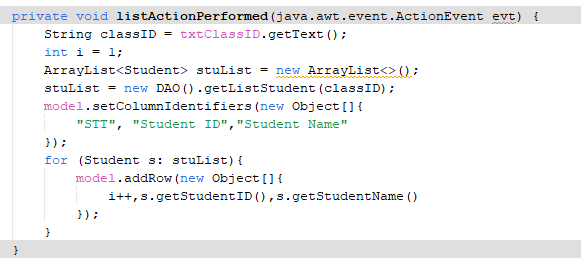
Here is some important code to define the button's function and populate data to the table.



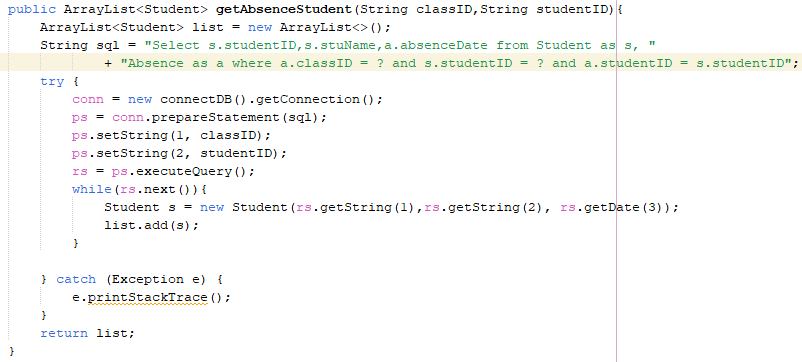


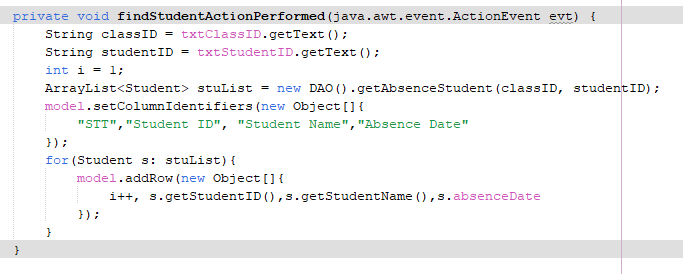
**fig 17: Code for Insert function in DAO.java class and View.java class.**

****

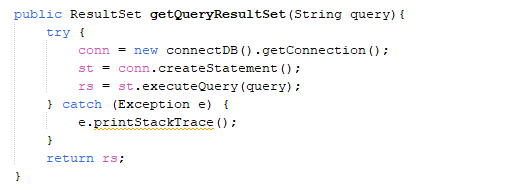
****

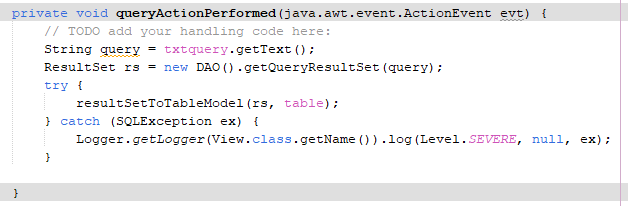
**fig 18: Code for ‘Student List’ function in DAO.java class and View.java class.**

****

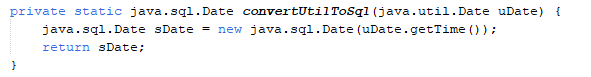
****

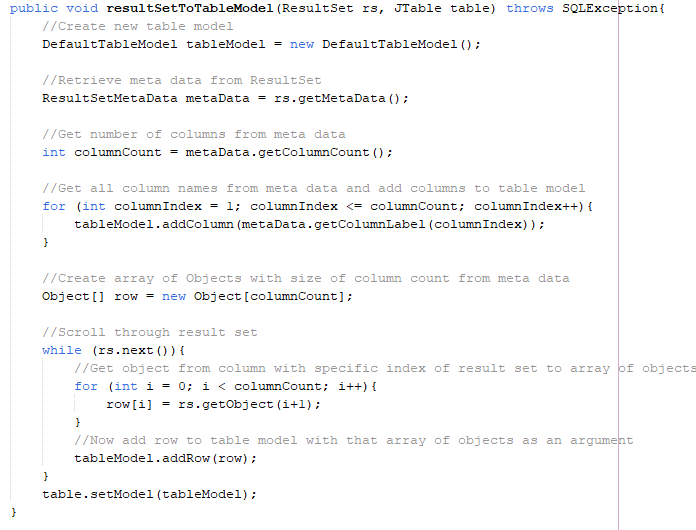
**fig 19: Code for ‘Find’ function in DAO.java class and View.java class.**

****

****

**fig 20: Code for ‘Find’ function in DAO.java class and View.java class.**

****

****

**fig 21: Two functions converting util.date to sql.date and populate data from result set to table .**

**Part 4**

**CONCLUSION**

In conclusion, our group designed and developed a simple database for attendance management. From this database, we can connect to the java platform and implement some function to populate data from database to GUI. Therefore, we can understand SQL knowledge, converting ERD to Relational Model, normalization of database, or how data to be stored in an application.

**Contribution:**

| **Name** | **Contribution** |
| --- | --- |
| Tuấn | 20% |
| Tân | 20% |
| Ánh | 20% |
| Khang | 20% |
| Huy | 20% |

**Part 5**

**REFERENCES**

1. What is a database, Oracle, from :

https://www.oracle.com/database/what-is-database/

1. attendance management, part 2 from:

https://notesformsc.org/attendance-management-part-2/

1. “What is an ER Diagram”, LucidChart, from:

https://www.lucidchart.com/pages/er-diagrams#section\_5

1. How to convert ER Diagram to relational Database ,Learn Database, from:

https://www.learndb.com/databases/how-to-convert-er-diagram-to-relational-database

1. Delete cascade and Update Cascade in SQL server foreign key, SQLshack, from:

https://www.sqlshack.com/delete-cascade-and-update-cascade-in-sql-server-foreign-key/