## Table of Contents

## Overview

### *Interface*

In this project, our group will use the **command line interface** as the user interface for the users to interact with the automated examination system.

When the user launches the application, the user will first enter the “login” menu. The login menu will require the user to enter the user_id. That user_id is identical to the teacher_id(starting with “t”)/student_id(starting with “s”) (refer to the relational schema/ER diagram), the system will identify if it’s a teacher or student ID by the first letter of their ID and enter “student mode” or “teacher mode”.

After logging in, user will be in the main menu of either the “student mode” or the “teacher mode”. If user is in the “student mode” main menu, the user can choose two functions “Take Taker” or “Record Checker”; If the user is in the “teacher mode” main menu, the user can choose three functions “Test Designer”, “Analysis Report”, and “Test Marker”. These functions are based on the requirement document of the project. *Detailed explanation of the function is explained in page 4.*

The password of the login page we assigned to students will be in the format of “abcd” + the last non-zero digit(s) of their student ID and the password we assigned to teachers will be in the format of “abcdt” + the last non-zero digit(s) of their teacher ID.

### *Programming Language/Platforms*

This project will be implemented by Java, using the Java JDBC API, introduced in Lab 5.

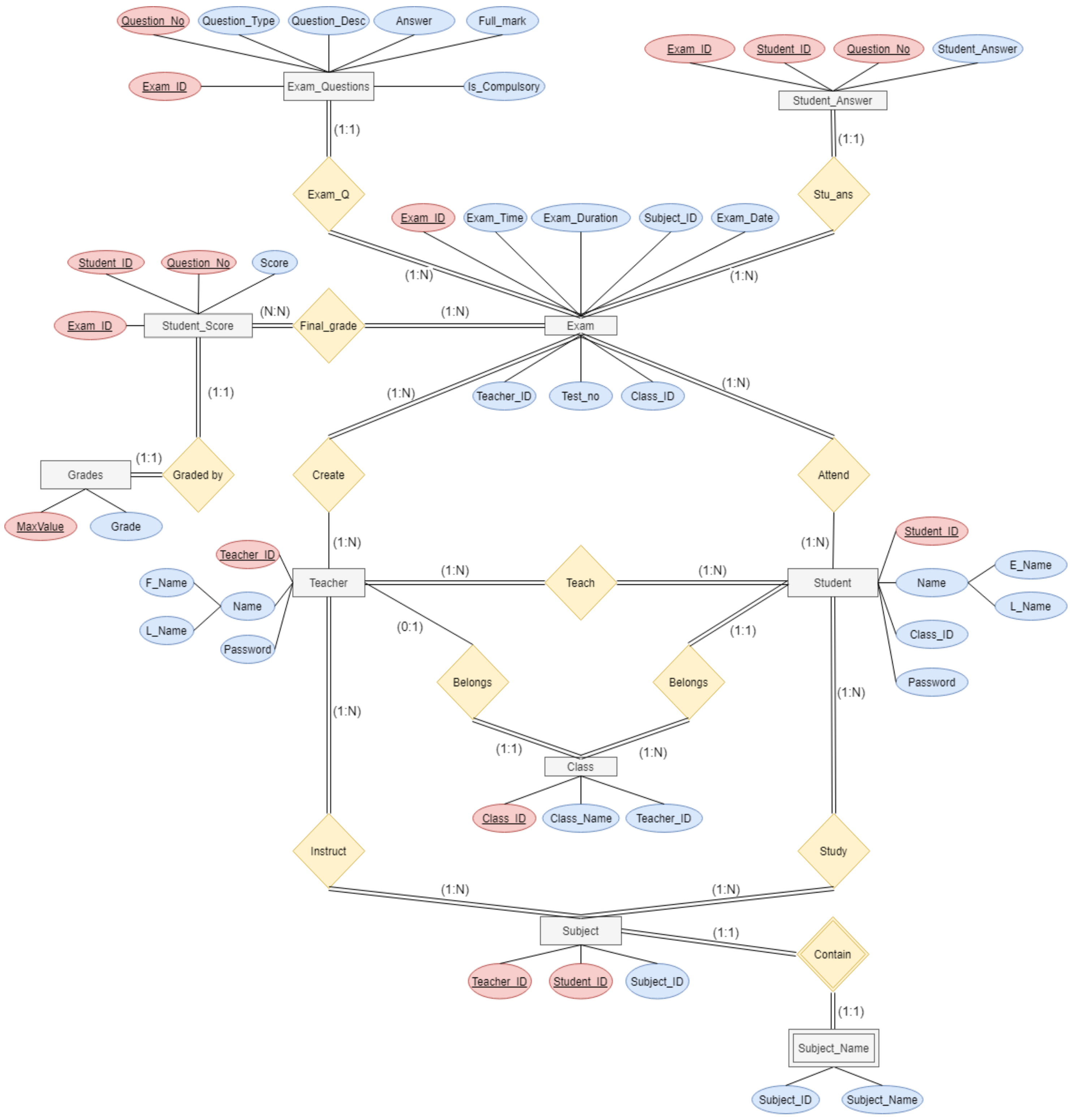
The program will be run using PolyU COMP’s Apollo’s server, the database will also be hosted on the apollo server.

The user interface, logic checking, exam grading functions etc... will be implemented using Java’s functions. When the program needs to access the database, the program will access the JDBC API and control the database using SQL commands.

## Schedule

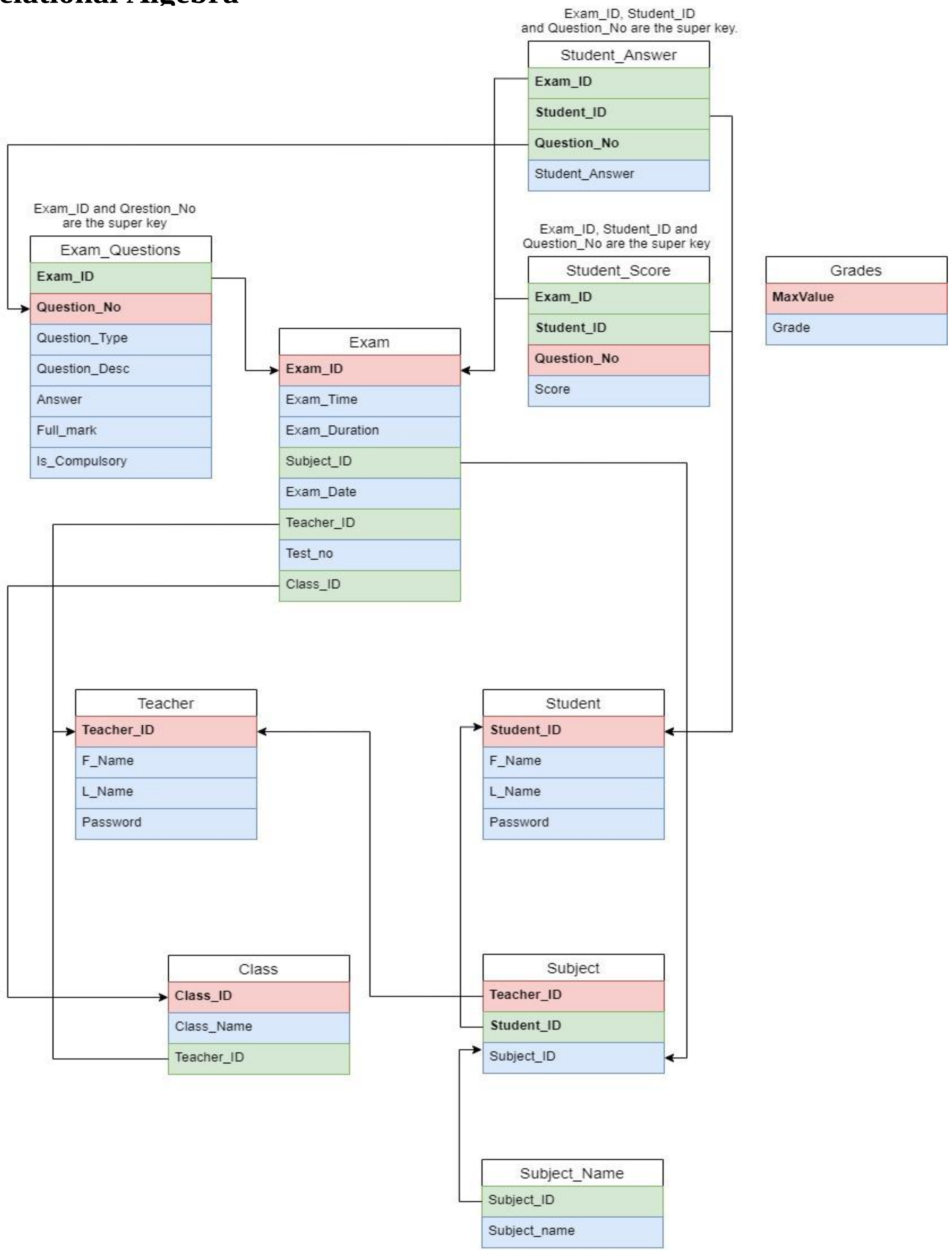
| Date | Event |
|---|---|
| 12 Novermber 2020 – 15 November 2020 | ER diagram |
| 16 November 2020 | Project plan |
| 17 November 2020 – 19 November 2020 | Relational schema |
| 20 November 2020 – 21 November 2020 | Database Implementation |
| 22 November 2020 – 24 November 2020 | Test Designer and Test Taker functions implementation |
| 25 Novermber 2020 | Test Designer and Test Taker interface framework implementation |
| 26 November 2020 | Final check for relational schema |
| 26 November 2020 – 29 Novermber 2020 | Record checker, Test Marker and Analysis Report functions implementation |
| 30 November 2020 | Record checker, Test Marker and Analysis Report interface framework implementation |
| 1 December 2020 – 3 December 2020 | Project report and user guide |
| 4 December 2020 | Testing data and analysis report |
| 4 December | Presentation |

## ER Diagram



The color peach represents Entity. The yellow represents relationship. The red represents super key and the blue represents attribute.

## Relational Algebra



The red represents super key, the blue represents attribute, and the green represents foreign key.

## SQL Description

We separate all entities into three main type: Human entities, Class and subject entities and Exam entities.

For human entities, which contain Teacher and Student.

***Entity Teacher***

It stores the basic information of each teacher.

$\boxed{\text{Teacher\_ID:}}$ The ID of each teacher. It is unique and it also be used for the username of teachers.

Name:

$\boxed{\text{F\_Name:}}$ First name.

$\boxed{\text{L\_Name:}}$ Last name.

$\boxed{\text{Password:}}$ Password of each teacher's account.

***Entity Student***

It stores the basic information of each student and it has the similar structure of entity teacher.

For Class and subject entities, which contain Class, Subject and Subject_Name.

***Entity Class***

$\boxed{\text{Class\_ID:}}$ It is assigned by the system and it is unique.

$\boxed{\text{Class\_Name:}}$ E.g. 4A, 4B, 5C.

$\boxed{\text{Teacher\_ID:}}$ The ID of each class teacher. Foreign key and it mapped to Teacher.Teacher_ID.

***Entity Subject***

The main purpose is to see each student's corresponding teacher in each subject.

$\boxed{\text{Teacher\_ID:}}$ The ID of teacher who instruct particular subject. Foreign key and it mapped to Teacher. Teacher_ID.

$\boxed{\text{Student\_ID:}}$ The ID of teacher who study particular subject. Foreign key and it mapped to Student.Student_ID.

$\boxed{\text{Subject\_ID:}}$ ID of subject.

***Entity Subject_Name***

$\boxed{\text{Subject\_ID:}}$ ID of subject. Foreign key and it mapped to Subject. Subject_ID.

$\boxed{\text{Subject\_Name:}}$ E.g. Economics, ICT.

For Exam entities, which contain Exam, Exam_Questions, Student_Answer, Student_Score and Grades.

***Entity Exam***

Exam_ID: It is assigned by the system and it is unique.

Exam_Time: The starting time of the exam. It is in time type.

Exam_Duration: It all be transferred into minutes.

Subject_ID: It states which subject it is for each exam. Foreign key and it mapped to Subject. Subject_ID.

Exam_Date: It is in date type.

Teacher_ID: The ID of teacher who hold the exam. Foreign key and it mapped to Teacher. Teacher_ID.

Test_no: Test number which is assigned by each teacher.

Class_ID: It states which class will be participated in each exam. Foreign key and it mapped to Class.Class_ID.

***Entity Student_Score***

Exam_ID: Which exam is this answer belongs to. Foreign key and it mapped to Exam. Exam_ID.

Student_ID: Which student is this answer belongs to. Foreign key and it mapped to Student.Student_ID.

Question_No: Which question number is the score belongs to.

Score: The score itself.

***Entity Grades***

The conversion of Marks

MaxValue: The maximum value of each grade. E.g. A+ would be 100 and A would be 95

Grade: The grading. E.g. A+, A

***Entity Exam_Questions***

Exam_ID: It is assigned by the system and it is unique. Foreign key and it mapped to Exam. Exam_ID.

Question_No: The question number of each exam. E.g. Q1 for ICT exam.

Question_Type: 0 stands for multiple-choice. 1 stands for fill in the blank and 2 stands for standard full-length test questions.

Question_Desc: Question description. The question itself.

Answer: The answer for each question.

Full_mark: Full mark for each question.

Is_Compulsory: Boolean type. True stands for compulsory question. Vice versa.

***Entity Student_Answer***

It stores the answers which is answered by student.

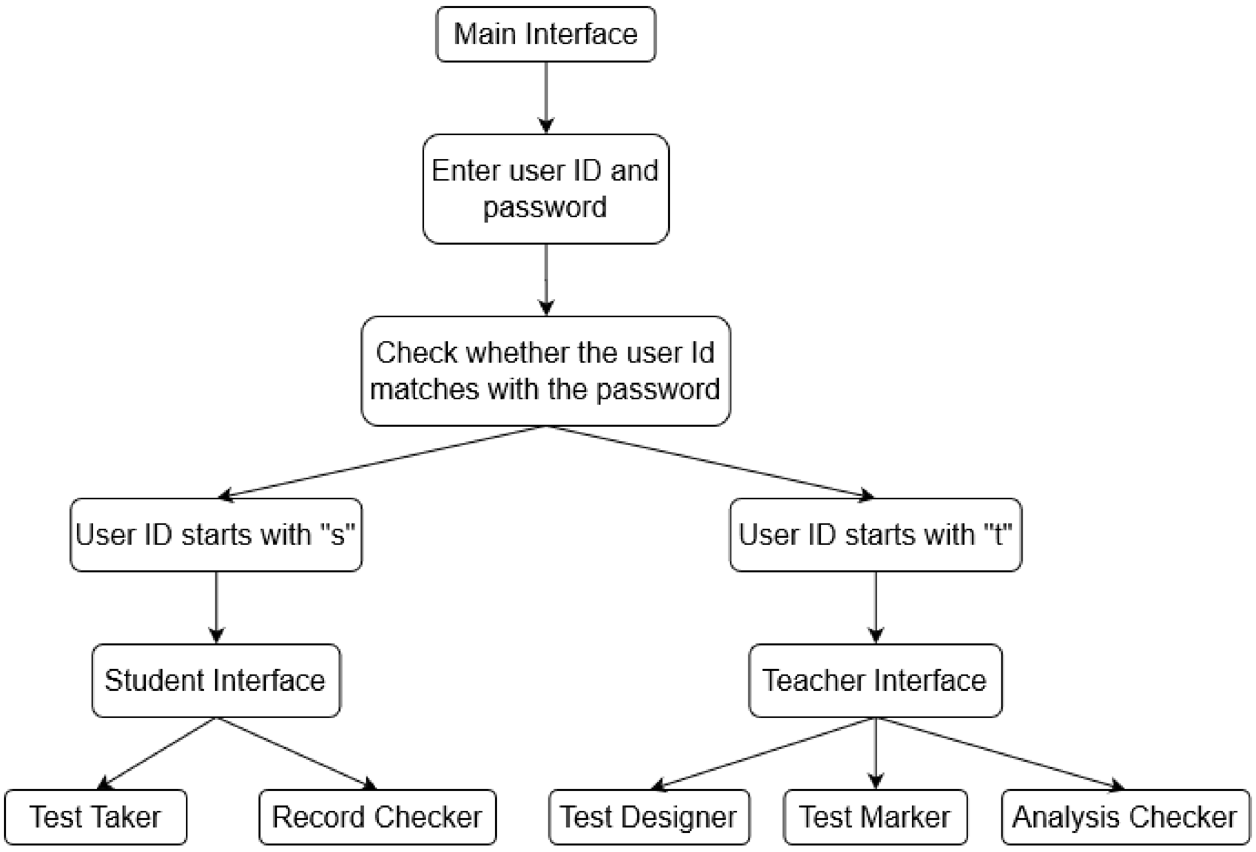
Exam_ID: Which exam is this answer belongs to. Foreign key and it mapped to Exam. Exam_ID.

Student_ID: The ID of student who wrote this answer. Foreign key and it mapped to Student.Student_ID.

Question_No: Which question number is the answer belongs to. Foreign key and it mapped to Exam_Questions.Question_No.

Student_Answer: The answer itself.

## Java JDBC Description

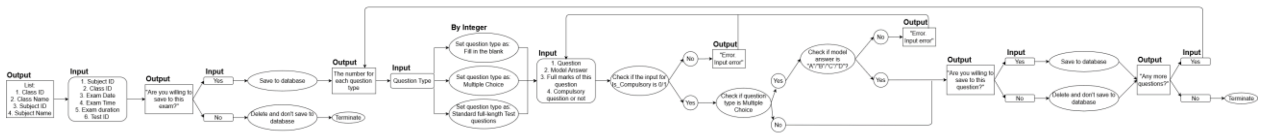


## Java JDBC Functions

### *Test Designer*

The Test Designer function allows teachers to create an exam paper. The usage is similar to that specified in the project requirement document. When the user first enters the function, the interface will require the user to enter the general information of the exam, including the class ID, subject ID, test number, examination date, examination starting time and the duration of the examination.

After that, the command line interface will allow the user to start setting questions, including the question type, question, model answer, the full marks of the question and whether the question is compulsory or not.   Every time a question is added, the database storing the table of exam questions will be updated.


Output
List: 1. Class ID 2. Class Name 3. Subject ID 4. Subject Name
Input
1. Subject ID 2. Class ID 3. Exam Date 4. Exam Time 5. Exam duration 6. Test ID
Output
"Are you willing to save to this exam?"
Input
Yes
No
Save to database
Delete and don't save to database
Terminate
Output
The number for each question type
Input
Question Type
By Integer
Set question type as: Fill in the blank
Set question type as: Multiple Choice
Set question type as: Standard full-length Test questions
Input
1. Question 2. Model Answer 3. Full marks of this question 4. Compulsory question or not
Check if the input for Is_Compulsory is 0/1
No
Yes
Output
"Error. Input error"
Check if question type is Multiple Choice
Yes
No
Check if model answer is "A"/"B"/"C"/"D"?
No
Yes
Output
"Error. Input error"
Output
"Are you willing to save to this question?"
Input
Yes
No
Save to database
Delete and don't save to database
Output
"Any more questions?"
Input
Yes
No
Terminate

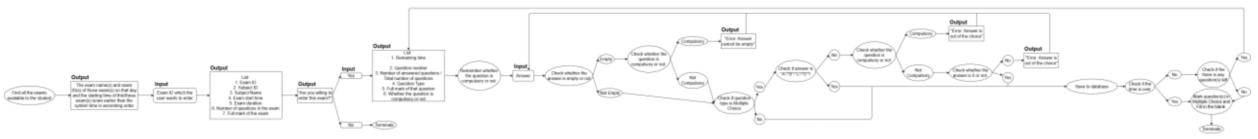
### *Test Taker*

The Test Taker function is only accessed by students. When the user enters that function, the function will query the class ID and student ID from the student, then it will use the student ID to query all the subject ID(s) which the student take.  Finally, it will use the class ID and the subject ID to query all the exams coming up.

When the examination list is queried, the system will check the date and the start time of the exam. Those examination(s) which the examination date equals to the system date and the specified start time is earlier than the system time will be printed out in the ascending order with their exam ID(s).  Students will be required to input the exam ID which corresponds to the exam they want to take and enter to the related examination information page.

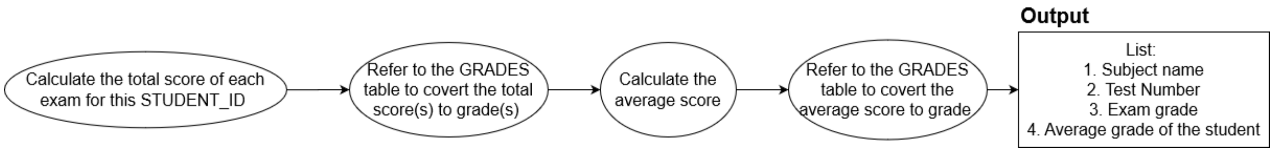
In the examination information page, the system will print out the exam ID, subject ID, subject name, the designated examination start time, the duration of the exam, the number of questions in this exam and the full mark of the examination.  Finally, we will ask the students to confirm whether they want to take the exam now.

During the examination, a timer will be provided first for showing the remaining time left for this examination from the start of that question.  The timer will count until it exceeds the stated exam duration in the database. When the timer exceeds the duration, a break operation will be triggered and the system will end the exam.  For the question part, the system will constantly query the question number, question description, the question type and whether the question is compulsory or not.  The system will print out the question number which the student is currently in, the number of questions they answered, the total number of questions in this exam, the question type, the full mark of this question and whether the question is compulsory or not. Relevant answers will be prompted for users to input according to the question type. If the question is compulsory, the students must input something. Otherwise, the error message will be printed and return back to the question. On the other hand, no input will be classified as skipping the question if the question is not compulsory.


Find all the exams available to the student
Output
The exam name(s) and exam ID(s) of those exam(s) on that day and the starting time of this/these exam(s) is/are earlier than the system time in ascending order
Input
Exam ID which the user wants to enter
Output
List:
1. Exam ID
2. Subject ID
3. Subject Name
4. Exam start time
5. Exam duration
6. Number of questions in the exam
7. Full mark of the exam
Output
"Are you willing to enter this exam?"
Input
Yes
No
Terminate
Output
List:
1. Remaining time
2. Question number
3. Number of answered questions / Total number of questions
4. Question Type
5. Full mark of that question
6. Whether the question is compulsory or not
Remember whether the question is compulsory or not
Input
Answer
Check whether the answer is empty or not
Empty
Not Empty
Check whether the question is compulsory or not
Compulsory
Not Compulsory
Output
"Error: Answer cannot be empty"
Check if question type is Multiple Choice
Yes
No
Check if answer is "A"/"B"/"C"/"D"
No
Yes
Check whether the question is compulsory or not
Compulsory
Not Compulsory
Output
"Error: Answer is out of the choice"
Check whether the answer is 0 or not
No
Yes
Output
"Error: Answer is out of the choice"
Save to database
Check if the time is over
No
Yes
Check if the there is any question(s) left
Yes
No
Mark question(s) in Multiple Choice and Fill in the blank
Terminate

### *Record Checker*

The Record Checker function can only access by the student. This function will query the score of each exam the student takes by the STUDENT_ID and summing up all the scores in each question of each EXAM_ID which the STUDENT_ID is linked to, total mark of each exam will be converted to grade according to the stated grade table in the database. Then, the system will print out the subject ID, subject name, test number, grade and the mean, mode and median grades of that examination for each exam they took.


Calculate the total score of each exam for this STUDENT_ID
Refer to the GRADES table to covert the total score(s) to grade(s)
Calculate the average score
Refer to the GRADES table to covert the average score to grade
Output
List:
1. Subject name
2. Test Number
3. Exam grade
4. Average grade of the student

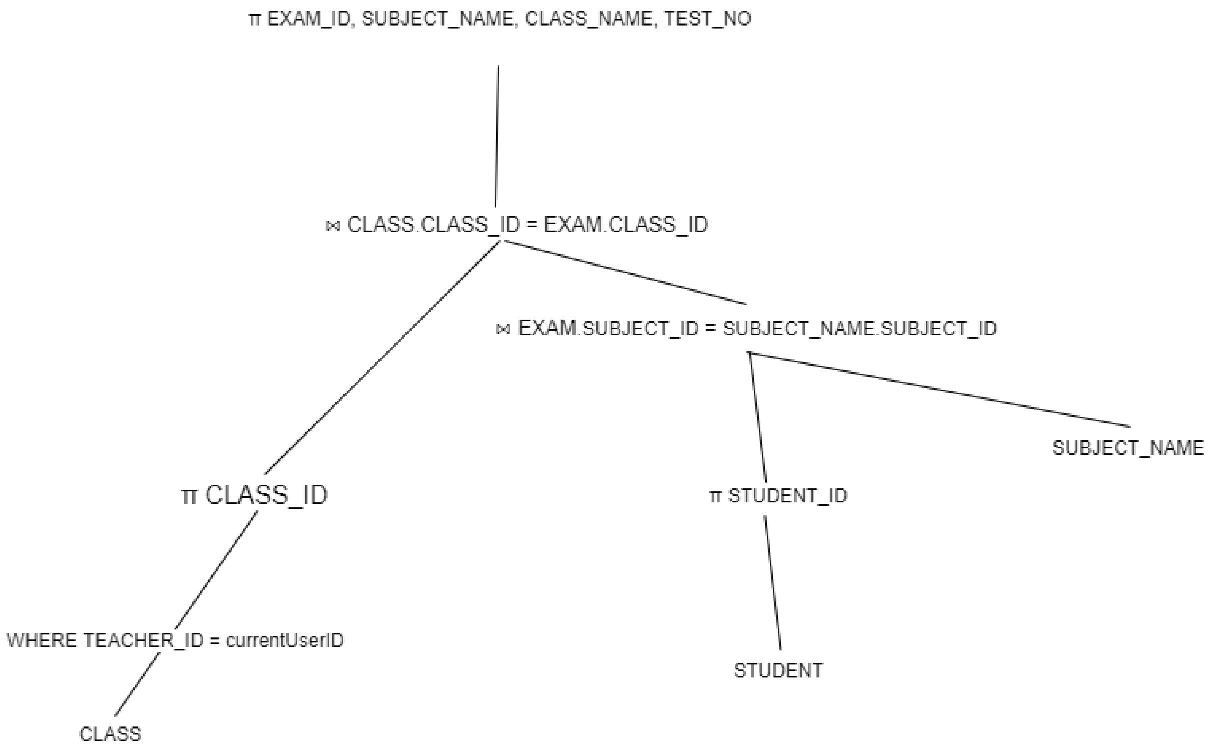
***Analysis Checker***

The Analysis Check function can be divided into 3 parts, which is student performance analysis, subject result analysis and class result analysis.

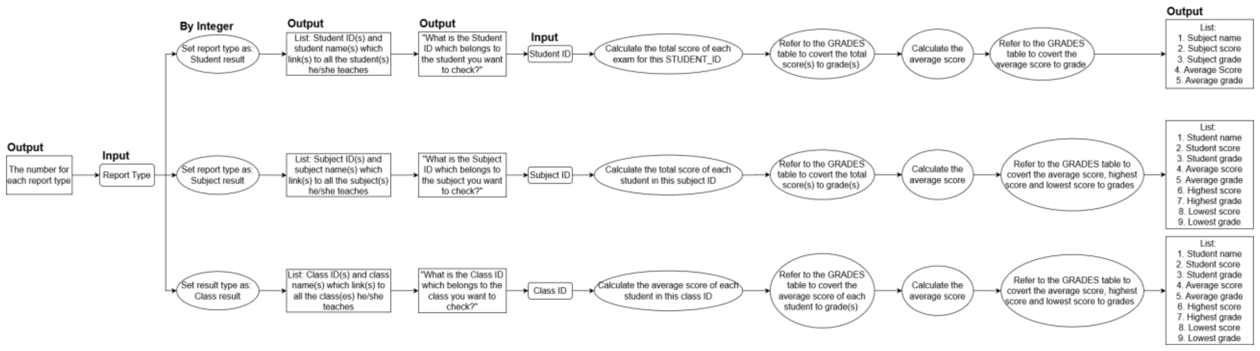
Firstly, in terms of student performance analysis, this function will prompt the user to enter a STUDENT_ID which allows the system to generate a report of that student. The report will include the grade and the total mark of that student (searching by the STUDENT_ID) in each exam that he/she took, and an average score of the student will be calculated and printed.

Secondly, in terms of subject result analysis, this function will show all the student(s)' grade(s) and total mark(s) of all the exam(s) of the subject which linked to the SUBJECT_ID the user inputs. In addition, the average, highest and lowest score will be calculated and printed.

For optimizing the process of getting data from database. We use query optimization to optimize all of our query in this program including this Analysis Checker. The following is the example of query optimization in selecting all of the student who are taught by the current user:


π EXAM_ID, SUBJECT_NAME, CLASS_NAME, TEST_NO
⋈ CLASS.CLASS_ID = EXAM.CLASS_ID
⋈ EXAM.SUBJECT_ID = SUBJECT_NAME.SUBJECT_ID
π CLASS_ID
π STUDENT_ID
SUBJECT_NAME
WHERE TEACHER_ID = currentUserID
STUDENT
CLASS

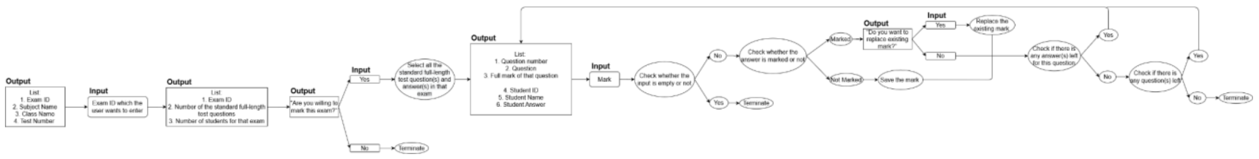
Finally, in terms of class result analysis, this will provide the result of all the students in the selected class based on the CLASS_ID the user inputs, similar to the above function which the basic statistics will be provided in the report.


Output
The number for each report type
Input
Report Type
By Integer
Set report type as: Student result
Output
List: Student ID(s) and student name(s) which link(s) to all the student(s) he/she teaches
Output
"What is the Student ID which belongs to the student you want to check?"
Input
Student ID
Calculate the total score of each exam for this STUDENT_ID
Refer to the GRADES table to covert the total score(s) to grade(s)
Calculate the average score
Refer to the GRADES table to covert the average score to grade
Output
List: 1. Subject name 2. Subject score 3. Subject grade 4. Average Score 5. Average grade
Set report type as: Subject result
List: Subject ID(s) and subject name(s) which link(s) to all the subject(s) he/she teaches
"What is the Subject ID which belongs to the subject you want to check?"
Subject ID
Calculate the total score of each student in this subject ID
Refer to the GRADES table to covert the total score(s) to grade(s)
Calculate the average score
Refer to the GRADES table to covert the average score, highest score and lowest score to grades
List: 1. Student name 2. Student score 3. Student grade 4. Average score 5. Average grade 6. Highest score 7. Highest grade 8. Lowest score 9. Lowest grade
Set result type as: Class result
List: Class ID(s) and class name(s) which link(s) to all the class(es) he/she teaches
"What is the Class ID which belongs to the class you want to check?"
Class ID
Calculate the average score of each student in this class ID
Refer to the GRADES table to covert the average score of each student to grade(s)
Calculate the average score
Refer to the GRADES table to covert the average score, highest score and lowest score to grades
List: 1. Student name 2. Student score 3. Student grade 4. Average score 5. Average grade 6. Highest score 7. Highest grade 8. Lowest score 9. Lowest grade

***Test Marker***

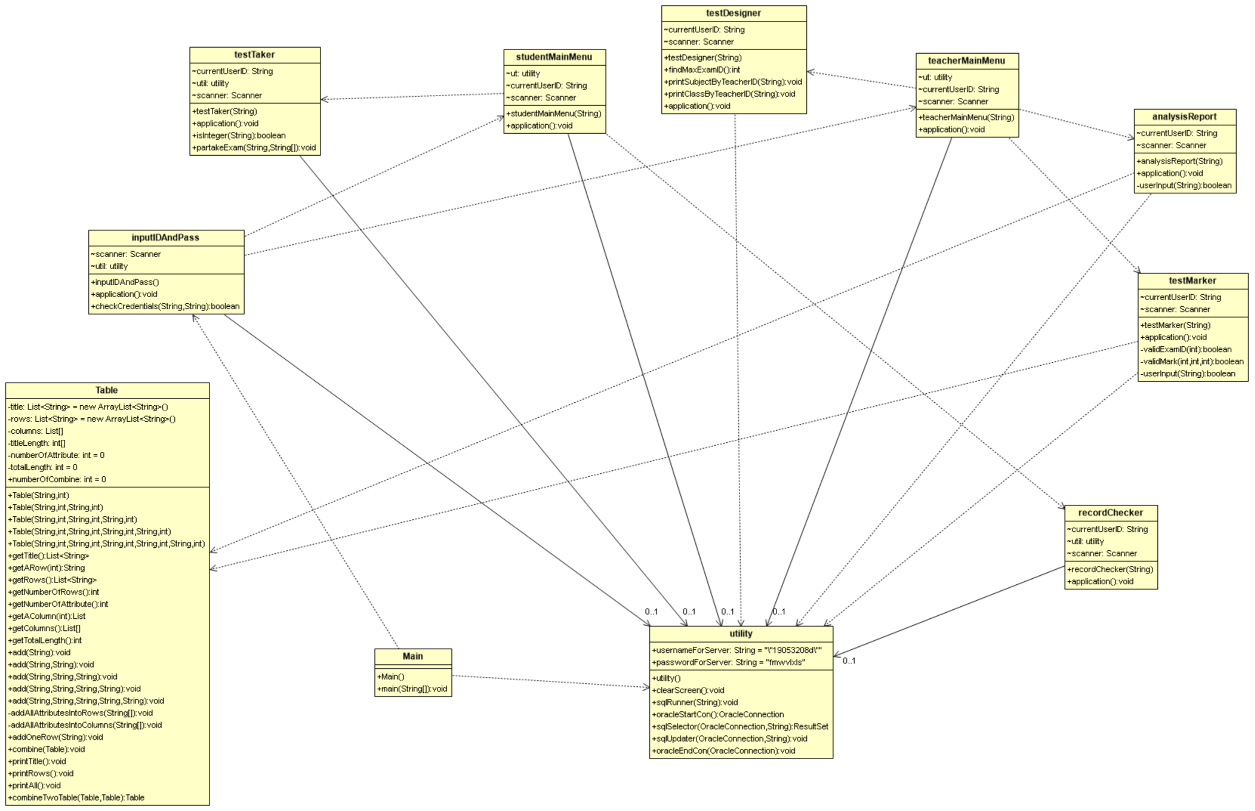
The Test Marker function enables teachers to grade the standard full-length test questions. Since the standard full-length test questions cannot be graded by the system automatically.

The command line interface will list all the exam(s) which need(s) to be marked by the teacher with the EXAM_ID provided and he/she needs to input the EXAM_ID which he/she wants to mark. Then, the system will show the exam ID, number of the standard full-length test questions and number of students who took the exam and ask whether the teacher are willing to mark it now. If the user confirms to mark it, the interface will show the question number, the full question, the full marks of that question and the model answer to that question which set by him/her in the *Test Designer* function for each question in the exam. For students' answers, the interface will show the student name and the student number and the answer that student wrote during his/her test. As a result, the teacher can easily mark the work of the student and input the score to the system, which will be used for calculating the total mark of the examination for each student. For the ease of use, the system will show all the answers to the same question one-by-one, in other words same question but answers from different students, which can facilitate the teacher on the grading process.



Output
List: 1. Exam ID 2. Subject Name 3. Class Name 4. Test Number
Input
Exam ID which the user wants to enter
Output
List: 1. Exam ID 2. Number of the standard full-length test questions 3. Number of students for that exam
Output
"Are you willing to mark this exam?"
Input
Yes
No
Terminate
Select all the standard full-length test question(s) and answer(s) in that exam
Output
List: 1. Question number 2. Question 3. Full mark of that question 4. Student ID 5. Student Name 6. Student Answer
Input
Mark
Check whether the input is empty or not
No
Yes
Terminate
Check whether the answer is marked or not
Marked
Not Marked
Output
"Do you want to replace existing mark?"
Input
Yes
No
Replace the existing mark
Save the mark
Check if there is any answer(s) left for this question
Yes
No
Check if there is any question(s) left
Yes
No
Terminate

## Class diagram for Java JDBC file



*All of the images in this document are attached in the report_pics folder for your reference.