

**Faculty of Computing**

**SCSD2523**

**Database**

***2015/2016***

***SESSION 08***

***SHARIN HAZLIN BINTI HUSPI 11809***

#### **Project Task 3: Logical Design & System Development**

YAP YOONG SENG

A14CS0150

MOHAMMAD ZULKIFLI BIN ZAINUL

A14CS0173

QURATUL AIN NABIHAH BINTI MOHD HIZAM

A14CS0126

# **STUDENT SUBJECT REGISTRATION SYSTEM**

## The Conceptual Design (from Task 2)

In this student subject registration system, there are 4 entities are involved. The entities are including Student, Class, Subject and Staff.

In the Student entity, the attributes are student name, matric number, address, class name, contact number, student ID, identity card number, e-mail and subject code. However, there are some attributes that are containing similar data will be grouped under an attribute. For example, the first name and last name of a student are grouped under student name attribute. The matric number is the primary key of Student entity. The address attribute contains street, city, postcode and state. The class name is foreign key for Class entity which originally from Class entity as primary key. It represents the class that the particular student will be entered. For instance, the form 4 science stream student is arranged into either 4A, 4B, or 4C class, the form 3 student that has greater rank is arranged into 3A or 3B class. In the contact number attribute, it has mobile phone number and home telephone number. The student ID is the account name of the student that login into online system for administration of timetable, tuition fees, and examination results. The subject code is foreign key in the Class entity which is from Subject entity as primary key. The subject code is a multivalued attribute that holds the code of subjects that particular student is taking. A student can takes at least 8 subjects and maximum of 12 subjects.

Besides that, class name, number of student in class, staff number (teacher) and staff number (staff) are the attributes for Class entity. The class name is the primary key, while staff number (teacher) and staff number (staff) are the foreign keys.

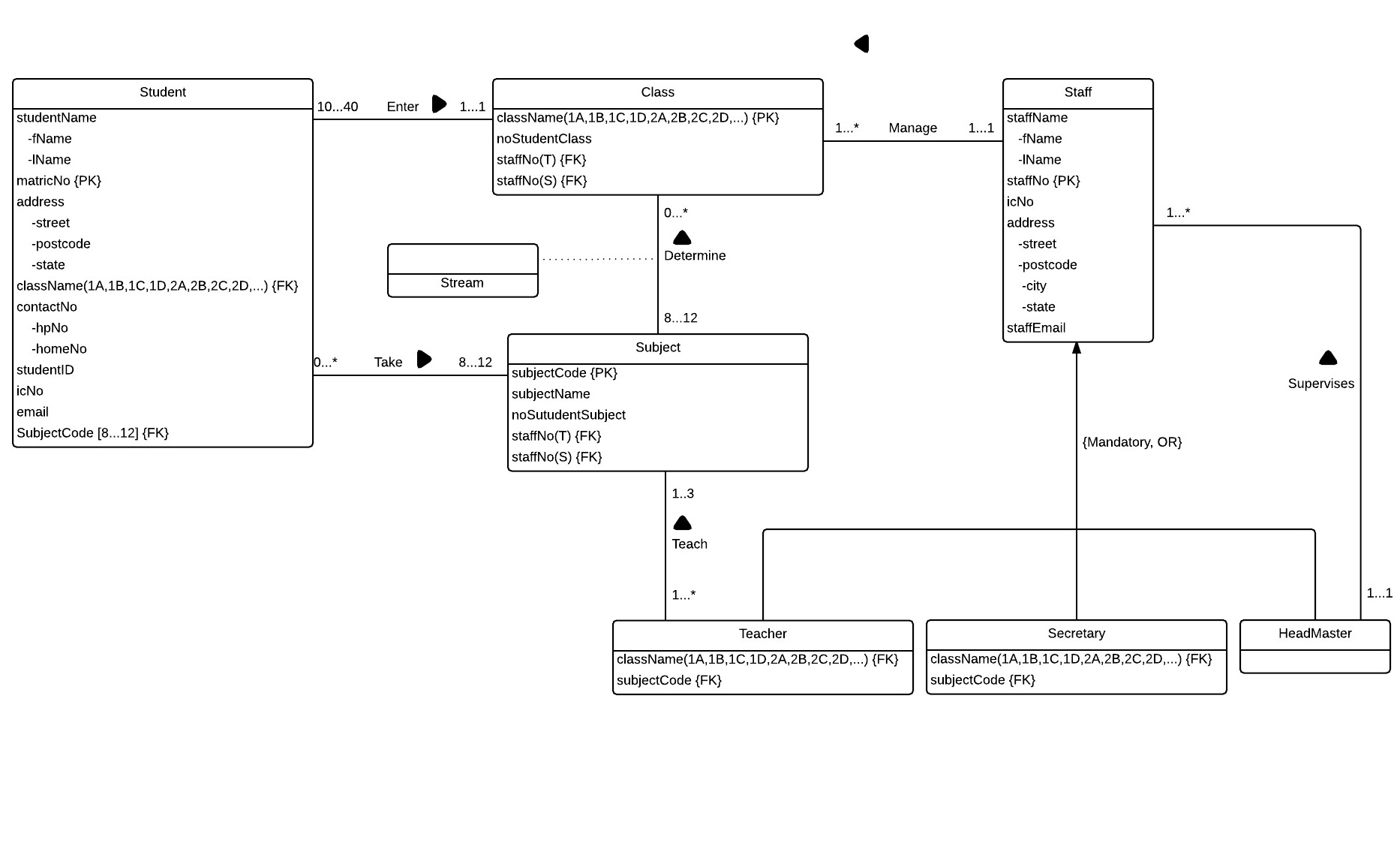
For the Subject entity, it has subject code, subject name, number of student that taking the subject, staff number (teacher) and staff number (staff) attributes. The subject code is obviously a primary key in Subject entity, complementing the staff number (teacher) and staff number (staff) as foreign keys.

The another prime entity is Staff entity. It has a superclass Staff and also 3 subclasses which are including Teacher, Secretary, and Headmaster. The Staff superclass undergoes generalization, containing attributes of staff name (first name and last name), staff number, identity card number, address (street, city, postcode and state) and staff e-mail. The staff number is primary key in the entity. Meanwhile in the subclasses, the Teacher holds subject code and class name attributes, as foreign keys from Subject and Class entities respectively. The Secretary and Headmaster subclasses got no attribute. There are 2 constraints are applied in this entity between superclass and subclasses, that are participation constraint and disjoint constraint. The participation constraint involves mandatory which is the Staff superclass must be member of Teacher, Secretary and Headmaster subclasses. While for disjoint constraint, it is non-disjoint, means the Staff entity occurrence can be a member of more than a subclass.

In term of entity relationships, a student can only enter to a class. There is limitation for number of student in class, indicating that there are minimum 10 students up to 40 students in a class. For the Student and Class entities, a student takes at least 8 subjects and maximum 12 subjects. A particular subject can be taken by no student or many students. Regarding to the relationship between Subject and Class entities, a subject determines the class by its stream. For example, the Biology subject belongs to science stream, therefore it identifies the student to enter the class A, B or C of form 4 and 5. The Economy and Art subjects belongs to account and art stream separately, so apply to class D, E, F (account stream class), and I, J, K (art stream class) respectively. A class also is determined by its subjects following the stream. For instance, a student which is taking 9 subjects that including Malay language, English, Mathematics, History, Moral, Additional Mathematics, Biology, Chemistry and Physics, the student is arranged to enter class A, B or C. The Staff and Class also have their relationship. A staff manages a class up to many classes, while a class only can be managed by a staff. The Teacher and Subject entities, a teacher teaches one to three subjects, and a subject can be taught by minimum 1 teacher up to many teachers. A headmaster supervises 1 to many staffs, but a staff only can be supervised by one and only headmaster.

For the entity integrity for our ERD database is that none of our primary key is null. it. All this primary key matricNo, subjectCode, className and staffNo in our ERD must have value in it. In our Student table we got 2 foreign key which is className that reference with the Class table and subjectCode that reference from Subject table. For Class table we got two foreign which is staffNo(T) for the Teacher table from the Staff subclass and staffNo(S) for the Secretary tbale from the Staff subclass. In the Teacher table we got subjectCode and also className as a foreign key same as Secretary table also got same foreign key. The Subject table we got staffNo(T) for the teacher so we know what kind of subject this teacher teach and staffNo(S) for secretary so we know which secretary that handle particular subject.

**ERD for the Subject Registration System**



## The Normalization process from conceptual design

The normalization process that producing a set of relations with desirable properties as well as satisfying the data requirements of Student Subject Registration System is applied. The normalization approach will be used is validation technique.

BCNF

3NF

2NF

1NF

### 1NF

The first step in normalization process is identifying functional dependencies in SubjectRegistration at first normal form (1NF) phase. For the functional dependencies, the matric ID of student, class name and teacher’s staff number determine the student’s name, stream name, subject code that student taking the subject, subject nameand teacher’s name. While for partial dependencies, the class name determines the stream of class, for example 5 Perdana is science stream class. The subject code determines the subject name, for instance TR111 is Malay Language. The staff number also can determine a teacher’s name. The alternative dependency is the staff number can determine the subject code and class name, this means that a teacher that teaching a particular subject along with which class can be identified by tracking the staff number.

The 1NF relation with instance of the relations:

SubjectRegistration (matricID, studentName, className, streamName, subjectCode, subjectName, staffNo, staffName)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| matricID | studentName | className | streamName | subjectCode | subjectName | staffNo | staffName |

### 2NF

The following is the listing of functional dependencies based on 1NF:

FD6: matricID, className, subjectCode, staffNo -> studentName, streamName, subjectName, staffName

FD1: matricID -> studentName

FD2: className -> streamName

FD3: subjectCode -> subjectName

FD4: staffNo -> staffName

FD5: staffNo -> subjectCode, className

The primary keys are including matric ID of student, class name, subject code and teacher’s staff number. The subsets of primary key are class name, subject code and teacher’s staff number. Since FD2, FD3, and FD4 are partial dependencies, they should be isolated from the relation, create and name new relations for each of partial FDs respectively.

The 2NF relations with instances of the relations:

(NOTE: some of the lower prior attributes like password, address, email and others will be added, 1NF did not added for the ease of clearer illustration and avoid overwhelming the 1NF)

Student (matricID, studentName, password, icNumber, jantina, agama, kaum, street, state, hpNo, email, className, streamName)

Class (className, streamName)

Subject (subjectCode, subjectName)

Teacher (staffNo, staffName, password, icNumber, street, state, hpNo, email, classSubTCode, className, subjectCode)

### 3NF

The following is the listing of functional dependencies based on 2NF:

Student (matricID, studentName, password, icNumber, jantina, agama, kaum, street, state, hpNo, email, className, streamName)

FD6: matricID, className, subjectCode, staffNo -> studentName, streamName, subjectName, staffName

FD1: matricID -> studentName

FD2: className -> streamName

Class (className, streamName)

FD2: className -> streamName

Subject (subjectCode, subjectName)

FD3: subjectCode -> subjectName

Teacher (staffNo, staffName, password, icNumber, street, state, hpNo, email, classSubTCode, className, subjectCode)

FD4: staffNo -> staffName

FD5: staffNo -> subjectCode, className

FD7: staffNo -> classSubTCode

FD8: classSubTCode -> staffNo, subjectCode, className

The Teacher relation is enhanced by adding classSubTCode, className, subjectCode attributes. In the Student relation, the FD2 represents the class name determines the stream name. Since it is already appeared in Class relation, the streamName in Student will be removed, leaving className as foreign key. The stream name will also be separated to create a new relation named Stream that containing stream name attribute, the stream name in Class relation acts as foreign key. In the Teacher relation, the FD8 staffNo, subjectCode, className are transitive dependency on classSubTCode via staffNo. Therefore, all attributes from FD8 will be separated to create a new relation Class\_Sub\_Teacher, classSubTCode as primary key while staffNo, subjectCode, className act as foreign keys.

These are the final set of normalized relations in the 3NF derived from the SubjectRegistration relation.

Student (matricID, studentName, password, icNumber, jantina, agama, kaum, street, state, hpNo, email, className)

Class (className, streamName)

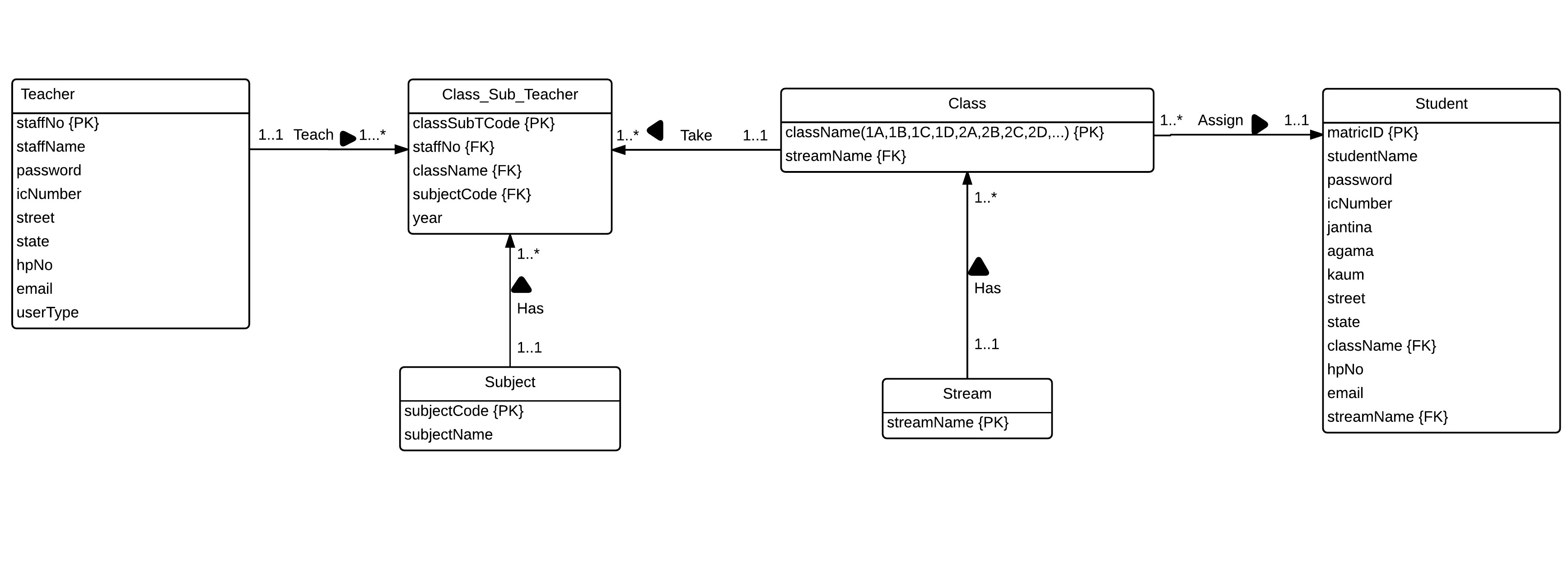
Stream (streamName)

Class\_Sub\_Teacher (classSubTCode, staffNo, subjectCode, className, year)

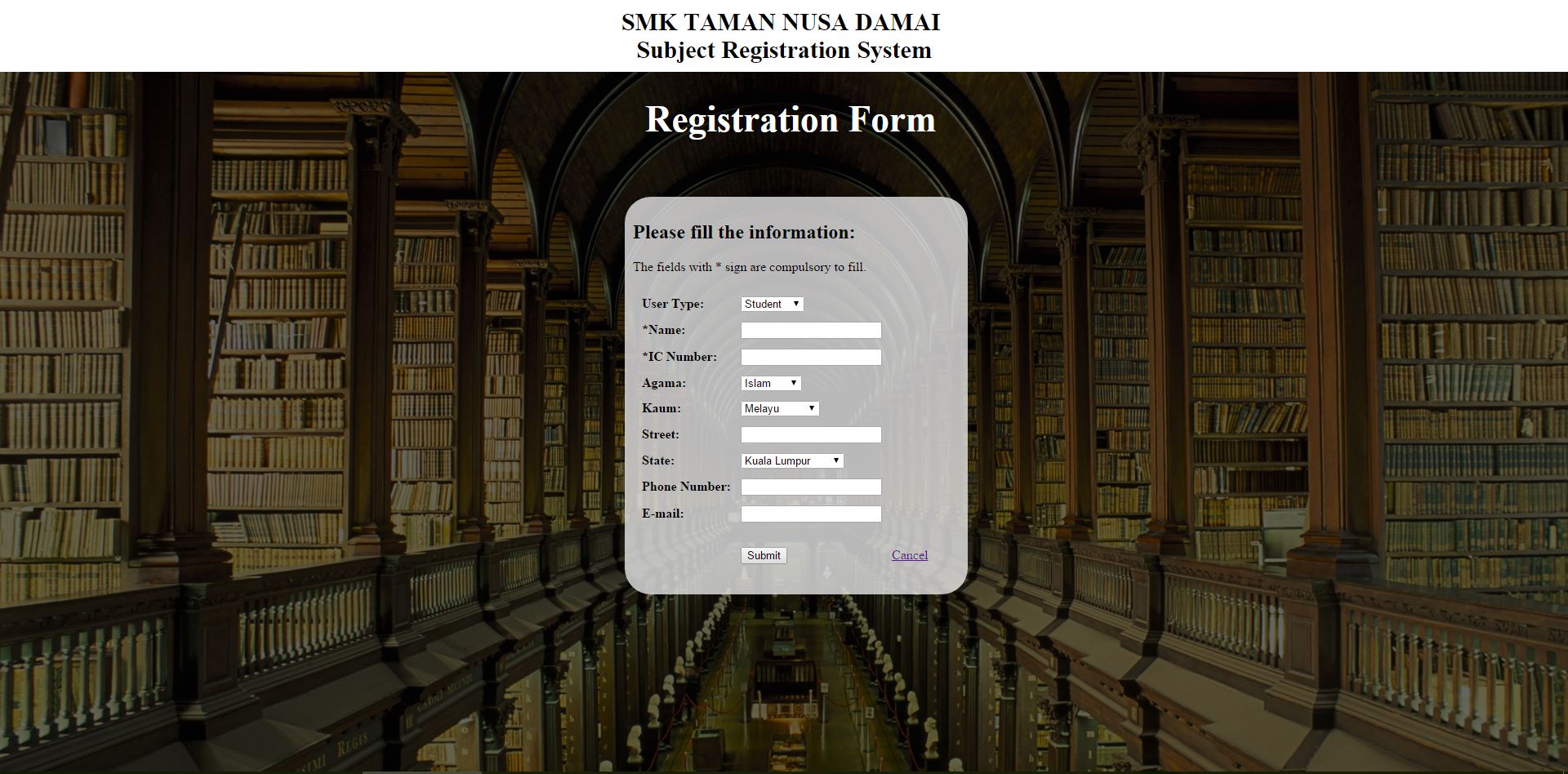
Subject (subjectCode, subjectName)

Teacher (staffNo, staffName, password, icNumber, street, state, hpNo, email)

## Logical design of DBMS



## Examples of queries



*This interface shows the registration for the Subject Registration System.*

$sql = "insert into student(matricID, studentName, password, icNumber, agama , kaum , street , state, hpNo ,email,userType) " .

"values('$usercombine', '$name', '$ic', '$ic', '$agama', '$kaum', '$street', '$state', '$mobile', '$email', '$userType')" ;

Inserting the name, IC number, religion, race, street, state, phone number and email into table Student.



*This interface shows the login page of the Subject Registration System.*

if($user == "T")

{

$result = mysqli\_query($conn,"select staffNo,staffName,password from teacher where staffNo='".$userid."' and password ='".$pass."'");

}

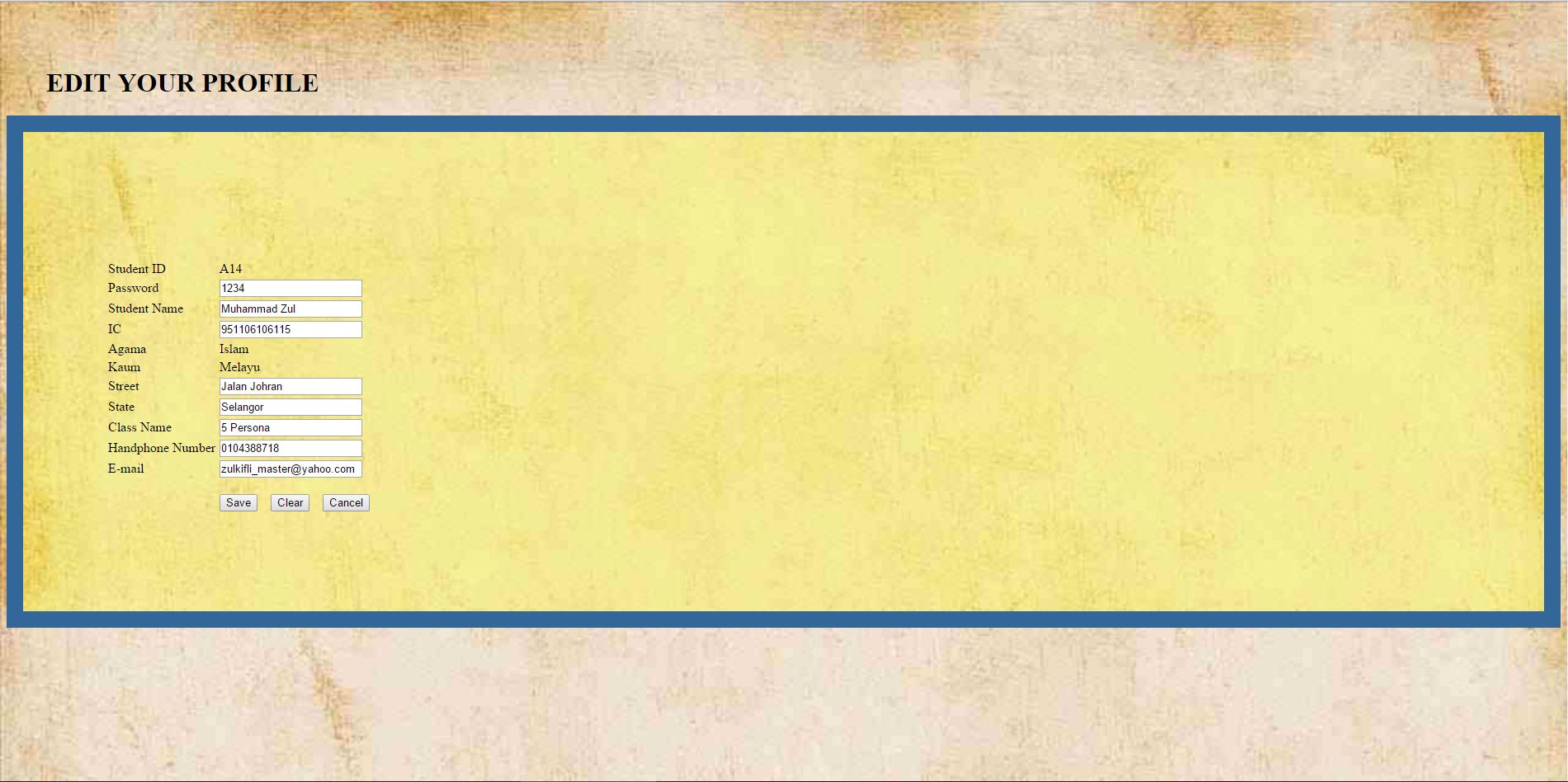
if($user == "A")

{

$result = mysqli\_query($conn,"select matricID,studentName,password from student where matricID='".$userid."' and password ='".$pass."'");

}

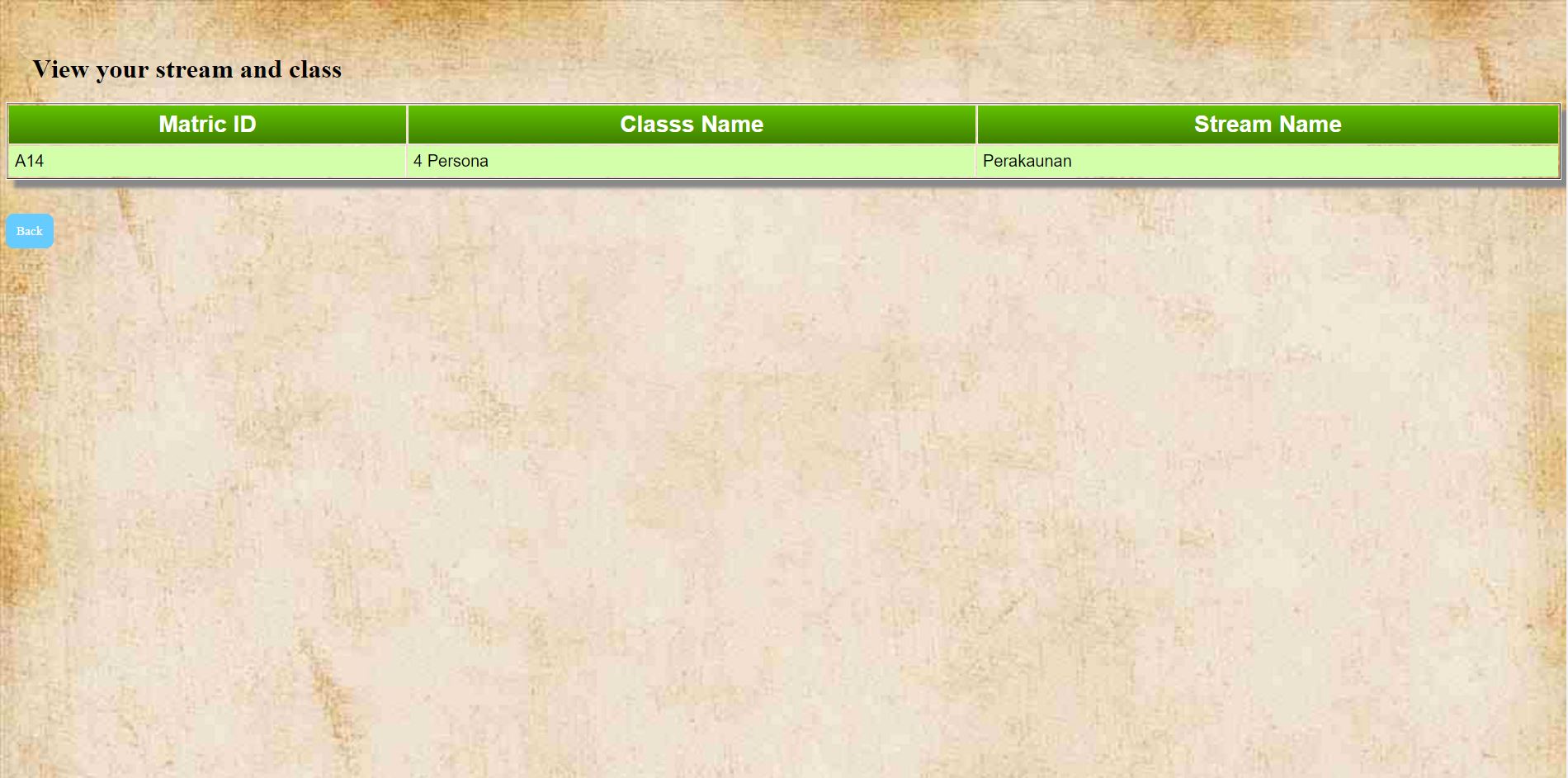
Select from table Student or Teacher where matricID or staffNo match the username and password entered from interface match the password from database.

**

*This interface shows the profile edit of a student.*

UPDATE student SET password = '".$password."', studentName = '".$Name."', icNumber = '".$IC."', street = '".$street."', state= '".$state."', hpNo= '".$mobile."' , email= '".$email."' WHERE matricID= '".$userid."'";

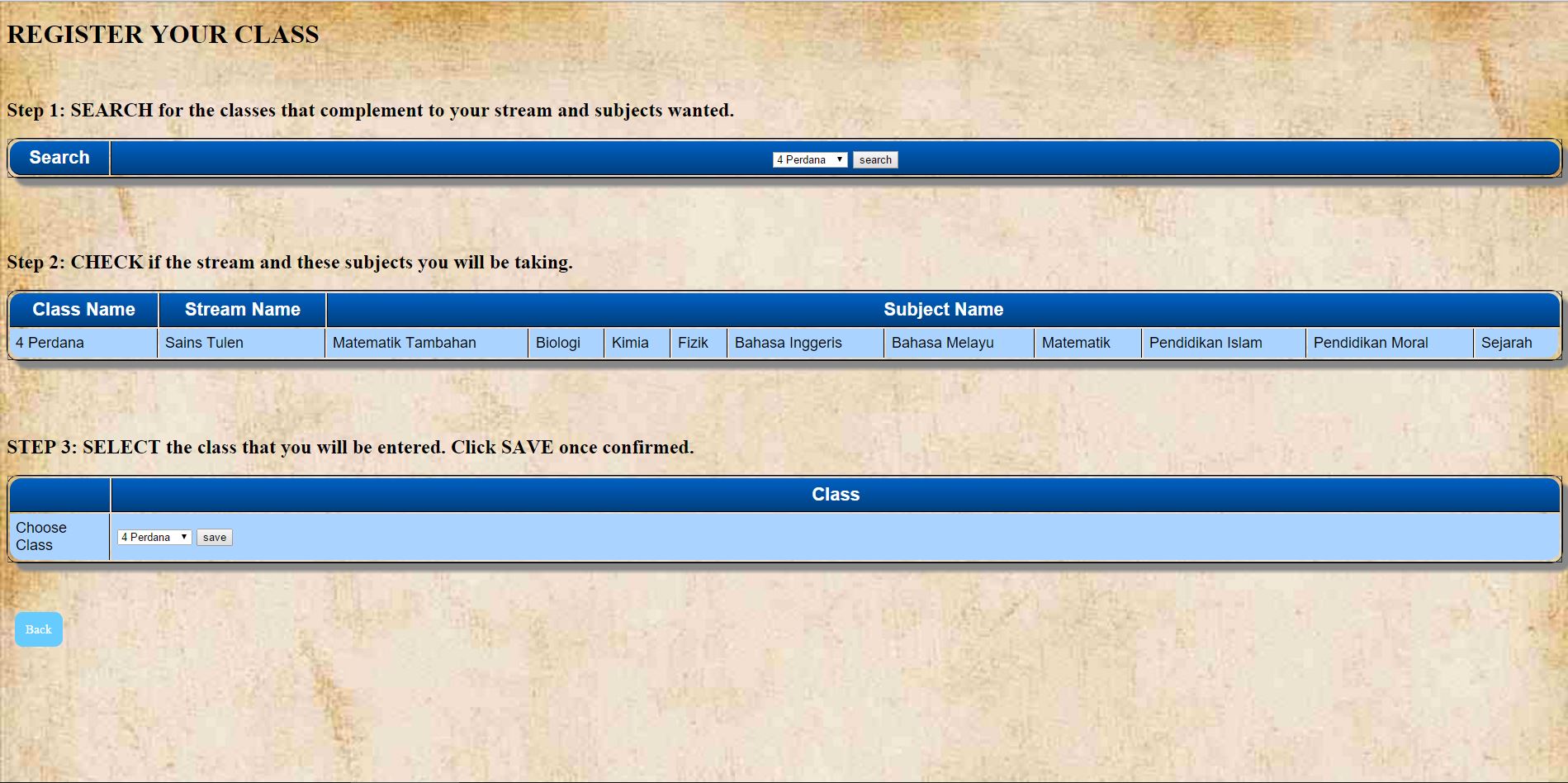
Update the table Student that password, student name, IC number, street, state, phone number and email from interface into tuple database of Student where matricID of interface matches the matricID in database.



*This interface shows the viewing stream and class of a student.*

SELECT matricID,className,streamName FROM student where matricID='".$matricID."'

Select matric ID, class name and stream name from table Student where matricID of interface matches the matricID in database.



*This interface shows the registration of subjects based on class of a student.*

In STEP 1:

SELECT subjectCode FROM class\_sub\_teacher where className='".$class."'

Select subject code from Class\_Sub\_Teacher where class name from interface matches the class name in database.

In STEP 2:

SELECT className,streamName FROM class where className='".$class."'

Select the class name and stream name from table Class where class name from search in STEP 1 matches the class name in database.

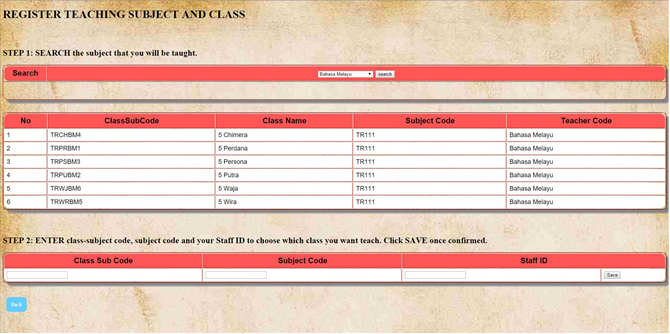
SELECT subjectName FROM subject WHERE subjectCode = '".$subjectcode."'

Select the subject name from table Subject where subject code from search in STEP 1 matches the subject code in database.

In STEP 3:

UPDATE student SET className = '".$class."',streamName = '".$stream."' where matricID = '".$matric."'

Update the class name from interface into table Student and automatically update the stream name based on class where matricID of interface matches the matricID in database.



*This interface shows the registration of teaching subject and class of a teacher.*

In STEP 1:

SELECT class\_sub\_teacher.classSubTCode, class\_sub\_teacher.className, class\_sub\_teacher.subjectCode, subject.subjectName, class\_sub\_teacher.staffNo FROM class\_sub\_teacher INNER JOIN subject ON class\_sub\_teacher.subjectCode=subject.subjectCode WHERE class\_sub\_teacher.subjectCode='".$subject."' ORDER BY class\_sub\_teacher.classSubTCode

In STEP 2:

UPDATE class\_sub\_teacher SET staffNo = '".$staff."' where classSubTCode = '".$streamc."'

select className from class\_sub\_teacher where classSubTCode='".$streamc."'

UPDATE teacher SET classSubTCode = '".$streamc."', subjectCode = '".$subjectcode."' , className = '".$class."' where staffNo = '".$staff."'