**Student Registration Database Management System**

**Project Implementation Report**

**IST659M003**

**Karan Ashar**

**Dhwani Gandhi**

**CONTENTS**

1. **Project Summary**
2. **Entity and Attribute Table**
3. **Entity Relationship Diagram**
4. **Database System Architecture**
5. **SQL Scripts for Creating Tables and Inserting Values**
6. **Output for Create and Insert SQL Scripts**
7. **Major Data Questions**
8. **Interfaces**
9. **Reports**
10. **Triggers**

**PROJECT SUMMARY**

Navlakhi® Tutorials is an educational service running for the past 25 years in Mumbai, India. It is a renowned organization in the Education Sector specializing in Engineering courses. There are approximately 100 students who enroll in the tutorials every semester. As of today, there is no automated database management system to store the details of the incoming students.

Currently the process of registering is manual. The data is stored in various Excel Spreadsheets. The students need to come over or call to register themselves. To know which subjects are available for the current semester they must do the same.

A relational database will be able to store the data of the incoming students who want to register for the tutorials automatically and in a logical format. There will be a student entity storing information about the student registering for the tutorials. The data stored would consist of his name, phone number, the college he goes to, the branch he is studying in and his email. The database would include a table of all those colleges across Mumbai selected by the tutorials and the respective branches available in those colleges. The database will also include all the subjects available in each branch at each college. We create another entity to store the semester when the subject is available. This is then connected with the student username so that we can know about all the subjects the student has ever taken up in the tutorials. We have also created a payment entity to keep a track of the payment made by each student for an individual subject.

Data can be drawn out of the system easily regarding who is registered for which subject. The students can now register for the course without any difficulty.

The users of the system are categorized into two groups:

1. Students: Enroll for the tutorials and register for subjects by viewing the available subjects.
2. Admin: Enters payment information, views reports and updates data when needed.

The following business rules were followed while developing the project.

1. A student needs to register only once unless his/her college or branch changes.
2. New registrations are done every six months.
3. If a student fails a subject and he takes up that subject at the tutorials again, he need not pay for the subject again i.e. he can continue that subject free of cost.
4. The payment should be made for each subject individually by a student.
5. The student registration is confirmed only once the payment is done.

**ENTITY AND ATTRIBUTE TABLE**

1. **College Table:** Stores details about specific engineering colleges in Mumbai

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name:**  **College** | **Attribute Name** | **Field Type** | **Required** | **Description** |
| Primary Key | College\_ID | INTEGER | Yes | Unique Identifier for the College |
| Other Attributes | Name | VARCHAR(100) | Yes | Name of the College |
|  | Address\_Line\_1 | VARCHAR(100) | Yes | Address Line 1 of the College |
|  | Address\_Line\_2 | VARCHAR(100) | No | Address Line 2 of the College |
|  | Zip\_Code | NUMERIC(7,0) | Yes | Zip Code of the College |
|  | College\_Status | CHAR(3) | Yes | Yes-if currently active, No-if college has shut down |

1. **Branch Table:** Stores details about different branches that are available in different engineering colleges.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name:**  **Branch** | **Attribute Name** | **Field Type** | **Required** | **Description** |
| Primary Key | Branch\_ID | INTEGER | Yes | Unique Identifier for the Branch |
| Other Attributes | Name | VARCHAR(100) | Yes | Name of the Branch |
|  | Branch\_Status | CHAR(3) | Yes | Yes-if currently active, No-if branch has shut down |

1. **Subject:** Stores details about every subject the tutorials has ever taught.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name:**  **Subject** | **Attribute Name** | **Field Type** | **Required** | **Description** |
| Primary Key | Subject\_ID | INTEGER | Yes | Unique Identifier for the College |
| Other Attributes | Name | VARCHAR(100) | Yes | Name of the Subject |
|  | Branch\_Status | CHAR(3) | Yes | Yes-if currently active, No-if subject has shut down |

1. **College\_Branch:** An associate table containing the branch and college

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name:**  **College\_Branch** | **Attribute Name** | **Field Type** | **Required** | **Description** |
| Primary Key / Foreign Key 1 | College\_ID | INTEGER | Yes | Unique Identifier for the College |
| Primary Key /  Foreign Key 2 | Branch\_ID | INTEGER | Yes | Unique Identifier for the Branch |

1. **Branch\_Subject:** Associate table containing semester college branch and subject

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name:**  **Branch\_Subject** | **Attribute Name** | **Field Type** | **Required** | **Description** |
| Primary Key | Semester | NUMERIC(1) | Yes | Semester in which the subject is taught |
| Primary Key / Foreign Key 1 | College\_ID | INTEGER | Yes | Unique Identifier for the College |
| Primary Key /  Foreign Key 1 | Branch\_ID | INTEGER | Yes | Unique Identifier for the Branch |
| Primary Key / Foreign Key 2 | Subject\_ID | INTEGER | Yes | Unique Identifier for the Subject |

1. **Student:** Stores information about a student.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name:**  **Student** | **Attribute Name** | **Field Type** | **Required** | **Description** |
| Primary Key | U\_Name | VARCHAR (30) | Yes | Student’s first name, his/her birthday and N where N is 1 for the 1st time he registers; if his/her college or branch changes he registers the 2nd time where N is 2, example: dhwaniDDMMYYYYN |
| Foreign Key 1 | College\_ID | INTEGER | Yes | College ID of the student he/she goes to |
| Foreign Key 1 | Branch\_ID | INTEGER | Yes | Branch ID of the student he/she goes to |
| Other Attributes | First\_Name | CHAR(20) | Yes | First Name of the student |
|  | Last\_Name | CHAR(20) | Yes | Last Name of the student |
|  | Status\_1 | CHAR(3) | Yes | Status which defines whether the student is a current student or not. Status is changed by the faculty member. (Yes-If current student, No-Otherwise) |
|  | Status\_2 | CHAR(3) | Yes | Status which defines whether the student is a previous student or not. Status is changed by the faculty member. (Yes-If previous student, No-Otherwise) |
|  | Phone\_Number | VARCHAR(15) | Yes | Phone number of the student |
|  | Email | VARCHAR(35) | Yes | Email address of the student |

1. **Student Registration:** Table containing details about student registration.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name:**  **Student Registration** | **Attribute Name** | **Field Type** | **Required** | **Description** |
| Primary Key / Foreign Key 1 | Semester | NUMERIC(1) | Yes | Semester in which the subject is taught |
| Primary Key / Foreign Key 1 | College\_ID | INTEGER | Yes | Unique Identifier for the College |
| Primary Key /  Foreign Key 1 | Branch\_ID | INTEGER | Yes | Unique Identifier for the Branch |
| Primary Key / Foreign Key 1 | Subject\_ID | INTEGER | Yes | Unique Identifier for the Subject |
| Primary Key /  Foreign Key 2 | U\_Name | VARCHAR(30) | Yes | Unique Identifier for the Student |
| Other Attributes | Date | DATETIME | Yes | The date and time when the registration form was filled |
|  | Student\_Status | CHAR(3) | Yes | Yes-If the student has made the payment for this subject, No-If the student is still left to make the payment. |

1. **Payment:** Table containing details about the payments made.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name:**  **Payment** | **Attribute Name** | **Field Type** | **Required** | **Description** |
| Primary Key | Payment\_ID | INTEGER | Yes | Unique Identifier for each Payment |
| Primary Key / Foreign Key 1 | Semester | NUMERIC(1) | Yes | Semester in which the subject is taught |
| Primary Key / Foreign Key 1 | College\_ID | INTEGER | Yes | Unique Identifier for the College |
| Primary Key /  Foreign Key 1 | Branch\_ID | INTEGER | Yes | Unique Identifier for the Branch |
| Primary Key / Foreign Key 1 | Subject\_ID | INTEGER | Yes | Unique Identifier for the Subject |
| Primary Key /  Foreign Key 1 | U\_Name | VARCHAR(30) | Yes | Unique Identifier for the Student |
| Other Attributes | Date | DATETIME | Yes | The date and time when the payment was made |
|  | Amount | NUMERIC(8,2) | Yes | Total Amount of the payment made |

**ENTITY RELATIONSHIP DIAGRAM**



**DATABASE SYSTEM ARCHITECTURE**

We used the following tools to create and implement this project:

1. **SQL Server:** This was used as the database which stored all the tables and their data. We created the tables using SQL queries. We also used SQL Server to create and deploy triggers.
2. **MS Visio:** We created the Entity Relationship Diagram using MS Visio. We created Entities and defined their attributes along with the keys. We established relationships and dependencies between entities.
3. **MS Access:** The interface of the system was created using MS Access. We created forms that could take user input. We also generated reports using MS Access.

**SQL SCRIPTS FOR CREATING TABLES AND INSERTING VALUES**

**CREATING TABLES:**

We created tables in SQL Server using the following SQL queries:

1. **College Table**

CREATE TABLE College(

College\_ID INTEGER PRIMARY KEY,

Name VARCHAR(100) NOT NULL,

Address\_Line\_1 VARCHAR(100) NOT NULL,

Address\_Line\_2 VARCHAR(100),

Zip\_Code NUMERIC(7,0) NOT NULL,

College\_Status CHAR (3) NOT NULL CHECK(College\_Status IN ('Yes','No'))

);

1. **Branch Table**

CREATE TABLE Branch (

Branch\_ID INTEGER PRIMARY KEY,

Name VARCHAR(100) NOT NULL,

Branch\_Status CHAR (3) NOT NULL CHECK(Branch\_Status IN ('Yes','No'))

);

1. **Subject Table**

CREATE TABLE Subject(

Subject\_ID INTEGER PRIMARY KEY,

Name VARCHAR(100) NOT NULL,

Subject\_Status CHAR (3) NOT NULL CHECK(Subject\_Status IN ('Yes','No'))

);

1. **College\_Branch Table**

CREATE TABLE College\_Branch(

Branch\_ID INTEGER NOT NULL,

College\_ID INTEGER NOT NULL,

CONSTRAINT College\_Branch\_Pk PRIMARY KEY(Branch\_ID,College\_ID),

CONSTRAINT College\_Branch\_Fk1 FOREIGN KEY (Branch\_ID) REFERENCES Branch(Branch\_ID),

CONSTRAINT College\_Branch\_Fk2 FOREIGN KEY (College\_ID) REFERENCES College(College\_ID)

);

1. **Branch\_Subject Table**

CREATE TABLE Branch\_Subject(

Semester NUMERIC(1) NOT NULL,

Subject\_ID INTEGER NOT NULL,

Branch\_ID INTEGER NOT NULL,

College\_ID INTEGER NOT NULL,

CONSTRAINT Branch\_Subject\_Pk PRIMARY KEY(Semester,Subject\_ID,Branch\_ID,College\_ID),

CONSTRAINT Branch\_Subject\_Fk1 FOREIGN KEY (Subject\_ID) REFERENCES Subject(Subject\_ID),

CONSTRAINT Branch\_Subject\_Fk2 FOREIGN KEY (Branch\_ID,College\_ID) REFERENCES College\_Branch (Branch\_ID,College\_ID)

);

1. **Student Table**

CREATE TABLE Student(

U\_Name VARCHAR(30) PRIMARY KEY,

First\_Name CHAR(20) NOT NULL,

Last\_Name CHAR(20) NOT NULL,

Status\_1 CHAR (3) NOT NULL CHECK(Status\_1 IN ('Yes','No')) DEFAULT 'No',

Status\_2 CHAR (3) NOT NULL CHECK(Status\_2 IN ('Yes','No')) DEFAULT 'No',

Branch\_ID INTEGER NOT NULL,

College\_ID INTEGER NOT NULL,

Phone\_Number VARCHAR(15) NOT NULL,

Email VARCHAR(35) NOT NULL,

CONSTRAINT Student\_Fk1 FOREIGN KEY (Branch\_ID,College\_ID) REFERENCES College\_Branch (Branch\_ID,College\_ID)

);

1. **Student\_Registration Table**

CREATE TABLE Student\_Registration(

Semester NUMERIC(1) NOT NULL,

Subject\_ID INTEGER NOT NULL,

Branch\_ID INTEGER NOT NULL,

College\_ID INTEGER NOT NULL,

U\_Name VARCHAR(30) NOT NULL,

Registration\_Date DATETIME DEFAULT GETDATE() NOT NULL,

Student\_Status CHAR (3) NOT NULL CHECK(Student\_Status IN ('Yes','No')) DEFAULT 'No',

CONSTRAINT Student\_Registration\_Pk PRIMARY KEY (Semester,Subject\_ID,Branch\_ID,College\_ID,U\_Name),

CONSTRAINT Student\_Registration\_Fk1 FOREIGN KEY (Semester,Subject\_ID,Branch\_ID,College\_ID) REFERENCES Branch\_Subject (Semester,Subject\_ID,Branch\_ID,College\_ID),

CONSTRAINT Student\_Registration\_Fk2 FOREIGN KEY (U\_Name) REFERENCES Student (U\_Name)

);

1. **Payment Table**

CREATE TABLE Payment(

Payment\_ID INTEGER NOT NULL,

Semester NUMERIC(1) NOT NULL,

Subject\_ID INTEGER NOT NULL,

Branch\_ID INTEGER NOT NULL,

College\_ID INTEGER NOT NULL,

U\_Name VARCHAR(30) NOT NULL,

Payment\_Date DATETIME DEFAULT GETDATE() NOT NULL,

Amount NUMERIC(8,2) NOT NULL,

CONSTRAINT Payment\_Pk PRIMARY KEY(Payment\_ID,Semester,Subject\_ID,Branch\_ID,College\_ID,U\_Name),

CONSTRAINT Payment\_Fk1 FOREIGN KEY (Semester,Subject\_ID,Branch\_ID,College\_ID,U\_Name) REFERENCES Student\_Registration(Semester,Subject\_ID,Branch\_ID,College\_ID,U\_Name)

);

**INSERTING VALUES:**

We used the following SQL queries to insert data in the tables created using the above queries:

1. **College Table**

INSERT INTO College(College\_ID,Name,Address\_Line\_1,Zip\_Code,College\_Status)

VALUES ('1','K J Somaiya College of Engineering','Sion','400100','Yes');

INSERT INTO College(College\_ID,Name,Address\_Line\_1,Zip\_Code,College\_Status)

VALUES ('2','Vidyalankar Institute of Technology','Dadar','400001','Yes');

1. **Branch Table**

INSERT INTO Branch

VALUES ('1','Information Technology','Yes');

INSERT INTO Branch

VALUES ('2','Electronics Engineering','Yes');

1. **Subject Table**

INSERT INTO Subject

VALUES ('1','Mathematics 1','No');

INSERT INTO Subject

VALUES ('2','Mathematics 2','Yes');

1. **College\_Branch Table**

INSERT INTO College\_Branch

VALUES ('1','1');

INSERT INTO College\_Branch

VALUES ('1','2');

1. **Branch\_Subject Table**

INSERT INTO Branch\_Subject

VALUES ('1','1','1','1');

INSERT INTO Branch\_Subject

VALUES ('1','1','1','2');

1. **Student Table**

INSERT INTO Student(U\_Name,First\_Name,Last\_Name,Branch\_ID,College\_ID,Phone\_Number,Email)

VALUES ('abhishek281219961','Abhishek','Ashar','5','10','9821311210','abhishek@gmail.com');

INSERT INTO Student(U\_Name,First\_Name,Last\_Name,Branch\_ID,College\_ID,Phone\_Number,Email)

VALUES ('anandi291219961','Anandi','Ashar','2','4','7977372426','anandiashar@gmail.com');

1. **Student\_Registration Table**

INSERT INTO Student\_Registration(Semester,Subject\_ID,Branch\_ID,College\_ID,U\_Name)

VALUES ('1','1','13','10','chaitanya010519971');

INSERT INTO Student\_Registration(Semester,Subject\_ID,Branch\_ID,College\_ID,U\_Name)

VALUES ('1','1','3','10','dhwani060219971');

1. **Payment Table**

INSERT INTO Payment(Payment\_ID,U\_Name,College\_ID,Branch\_ID,Subject\_ID,Semester,Amount)

VALUES ('24','twara191219951','18','1','1','1','7000');

INSERT INTO Payment(Payment\_ID,U\_Name,College\_ID,Branch\_ID,Subject\_ID,Semester,Amount)

VALUES ('25','zainab260719971','34','11','8','1','7000');

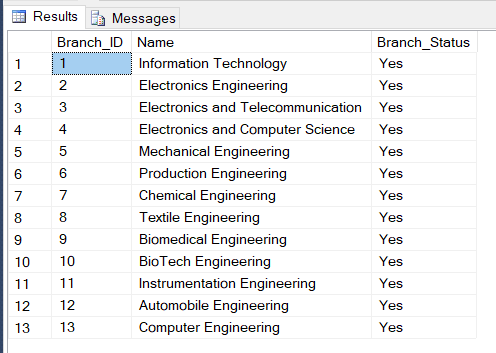
**OUTPUT FOR CREATE AND INSERT SQL SCRIPTS:**

The below tables contain data which was inserted using SQL scripts, Access forms and triggers

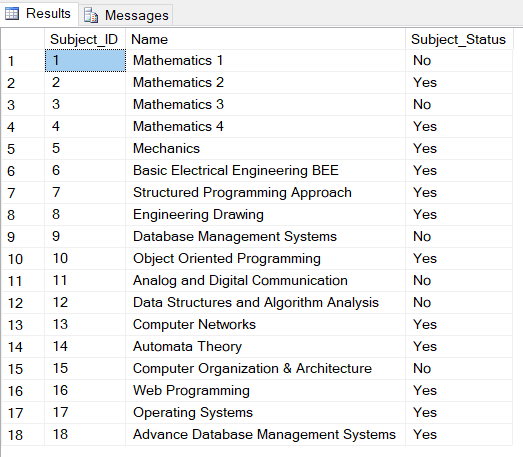
1. **College Table**



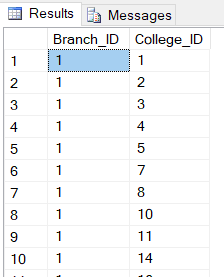
1. **Branch Table**



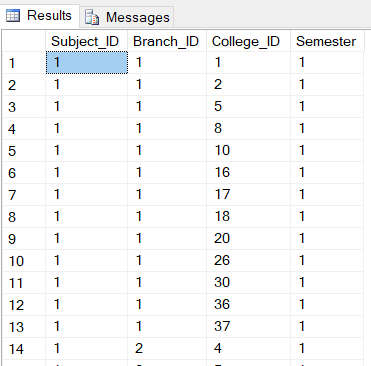
1. **Subject Table**



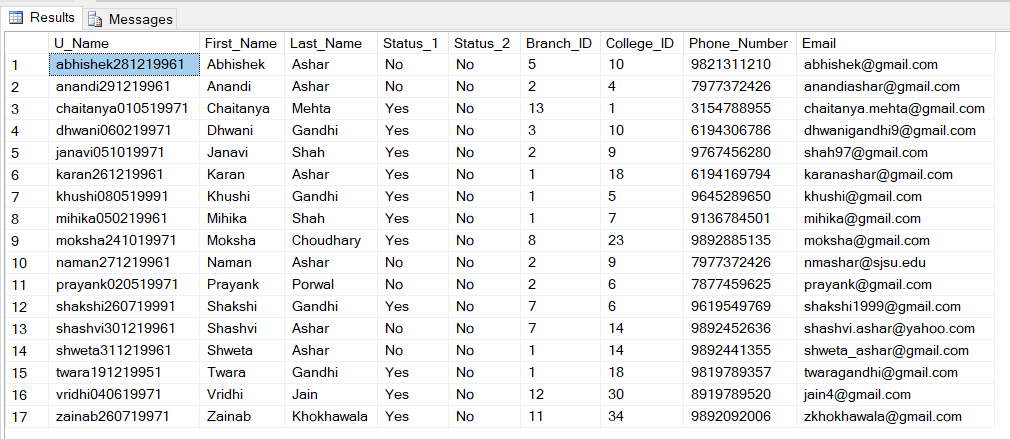
1. **College\_Branch Table**



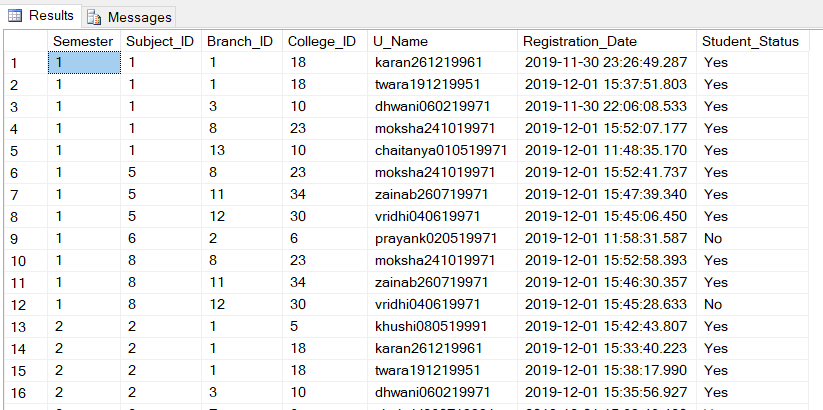
1. **Branch\_Subject Table**



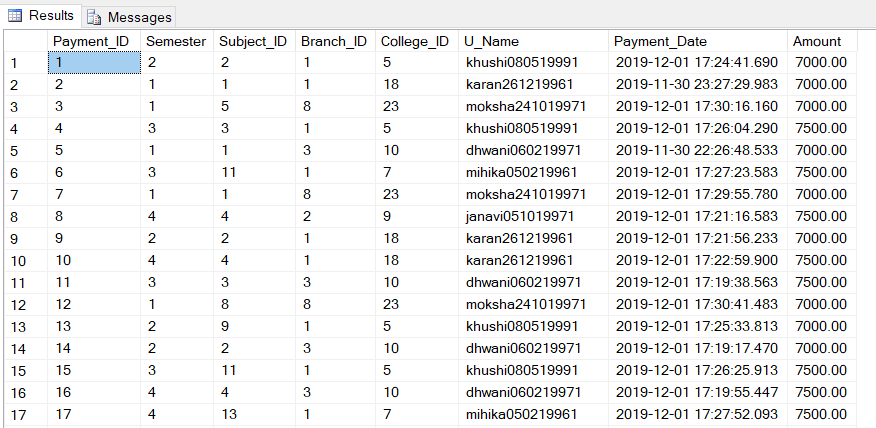
1. **Student Table**



1. **Student\_Registration Table**



1. **Payment Table**



**MAJOR DATA QUESTIONS**

We answered the major data questions using SQL queries in MS Access. We created forms and reports to display the same to the user for easy understanding.

1. Who are the currently enrolled students in the tutorials?

**SQL Query:**

SELECT dbo\_Student1.[First\_Name], dbo\_Student1.[Last\_Name], dbo\_Student1.[Phone\_Number], dbo\_Student1.[Email]

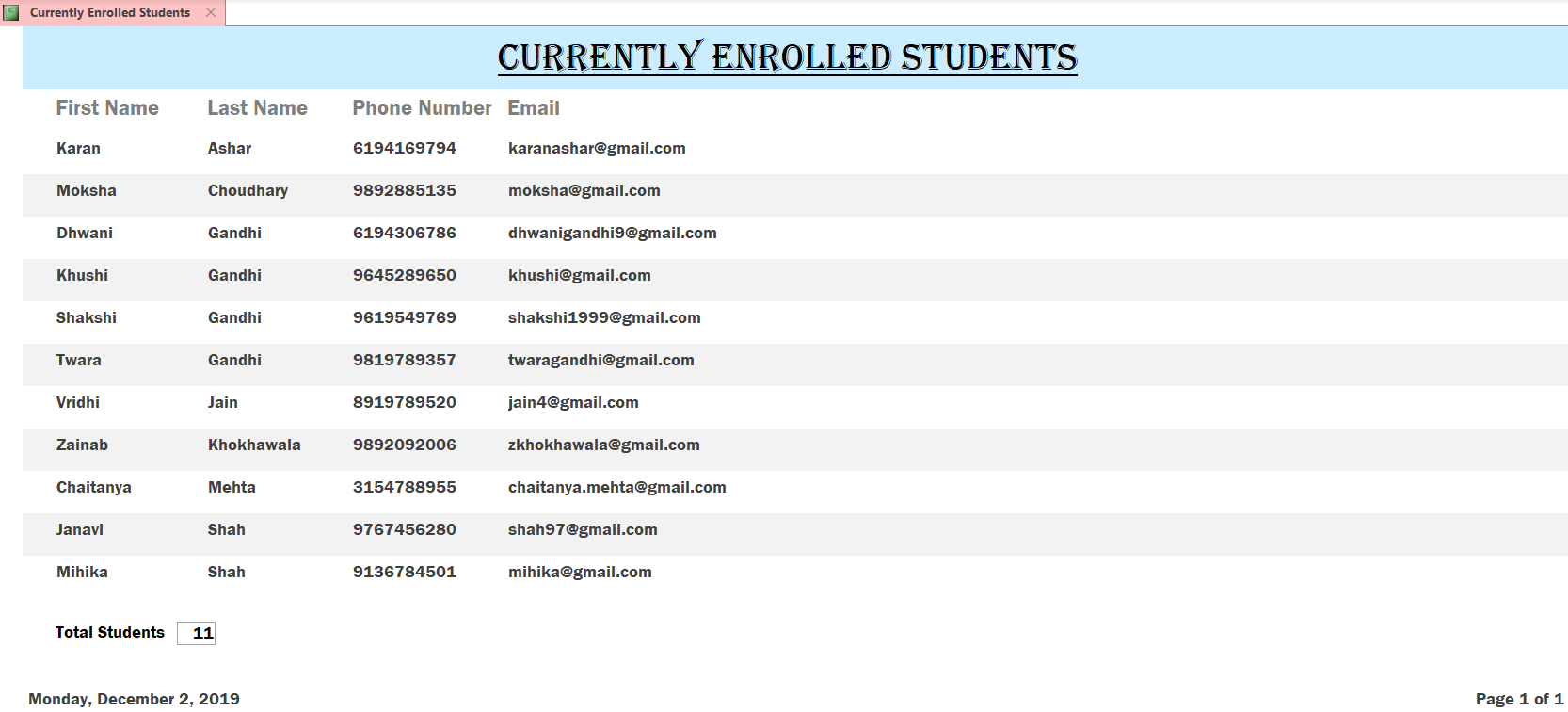
FROM dbo\_Student1

WHERE dbo\_Student1.[Status\_1] = 'Yes';

**OUTPUT(Datasheet)**



**OUTPUT(Report)**



1. How many students are currently enrolled in a particular subject?

**SQL Query:**

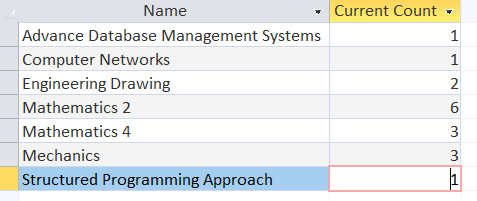
SELECT dbo\_Subject1.Name, COUNT(dbo\_Student\_Registration.Subject\_ID) AS [Current Count]

FROM dbo\_Student\_Registration LEFT JOIN dbo\_Subject1 ON dbo\_Student\_Registration.[Subject\_ID] = dbo\_Subject1.[Subject\_ID]

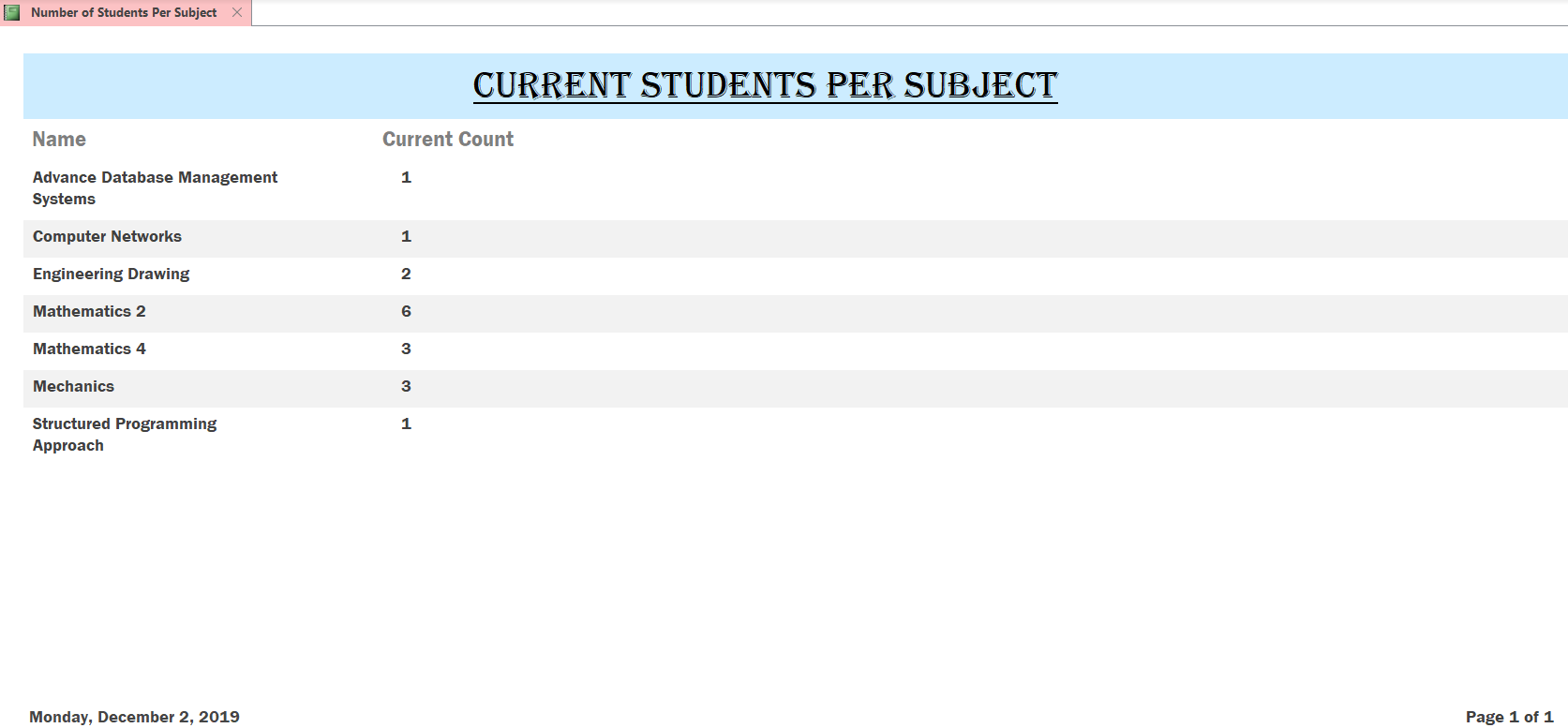
WHERE dbo\_Student\_Registration.Student\_Status='Yes' AND dbo\_Subject1.Subject\_Status='Yes'

GROUP BY dbo\_Subject1.Name;

**OUTPUT(Datasheet)**



**OUTPUT(Report)**



1. Which subjects are taught by the tutorials?

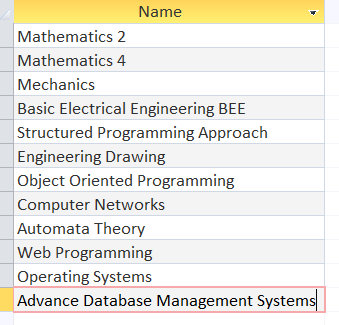
**SQL Query:**

SELECT dbo\_Subject1.[Name]

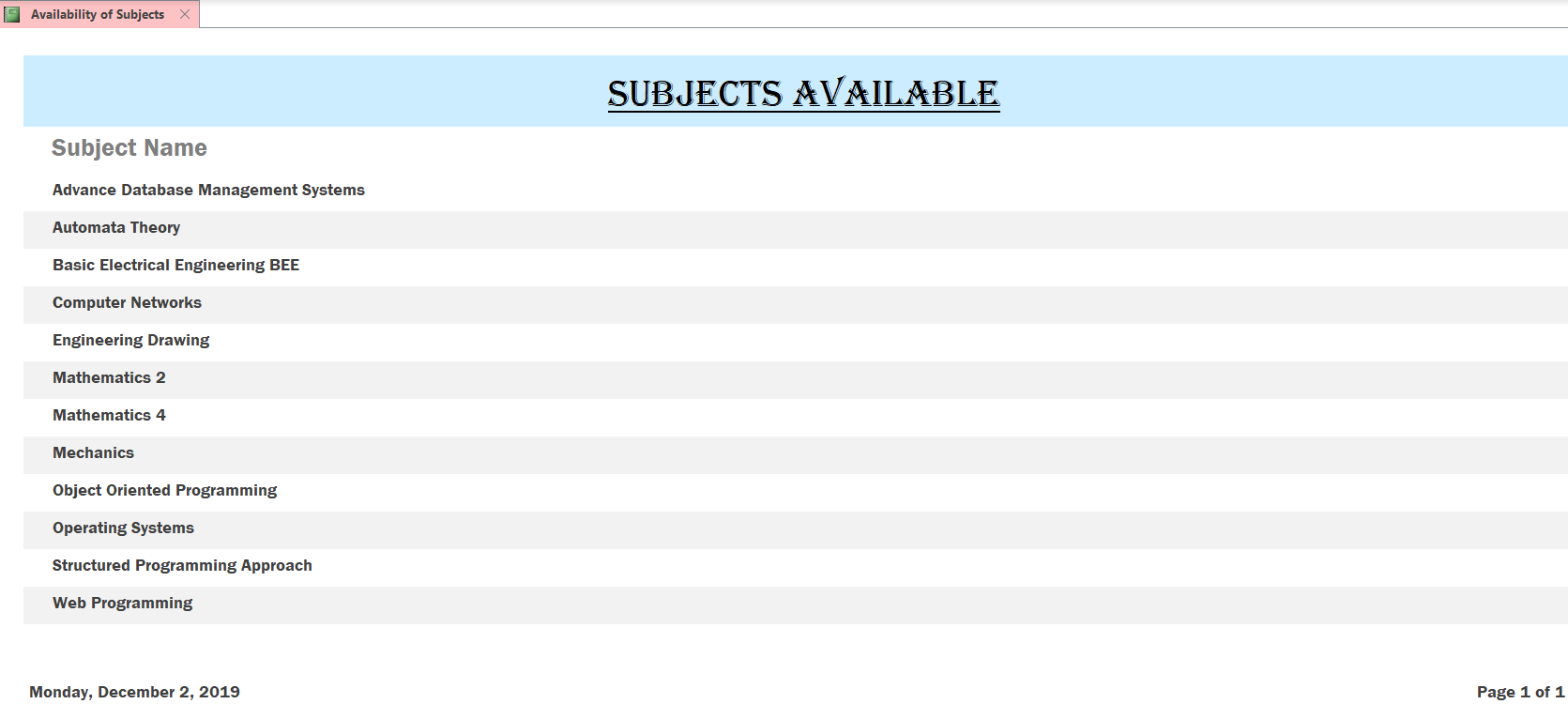
FROM dbo\_Subject1

WHERE dbo\_Subject1.[Subject\_Status] = 'Yes';

**OUTPUT(Datasheet)**



**OUTPUT(Report)**



1. What is the total payment received till date?

**SQL Query:**

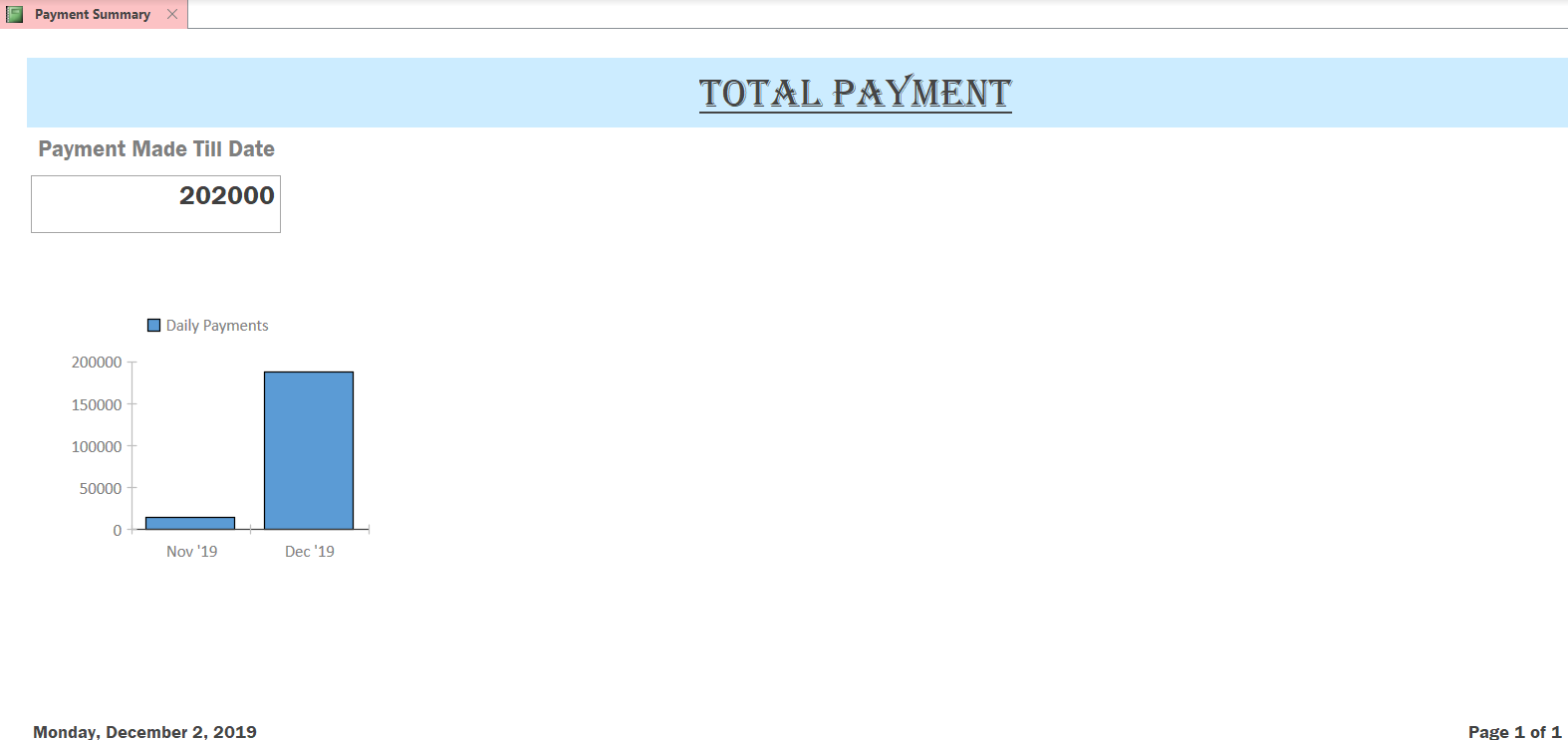
SELECT DISTINCTROW Sum([dbo\_Payment].[Amount]) AS [Sum Of Amount]

FROM dbo\_Payment;

**OUTPUT(Datasheet)**



**OUTPUT(Report)**



1. Who are the students who have registered but are left to pay?

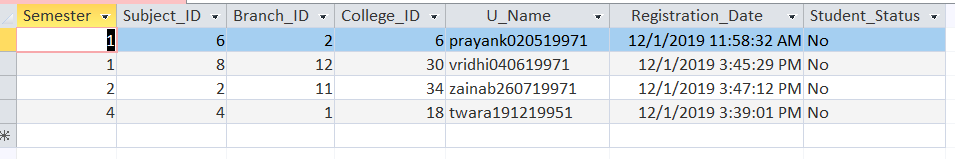
**SQL Query:**

SELECT dbo\_Student\_Registration.\*

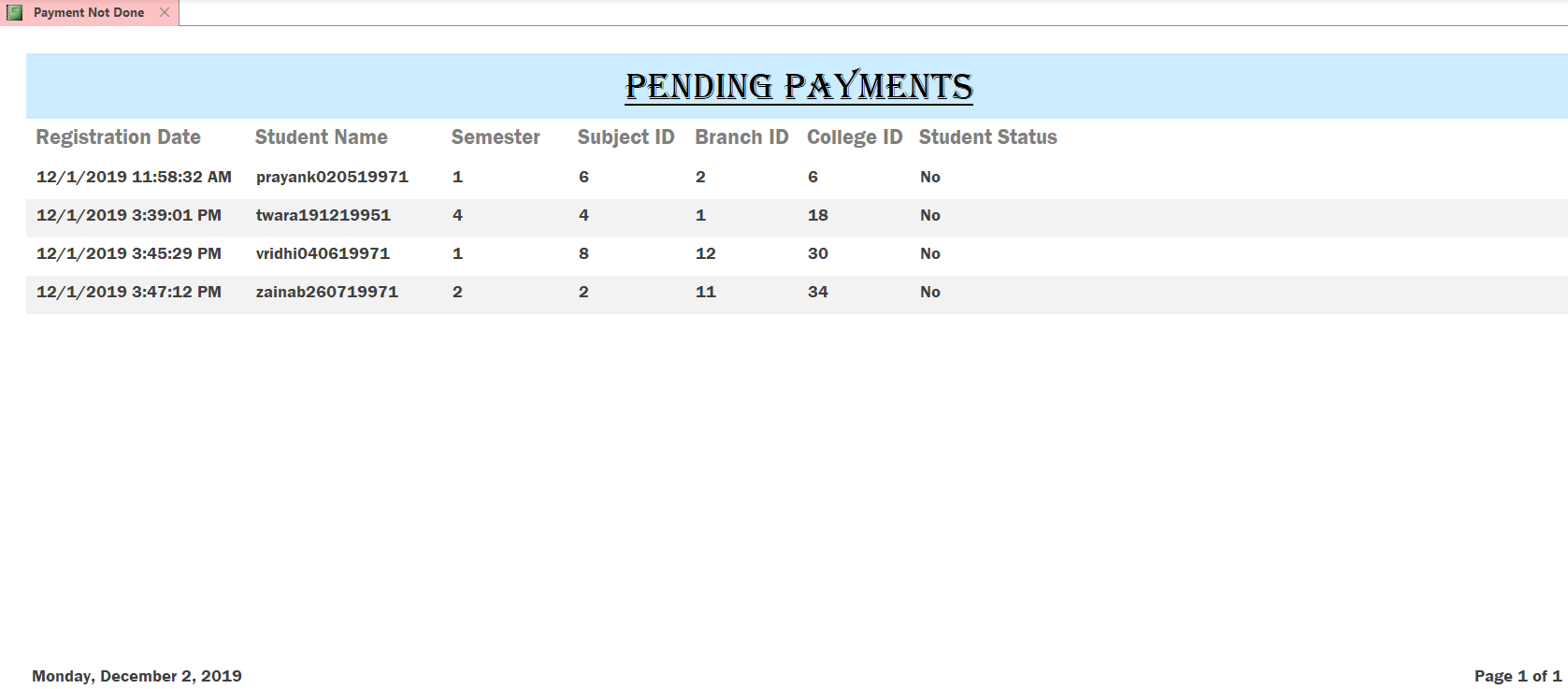
FROM dbo\_Student\_Registration LEFT JOIN dbo\_Payment ON (dbo\_Student\_Registration.U\_Name=dbo\_Payment.U\_Name) AND (dbo\_Student\_Registration.College\_ID=dbo\_Payment.College\_ID) AND (dbo\_Student\_Registration.Branch\_ID=dbo\_Payment.Branch\_ID) AND (dbo\_Student\_Registration.Subject\_ID=dbo\_Payment.Subject\_ID) AND (dbo\_Student\_Registration.Semester=dbo\_Payment.Semester)

WHERE dbo\_Payment.Payment\_ID IS NULL;

**OUTPUT(Datasheet)**

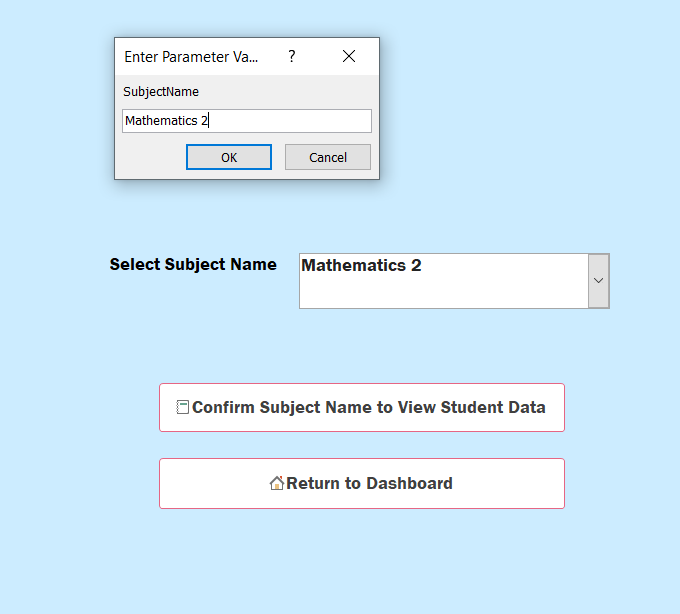


**OUTPUT(Report)**



1. Who are the students currently studying a particular subject?

Here we entered ‘Mathematics 2’ subject in the input form. The query takes the user input from the form and gives the student data of the selected subject.



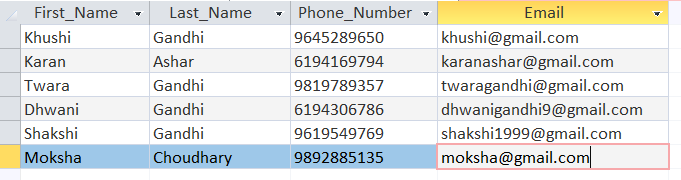
**SQL Query:**

SELECT dbo\_Student1.First\_Name, dbo\_Student1.Last\_Name, dbo\_Student1.Phone\_Number, dbo\_Student1.Email

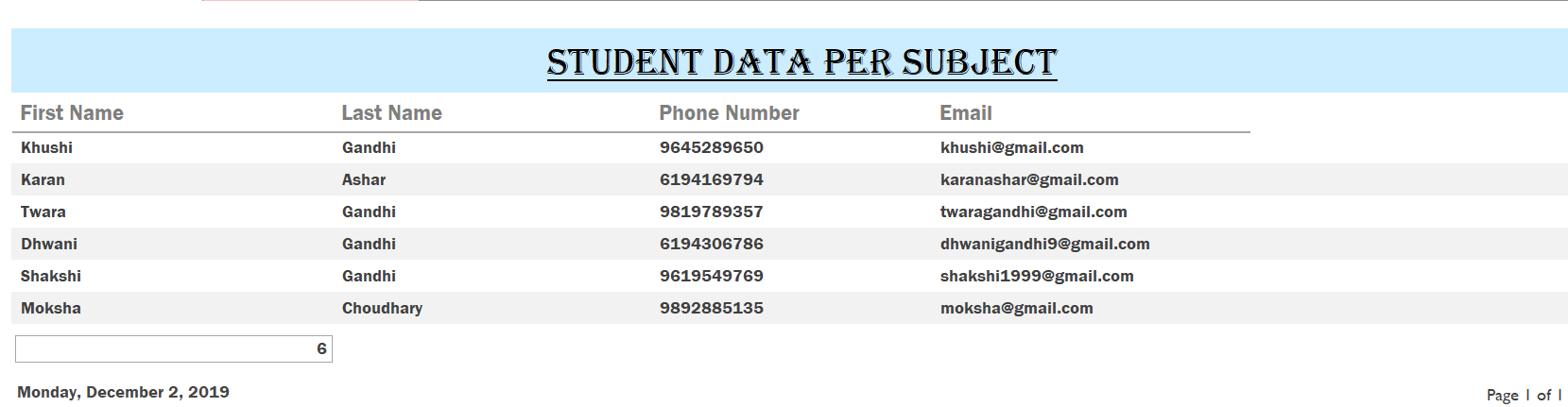
FROM (dbo\_Student\_Registration INNER JOIN dbo\_Student1 ON dbo\_Student1.U\_Name = dbo\_Student\_Registration.U\_Name) INNER JOIN dbo\_Subject1 ON dbo\_Student\_Registration.Subject\_ID = dbo\_Subject1.Subject\_ID

WHERE dbo\_Student\_Registration.Student\_Status = 'Yes' AND SubjectName=dbo\_Subject1.Name;

**OUTPUT(Datasheet)**



**OUTPUT(Report)**



**INTERFACES**

The users of the system are categorized into two groups:

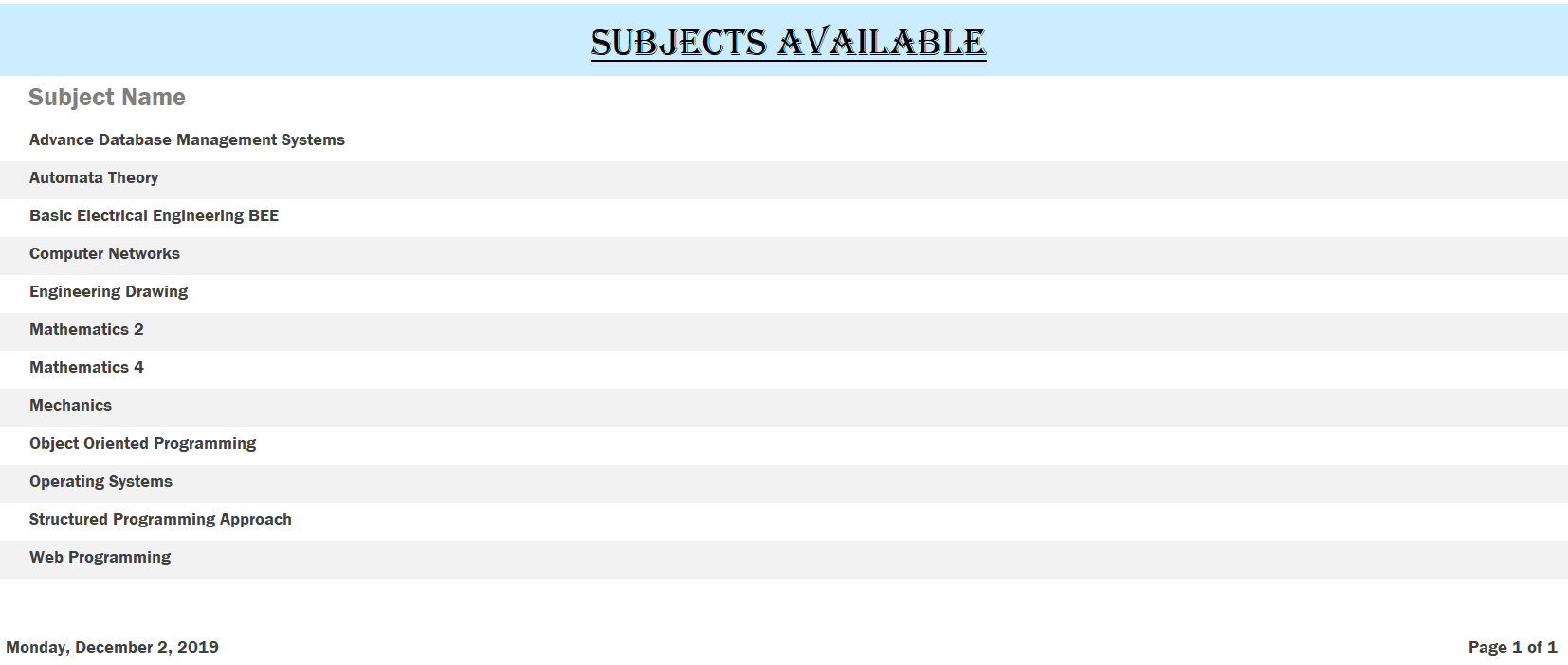
1. Students: Enroll for the tutorials and register for subjects by viewing the available subjects.
2. Admin: Enters payment information, views reports and updates data when needed.

Different dashboards are created for the two types of users.

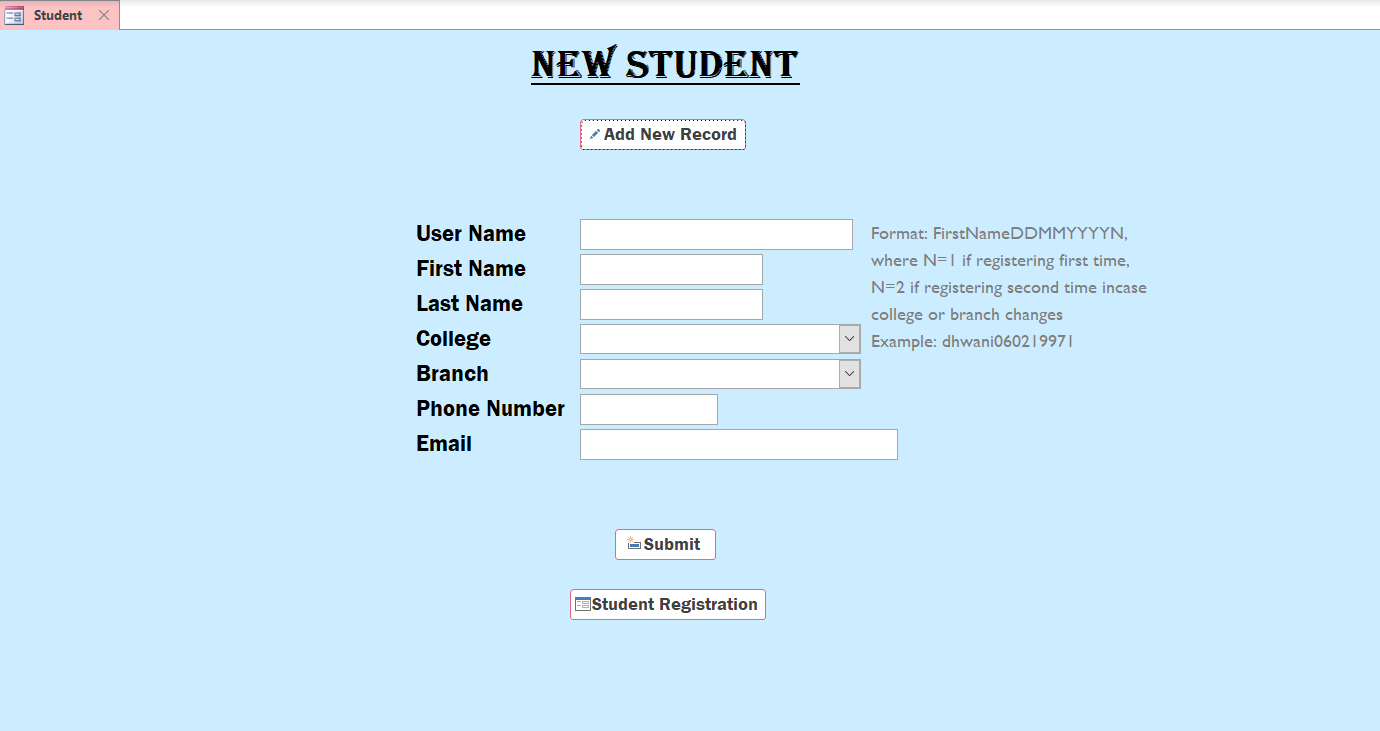
**Student Dashboard**



The student can view the subjects currently available.

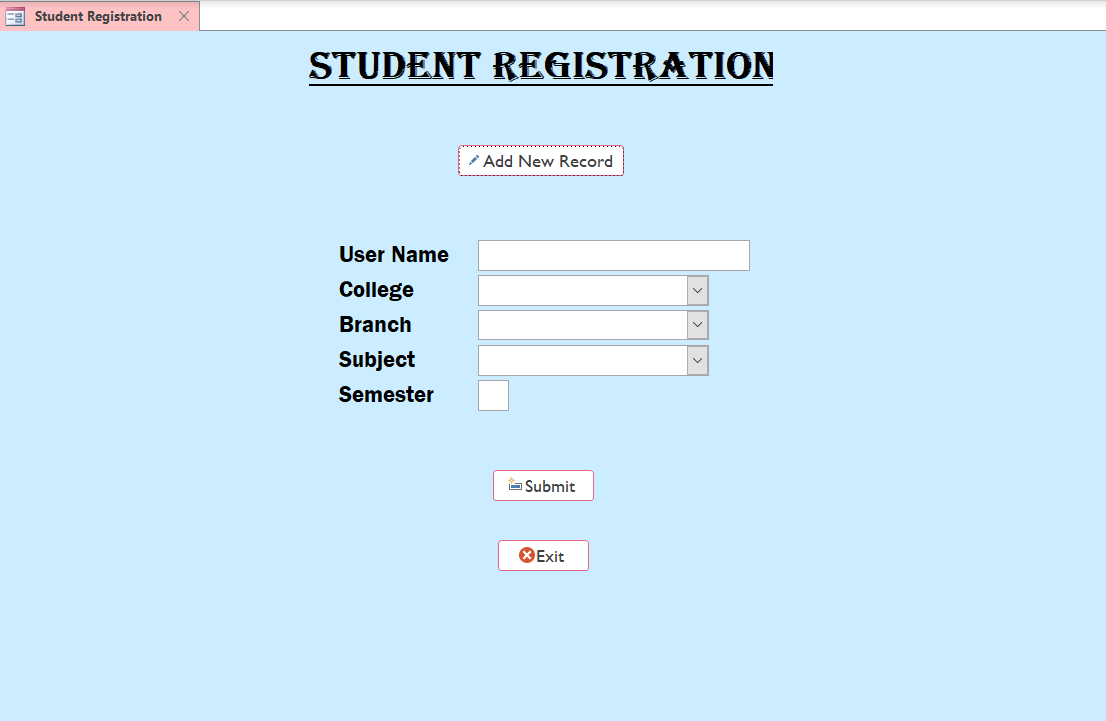


For a student to register for a subject, he/she must be an enrolled student in this portal. Thus, there is an option to register for new students. If the user selects this, he is redirected to a form that takes the student information as input.



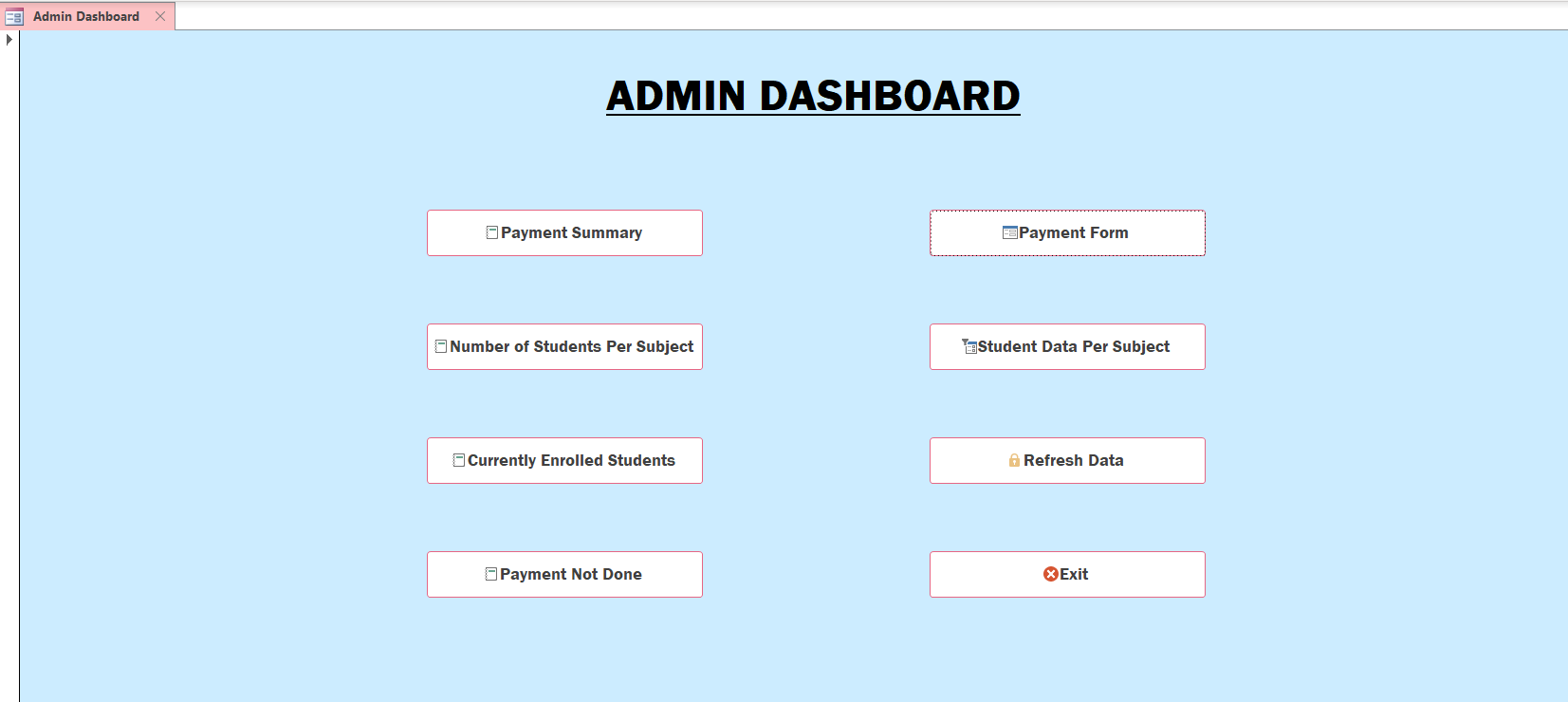
This new student can now directly register for a subject by clicking on Student Registration. In the previous form if he/she is an already existing student he can register for the subject by clicking on existing student.

In both these cases, he/she is redirected to the student registration form where the student can select the subject he wants to register for.



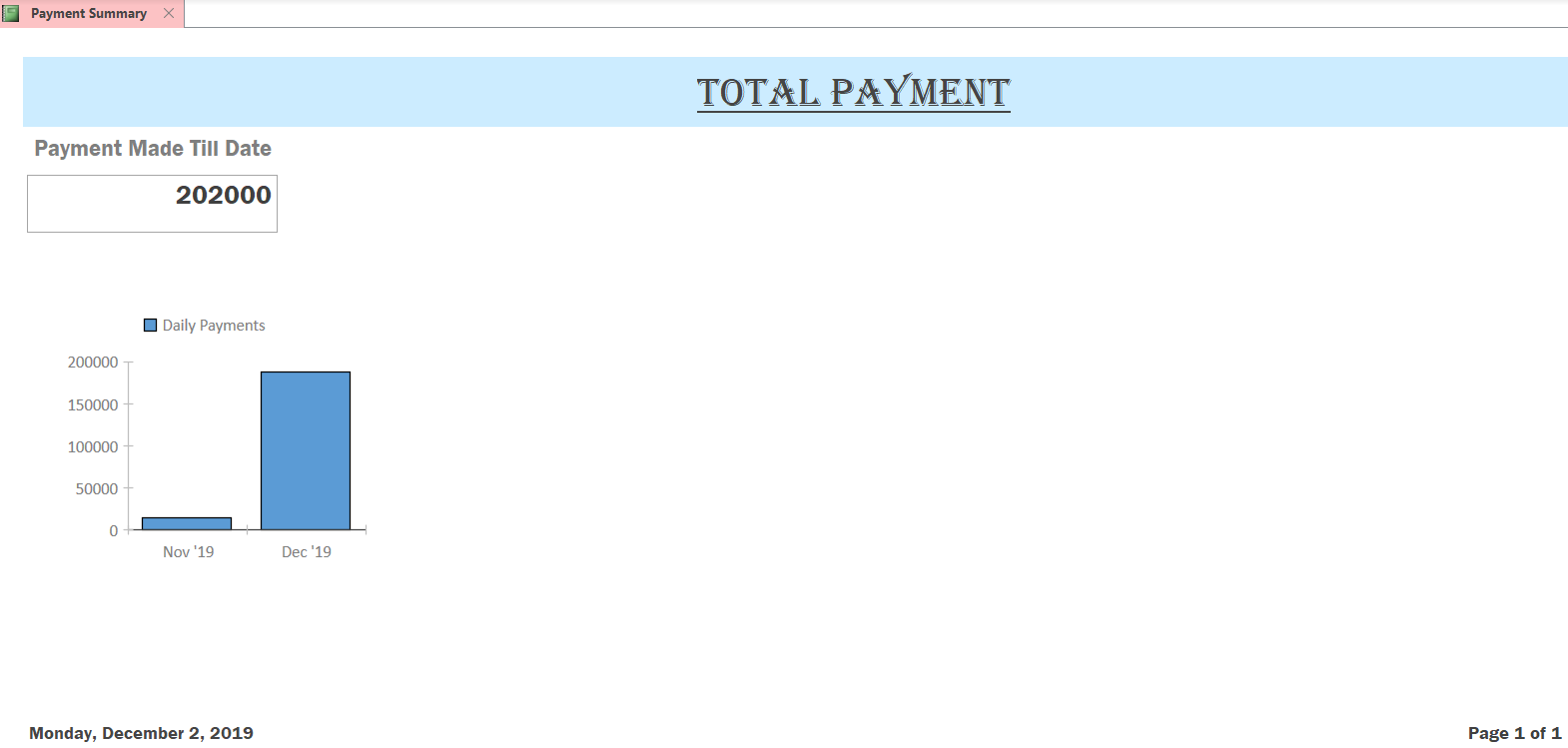
If the student needs to register for more than one subject, he can click on Add New Record. Once done the student is ready to go and can now exit the application.

**Admin Dashboard**

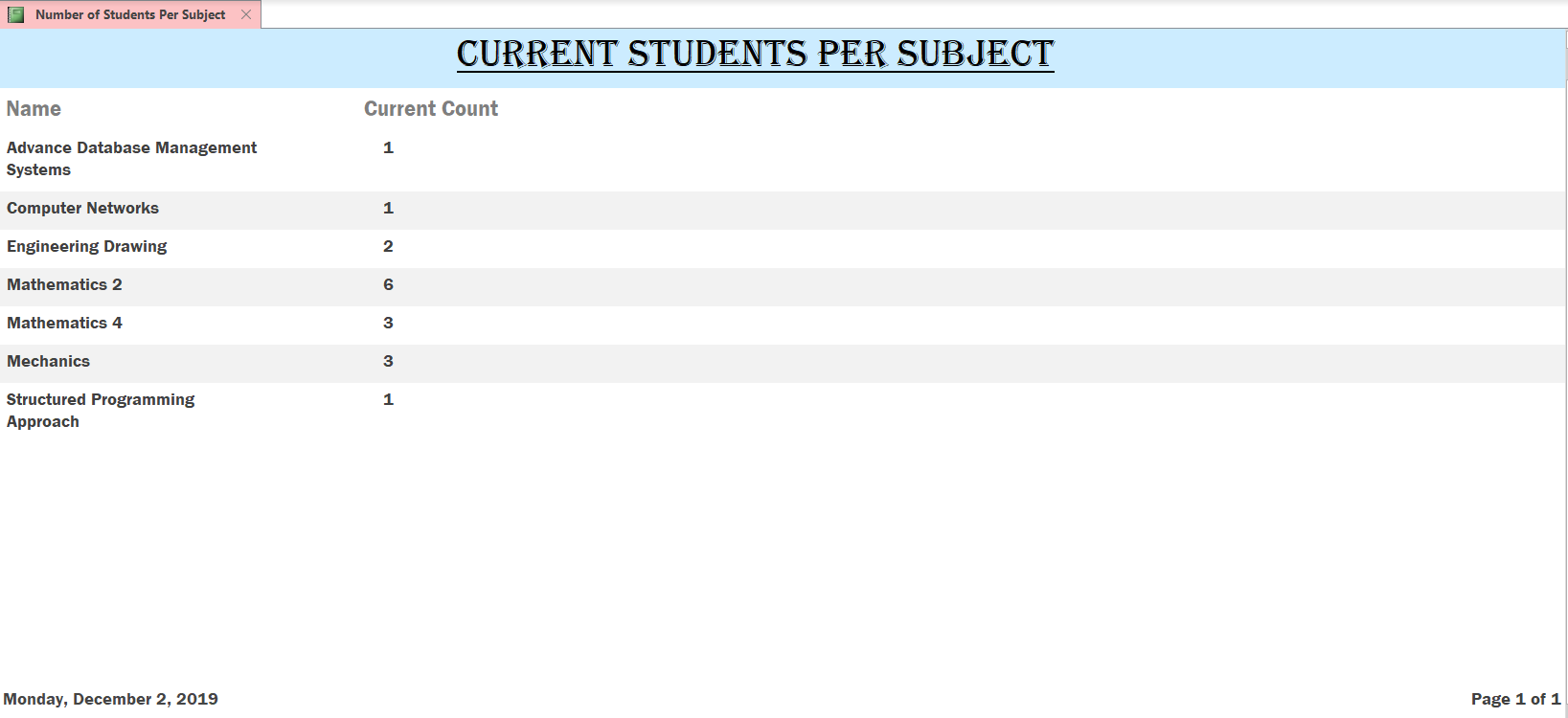


Here the admin is provided with eight options:

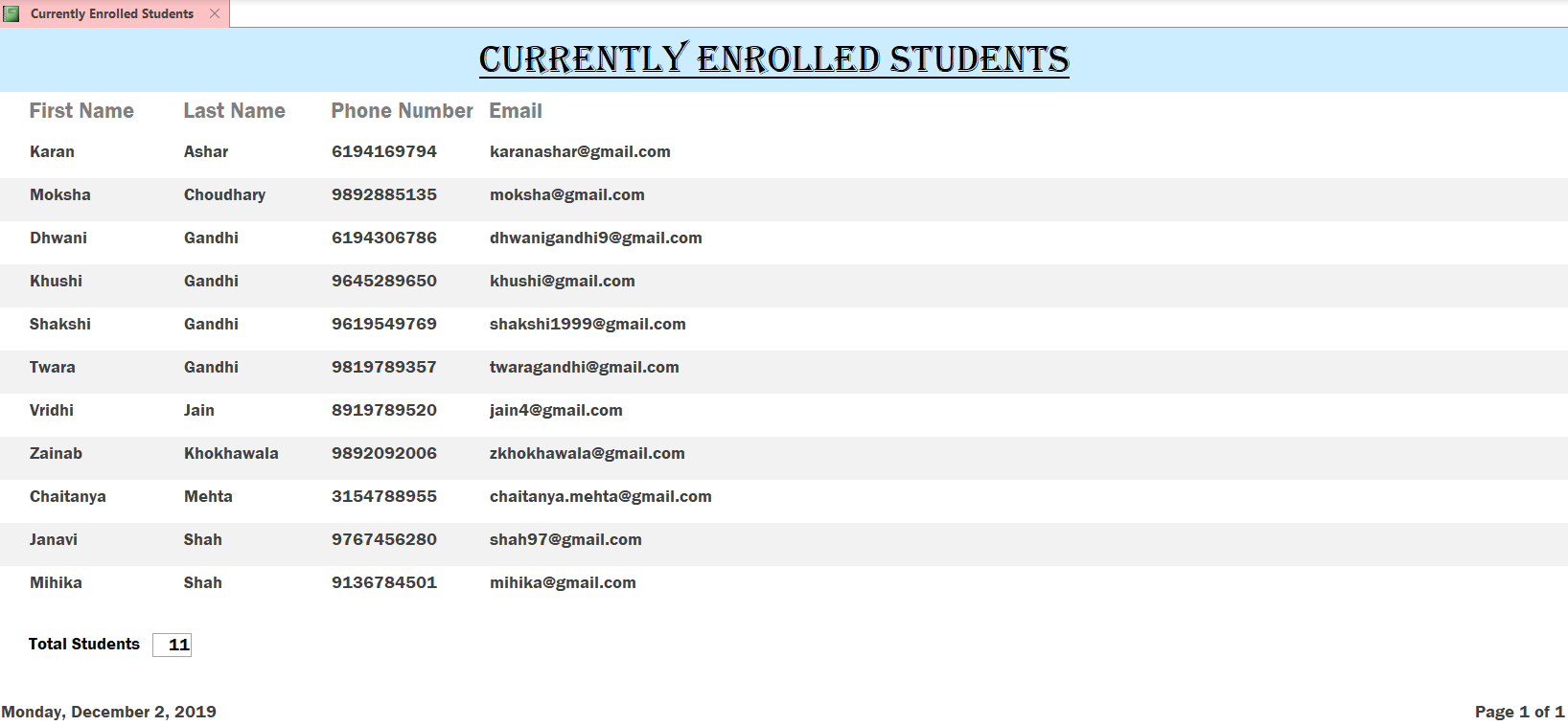
1. Payment Summary: User can view the payment summary till date.



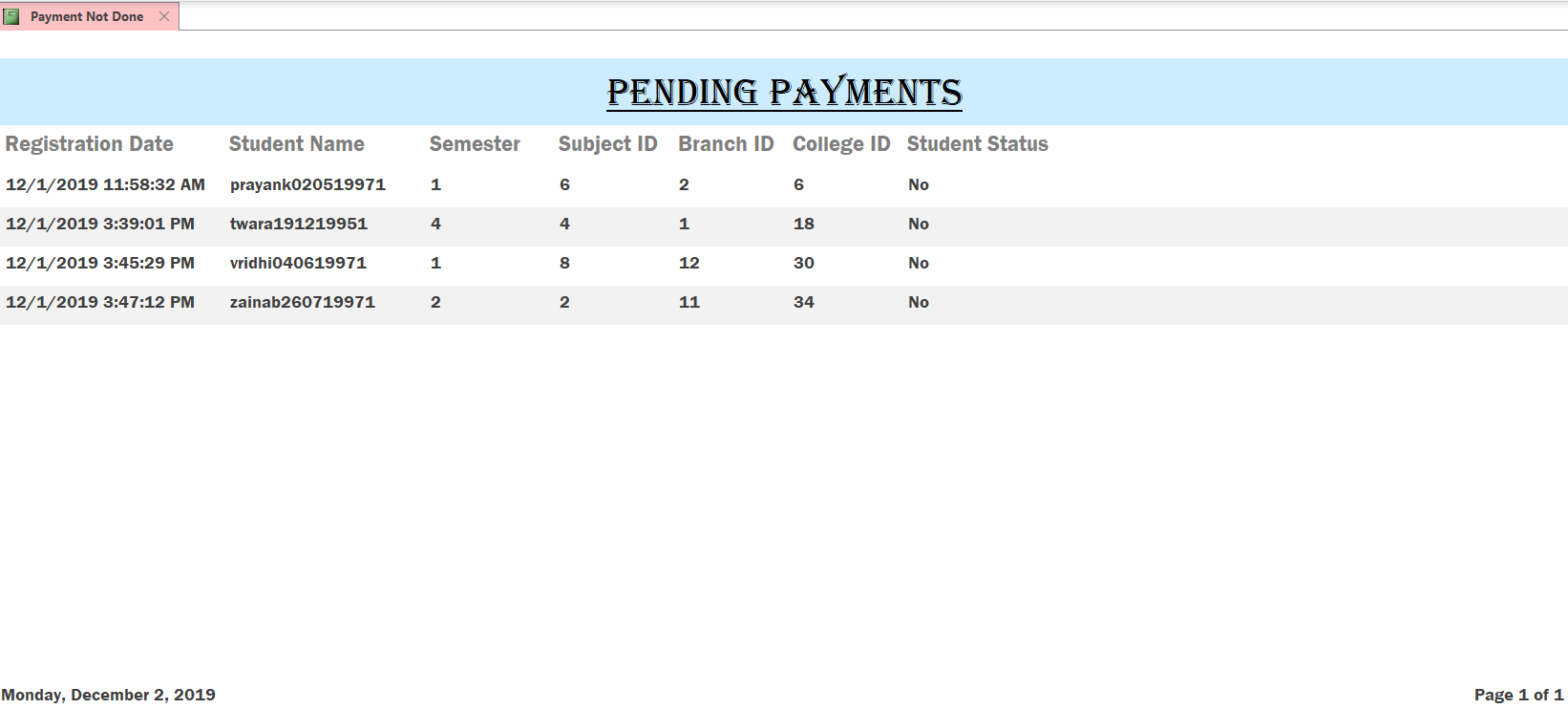
1. Number of Students Per Subject: User can view the number of students in each subject.



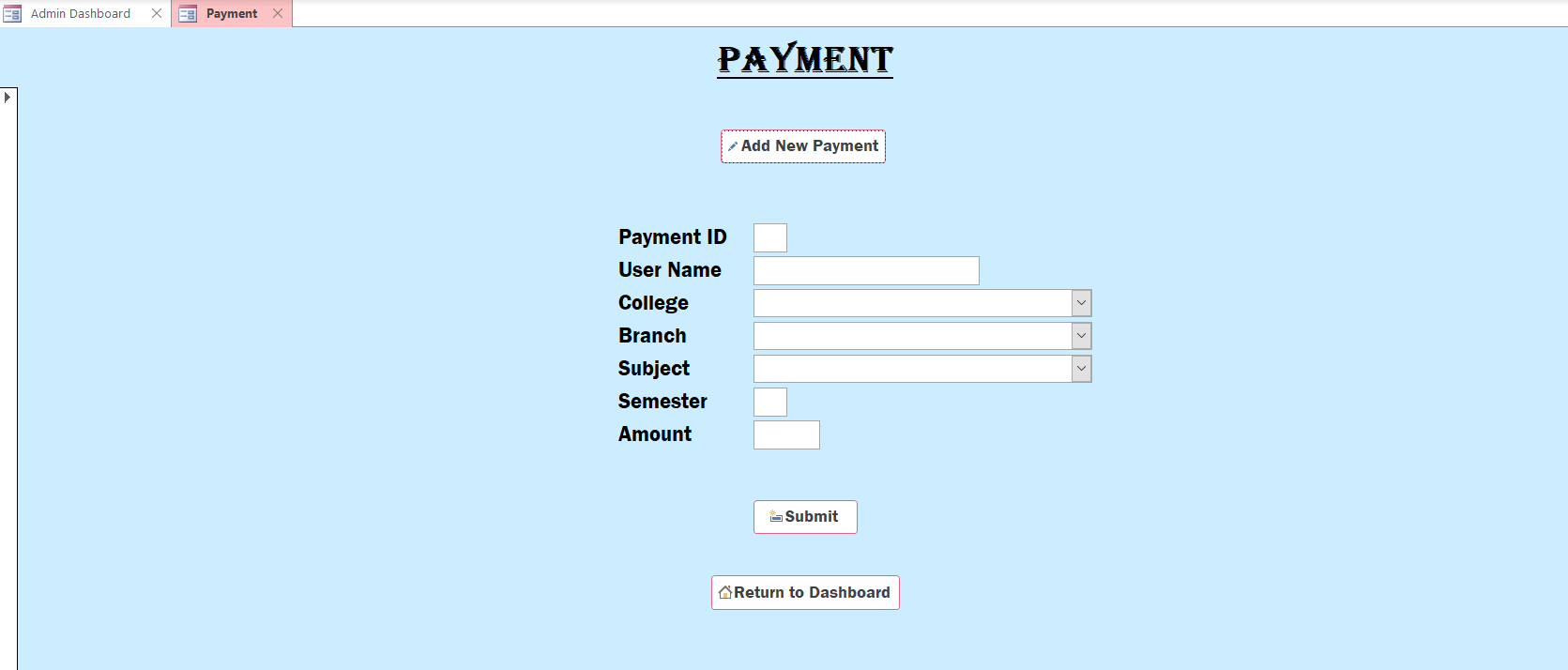
1. Currently Enrolled Students: User can view the student information of each student currently enrolled for the subjects and have paid.



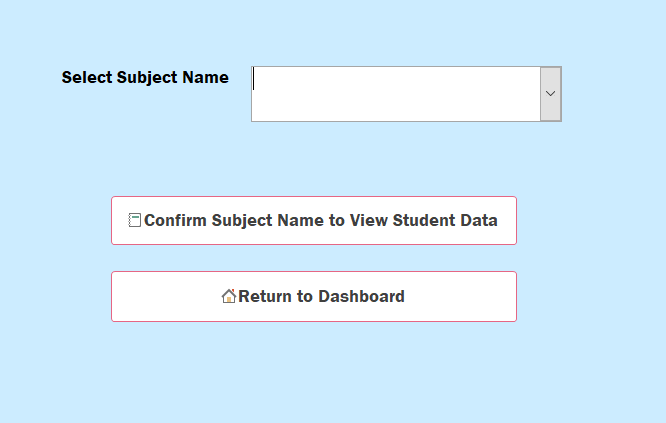
1. Payment Not Done: User can view information of those students who have registered for the subjects but have not paid.



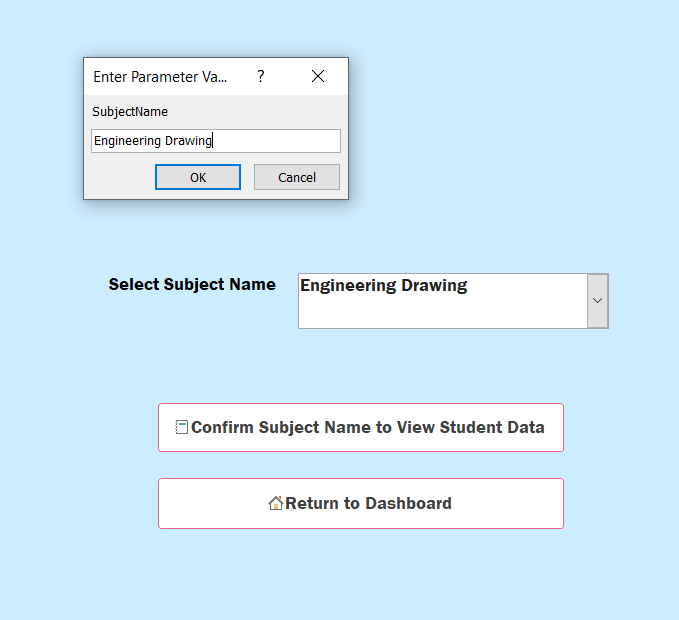
1. Payment Form: User is redirected to the payment form where he can input details of the student who is paying.



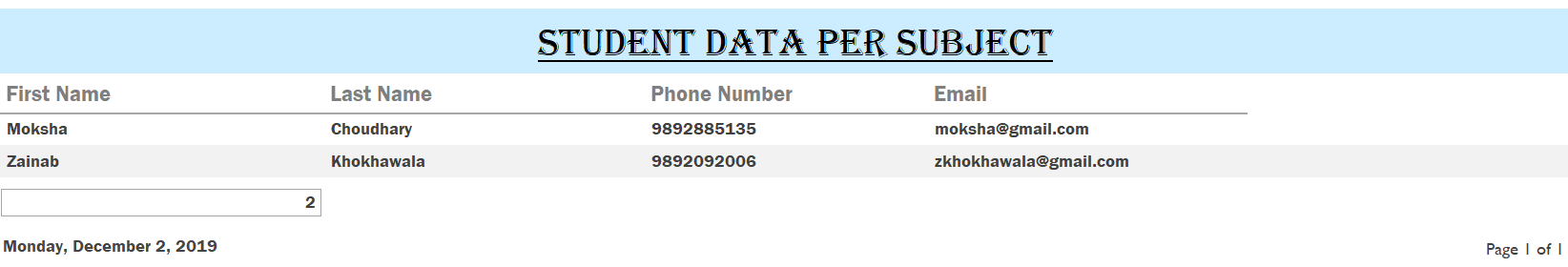
1. Student Data Per Subject: User can view the student data according to the subject he selects. It redirects the user to a form where he can select the subject.



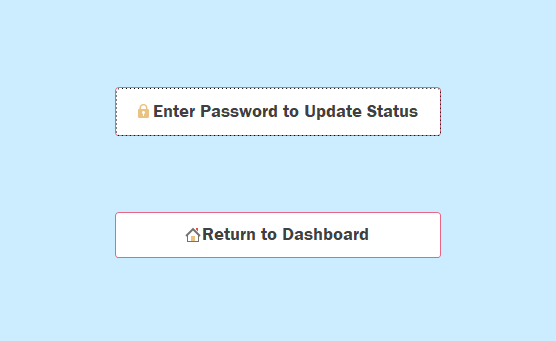
If the user wants the data for ‘Engineering Drawing’ subject he selects that subject and types it down once more for confirmation.



The user can then view the report of the students registered for that subject.



1. Refresh Data: User can update the status of the registered students to ‘No’ once the semester ends. As soon as he clicks this button he is redirected to another form. He needs to enter his password for security purposes so as to not change the data unnecessarily. Once the portal verifies the password provided it runs this query and updates the status.

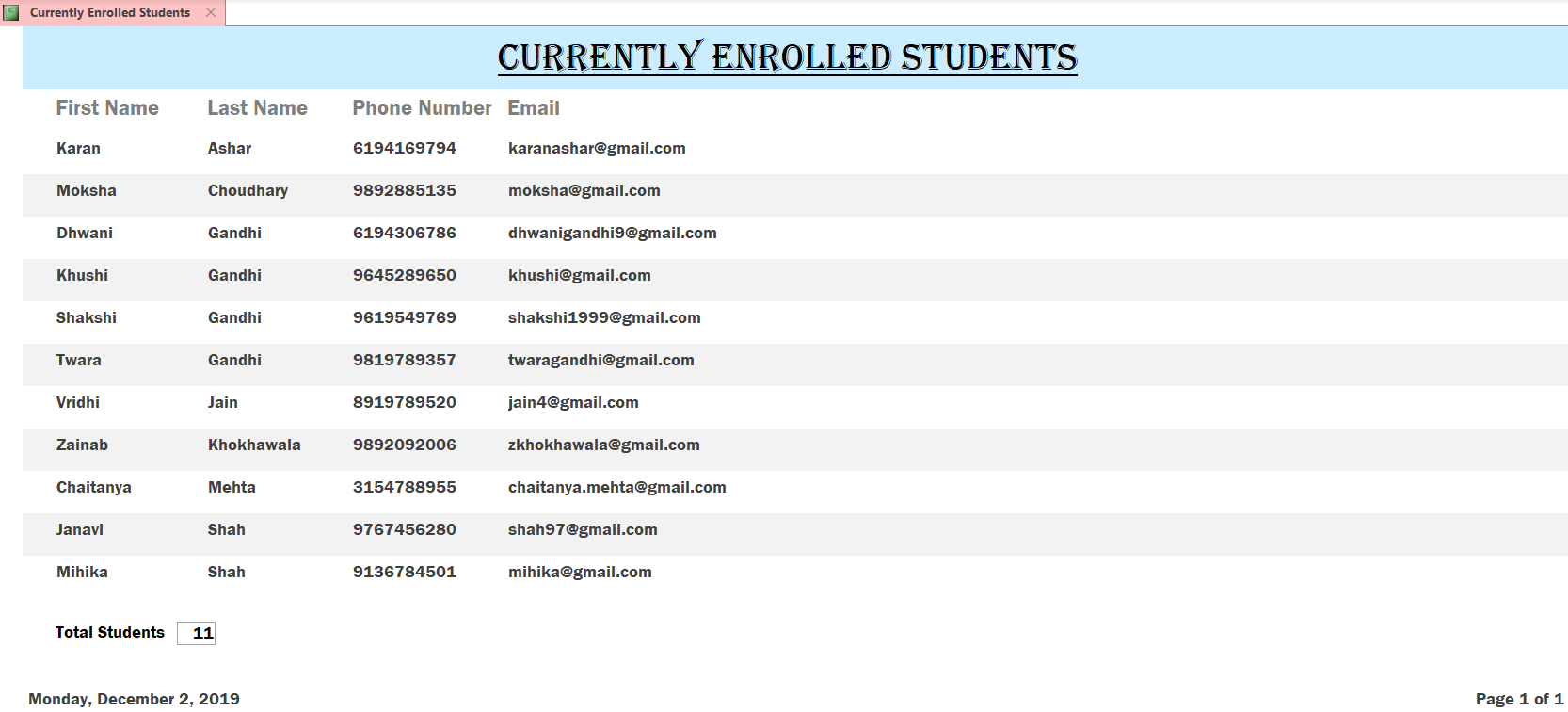


1. Exit: User can exit the application using this button.

**REPORTS**

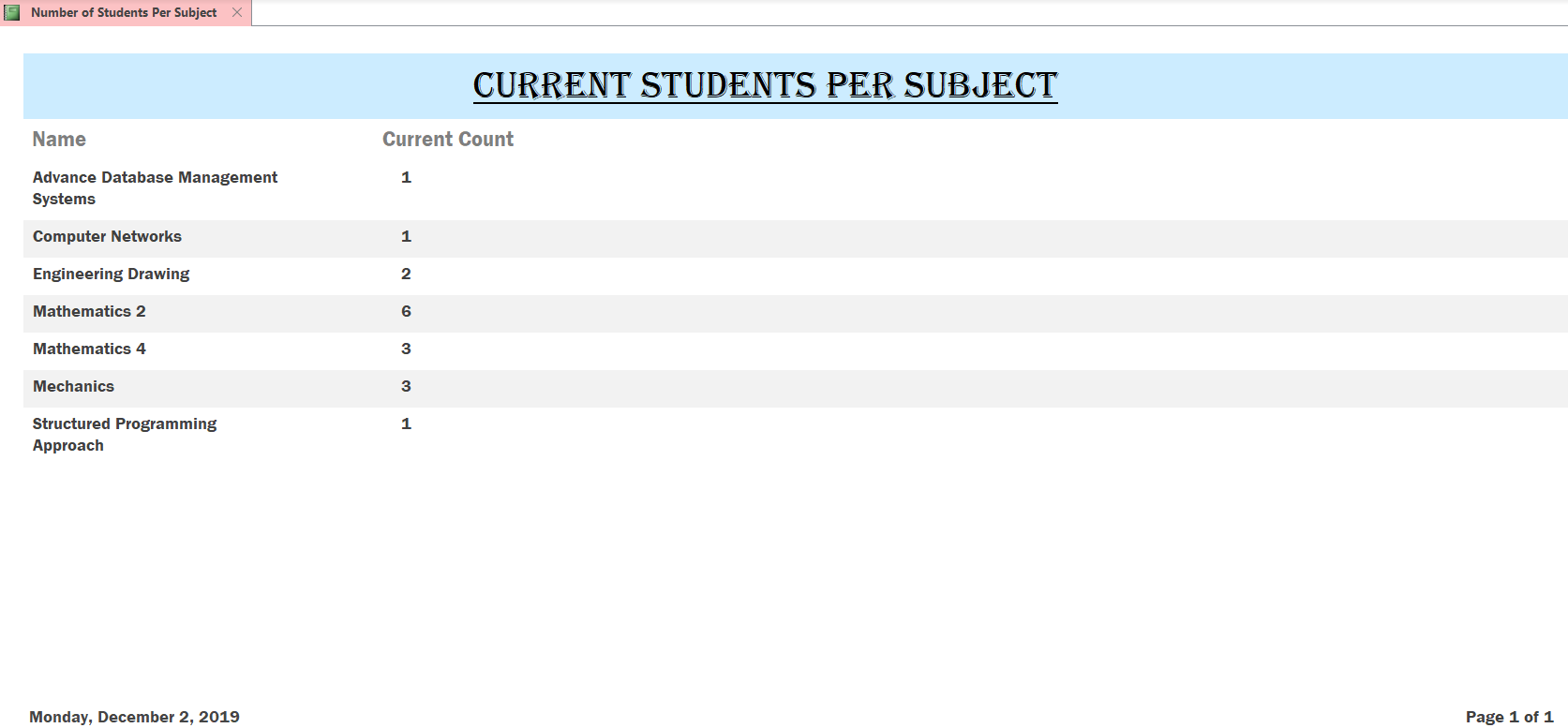
1. Currently Enrolled Students:

It displays the details of all those students who are currently studying in the tutorials.



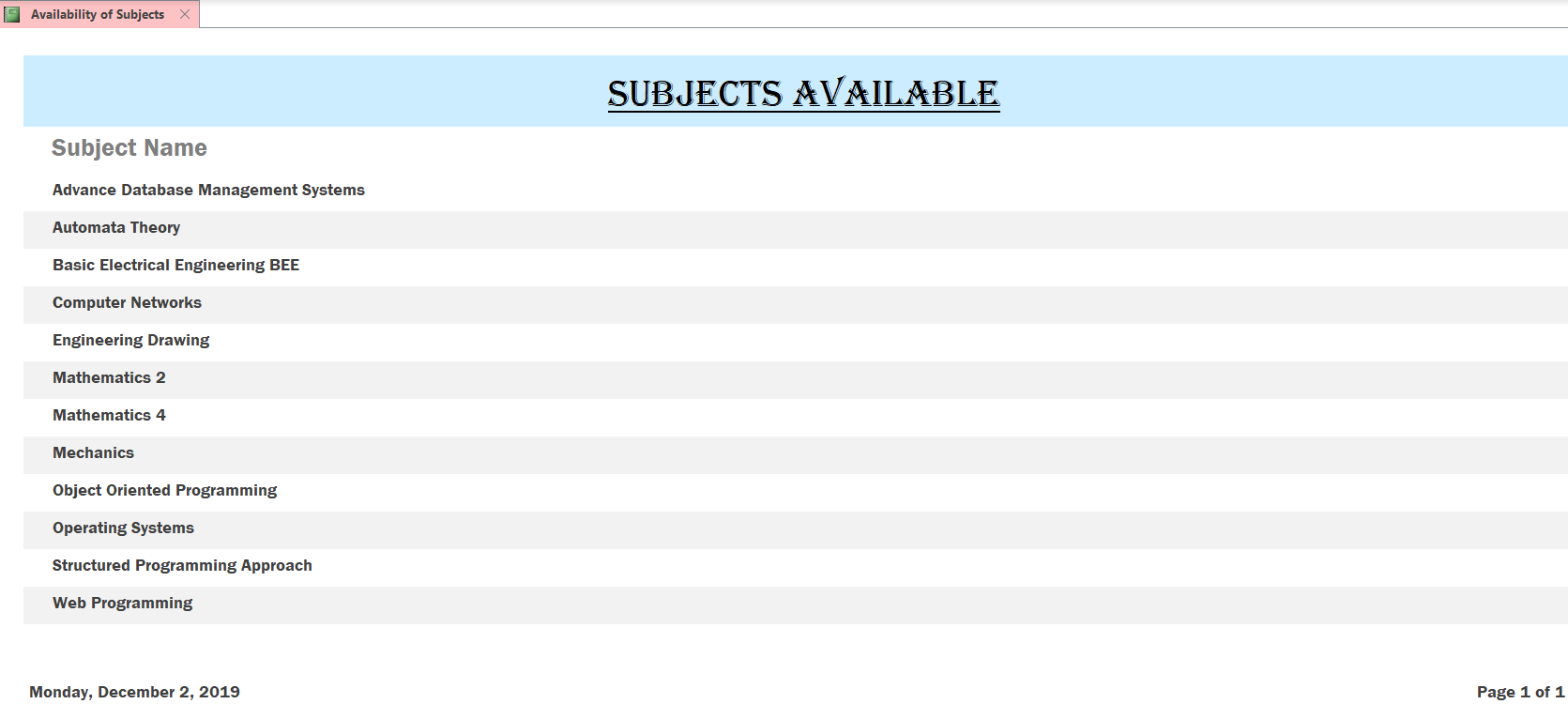
1. Number of Students per Subject:

This report displays the count of the students for each subject.



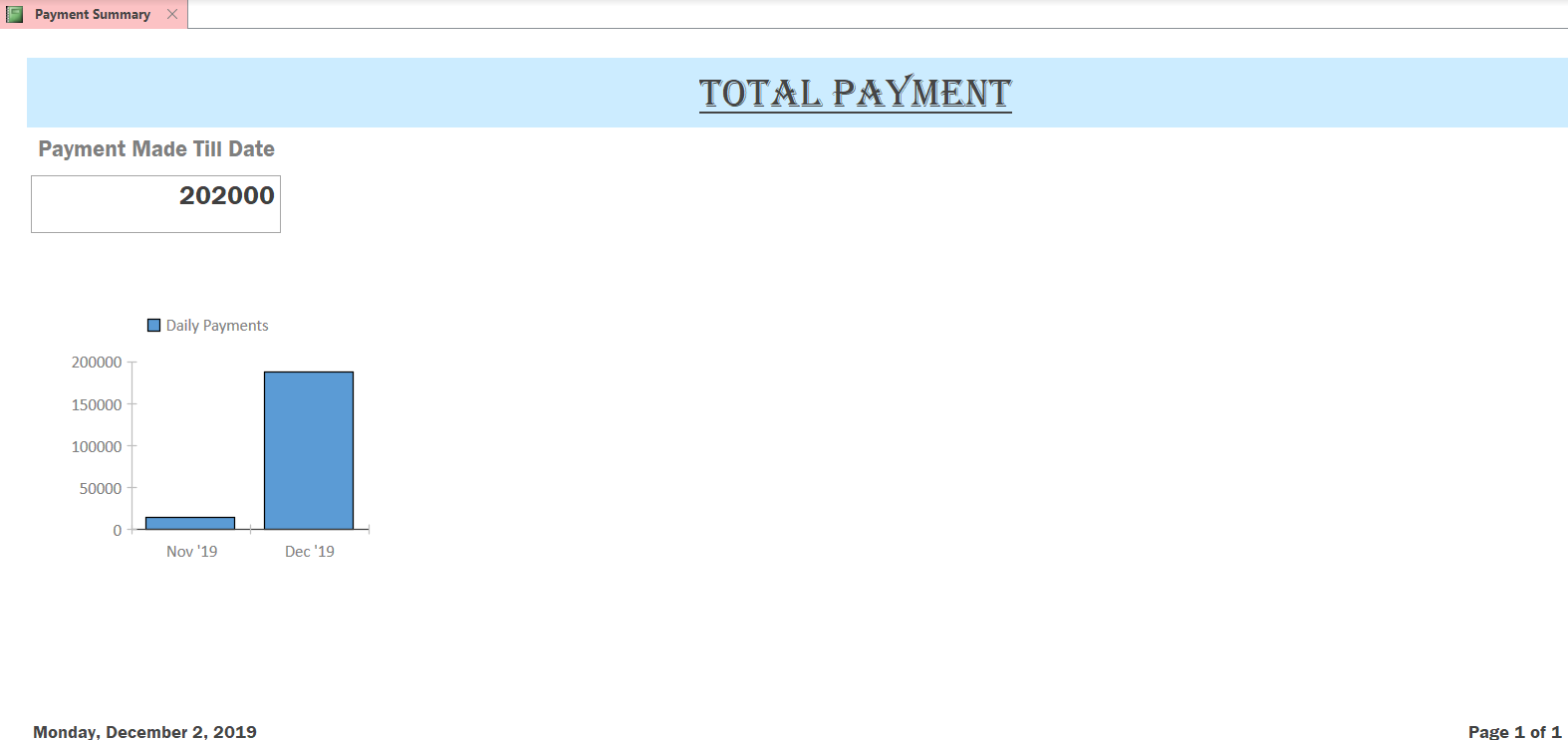
1. Subjects Available:

This displays those subjects that are taught by the tutorials.



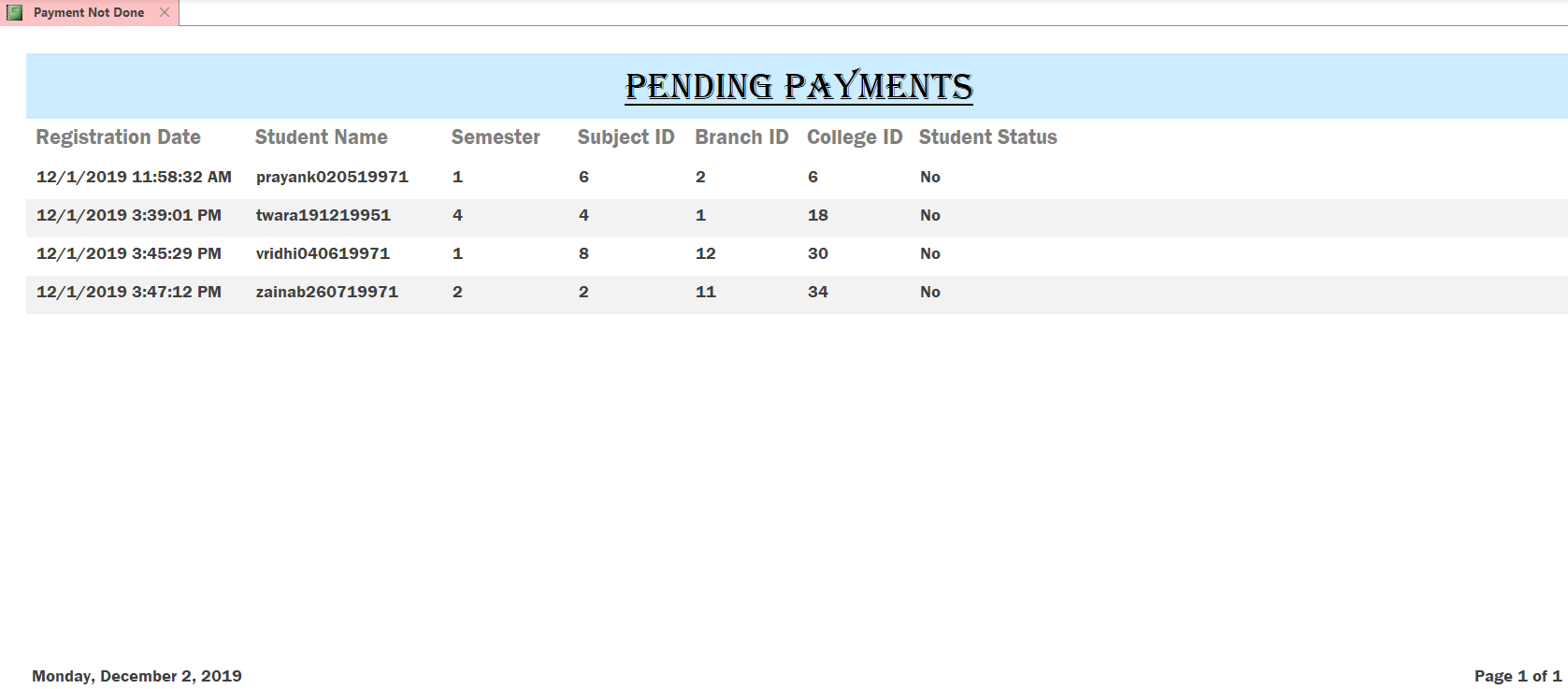
1. Total Payment:

This report displays the total amount in rupees earned by the tutorials till date. It also displays a graph that shows the payments made monthly.



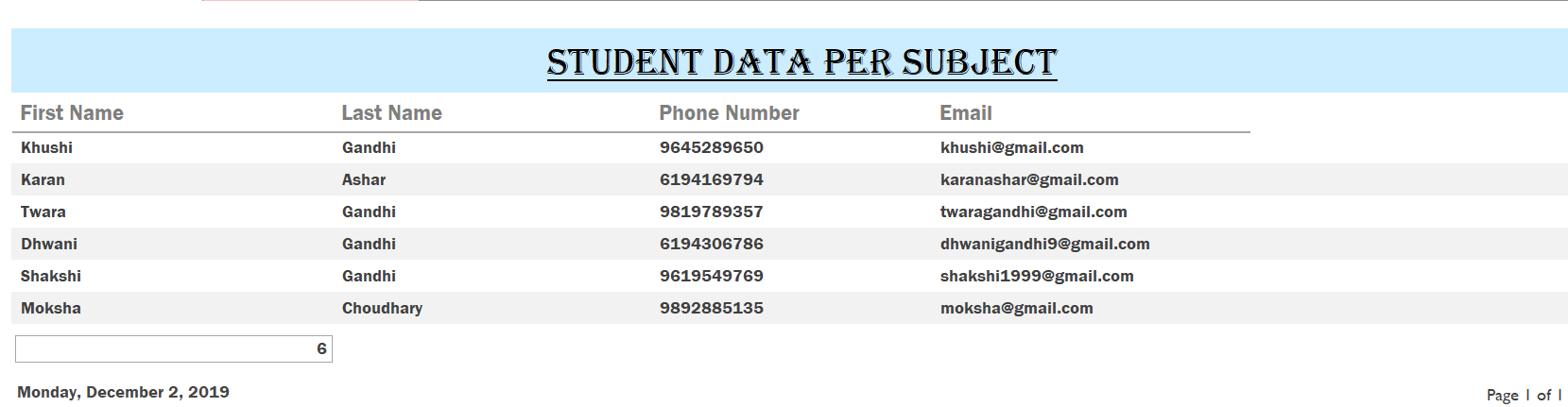
1. Pending Payments:

This generates a report consisting of student data of those students who have registered for the tutorials but have not paid yet.



1. Student Data per Subject:

This report keeps changing according to the subject selected. It displays data of only those students who are studying that selected subject.



**TRIGGERS**

**Without Trigger:**

By default, the student\_status in the student registration table and status\_1 in student table is ‘No’. The admin needs to keep changing this status manually once the payment is done by the student for each student.

**With Trigger:**

Here as soon as an entry is made in the payment table, the corresponding entry in the student\_registration table and the student table are matched, and the respective status are changed to ‘Yes’

CREATE TRIGGER Update\_Status

ON Payment

FOR INSERT AS

BEGIN

UPDATE Student\_Registration

SET Student\_Registration.Student\_Status='Yes'

FROM Student\_Registration

INNER JOIN Payment ON

Student\_Registration.Semester=Payment.Semester AND

Student\_Registration.Subject\_ID=Payment.Subject\_ID AND

Student\_Registration.Branch\_ID=Payment.Branch\_ID AND

Student\_Registration.College\_ID=Payment.College\_ID AND

Student\_Registration.U\_Name=Payment.U\_Name

END;

CREATE TRIGGER Update\_Status1

ON Payment

FOR INSERT AS

BEGIN

UPDATE Student

SET Student.Status\_1='Yes'

FROM Student

INNER JOIN Payment ON

Student.U\_Name=Payment.U\_Name

END;