IN PARTNERSHIP WITH PLYMOUTH UNIVERSITY

Name: Ranasinghe Ranasinghe
Student Reference Number:10903090

Module Code: PUSL2019	Module Name: Information Management and Retrieval
Coursework Title: Database for NSBM Green University	
Deadline Date: 16/01/2024	Member of staff responsible for coursework: Mr Naji Saravanabavan
Programme: Bsc(Honours) Computer Security	

Please note that University Academic Regulations are available under Rules and Regulations on the University website www.plymouth.ac.uk/studenthandbook.

Group work: please list all names of all participants formally associated with this work and state whether the work was undertaken alone or as part of a team. Please note you may be required to identify individual responsibility for component parts.

	Group Number	B
	Student ID	Name
1	10899343	Priyantha Ranasinghe
2	10899233	Dodampe Nimna
3	10899228	Rankira Kosgollage
4	10903090	Ranasinghe Ranasinghe
5	10898748	Gardihewage Dayarathne
6	10900326	Withanage Mel
7	10817967	CDM Fernando
8	10822745	Abeysinghe Herath

We confirm that we have read and understood the Plymouth University regulations relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.

Signed on behalf of the group: RAC Ranasinghe

Individual assignment: ***I confirm that I have read and understood the Plymouth University regulations relating to Assessment Offences and that I am aware of the possible penalties for any breach of these regulations. I confirm that this is my own independent work.*** Signed:
Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence. I *have used/not used translation software. If used, please state name of software……
Overall mark _____% Assessors Initials _____ Date_____

*Please delete as appropriateSci/ps/d:/students/cwkfrontcover/2013/14

Due to the quality decrease of the images, the links related to the images are attached at the end of each diagram.

Table of Contents

Introduction

In the database created for NSBM Green University our team gathered important details related to the university. Below is the complete scenario we created with the gathered information.

Any person can apply for NSBM university as a student, alumnus, or employee. First, the person has to give their educational qualifications, guardianship, and extra qualifications details. If the person is an employee, they have a process to go through to work as an admin employee, an academic employee, or a student assistant which can be either a research assistant or a teaching assistant.

If the person is a student, they are to be registered through the office, When the students are successfully registered, the university provides them with their access details and through these access details, the student can access their logins. Once that is done, the office provides the students with a membership to the library.

Next, the students are enrolled in their courses through the office when they pay their course fees. Courses also have many degrees programs, and these courses are offered by a faculty and departments. Each one of the faculties have many lecture halls and a faculty can have many departments. All courses have modules, and these modules have instructors as well as students. The instructors teach modules to all of the students. Instructors are also informed of details by the NSBM university. When the modules are taught to the students, their attendance is also checked. A student can also tutor a module to another student, a student can be either a tutor or a tutee.

Each module has one exam. All the students must participate to all of the exams. If a student’s grade or attendance is a failure/low, they must re-participate in the repeat exam after registering through the office by paying the repeat fee. All of the exams are conducted by the exam division. The exam divisions are informed of all the exam details through NSBM University. If a student cannot sit for the exam because of a health reason they must submit their health records to the exam divisions.

NSBM also have events, clubs, and sports and a hostel. An event can sell tickets and the students can buy one ticket per each and through those tickets, they can participate in the events. Students get the opportunity to join clubs and sports that they are interested in. Students also get the choice to live in the hostel.

Once the students pass all their exams and complete the duration of their degree program the student's GPA is calculated, and the students achieve their graduation and their degree program.

EER Diagram.

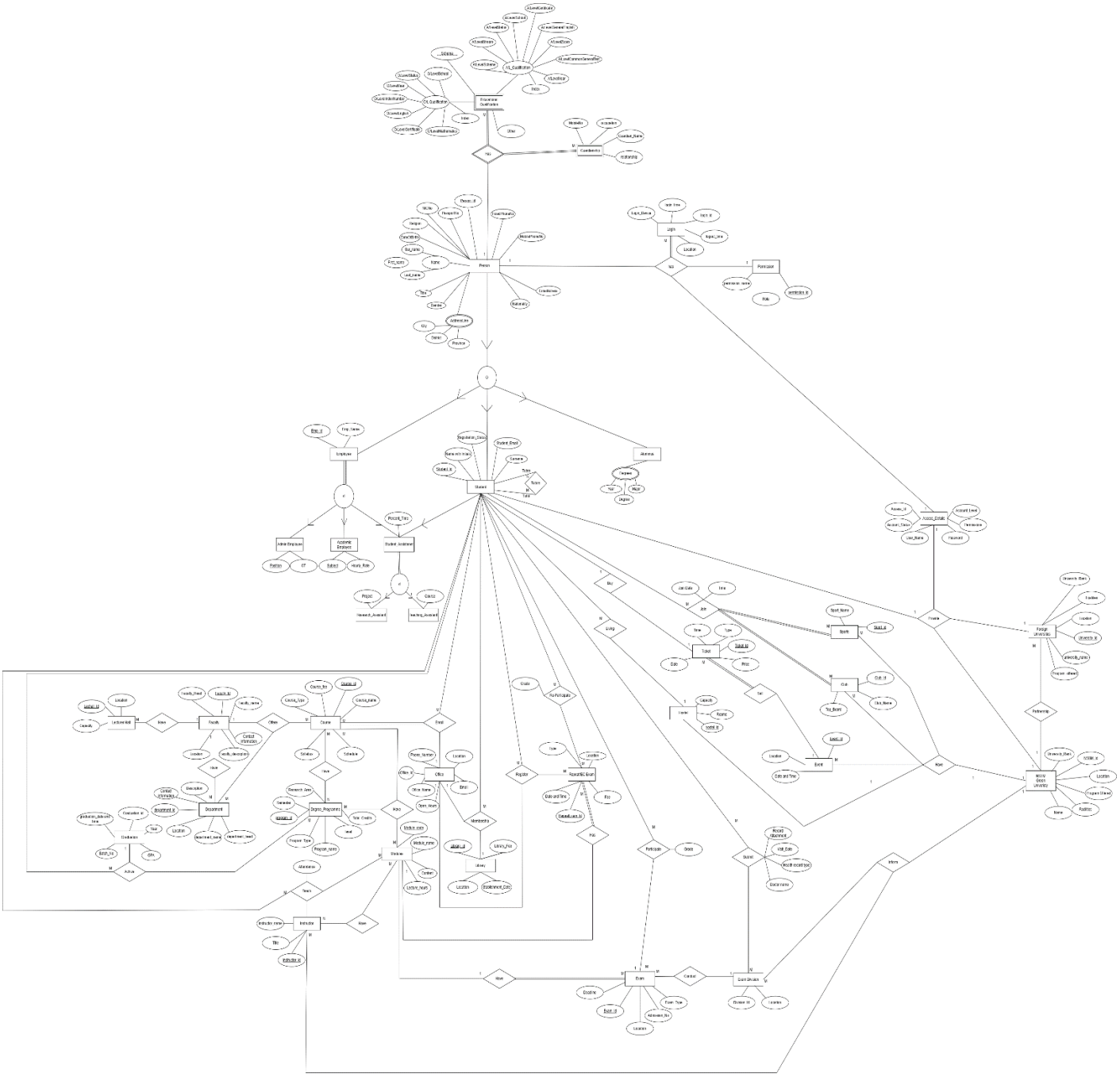


diagram link -

https://drive.google.com/file/d/1EvSeelpyga4pM45MyQR_dGx56CHVJmUW/view?usp=sharing

Assumptions

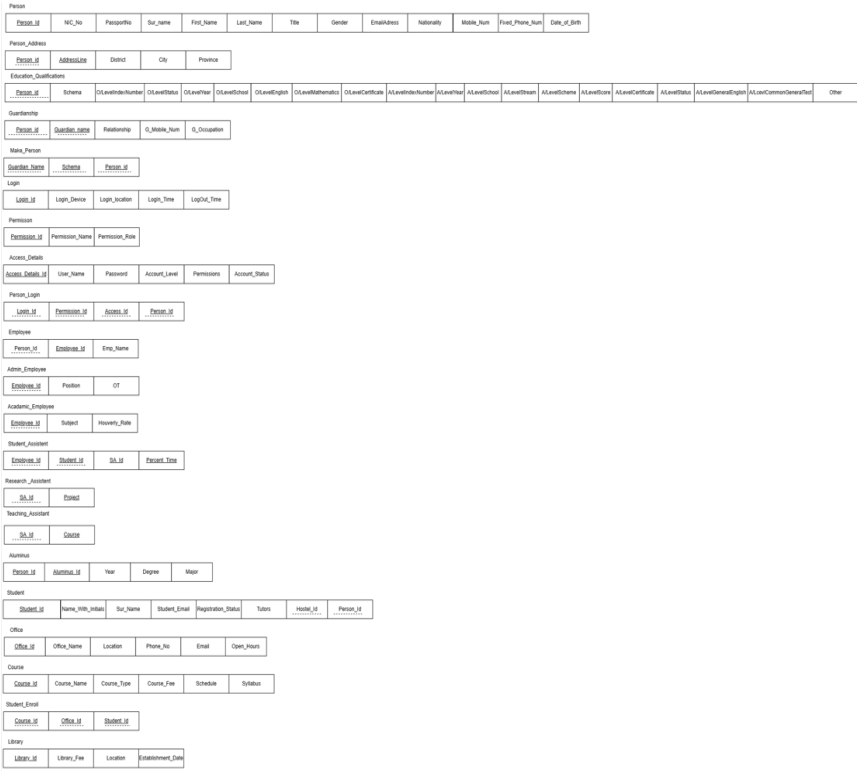
- Once the student is registered through the office, NSBM university, or the foreign universities that they are registered with will provide the access details.
- There is a process within the student entity to store and manage these access details.
- Since NSBM and Foreign universities have a partnership, they offer more diverse degree programs.
- Tutor, Tutee Relationship represents the tutoring relationship between students. A student can be a tutor or a tutee, many students can tutor many students.
- Department, Faculty, and Lecture Hall represent the academic structure of the university.
- Hostel, Event, Club, and Sport represent different aspects of campus life.
- Once the student receives their access details, they can log in to their student accounts.
- There is a relationship to manage the multiple logins and associated permissions for a person.
- There is a process to manage the calculation of a student's GPA.
- There is a process related to the alumnus.
- There is a process to manage the sale of tickets for events.
- There is a separate relationship to manage the library membership details for students.
- There is a separate process to manage health records.
- There is a relationship to indicate whether a student is living in the hostel.

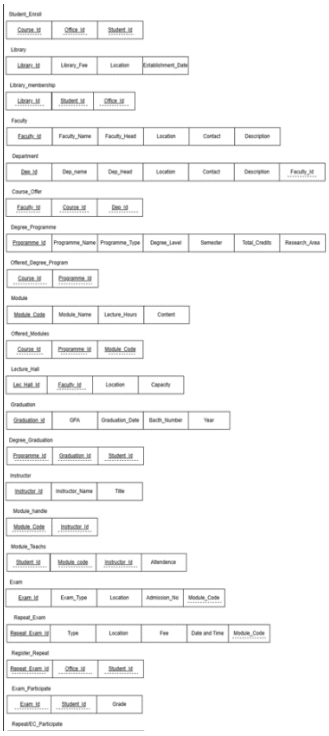
Relational Mapping

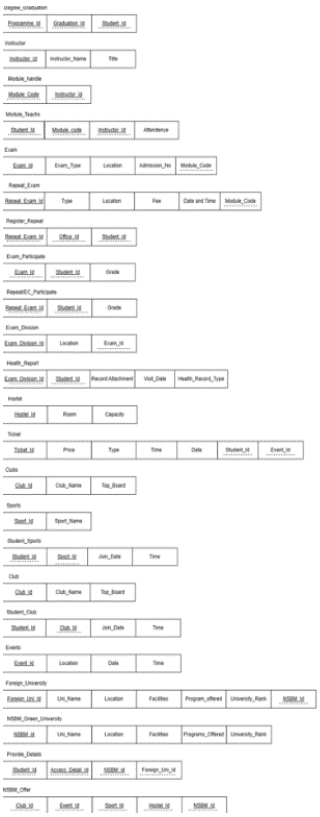
Schema link - https://drive.google.com/file/d/1H3VawE3BIL32IlDTKOyINg_eCYBNAA8p/view?usp=sharing

Normalization

1st Normalization Form

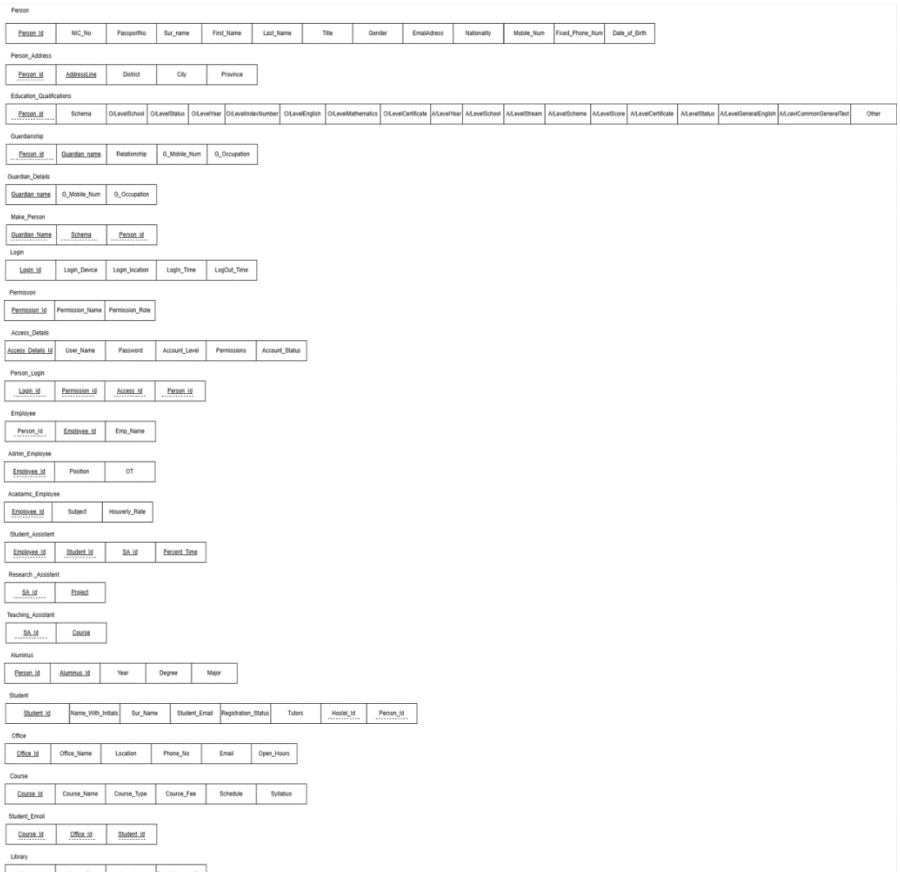


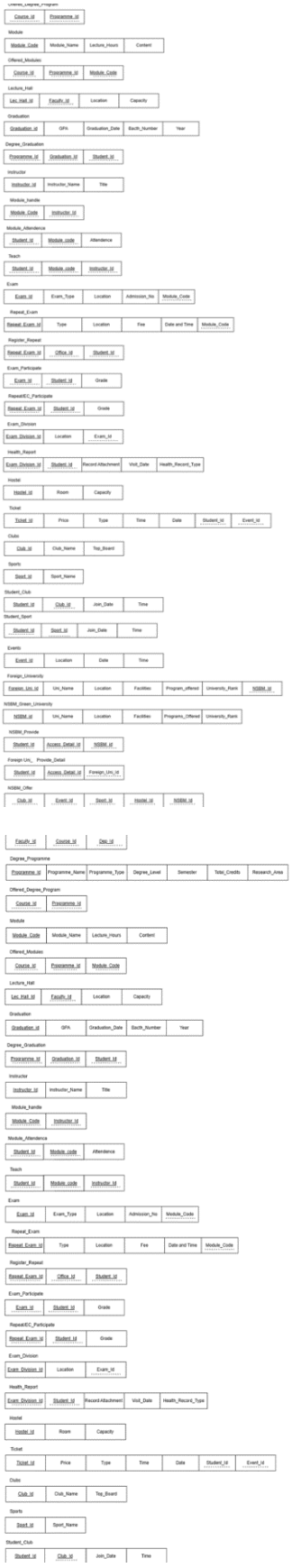




1NF Table link –
https://drive.google.com/file/d/1aT86NmyfivQwIVbZCPKFEdctztpswzj9/view?usp=sharing

2nd Normalization Form (Removed Functional Dependencies)

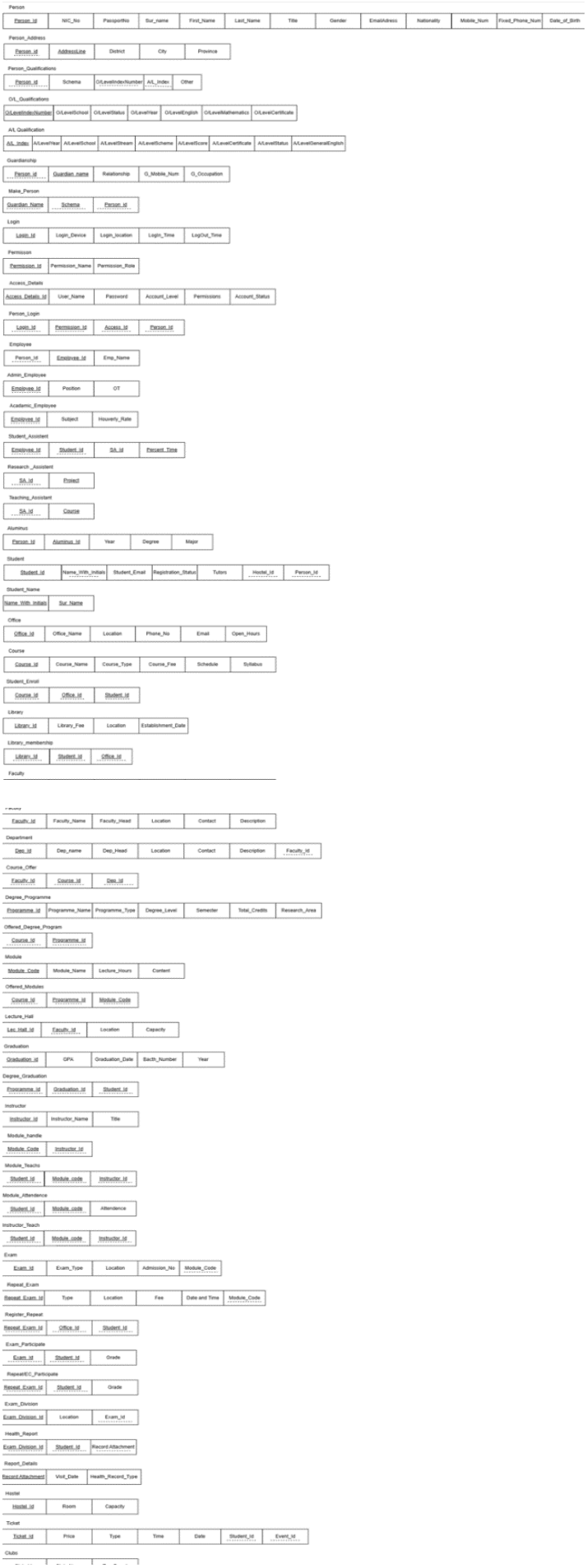


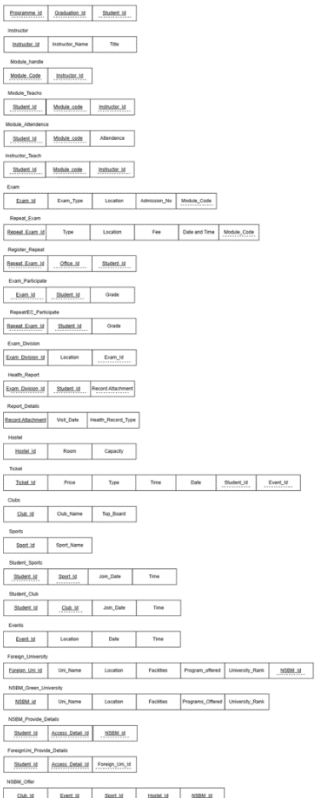


2NF Table link –

https://drive.google.com/file/d/1mCpQ-4OaxcbS4GK4luC2Gfp_2N6Seagk/view?usp=sharing

3rd Normalization Form (Removed Transitive Dependencies)





3NF table link - https://drive.google.com/file/d/1g5YrNAYGvcIqR5RgWd70eNRXxz-Nf1BE/view?usp=sharing

Data Dictionary

Person

FieldName	Datatype	FieldLength	Constraint	Description
Person_ID	int	10	PrimaryKey	ID number of the person
NIC_NO	varchar	15	NotNull	National Identification Number
PassportNo	varchar	50	NotNull	Passport Number
Sur_Name	varchar	50	NotNull	Surname of the student
First_Name	varchar	50	NotNull	First name of the student
Last_Name	varchar	50	NotNull	Last name of the student
Title	varchar	20	NotNull	Job title
Gender	char	1	NotNull	F- Female, M- Male, N- Unknown (null)
EmailAddress	varchar	50	NotNull	Address of the person
Nationality	varchar	15	NotNull	Nationality
Mobile_Num	number	10	NotNull	Mobile number of the person
Fixed_Phone_Num	number	10	NotNull	Fixed phone number of the person
Date_of_Birth	date	15	NotNull	Date of The Birth

Person_Address

FieldName	Datatype	FieldLength	Constraint	Description
Person_ID	int	10	PrimaryKey, ForeignKey	ID number of the person
City	varchar	50	NotNull	City
District	varchar	50	NotNull	District
Province	varchar	50	NotNull	Province

Person_Qualification

FieldName	Datatype	FieldLength	Constraint	Description
Person_ID	int	10	PrimaryKey,ForeignKey	ID number of the person
O/LevelIndexNumber	int	10	ForeignKey	Ordinary Level index number
A/L_Index	int	10	ForeignKey	Advanced Level index number
Other	varchar	50	NotNull	Other informations

O/L_Qualification

FieldName	Datatype	FieldLength	Constraint	Description
O/LevelIndexNumber	int	10	PrimaryKey	Index number of Ordinary Level
O/LevelSchool	varchar	50	NotNull	The school where O/Level was conducted
O/LevelStatus	varchar	10	NotNull	Ordinary Level Status
O/LevelYear	date	4	NotNull	The year that exam was written
O/LevelEnglish	varchar	10	NotNull	Status of O/L English paper
O/LevelMathematics	varchar	10	NotNull	Status of O/L Maths paper
O/LevelCertificate	varchar	10	NotNull	Ordinary Level certificate issued by government

A/L_Qualification

FieldName	Datatype	FieldLength	Constraint	Description
A/L_Index	int	15	PrimaryKey	Index number of Advanced Level
A/LevelYear	date	4	NotNull	The year that exam was written
A/LevelSchool	varchar	50	NotNull	The school where A/Level was conducted
A/LevelStreame	varchar	10	NotNull	
A/LScheme	varchar	10	NotNull	
A/LevelScore	int	10	NotNull	A/Level Z -score
A/LevelCertificate	varchar	10	NotNull	Advanced Level certificate issued by government
A/LevelStatus	varchar	10	NotNull	Advanced Level Status
A/LevelGeneralEnglish	varchar	10	NotNull	Status of A/L English subject

Guardianship

FieldName	Datatype	FieldLength	Constraint	Description
Person_ID	int	10	PrimaryKey,ForeignKey	ID number of the person
Guardian_Name	varchar	50	NotNull	Name of the guardian
Relationship	varchar	50	PrimaryKey	Relationship with the person
G_Mobile_Num	varchar	10	NotNull	Guardian mobile number
G_Occupation	varchar	15	NotNull	Occupation of the guardian

Extra_Qualification

FieldName	Datatype	FieldLength	Constraint	Description
Person_Id	int	10	PrimaryKey	Id number of the person
LanguageSkills	varchar	15	NotNull	Language skills that person has
Sports	varchar	15	PrimaryKey	Sports that the person had play

Make_Person

FieldName	Datatype	FieldLength	Constraint	Description
Guardian_Relationship	varchar	50	PrimaryKey,ForeignKey	Name of the guardian
Indexes	int	15	PrimaryKey,ForeignKey	
Person_Id	int	10	PrimaryKey,ForeignKey	Id number of the person
Sports	varchar	15	PrimaryKey,ForeignKey	

Login

FieldName	Datatype	FieldLength	Constraint	Description
Login_Id	varchar	20	PrimaryKey	Login ID
Login_Device	varchar	10	NotNull	Login device
Login_location	varchar	30	NotNull	Login location
Login_Time	time	8	NotNull	Login time
Logout_Time	time	8	NotNull	Logout time

Employee

FieldName	Datatype	FieldLength	Constraint	Description
Person_ID	int	10	Foreign Key	ID number of the person
Employee_ID	int	10	PrimaryKey	ID number of the employee
Emp_Name	varchar	100	NotNull	Name of the employee

Admin_Employee

FieldName	Datatype	FieldLength	Constraint	Description
Employee_ID	int	10	PrimaryKey,Foreign Key	ID number of the Admin_Employee
Position	varchar	50	NotNull	Position of the Admin_Employee
OT	varchar	50	NotNull	OT

Acadamic_Employee

FieldName	Datatype	FieldLength	Constraint	Description
Employee_ID	int	10	PrimaryKey,Foreign Key	ID number of the employee
Subject	varchar	50	NotNull	Name of the subject
Hourly_Rate	varchar	50	NotNull	Hourly rate of a Acadamic Employee

Student_Assistant

FieldName	Datatype	FieldLength	Constraint	Description
Employee_ID	int	10	PrimaryKey,Foreign Key	ID number of the employee
Student_ID	int	10	PrimaryKey,Foreign Key	ID number of the student
SA_ID	int	10	PrimaryKey	ID number of the
Precent_Time	varchar	20	PrimaryKey	Precent time of a student assistant

Research _Assistant

FieldName	Datatype	FieldLength	Constraint	Description
SA_ID	int	10	PrimaryKey,Foreign Key	ID number of the Research_Assistant
Project	varchar	50	PrimaryKey	Name of the project

Research _Assistant

FieldName	Datatype	FieldLength	Constraint	Description
SA_ID	int	10	PrimaryKey,Foreign Key	ID number of the Research_Assistant
Project	varchar	50	PrimaryKey	Name of the project

Teaching _Assistant

FieldName	Datatype	FieldLength	Constraint	Description
SA_ID	int	10	PrimaryKey,Foreign Key	ID number of the Teaching _Assistant
Course	varchar	50	PrimaryKey	Name of the course

Aluminus

FieldName	Datatype	FieldLength	Constraint	Description
Person_ID	int	10	PrimaryKey	ID number of the person
Aluminus_ID	int	10	PrimaryKey	ID number of the Aluminus
Year	Date	10	NotNull	Year of the Aluminus
Degree	varchar	100	NotNull	Degree of the Aluminus
Major	varchar	100	NotNull	Major of the Aluminus

Student

FieldName	Datatype	FieldLength	Constraint	Description
Student_Id	int	10	PrimaryKey	Id number of the student
Name_With_initials	varchar	50	Foreign Key	student name with initials
Student_Email	varchar	50	NotNull	Email of the student
Registration_Status	varchar	10	NotNull	Registration status
Tutors	varchar	10	NotNull	Tutors assigned for student
Hostel_Id	varchar	10	Foreign Key	student hostel Id number
Person_Id	varchar	10	Foreign Key	Id number of the person

Student_Name

FieldName	Datatype	FieldLength	Constraint	Description
Name_with_initials	varchar	50	PrimaryKey	Student name with initials
Sur_Name	varchar	20	PrimaryKey	Surname of the student

Office

FieldName	Datatype	FieldLength	Constraint	Description
office_ID	int	10	PrimaryKey	Id number of the office
Office_Name	varchar	50	NotNull	Name of the office
Location	varchar	100	NotNull	Location of the office
Phone_No	number	10	NotNull	Contact number of the office
Email	varchar	100	NotNull	Email of the office
Open_Hours	varchar	4	NotNull	Open hours

Course

FieldName	Datatype	FieldLength	Constraint	Description
Course_ID	int	10	PrimaryKey	Id number of the course
Course_Name	varchar	100	NotNull	Name of the course
Course_Type	varchar	50	NotNull	Type of the course
Course_Fee	varchar	30	NotNull	Fee of the course
Schedule	varchar	50	NotNull	Schedule of the course
Syllabus	varchar	50	NotNull	Syllabus

Student_Enroll

FieldName	Datatype	FieldLength	Constraint	Description
Course_ID	int	10	PrimaryKey,Foreign Key	Id number of the course
Office_ID	int	10	PrimaryKey,Foreign Key	Id number of the office
Student_ID	int	10	PrimaryKey,Foreign Key	Id number of the student

Library

FieldName	Datatype	FieldLength	Constraint	Description
Library_ID	int	10	PrimaryKey	Id number of the library
Library_Fee	varchar	20	NotNull	Fee of library
Location	varchar	100	NotNull	Location of the library
Establishment_Date	date	10	NotNull	Established date of the library

Library_Membership

FieldName	Datatype	FieldLength	Constraint	Description
Library_ID	int	10	PrimaryKey,Foreign Key	Id number of the library
Student_ID	int	10	PrimaryKey,Foreign Key	Id number of the student
Office_ID	int	10	PrimaryKey,Foreign Key	Id number of the office

Faculty

FieldName	Datatype	FieldLength	Constraint	Description
Faculty_ID	int	10	PrimaryKey	Id number of the faculty
Faculty_Name	varchar	50	NotNull	Name of the faculty
Faculty_Head	varchar	100	NotNull	Name of the faculty head
Location	varchar	100	NotNull	Location of the faculty
Contact	number	10	NotNull	Contact number of the library
Description	varchar	100	NotNull	Description

Department

FieldName	Datatype	FieldLength	Constraint	Description
Dep_ID	int	10	PrimaryKey	Id number of the department
Dep_Name	varchar	100	NotNull	Name of the department
Dep_Head	varchar	100	NotNull	Department head
Location	varchar	100	NotNull	Location of the department
Contact	number	10	NotNull	Contact number of the department
Description	varchar	100	NotNull	Description
Faculty_ID	int	10	Foreign Key	Id number of the faculty

Course_Offer

FieldName	Datatype	FieldLength	Constraint	Description
Faculty_ID	int	10	PrimaryKey,Foreign Key	Id number of the faculty
Course_ID	int	10	PrimaryKey,Foreign Key	Id number of the course
Dep_ID	int	10	PrimaryKey,Foreign Key	Id number of the department

Degree_Programme

FieldName	Datatype	FieldLength	Constraint	Description
Programme_ID	int	10	PrimaryKey	Id number of the programme
Programme_Name	varchar	100	NotNull	Name of the degree programme
Programme_Type	varchar	50	NotNull	Type of the programme
Degree_Level	varchar	50	NotNull	Level of the degree
Semester	varchar	20	NotNull	Semester
Total_Credits	varchar	5	NotNull	Total credits for degree programme
Research_Area	varchar	100	NotNull	Research area

Offered_Degree_Program

FieldName	Datatype	FieldLength	Constraint	Description
Course_ID	int	10	PrimaryKey,Foreign Key	Id number of the course
Programme_ID	int	10	PrimaryKey,Foreign Key	Id number of the programme

Module

FieldName	Datatype	FieldLength	Constraint	Description
Module_Code	varchar	10	PrimaryKey	Module code of the assigned modules
Module_Name	varchar	50	NotNull	Name of the module
Lecture_Hours	int	08	NotNull	Lecture hours allocated for each module
Content	varchar	20	NotNull	Content of the module

Offered_Modules

FieldName	Datatype	FieldLength	Constraint	Description
Course_Id	Int	10	PrimaryKey	Id number of the course
Programme_Id	int	10	PrimaryKey	Id number of the program
Module_Code	varchar	10	PrimaryKey	Module code

Lecture_Hall

FieldName	Datatype	FieldLength	Constraint	Description
Lec_Hall_Id	varchar	5	PrimaryKey	Id of the lecture hall
Faculty_Id	varchar	2	PrimaryKey	Id of the faculty
Location	varchar	20	NotNull	Location of the lecture hall
Capacity	varchar	1000	NotNull	Capacity of the lecture hall

Graduation

FieldName	Datatype	FieldLength	Constraint	Description
Graduation_Id	varchar	10	PrimaryKey	Graduation Id
GPA	int	10	NotNull	Grade point average of the graduated student
Graduation_Date	date	10	NotNull	Date of the graduation
Batch_Number	varchar	10	NotNull	Batch number of the graduated student
Year	date	4	NotNull	Year of the graduation

Degree_Graduation

FieldName	Datatype	FieldLength	Constraint	Description
Programme_Id	int	10	PrimaryKey	Id number of the program
Graduation_Id	int	10	PrimaryKey	Id number of the graduation
Student_Id	int	10	PrimaryKey	Id number of the student

Instructor

FieldName	Datatype	FieldLength	Constraint	Description
Instructor_Id	int	10	PrimaryKey	Id number of the instructor
Instructor_Name	varchar	50	NotNull	Name of the instructor
Title	varchar	50	NotNull	Job title of the instructor

Module_handle

FieldName	Datatype	FieldLength	Constraint	Description
Module_Code	varchar	10	PrimaryKey	Module code
Instructor_Id	int	10	PrimaryKey	Id number of the instructor

Module_Teaches

FieldName	Datatype	FieldLength	Constraint	Description
Student_Id	int	10	PrimaryKey	Id number of the student
Module_Code	varchar	10	PrimaryKey	Module code
Instructor_Id	int	10	PrimaryKey	Id number of the instructor

Module_Attendance

FieldName	Datatype	FieldLength	Constraint	Description
Student_Id	int	10	PrimaryKey	Id number of the student
Module_Code	varchar	10	PrimaryKey	module code
Attendance	boolean	1	NotNull	Attendance for the modules

Intstructor_Teach

FieldName	Datatype	FieldLength	Constraint	Description
Student_ID	int	10	PrimaryKey,Foreign Key	Id number of the student
Module_Code	varchar	50	PrimaryKey,Foreign Key	Module code
Instructor_ID	int	10	PrimaryKey,Foreign Key	Id number of the instructor

Exam

FieldName	Datatype	FieldLength	Constraint	Description
Exam_ID	int	10	PrimaryKey	Id number of the exam
Exam_Type	varchar	50	NotNull	Type of the exam
Location	varchar	100	NotNull	Exam location
Addmission_No	int	10	NotNull	Addmission number
Module_Code	varchar	50	Foreign Key	Code of the module

Repeat_Exam

FieldName	Datatype	FieldLength	Constraint	Description
Repeat_Exam_ID	int	10	PrimaryKey	Id number of the repeat exam
Type	varchar	50	NotNull	Type of the exam
Location	varchar	100	NotNull	Location
Fee	varchar	20	NotNull	Exam Fee
Date and Time	date	20	NotNull	Date and time
Module_Code	varchar	50	Foreign Key	Code of the module

Register_Repeat

FieldName	Datatype	FieldLength	Constraint	Description
Repeat_Exam_ID	int	10	PrimaryKey,Foreign Key	Id number of the repeat exam
Office_ID	int	10	PrimaryKey,Foreign Key	Id number of the office
Student_ID	int	10	PrimaryKey,Foreign Key	Id number of the student

Exam_Participate

FieldName	Datatype	FieldLength	Constraint	Description
Exam_ID	int	10	PrimaryKey,Foreign Key	Id number of the exam
Student_ID	int	10	PrimaryKey,Foreign Key	Id number of the student
Grade	varchar	5	NotNull	Grade

Exam_Participate

FieldName	Datatype	FieldLength	Constraint	Description
Exam_ID	int	10	PrimaryKey,Foreign Key	Id number of the exam
Student_ID	int	10	PrimaryKey,Foreign Key	Id number of the student
Grade	varchar	5	NotNull	Grade

Repeat/EC_Participate

FieldName	Datatype	FieldLength	Constraint	Description
Repeat_Exam_ID	int	10	PrimaryKey,Foreign Key	Id number of the repeat exam
Student_ID	int	10	PrimaryKey,Foreign Key	Id number of the student
Grade	varchar	5	NotNull	Grade

Exam_Division

FieldName	Datatype	FieldLength	Constraint	Description
Exam_Division_ID	int	10	PrimaryKey	Id number of the exam devision
Location	varchar	100	NotNull	Location of the exam devision
Exam_ID	int	10	Foreign Key	Id number of the exam

Health_Report

FieldName	Datatype	FieldLength	Constraint	Description
Exam_Division_ID	int	10	PrimaryKey,Foreign Key	Id number of the exam devision
Student_ID	int	10	PrimaryKey,Foreign Key	Id number of the student
Record_Attachment	varchar	50	Foreign Key	Record Attachment

Report_Details

FieldName	Datatype	FieldLength	Constraint	Description
Record_Attachment	varchar	50	PrimaryKey	Record Attachment
Visit_Date	date	date	NotNull	Visit date
Health_Research_Type	varchar	100	NotNull	Type of the health research

Report_Details

FieldName	Datatype	FieldLength	Constraint	Description
Record_Attachment	varchar	50	PrimaryKey	Record Attachment
Visit_Date	date	date	NotNull	Visit date
Health_Research_Type	varchar	100	NotNull	Type of the health research

Hostel

FieldName	Datatype	FieldLength	Constraint	Description
Hostel_ID	int	10	PrimaryKey	Id number of the hostel
Room	varchar	10	NotNull	Room
Capacity	varchar	20	NotNull	capacity

Ticket

FieldName	Datatype	FieldLength	Constraint	Description
Ticket_Id	int	10	PrimaryKey	Id of the ticket
Price	varchar	10	NotNull	Price of the ticket
Type	varchar	10	NotNull	Type of the ticket
Time	time	time	NotNull	Time of the event
Date	date	date	NotNull	Date of the event
Student_Id	int	10	ForeignKey	Id of the student(buyer)
Event_Id	int	10	ForeignKey	Id of the event

Clubs

FieldName	Datatype	FieldLength	Constraint	Description
Club_Id	int	10	PrimaryKey	Id of the club
Club_Name	varchar	50	NotNull	Name of the club
Top_Board	varchar	10	NotNull	

Sports

FieldName	Datatype	FieldLength	Constraint	Description
Sport_Id	int	10	PrimaryKey	Id of the sport
Sport_Name	varchar	50	NotNull	Name of the sport

Student_Sports

FieldName	Datatype	FieldLength	Constraint	Description
Student_Id	int	10	PrimaryKey, ForeignKey	Id of the student
Sport_Id	int	10	PrimaryKey, ForeignKey	Id of the sport
Join_Date	date	date	NotNull	Joinned date
Time	time	time	NotNull	Time it held

Student_Club

FieldName	Datatype	FieldLength	Constraint	Description
Student_Id	int	10	PrimaryKey	Id of the student
Club_Id	int	10	PrimaryKey, ForeignKey	Id of the club
Join_Date	date	date	NotNull	Joinned date
Time	time	time	NotNull	Time it held

Events

FieldName	Datatype	FieldLength	Constraint	Description
Event_Id	int	10	PrimaryKey	Id of the event
Location	varchar	20	NotNull	Location of the event
Date	date	date	NotNull	Starting date of the event
Time	time	time	NotNull	Time it held

Foreign_University

FieldName	Datatype	FieldLength	Constraint	Description
Foreign_Uni_Id	int	10	PrimaryKey	Id of the foreign university
Uni_Name	varchar	50	NotNull	Name of the foreign university
Location	varchar	50	NotNull	Location of the foreign university
Facilities	varchar	100	NotNull	Facilities that offers by the foreign university
Program_offered	varchar	100	NotNull	Programs offered by the foreign university
University_Rank	varchar	1000	NotNull	Rank of the univaersity
NSBM_Id	int	10	PrimaryKey, ForeignKey	Id of the NSBM

NSBM_Green_University

FieldName	Datatype	FieldLength	Constraint	Description
NSBM_Id	int	10	PrimaryKey	Id of the NSBM
Uni_Name	varchar	50	NotNull	Name of the university
Location	varchar	50	NotNull	Location of the university
Facilities	varchar	100	NotNull	Facilities offered by NSBM
Program_offered	varchar	100	NotNull	Programs offered by NSBM
University_Rank	varchar	1000	NotNull	NSBM university rank

NSBM_Provide_Details

FieldName	Datatype	FieldLength	Constraint	Description
Student_Id	int	10	PrimaryKey, ForeignKey	Id of the student
Access_Detail_Id	int	10	PrimaryKey, ForeignKey	Id of the access details
NSBM_Id	int	10	PrimaryKey, ForeignKey	Id of the NSBM

ForeignUni_Provide_Details

FieldName	Datatype	FieldLength	Constraint	Description
Student_Id	int	10	PrimaryKey, ForeignKey	Id of the student
Access_Detail_Id	int	10	PrimaryKey, ForeignKey	Id of the access details
Foreign_Uni_Id	int	10	PrimaryKey, ForeignKey	Foreign university id

NSBM_offer

FieldName	Datatype	FieldLength	Constraint	Description
Club_ID	int	10	PrimaryKey,Foreign Key	Id number of the club
Event_ID	int	10	PrimaryKey,Foreign Key	Id number of the event
Sport_ID	int	10	PrimaryKey,Foreign Key	Id number of the sport
Hostel_ID	int	10	PrimaryKey,Foreign Key	Id number of the hostel
NSBM_ID	int	10	PrimaryKey,Foreign Key	Id number of the NSBM

Data Dictionary Link - https://drive.google.com/file/d/1yl3X6Gd4dk4OPTKm3_dekPzG6y9jv6Wv/view?usp=sharing

SQL table statements with related Constraints

```
CREATE DATABASE UniversityDB
USE UniversityDB
CREATE TABLE Person (
    Person_ID INT PRIMARY KEY,
    NIC_NO VARCHAR(15) NOT NULL,
    PassportNo VARCHAR(50) NOT NULL,
    Sur_Name VARCHAR(50) NOT NULL,
    First_Name VARCHAR(50) NOT NULL,
    Last_Name VARCHAR(50) NOT NULL,
    Title VARCHAR(20) NOT NULL,
    Gender CHAR(1) NOT NULL CHECK (Gender IN ('F', 'M', 'N')),
    EmailAddress VARCHAR(50) NOT NULL,
    Nationality VARCHAR(15) NOT NULL,
    Mobile_Num INT NOT NULL,
    Fixed_Phone_Num INT NOT NULL,
    Date_of_Birth DATE NOT NULL
);
CREATE TABLE Person_Address (
    Person_ID INT PRIMARY KEY,
    City VARCHAR(50) NOT NULL,
    District VARCHAR(50) NOT NULL,
    Province VARCHAR(50) NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID)
);
CREATE TABLE O_Level_Qualification (
    O_Level_Index_Number INT PRIMARY KEY,
    O_Level_School VARCHAR(50) NOT NULL,
    O_Level_Status VARCHAR(10) NOT NULL,
    O_Level_Year DATE NOT NULL,
    O_Level_English VARCHAR(10) NOT NULL,
    O_Level_Mathematics VARCHAR(10) NOT NULL,
    O_Level_Certificate VARCHAR(10) NOT NULL
);
CREATE TABLE A_Level_Qualification (
    A_Level_Index INT PRIMARY KEY,
    A_Level_Year DATE NOT NULL,
    A_Level_School VARCHAR(50) NOT NULL,
    A_Level_Stream VARCHAR(10) NOT NULL,
    A_Level_Scheme VARCHAR(10) NOT NULL,
    A_Level_Score INT NOT NULL,
    A_Level_Certificate VARCHAR(10) NOT NULL,
    A_Level_Status VARCHAR(10) NOT NULL,
    A_Level_General_English INT NOT NULL
);


 CREATE TABLE Person_Qualifications (
     Person_ID INT PRIMARY KEY,
     O_Level_Index_Number INT,
     A_Level_Index INT,
     Other VARCHAR(50) NOT NULL,
     FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID),
     FOREIGN KEY (O_Level_Index_Number) REFERENCES
 O_Level_Qualification(O_Level_Index_Number),
     FOREIGN KEY (A_Level_Index) REFERENCES
 A_Level_Qualification(A_Level_Index)
 );
 CREATE TABLE Guardianship (
     Person_ID INT,
     Guardian_Name VARCHAR(50) NOT NULL,
     Relationship VARCHAR(50) NOT NULL,
     G_Mobile_Num VARCHAR(10) NOT NULL,
     G_Occupation VARCHAR(15) NOT NULL,
     PRIMARY KEY (Person_ID, Relationship),
     FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID)
 );
 CREATE TABLE Make_Person (
     Person_ID INT,
     Guardian_Name VARCHAR(50) NOT NULL,
     Relationship VARCHAR(50) NOT NULL,
     Indexes INT,
     PRIMARY KEY (Guardian_Name, Relationship, Indexes,
 Person_Id),
     FOREIGN KEY (Person_ID, Relationship) REFERENCES
 Guardianship(Person_ID, Relationship),
     FOREIGN KEY (Indexes) REFERENCES
 O_Level_Qualification(O_Level_Index_Number),
     FOREIGN KEY (Person_Id) REFERENCES Person(Person_ID)
 );
 CREATE TABLE Login (
     Login_Id VARCHAR(20) PRIMARY KEY,
     Login_Device VARCHAR(10) NOT NULL,
     Login_Location VARCHAR(30) NOT NULL,
     Login_Time TIME NOT NULL,
     Logout_Time TIME NOT NULL
 );
 CREATE TABLE Permission (
     Permission_ID INT PRIMARY KEY,
     Permission_Name VARCHAR(50) NOT NULL,
     Permission_Role VARCHAR(100) NOT NULL
 );
```

```
CREATE TABLE Access_Details (
    Access_Details_ID INT PRIMARY KEY,
    User_Name VARCHAR(20) NOT NULL,
    Password VARCHAR(10) NOT NULL,
    Account_Level VARCHAR(50) NOT NULL,
    Permissions VARCHAR(50) NOT NULL,
    Account_Status VARCHAR(50) NOT NULL
);
CREATE TABLE Person_Login (
   Login_Id VARCHAR(20),
    Permission_ID INT,
    Access_ID INT,
    Person_ID INT,
    PRIMARY KEY (Login_ID, Permission_ID, Access_ID, Person_ID),
    FOREIGN KEY (Login_ID) REFERENCES Login(Login_ID),
    FOREIGN KEY (Permission_ID) REFERENCES
Permission(Permission_ID),
    FOREIGN KEY (Access_ID) REFERENCES
Access_Details(Access_Details_ID),
    FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID)
);
CREATE TABLE Employee (
    Person_ID INT,
    Employee_ID INT PRIMARY KEY,
    Emp_Name VARCHAR(100) NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID)
);
CREATE TABLE Alumnus (
    Person_ID INT,
    Alumnus_ID INT PRIMARY KEY,
    Year DATE NOT NULL,
    Degree VARCHAR(100) NOT NULL,
    Major VARCHAR(100) NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID)
);
CREATE TABLE Admin_Employee (
    Employee_ID INT PRIMARY KEY,
    Position VARCHAR(50) NOT NULL,
    OT VARCHAR(50) NOT NULL,
    FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID)
);


CREATE TABLE Academic_Employee (
    Employee_ID INT PRIMARY KEY,
    Subject VARCHAR(50) NOT NULL,
    Hourly_Rate VARCHAR(50) NOT NULL,
    FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID)
);
CREATE TABLE Student_Name (
    Name_With_Initials VARCHAR(50) PRIMARY KEY,
    Sur_Name VARCHAR(20)
);
CREATE TABLE Hostel(
      Hostel_ID int NOT NULL,
      Room varchar(10) ,
      Capacity varchar(20) ,
      PRIMARY KEY (Hostel_ID)
);

CREATE TABLE Student (
    Student_Id INT PRIMARY KEY,
    Name_With_Initials VARCHAR(50) ,
    Student_Email VARCHAR(50) NOT NULL,
    Registration_Status VARCHAR(10) NOT NULL,
    Tutors VARCHAR(10) NOT NULL,
    Hostel_Id int REFERENCES Hostel(Hostel_Id),
    Person_Id FOREIGN KEY REFERENCES Person(Person_Id),
     FOREIGN KEY (Name_With_Initials) REFERENCES
Student_Name(Name_With_Initials)
);
CREATE TABLE Student_Assistant (
    Employee_ID INT,
    Student_ID INT,
    SA_ID INT PRIMARY KEY,
    Precent_Time VARCHAR(20) NOT NULL,
    -- Other attributes specific to Student_Assistant
    FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID),
    FOREIGN KEY (Student_ID) REFERENCES Student(Student_ID)
);
CREATE TABLE Research_Assistant (
    SA_ID INT PRIMARY KEY,
    Project VARCHAR(50),
    -- Other attributes specific to Research_Assistant
    FOREIGN KEY (SA_ID) REFERENCES Student_Assistant(SA_ID)
);
```

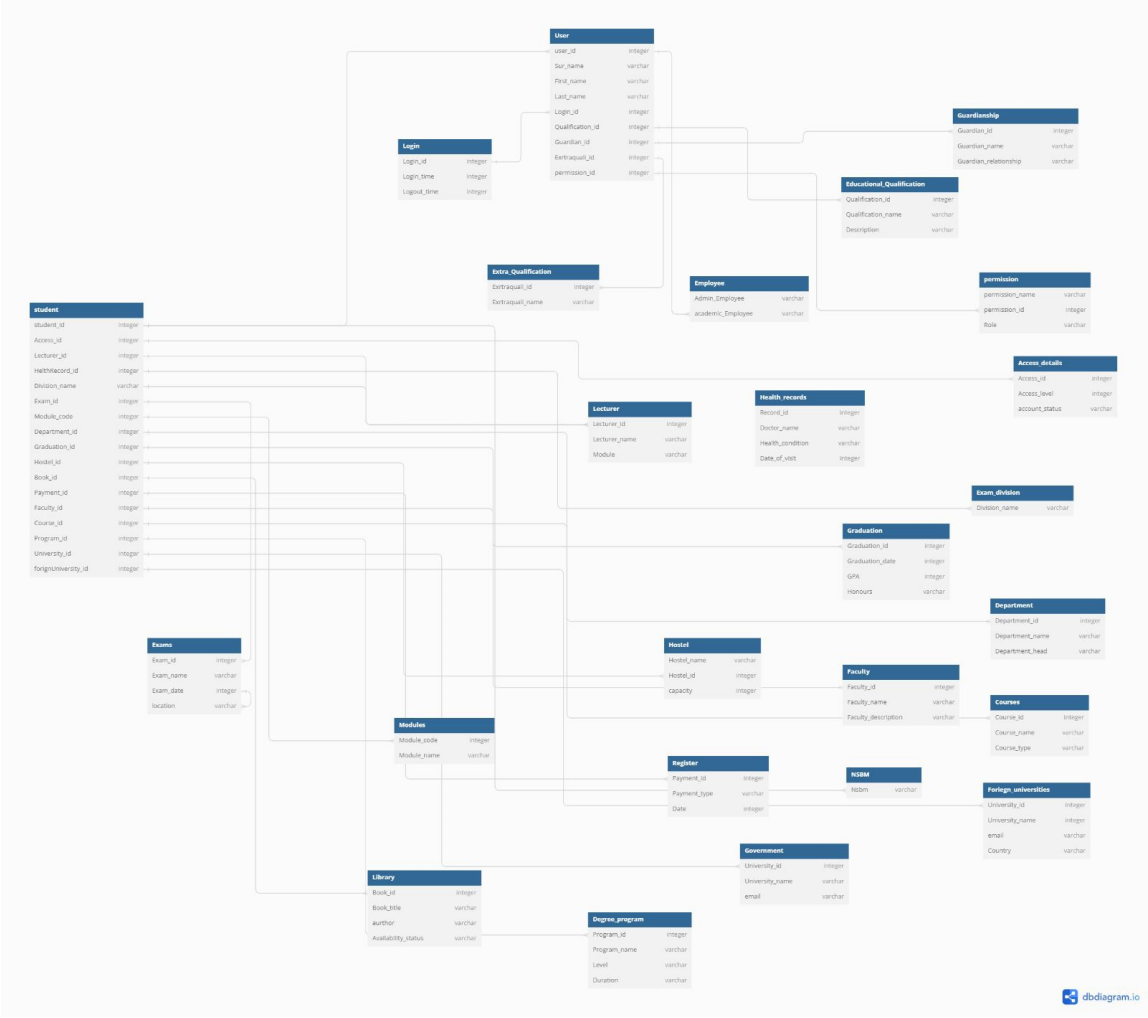
```
CREATE TABLE Teaching_Assistant (
    SA_ID INT PRIMARY KEY,
    Course VARCHAR(50),
    FOREIGN KEY (SA_ID) REFERENCES Student_Assistant(SA_ID)
);
CREATE TABLE Office (
    Office_ID INT PRIMARY KEY,
    Office_Name VARCHAR(50) NOT NULL,
    Location VARCHAR(100) NOT NULL,
    Phone_No VARCHAR(10) NOT NULL, -- Assuming it's a string,
adjust if needed
    Email VARCHAR(100) NOT NULL,
    Open_Hours VARCHAR(4) NOT NULL
);
CREATE TABLE Course (
    Course_ID INT PRIMARY KEY,
    Course_Name VARCHAR(100) NOT NULL,
    Course_Type VARCHAR(50) NOT NULL,
    Course_Fee VARCHAR(30) NOT NULL,
    Schedule VARCHAR(50) NOT NULL,
    Syllabus VARCHAR(50) NOT NULL
);
CREATE TABLE Student_Enroll (
    Course_ID INT,
    Office_ID INT,
    Student_ID INT,
    PRIMARY KEY (Course_ID, Office_ID, Student_ID),
    FOREIGN KEY (Course_ID) REFERENCES Course(Course_ID),
    FOREIGN KEY (Office_ID) REFERENCES Office(Office_ID),
    FOREIGN KEY (Student_ID) REFERENCES Student(Student_ID)
);
CREATE TABLE Library (
    Library_ID INT PRIMARY KEY,
    Library_Fee VARCHAR(20) NOT NULL,
    Location VARCHAR(100) NOT NULL,
    Establishment_Date DATE NOT NULL
);


 CREATE TABLE Library_Membership (
     Library_ID INT,
     Student_ID INT,
     Office_ID INT,
     PRIMARY KEY (Library_ID, Student_ID, Office_ID),
     FOREIGN KEY (Library_ID) REFERENCES Library(Library_ID),
     FOREIGN KEY (Student_ID) REFERENCES Student(Student_ID),
     FOREIGN KEY (Office_ID) REFERENCES Office(Office_ID)
 );
 CREATE TABLE Faculty (
     Faculty_ID VARCHAR(10) PRIMARY KEY,
     Faculty_Name VARCHAR(50) NOT NULL,
     Faculty_Head VARCHAR(100) NOT NULL,
     Location VARCHAR(100) NOT NULL,
     Contact_Number VARCHAR(10) NOT NULL, -- Assuming it's a
 string, adjust if needed
     Description VARCHAR(100) NOT NULL
 );
 CREATE TABLE Department (
     Dep_ID INT PRIMARY KEY,
     Dep_Name VARCHAR(100) NOT NULL,
     Dep_Head VARCHAR(100) NOT NULL,
     Location VARCHAR(100) NOT NULL,
     Contact_Number VARCHAR(10) NOT NULL, -- Assuming it's a
 string, adjust if needed
     Description VARCHAR(100) NOT NULL,
     Faculty_ID VARCHAR(10),
     FOREIGN KEY (Faculty_ID) REFERENCES Faculty(Faculty_ID)
 );
 CREATE TABLE Course_Offeres (
     Faculty_ID VARCHAR(10),
     Course_ID INT,
     Dep_ID INT,
     PRIMARY KEY (Faculty_ID, Course_ID, Dep_ID),
     FOREIGN KEY (Faculty_ID) REFERENCES Faculty(Faculty_ID),
     FOREIGN KEY (Course_ID) REFERENCES Course(Course_ID),
     FOREIGN KEY (Dep_ID) REFERENCES Department(Dep_ID)
 );
```

```
CREATE TABLE Degree_Programme (
    Programme_ID INT PRIMARY KEY,
    Programme_Name VARCHAR(100) NOT NULL,
    Programme_Type VARCHAR(50) NOT NULL,
    Degree_Level VARCHAR(50) NOT NULL,
    Semester VARCHAR(20) NOT NULL,
    Total_Credits VARCHAR(5) NOT NULL,
    Research_Area VARCHAR(100) NOT NULL
);
CREATE TABLE Offered_Degree_Program (
    Course_ID INT,
    Programme_ID INT,
    PRIMARY KEY (Course_ID, Programme_ID),
    FOREIGN KEY (Course_ID) REFERENCES Course(Course_ID),
    FOREIGN KEY (Programme_ID) REFERENCES
Degree_Programme(Programme_ID)
);
CREATE TABLE Module (
    Module_Code VARCHAR(10) PRIMARY KEY,
    Module_Name VARCHAR(50) NOT NULL,
    Lecture_Hours INT NOT NULL CHECK (Lecture_Hours >= 0),
    Content VARCHAR(20) NOT NULL
);
CREATE TABLE Offered_Modules (
    Course_Id INT,
    Programme_Id INT,
    Module_Code VARCHAR(10),
    PRIMARY KEY (Course_Id, Programme_Id, Module_Code),
    FOREIGN KEY (Course_Id) REFERENCES Course(Course_Id),
    FOREIGN KEY (Programme_Id) REFERENCES
Degree_Programme(Programme_Id),
    FOREIGN KEY (Module_Code) REFERENCES Module(Module_Code)
);
CREATE TABLE Lecture_Hall (
    Lec_Hall_Id VARCHAR(5) PRIMARY KEY,
    Faculty_Id VARCHAR(10),
    Location VARCHAR(20) NOT NULL,
    Capacity INT NOT NULL CHECK (Capacity >= 0),
    FOREIGN KEY (Faculty_Id) REFERENCES Faculty(Faculty_Id)
);

 CREATE TABLE Graduation (
     Graduation_Id INT PRIMARY KEY,
     GPA INT NOT NULL,
     Graduation_Date DATE NOT NULL,
     Batch_Number VARCHAR(10) NOT NULL,
     Year DATE NOT NULL
 );
 CREATE TABLE Degree_Graduation (
     Programme_Id INT,
     Graduation_Id INT,
     Student_Id INT,
     PRIMARY KEY (Programme_Id, Graduation_Id, Student_Id),
     FOREIGN KEY (Programme_Id) REFERENCES
 Degree_Programme(Programme_Id),
     FOREIGN KEY (Graduation_Id) REFERENCES
 Graduation(Graduation_Id),
     FOREIGN KEY (Student_Id) REFERENCES Student(Student_Id)
 );
 CREATE TABLE Instructor (
     Instructor_Id INT PRIMARY KEY,
     Instructor_Name VARCHAR(50) NOT NULL,
     Title VARCHAR(50) NOT NULL
 );
 CREATE TABLE Module_Handle (
     Module_Code VARCHAR(10),
     Instructor_Id INT,
     PRIMARY KEY (Module_Code, Instructor_Id),
     FOREIGN KEY (Module_Code) REFERENCES Module(Module_Code),
     FOREIGN KEY (Instructor_Id) REFERENCES
 Instructor(Instructor_Id)
 );
 CREATE TABLE Module_Teaches (
     Student_Id INT,
     Module_Code VARCHAR(10),
     Instructor_Id INT,
     PRIMARY KEY (Student_Id, Module_Code, Instructor_Id),
     FOREIGN KEY (Student_Id) REFERENCES Student(Student_Id),
     FOREIGN KEY (Module_Code, Instructor_Id) REFERENCES
 Module_Handle(Module_Code, Instructor_Id)
 );
```

```
CREATE TABLE Module_Attendance (
    Student_Id INT,
    Module_Code VARCHAR(10),
    Attendance INT NOT NULL,
    PRIMARY KEY (Student_Id, Module_Code),
    FOREIGN KEY (Student_Id) REFERENCES Student(Student_Id),
     FOREIGN KEY (Module_Code) REFERENCES Module(Module_Code)
);
CREATE TABLE Instructor_Teach (
    Student_ID INT,
    Module_Code VARCHAR(10),
    Instructor_ID INT,
    PRIMARY KEY (Student_ID, Module_Code, Instructor_ID),
    FOREIGN KEY (Student_ID) REFERENCES Student(Student_ID),
    FOREIGN KEY (Module_Code) REFERENCES Module(Module_Code),
    FOREIGN KEY (Instructor_ID) REFERENCES
Instructor(Instructor_ID)
);
-- Create Exam table
CREATE TABLE Exam (
    Exam_ID int PRIMARY KEY,
    Exam_Type varchar(50) NOT Null,
    Location varchar(10) Not Null,
    Addmission_No int Not Null,
    Module_code varchar(10),
    FOREIGN KEY (Module_code) REFERENCES Module(Module_code)
);
CREATE TABLE Repeat_Exam (
    Repeat_Exam_ID int PRIMARY KEY,
    Type varchar(50) Not Null,
    Location varchar(100) Not Null,
    Fee varchar(20) Not Null,
     DateandTime date Not Null,
    Module_code varchar(10),
    FOREIGN KEY (Module_code) REFERENCES Module(Module_Code)
);
CREATE TABLE Register_Repeat (
    Repeat_Exam_ID int,
    Office_ID int,
    Student_ID int,
    PRIMARY KEY (Repeat_Exam_ID, Office_ID, Student_ID),FOREIGN
KEY (Repeat_Exam_ID) REFERENCES Repeat_Exam(Repeat_Exam_ID),
    FOREIGN KEY (Office_ID) REFERENCES Office(Office_ID),
    FOREIGN KEY (Student_ID) REFERENCES Student(Student_ID)
);


CREATE TABLE Exam_Participate (
    Exam_ID int ,
    Student_ID int,
    Grade varchar(5) NOT NULL,
    PRIMARY KEY (Exam_ID, Student_ID),
     FOREIGN KEY (Exam_ID) REFERENCES Exam(Exam_ID),
    FOREIGN KEY (Student_ID) REFERENCES Student(Student_ID)
);
CREATE TABLE Repeat(
    Repeat_Exam_ID int ,
    Student_ID int,
     Grade varchar(5) NOT NULL,
    PRIMARY KEY (Repeat_Exam_ID,  Student_ID),
     FOREIGN KEY (Repeat_Exam_ID) REFERENCES
Repeat_Exam(Repeat_Exam_ID),
    FOREIGN KEY (Student_ID) REFERENCES Student(Student_ID)
);
CREATE TABLE Exam_Division (
    Exam_Division_ID int PRIMARY KEY,
    Location varchar(10) NOT NULL,
    Exam_ID int,
    FOREIGN KEY (Exam_ID) REFERENCES Exam(Exam_ID)
);
CREATE TABLE Health_Record (
    Exam_Division_ID INT,
    Student_ID INT,
    Record_Attachment VARCHAR(50),
    PRIMARY KEY (Exam_Division_ID, Student_ID),
    FOREIGN KEY (Exam_Division_ID) REFERENCES
Exam_Division(Exam_Division_ID),
    FOREIGN KEY (Student_ID) REFERENCES Student(Student_ID)
);
CREATE TABLE Report_Details(
     Record_Attachment varchar(50) NOT NULL,
     Visit_Date date ,
      Health_Research_Type varchar(10) ,
      PRIMARY KEY (Record_Attachment));
```

```
CREATE TABLE Events (
Event_ID INT NOT NULL,
Location VARCHAR (20) NOT NULL,
Date DATE NOT NULL,
Time TIME NOT NULL,
PRIMARY KEY (Event_ID)
);
CREATE TABLE Ticket(
      Ticket_Id int NOT NULL,
       Price varchar(10) ,
       Type varchar(10) ,
       Time time,
        Date date,
        Student_Id int NOT NULL,
        Event_Id int NOT NULL,
        PRIMARY KEY(Ticket_Id),
        FOREIGN KEY(Student_Id) REFERENCES Student(Student_Id),
        FOREIGN KEY (Event_Id) REFERENCES Events(Event_Id));

CREATE TABLE Clubs(
       Club_Id int NOT NULL,
       Club_Name varchar(50) NOT NULL,
       Top_Board varchar(50) NOT NULL,
        PRIMARY KEY (Club_Id));

CREATE TABLE Sports(
        Sport_Id int NOT NULL PRIMARY KEY,
        Sport_Name varchar(50)
);

CREATE TABLE  Student_Sports(
        Student_Id int NOT NULL,
        Sport_Id int NOT NULL,
        Join_Date date,
        Time time ,
         PRIMARY KEY (Student_Id, Sport_Id),
         FOREIGN KEY(Student_Id) REFERENCES
         Student(Student_Id),
         FOREIGN KEY(Sport_Id) REFERENCES Sports(Sport_Id)
            );


CREATE TABLE Student_Club(
        Student_Id int NOT NULL,
         Club_Id int NOT NULL,
         Join_Date date,
          Time time ,
          PRIMARY KEY (Student_Id, Club_Id)
);

CREATE TABLE NSBM_Green_University(
NSBM_Id INT NOT NULL PRIMARY KEY,
Uni_Name VARCHAR (50) NOT NULL,
Location VARCHAR (50) NOT NULL,
Facilities VARCHAR (100) NOT NULL,
Program_Offered VARCHAR (100) NOT NULL,
University_Rank VARCHAR (1000) NOT NULL
);
CREATE TABLE Foreign_University (
Foreign_Uni_Id INT NOT NULL PRIMARY KEY,
Uni_Name VARCHAR (50) NOT NULL,
Location VARCHAR (50) NOT NULL,
Facilities VARCHAR (100) NOT NULL,
Program_Offered VARCHAR (100) NOT NULL,
University_Rank VARCHAR (1000) NOT NULL,
NSBM_Id INT NOT NULL,
FOREIGN KEY(NSBM_Id) REFERENCES NSBM_Green_University(NSBM_Id)
);

CREATE TABLE NSBM_Provide_Details(
Student_Id INT NOT NULL,
Access_Detail_Id INT NOT NULL,
NSBM_Uni_Id INT NOT NULL,
PRIMARY KEY (Student_Id,Access_Detail_Id,NSBM_Uni_Id),
FOREIGN KEY(Student_Id) REFERENCES Student (Student_Id),
FOREIGN KEY (Access_Detail_Id) REFERENCES
Access_Details(Access_Details_ID),
FOREIGN KEY (NSBM_Uni_Id) REFERENCES
NSBM_Green_University(NSBM_Id)
);


CREATE TABLE Foreign_Uni_Details (
Student_ID INT NOT NULL,
Access_Detail_ID INT NOT NULL,
Foreign_Uni_ID INT NOT NULL,
PRIMARY KEY (Student_Id,Access_Detail_Id,Foreign_Uni_Id),
FOREIGN KEY(Student_Id) REFERENCES Student (Student_Id),
FOREIGN KEY (Access_Detail_Id) REFERENCES
Access_Details( Access_Details_ID),
FOREIGN KEY (Foreign_Uni_Id) REFERENCES
Foreign_University(Foreign_Uni_Id)
);

CREATE TABLE NSBM_Offer (
Club_ID INT NOT NULL,
Event_ID INT NOT NULL,
Sport_ID INT NOT NULL,
Hostel_ID INT NOT NULL,
NSBM_ID INT NOT NULL,
PRIMARY KEY (Club_Id,Event_Id,Sport_Id,Hostel_Id,NSBM_Id),
FOREIGN KEY(Club_Id) REFERENCES Clubs(Club_Id),
FOREIGN KEY(Event_Id) REFERENCES Events(Event_Id),
FOREIGN KEY(Sport_Id) REFERENCES Sports(Sport_Id),
FOREIGN KEY(Hostel_Id) REFERENCES Hostel(Hostel_Id),
FOREIGN KEY(NSBM_Id) REFERENCES NSBM_Green_University(NSBM_Id)
);
```

Database Diagram



Data diagram link-

https://dbdiagram.io/d/Copy-of-Untitled-Diagram-65a64223ac844320ae022674

Sample Records

```
INSERT INTO Person (Person_ID, NIC_NO, PassportNo, Sur_Name,
First_Name, Last_Name, Title, Gender, EmailAddress, Nationality,
Mobile_Num, Fixed_Phone_Num, Date_of_Birth)
VALUES
    (1, '11223344556', 'XYZ123', 'Adams', 'Jack', 'Johnson',
'Mr.', 'M', 'jack.johnson@email.com', 'US', 745678901, 76543210,
'1985-02-20'),
    (2, '22334455667', 'ABC987', 'Turner', 'Sophie', 'Taylor',
'Miss', 'F', 'sophie.taylor@email.com', 'UK', 756789012,
65432109, '1990-07-15'),
    (3, '34567890123', 'GHI789', 'Williams', 'Bob', 'Miller',
'Dr.', 'M', 'bob.miller@email.com', 'CA', 745678901, 76543210,
'1978-11-10'),
    (4, '45678901234', 'JKL012', 'Brown', 'Emily', 'Taylor',
'Miss', 'F', 'emily.taylor@email.com', 'AU', 756789012, 65432109,
'1995-03-25'),
    (5, '56789012345', 'MNO345', 'Clark', 'David', 'Anderson',
'Mr.', 'M', 'david.anderson@email.com', 'NZ', 767890123,
54321098, '1980-09-12'),
    (6, '67890123456', 'PQR678', 'Miller', 'Sophia', 'Smith',
'Mrs.', 'F', 'sophia.smith@email.com', 'IN', 768901234, 43210986,
'1992-07-08'),
    (7, '78901234567', 'STU901', 'Jones', 'Michael', 'Johnson',
'Mr.', 'M', 'michael.johnson@email.com', 'FR', 789023456,
72198765, '1987-04-15'),
    (8, '89012345678', 'VWX123', 'Anderson', 'Olivia', 'Brown',
'Ms.', 'F', 'olivia.brown@email.com', 'DE', 701234567, 71098754,
'1975-12-30'),
    (9, '90123456789', 'YZA456', 'Taylor', 'Daniel', 'White',
'Mr.', 'M', 'daniel.white@email.com', 'ES', 701234568, 70987643,
'1998-02-18'),
    (10, '01234567890', 'BCD789', 'Smith', 'Ava', 'Clark',
'Miss', 'F', 'ava.clark@email.com', 'IT', 723567890, 76543213,
'1983-06-28');
```

```
INSERT INTO Person_Address (Person_ID, City, District, Province)
VALUES
    (1, 'New York City', 'Manhattan', 'New York'),
    (2, 'London', 'Westminster', 'England'),
    (3, 'Toronto', 'Downtown', 'Ontario'),
    (4, 'Sydney', 'CBD', 'New South Wales'),
    (5, 'Auckland', 'Central', 'North Island'),
    (6, 'Mumbai', 'South Mumbai', 'Maharashtra'),
    (7, 'Paris', 'City Center', 'Ile-de-France'),
    (8, 'Berlin', 'Mitte', 'Berlin'),
    (9, 'Madrid', 'Sol', 'Community of Madrid'),
    (10, 'Rome', 'Historic Center', 'Lazio');


-- Alter the table definition to allow for longer certificate
names
ALTER TABLE O_Level_Qualification
ALTER COLUMN O_Level_Certificate VARCHAR(50);

INSERT INTO O_Level_Qualification (O_Level_Index_Number,
O_Level_School, O_Level_Status, O_Level_Year, O_Level_English,
O_Level_Mathematics, O_Level_Certificate)
VALUES
    (1, 'High School A', 'Passed', '2005-06-01', 'A', 'B',
'Distinction'),
    (2, 'High School B', 'Passed', '2006-06-01', 'B', 'A',
'Distinction'),
    (3, 'High School C', 'Failed', '2007-06-01', 'C', 'D',
'Participation'),
    (4, 'High School D', 'Passed', '2008-06-01', 'A', 'A',
'Distinction'),
    (5, 'High School E', 'Passed', '2009-06-01', 'B', 'B',
'Distinction'),
    (6, 'High School F', 'Passed', '2010-06-01', 'A', 'C',
'Distinction'),
    (7, 'High School G', 'Failed', '2011-06-01', 'D', 'F',
'Participation'),
    (8, 'High School H', 'Passed', '2012-06-01', 'A', 'B',
'Distinction'),
    (9, 'High School I', 'Passed', '2013-06-01', 'B', 'A',
'Distinction'),
    (10, 'High School J', 'Passed', '2014-06-01', 'A', 'A',
'Distinction');


-- Delete the A_Level_Scheme column from A_Level_Qualification
table
ALTER TABLE A_Level_Qualification
DROP COLUMN A_Level_Scheme;

-- Alter the table definition to allow for longer certificate
names
ALTER TABLE A_Level_Qualification
ALTER COLUMN A_Level_Certificate VARCHAR(50);

INSERT INTO A_Level_Qualification (A_Level_Index, A_Level_Year,
A_Level_School, A_Level_Stream, A_Level_Score,
A_Level_Certificate, A_Level_Status, A_Level_General_English)
VALUES
    (1, '2010-06-01', 'College A', 'Science', 85, 'Distinction',
'Passed', 90),
    (2, '2011-06-01', 'College B', 'Arts', 78, 'Distinction',
'Passed', 88),
    (3, '2012-06-01', 'College C', 'Commerce', 92, 'Distinction',
'Passed', 85),
    (4, '2013-06-01', 'College D', 'Science', 88, 'Distinction',
'Passed', 87),
    (5, '2014-06-01', 'College E', 'Arts', 79, 'Distinction',
'Passed', 89),
    (6, '2015-06-01', 'College F', 'Commerce', 90, 'Distinction',
'Passed', 84),
    (7, '2016-06-01', 'College G', 'Science', 86, 'Distinction',
'Passed', 88),
    (8, '2017-06-01', 'College H', 'Arts', 82, 'Distinction',
'Passed', 91),
    (9, '2018-06-01', 'College I', 'Commerce', 91, 'Distinction',
'Passed', 86),
    (10, '2019-06-01', 'College J', 'Science', 89, 'Distinction',
'Passed', 89);
```

```
O_Level_Index_Number, A_Level_Index, Other)
VALUES
    (1, 1, 1, 'Diploma in Computer Science'),
    (2, 2, 2, 'Bachelor of Arts in Literature'),
    (3, 3, NULL, 'Did not pursue A-levels'),
    (4, 4, 3, 'Bachelor of Commerce in Accounting'),
    (5, 5, 4, 'Bachelor of Science in Biology'),
    (6, 6, 5, 'Bachelor of Arts in Sociology'),
    (7, 7, NULL, 'Did not pursue A-levels'),
    (8, 8, 6, 'Master of Business Administration'),
    (9, 9, 7, 'Master of Science in Physics'),
    (10, 10, 8, 'Doctor of Philosophy in Literature');

INSERT INTO Student_Name (Name_With_Initials, Sur_Name)
VALUES
    ('John A.', 'Doe'),
    ('Sophie T.', 'Turner'),
    ('Bob M.', 'Miller'),
    ('Emily T.', 'Brown'),
    ('David A.', 'Clark'),
    ('Sophia S.', 'Smith'),
    ('Michael J.', 'Johnson'),
    ('Olivia B.', 'Brown'),
    ('Daniel W.', 'White'),
    ('Ava C.', 'Clark');

INSERT INTO Hostel (Hostel_ID, Room, Capacity)
VALUES
    (1, '101', 'Double'),
    (2, '203', 'Single'),
    (3, '305', 'Double'),
    (4, '102', 'Single'),
    (5, '204', 'Double'),
    (6, '306', 'Single'),
    (7, '103', 'Double'),
    (8, '205', 'Single'),
    (9, '307', 'Double'),
    (10, '104', 'Single');


INSERT INTO Student (Student_Id, Name_With_Initials,
Student_Email, Registration_Status, Tutors, Hostel_Id, Person_Id)
VALUES
    (1, 'John A.', 'john.doe@student.com', 'Registered',
'Tutor1', 1, 1),
    (2, 'Sophie T.', 'sophie.turner@student.com', 'Registered',
'Tutor2', 2, 2),
    (3, 'Bob M.', 'bob.miller@student.com', 'Registered',
'Tutor3', 3, 3),
    (4, 'Emily T.', 'emily.taylor@student.com', 'Registered',
'Tutor4', 4, 4),
    (5, 'David A.', 'david.anderson@student.com', 'Registered',
'Tutor5', 5, 5),
    (6, 'Sophia S.', 'sophia.smith@student.com', 'Registered',
'Tutor6', 6, 6),
    (7, 'Michael J.', 'michael.johnson@student.com',
'Registered', 'Tutor7', 7, 7),
    (8, 'Olivia B.', 'olivia.brown@student.com', 'Registered',
'Tutor8', 8, 8),
    (9, 'Daniel W.', 'daniel.white@student.com', 'Registered',
'Tutor9', 9, 9),
    (10, 'Ava C.', 'ava.clark@student.com', 'Registered',
'Tutor10', 10, 10);


INSERT INTO Course (Course_ID, Course_Name, Course_Type,
Course_Fee, Schedule, Syllabus)
VALUES
    (1, 'Computer Science', 'Bachelor', '5000', 'Mon/Wed/Fri
10:00 AM', 'CS101'),
    (2, 'Business Administration', 'Bachelor', '6000', 'Tue/Thu
2:00 PM', 'BA201'),
    (3, 'Psychology', 'Bachelor', '5500', 'Mon/Fri 1:00 PM',
'PSY301'),
    (4, 'Mechanical Engineering', 'Bachelor', '7000', 'Wed/Fri
3:00 PM', 'ME401'),
    (5, 'English Literature', 'Bachelor', '4800', 'Tue/Thu 11:00
AM', 'EL501');

DELETE FROM Course

-- Update Course table to include Course_Level column
ALTER TABLE Course
ADD Course_Level VARCHAR(20) NOT NULL;

-- Update Course insertion with Course_Level values
INSERT INTO Course (Course_ID, Course_Name, Course_Type,
Course_Fee, Schedule, Syllabus, Course_Level)
VALUES
    (1, 'Computer Science Foundation', 'Foundation', '5000',
'Monday-Friday', 'CS101', 'Foundation'),
    (2, 'Business Administration OHD', 'OHD', '6000', 'Monday-
Wednesday', 'BA201', 'OHD'),
    (3, 'English Literature Undergraduate', 'Undergraduate',
'4500', 'Tuesday-Thursday', 'EL301', 'Undergraduate'),
    (4, 'Mathematics Certificate', 'Certificate', '5500',
'Wednesday-Friday', 'MA401', 'Certificate'),
    (5, 'Psychology OHD', 'OHD', '4800', 'Thursday-Saturday',
'PSY501', 'OHD');
```

```
INSERT INTO Library (Library_ID, Library_Fee, Location,
Establishment_Date)
VALUES
    (1, '100', 'Main Library, 1st Floor', '2000-01-15'),
    (2, '150', 'Science Library, Ground Floor', '2002-05-20'),
    (3, '120', 'Arts Library, 2nd Floor', '2001-09-10'),
    (4, '130', 'Business Library, 3rd Floor', '2004-03-25'),
    (5, '110', 'Engineering Library, 4th Floor', '2003-07-18');

INSERT INTO Faculty (Faculty_ID, Faculty_Name, Faculty_Head,
Location, Contact_Number, Description)
VALUES
    ('FAC001', 'Engineering Faculty', 'Prof. Smith', 'Engineering
Building', '1234567890', 'Leading in engineering education'),
    ('FAC002', 'Business Faculty', 'Dr. Johnson', 'Business
Center', '9876543210', 'Innovative business education'),
    ('FAC003', 'Arts Faculty', 'Dr. Miller', 'Arts Building',
'8765432109', 'Nurturing creativity in the arts'),
    ('FAC004', 'Science Faculty', 'Prof. Taylor', 'Science
Complex', '7654321098', 'Advancing scientific knowledge'),
    ('FAC005', 'Health Sciences Faculty', 'Dr. Anderson', 'Health
Sciences Center', '6543210987', 'Promoting health and well-
being');


INSERT INTO Department (Dep_ID, Dep_Name, Dep_Head, Location,
Contact_Number, Description, Faculty_ID)
VALUES
    (1, 'Computer Science Department', 'Dr. White', 'CS
Building', '1122334455', 'Cutting-edge computer science
education', 'FAC001'),
    (2, 'Business Administration Department', 'Prof. Turner',
'Business Block', '2233445566', 'Strategic business management',
'FAC002'),
    (3, 'English Department', 'Dr. Brown', 'Arts Block',
'3344556677', 'Literary excellence and language studies',
'FAC003'),
    (4, 'Mathematics Department', 'Prof. Clark', 'Math Building',
'4455667788', 'Advancing mathematical knowledge', 'FAC004'),
    (5, 'Psychology Department', 'Dr. Smith', 'Psychology
Center', '5566778899', 'Understanding human behavior', 'FAC005');

INSERT INTO Course_Offeres (Faculty_ID, Course_ID, Dep_ID)
VALUES
    ('FAC001', 1, 1),
    ('FAC002', 2, 2),
    ('FAC003', 3, 3),
    ('FAC004', 4, 4),
    ('FAC005', 5, 5);


INSERT INTO Degree_Programme (Programme_ID, Programme_Name,
Programme_Type, Degree_Level, Semester, Total_Credits,
Research_Area)
VALUES
    (1, 'Computer Science', 'Bachelor', 'Undergraduate', '8',
'120', 'Artificial Intelligence'),
    (2, 'Business Administration', 'Bachelor', 'Undergraduate',
'6', '90', 'Strategic Management'),
    (3, 'English Literature', 'Bachelor', 'Undergraduate', '6',
'100', 'Literary Criticism'),
    (4, 'Mathematics', 'Bachelor', 'Undergraduate', '8', '130',
'Pure Mathematics'),
    (5, 'Psychology', 'Bachelor', 'Undergraduate', '8', '110',
'Clinical Psychology');

INSERT INTO Offered_Degree_Program (Course_ID, Programme_ID)
VALUES
    (1, 1),
    (2, 2),
    (3, 3),
    (4, 4),
    (5, 5);

INSERT INTO Clubs (Club_Id, Club_Name, Top_Board)
VALUES
    (1, 'Chess Club', 'John Doe'),
    (2, 'Photography Club', 'Emily Smith'),
    (3, 'Debating Club', 'Michael Johnson'),
    (4, 'Dance Club', 'Olivia Brown'),
    (5, 'Environmental Club', 'Daniel White');
```

```
INSERT INTO Sports (Sport_Id, Sport_Name)
VALUES
    (1, 'Football'),
    (2, 'Basketball'),
    (3, 'Swimming'),
    (4, 'Table Tennis'),
    (5, 'Badminton');

INSERT INTO Student_Sports (Student_Id, Sport_Id, Join_Date,
Time)
VALUES
    (1, 1, '2022-02-01', '17:00'),
    (2, 3, '2022-03-15', '15:30'),
    (3, 2, '2022-01-10', '18:45'),
    (4, 5, '2022-04-20', '16:20'),
    (5, 4, '2022-02-28', '14:10');

INSERT INTO Student_Club (Student_Id, Club_Id, Join_Date, Time)
VALUES
    (1, 1, '2022-03-05', '19:30'),
    (2, 3, '2022-02-18', '20:15'),
    (3, 2, '2022-01-25', '17:45'),
    (4, 5, '2022-04-10', '18:00'),
    (5, 4, '2022-03-01', '16:40');


INSERT INTO NSBM_Green_University (NSBM_Id, Uni_Name, Location,
Facilities, Program_Offered, University_Rank)
VALUES
    (1, 'NSBM Green University', 'Colombo, Sri Lanka', 'State-of-
the-art campus', 'Engineering, Business, Arts', 'Top 10 in Sri
Lanka'),
    (2, 'Foreign University 1', 'Plymouth, UK', 'Modern
facilities', 'Engineering, Business, Computer Science', 'Top 50
in the UK'),
    (3, 'Foreign University 2', 'Victoria, Australia', 'Cutting-
edge infrastructure', 'Business, Science, Health Sciences', 'Top
30 in Australia');

INSERT INTO Foreign_University (Foreign_Uni_Id, Uni_Name,
Location, Facilities, Program_Offered, University_Rank, NSBM_Id)
VALUES
    (1, 'University of Plymouth', 'Plymouth, UK', 'State-of-the-
art facilities', 'Engineering, Business, Computer Science', 'Top
50 in the UK', 1),
    (2, 'Victoria University', 'Victoria, Australia', 'Modern
campus', 'Business, Science, Health Sciences', 'Top 30 in
Australia', 1);
```

Create Trigger

Password Generate Trigger

```
CREATE TRIGGER after_student_insert
ON Person
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;

    -- Temporary table to store inserted Person_ID values
    CREATE TABLE #InsertedIDs (Person_ID INT);

    -- Insert new Person_ID values into the temporary table
    INSERT INTO #InsertedIDs (Person_ID)
    SELECT Person_ID FROM INSERTED;

    DECLARE @default_password VARCHAR(10);

    -- Fetch data from the inserted Person
    SELECT TOP 1 @default_password = Default_Password
    FROM Student_Default_Passwords sdp
    WHERE EXISTS (SELECT 1 FROM #InsertedIDs i WHERE i.Person_ID = sdp.Person_ID);

    IF @default_password IS NULL
    BEGIN
        DECLARE @generated_password VARCHAR(255);
        EXEC GenerateRandomPassword 10, @generated_password OUTPUT;
        SET @default_password = @generated_password;
    END;
```

```
    DECLARE @default_account_level VARCHAR(50) = 'Student';
    DECLARE @default_permissions VARCHAR(50) = 'Basic';
    DECLARE @default_account_status VARCHAR(50) = 'Active';

    -- Insert corresponding record into Access_Details
    INSERT INTO Access_Details (Access_Details_ID, User_Name, [Password], Account_Level, Permissions, Account_Status)
    SELECT
        i.Person_ID,
        CONCAT(p.First_Name, p.Last_Name),
        @default_password,
        @default_account_level,
        @default_permissions,
        @default_account_status
    FROM #InsertedIDs i
    JOIN Person p ON i.Person_ID = p.Person_ID;

    -- Drop the temporary table
    DROP TABLE #InsertedIDs;
END;
```

Attendance Update Trigger

```
CREATE TRIGGER trg_UpdateAttendance
ON Module_Attendance
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @StudentId INT;
    DECLARE @ModuleCode VARCHAR(10);

    SELECT @StudentId = Student_Id, @ModuleCode = Module_Code FROM INSERTED;

    UPDATE Module_Attendance
    SET Attendance = ISNULL(Attendance, 0) + 1
    WHERE Student_Id = @StudentId AND Module_Code = @ModuleCode;
END;
```

Check all triggers in Database

```
sp_helptext trg_UpdateAttendance

-- List all triggers in the database
SELECT
    t.name AS TriggerName,
    OBJECT_NAME(parent_id) AS TableName,
    OBJECT_SCHEMA_NAME(parent_id) AS TableSchema,
    OBJECT_DEFINITION(t.object_id) AS TriggerDefinition
FROM
    sys.triggers t;
```

User-Define Functions

Age calculation Function

```sql
CREATE FUNCTION fn_CalculationAge(@DOB DATE)
RETURNS INT
AS
BEGIN
    DECLARE @Age INT
    SET @Age = YEAR(GETDATE()) - YEAR(@DOB) -
        CASE
            WHEN MONTH(GETDATE()) < MONTH(@DOB) OR
                (MONTH(GETDATE()) = MONTH(@DOB) AND DAY(GETDATE()) < DAY(@DOB))
                THEN 1
            ELSE 0
        END

    RETURN @Age
END;

-- SELECT UniversityDB.dbo.fn_CalculationAge('2004-03-20') AS Age;

SELECT
    Student.Student_Id,
    Student.Name_With_Initials,
    UniversityDB.dbo.fn_CalculationAge(Person.Date_of_Birth) AS Age
FROM
    Person
INNER JOIN
    Student ON Student.Person_Id = Person.Person_ID;
```

Student by Gender Function

```sql
CREATE FUNCTION fn_StudentByGender(@Gender NVARCHAR(1))
RETURNS TABLE
AS
RETURN (SELECT Student_Id, Name_With_Initials, Gender, Hostel_Id
        FROM
        Person
        INNER JOIN
        Student ON Student.Person_Id = Person.Person_ID
        WHERE Gender = @Gender)

SELECT * FROM fn_StudentByGender('F')
```

Get Name by Id Function

```sql
CREATE FUNCTION fn_GetNameByID(@id INT)
RETURNS NVARCHAR(30)
AS
BEGIN
      RETURN(SELECT Name_With_Initials FROM
              dbo.Student WHERE
              Student_Id = @id)
END

SELECT dbo.fn_GetNameById(1)

--Add Shemabinding Option
ALTER FUNCTION fn_GetNameByID(@id INT)
RETURNS NVARCHAR(30)
WITH SCHEMABINDING
AS
BEGIN
      RETURN(SELECT Name_With_Initials FROM
              dbo.Student WHERE
              Student_Id = @id)
END
drop table Student
```

Create Views

Student Enrolled Status

```
CREATE VIEW vwStudentEnrolledStatus
AS
SELECT
    S.Student_Id,
    S.Name_With_Initials,
    SE.Course_Id,
    SE.Office_Id
FROM
    Student S
JOIN
    Student_Enroll SE ON S.Student_Id = SE.Student_id;

SELECT * FROM vwStudentEnrolledStatus

drop VIEW vwStudentByFacultyId
```

Graduation View

```
CREATE VIEW GraduationDetails AS
SELECT
    G.Graduation_Id,
    G.GPA,
    G.Graduation_Date,
    G.Batch_Number,
    G.Year,
    DG.Programme_Id,
    DG.Student_Id,
    S.Name_With_Initials,
    S.Student_Email
FROM
    Graduation G
JOIN
    Degree_Graduation DG ON G.Graduation_Id = DG.Graduation_Id
JOIN
    Student S ON DG.Student_Id = S.Student_Id;


SELECT * FROM GraduationDetails
```

Attendance View

```
CREATE VIEW vw_AttendanceCalculation AS
SELECT
    M.Student_Id,
    S.Name_With_Initials AS Student_Name,
    M.Module_Code,
    M.Attendance,
    CASE
        WHEN 15 > 0 THEN (M.Attendance * 100.0 / 15)
        ELSE 0
    END AS Attendance_Percentage
FROM
    (SELECT
        MA.Student_Id,
        MA.Module_Code,
        COUNT(MA.Module_Code) AS Attendance
     FROM
        Module_Attendance MA
     GROUP BY
        MA.Student_Id, MA.Module_Code) M
JOIN
    Student S ON M.Student_Id = S.Student_Id;

SELECT * FROM vw_AttendanceCalculation;
```

Create Procedure

Get Student Count Procedure

```
CREATE PROC spGetStudentCount
@Student_Count INT OUTPUT
AS
BEGIN
     SELECT @Student_Count = COUNT(Student_Id) FROM Student
END

DECLARE @TotalCount INT
EXECUTE spGetStudentCount @TotalCount OUTPUT

PRINT @TotalCount
```

Attendance calculation procedure

```
CREATE PROCEDURE sp_AttendanceCalculation
    @TotalLectures INT
AS
BEGIN
    SELECT
        M.Student_Id,
        S.Name_With_Initials AS Student_Name,
        M.Module_Code,
        M.Attendance,
        CASE
            WHEN @TotalLectures > 0 THEN (M.Attendance * 100.0 / @TotalLectures)
            ELSE 0
        END AS Attendance_Percentage
    FROM
        (SELECT
            MA.Student_Id,
            MA.Module_Code,
            COUNT(MA.Module_Code) AS Attendance
         FROM
            Module_Attendance MA
         GROUP BY
            MA.Student_Id, MA.Module_Code) M
    JOIN
        Student S ON M.Student_Id = S.Student_Id;
END;

EXEC sp_AttendanceCalculation @TotalLectures = 15;
```

Generate random password procedure

```
CREATE PROCEDURE GenerateRandomPassword
    @PasswordLength INT,
    @GeneratedPassword VARCHAR(255) OUTPUT
AS
BEGIN
    DECLARE @Characters VARCHAR(94) = 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789!@#$%^&*()-_=+[]{}|;:,.<>?/';
    DECLARE @Password VARCHAR(255) = '';
    DECLARE @Index INT = 0;

    WHILE @Index < @PasswordLength
    BEGIN
        SET @Password = @Password + SUBSTRING(@Characters, CAST(FLOOR(1 + RAND() * 94) AS INT), 1);
        SET @Index = @Index + 1;
    END

    SET @GeneratedPassword = @Password;
END;

DECLARE @PasswordResult VARCHAR(255);
EXEC GenerateRandomPassword @PasswordLength = 10, @GeneratedPassword = @PasswordResult OUTPUT;
PRINT @PasswordResult;
```

Critical Appraisal

1. Database Design

Worthy Features-

- In Normalization the database adheres to normalization principles, ensuring efficient storage and minimizing data redundancy.
- For the Entity-Relationship Model, the ER diagram is used to represent relationships between different entities within the university system.

Shortcomings-

- Denormalization risks.
- Database design complexity.

Resolution-

- Optimize denormalization.
- Simplify database design.

2. Data Integrity

Worthy Features-

- Use of foreign keys to maintain referential integrity between tables.
- Implementation of constraints to ensure data consistency.

Shortcomings-

- Missing constraints where constraints are lacking which might lead to potential data integrity issues.
- Cascading deletion might pose risks in certain scenarios.

Resolution-

- Implement missing constraints by adding necessary constraints to maintain data integrity.
- Review cascading rules to ensure their rules align with university policies.

3. Performance

Worthy Features-

- Use of indexes to enhance query performance.
- Queries are optimized for efficiency.

Shortcomings-

- Indexing might be missing, potentially impacting query performance.
- Instances of inefficient queries.

Resolution-

- Adding indexes to improve query speed.
- Rewriting or optimizing queries for better performance.

4. **Security**

Worthy Features-

- Access Controls: Evaluate the implementation of access controls to restrict unauthorized access.
- Encryption: Highlight the use of encryption for sensitive data.

Shortcomings-

- Weak Access Controls: Address any weaknesses in access controls that could compromise data security.
- Inadequate Encryption: Evaluate if sensitive data is adequately encrypted.

Resolution-

- Enhance Access Controls: Recommend strengthening access controls.
- Implement Stronger Encryption: If necessary, propose stronger encryption methods.

5. **Scalability**

Worthy Features-

- Scalability measures with the university's growth.

Shortcomings-

- Scalability challenges.

Resolution-

- Scalability planning - enhance scalability by partitioning large tables or optimizing queries.

6. **User Interface**

Worthy Features-

- User-friendly interface which makes it easy to use.

Shortcomings-

- Areas were met where the user interface might have been too complex for end-users.

Resolution-

- Interface simplification.

Future comments

- Can be integration with external systems-
  Opportunities for integration with other university systems or external platforms.
  Implementing APIs for seamless data exchange with other tools used in the university.

- Advanced the reporting and analytics –

Using advanced reporting features for more in-depth analysis by integrating business intelligence tools or developing more sophisticated management reports.

- Adapting of New Technologies-
  Exploring to adapt emerging technologies that could benefit the university management system.

- Developing a Feedback Mechanism-
  Implementing a mechanism for collecting feedback from users to use the feedback to identify areas of improvement and prioritize future development efforts.

Data Definition Language (DDL)

Create, Drop, Alter were used with in the sample records.

Data Manipulation Language (DML)

Insert, Update, Delete were used within the sample records.

Database .bak file link

https://github.com/sasankanimesh/UniversityDB.git

Group Contribution

	Plymouth ID	Name (As appeared on DLE)	Contributed section
1	10899343	Priyantha Ranasinghe	Section 1,2,3
2	10899233	Dodampe Nimna	Section 1,2
3	10899228	Rankira Kosgollage	Section 1,2,4
4	10903090	Ranasinghe Ranasinghe	Section 1,2
5	10898748	Gardihewage Dayarathne	Section 1,2
6	10900326	Withanage Mel	Section 1,2
7	10817967	CDM Fernando	Section 2
8	10822745	Abeysinghe Herath	Section 2