PROJECT-2

1. *Design:*

In Project-2, I have used the Database solution of Project-1 created by Professor.

Below are the reasons,

1. I was not satisfied with my design in Project 1, and there were too many changes I needed to make after looking at the instructor solution.
2. Since I had created more tables in project-1, the complexity went beyond the scope of the business problem.
3. Since this project carries more than one-third of the total Grade, it is considered as crucial to me and I didn’t want the mistakes of Project-1 to reflect in this project.

***Creation of Schema:***

CREATE SCHEMA pr2;

GO

*2. Table Creation:*

CREATE TABLE pr2.BenefitSelection (

BenefitSelectionID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1),

BenefitSelection VARCHAR(10) NOT NULL

);

CREATE TABLE pr2.Grades (

GradeID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, Grade VARCHAR(3) NOT NULL

);

CREATE TABLE pr2.DayOfWeek (

DayOfWeekID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1),

Text VARCHAR(10) NOT NULL

);

CREATE TABLE pr2.StudentGradingStatus (

StudentStatusID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, StudentStatus VARCHAR(10) NOT NULL

);

CREATE TABLE pr2.SemesterText (

SemesterTextID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, SemesterText VARCHAR(20) NOT NULL

);

CREATE TABLE pr2.SemesterInfo (

SemesterID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, Semester INTEGER NOT NULL REFERENCES pr2.SemesterText(SemesterTextID)

, Year CHAR(4) NOT NULL

, FirstDay DATE NOT NULL

, LastDay DATE NOT NULL

);

CREATE TABLE pr2.College (

CollegeID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, CollegeName VARCHAR(50) NOT NULL

);

CREATE TABLE pr2.Buildings (

BuildingID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, BuildingName VARCHAR(50) NOT NULL

);

CREATE TABLE pr2.ProjectorInfo (

ProjectorID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, ProjectorText VARCHAR(50) NOT NULL

);

CREATE TABLE pr2.Addresses (

AddressID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, Street1 VARCHAR(200) NOT NULL

, Street2 VARCHAR(200)

, City VARCHAR(100) NOT NULL

, State CHAR(2) NOT NULL

, ZIP CHAR(5) NOT NULL

);

CREATE TABLE pr2.StudentStatus (

StudentStatusID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, StudentStatus VARCHAR(20) NOT NULL

);

CREATE TABLE [pr2].[AreaOfStudy] (

[AreaOfStudyID] INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, [StudyTitle] VARCHAR (20) NOT NULL

, [CollegeID] INTEGER NOT NULL REFERENCES pr2.College(CollegeID)

);

CREATE TABLE pr2.People (

PersonID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, NTID VARCHAR(10) NOT NULL

, FirstName VARCHAR(50) NOT NULL

, LastName VARCHAR(50) NOT NULL

, Password VARCHAR(200)

, DOB DATE NOT NULL

, SSN CHAR(9)

, HomeAddress INTEGER NOT NULL REFERENCES pr2.Addresses(AddressID)

, LocalAddress INTEGER REFERENCES pr2.Addresses(AddressID)

, IsActive BIT NOT NULL

);

CREATE TABLE pr2.StudentInfo (

StudentID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, PersonID INTEGER NOT NULL REFERENCES pr2.People(PersonID),

StudentStatusID INTEGER REFERENCES pr2.StudentStatus(StudentStatusID)

);

CREATE TABLE pr2.StudentAreaOfStudy (

AreaOfStudyID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, StudentID INTEGER NOT NULL REFERENCES pr2.StudentInfo(StudentID)

, AreaID INTEGER NOT NULL REFERENCES pr2.AreaOfStudy(AreaOfStudyID)

, IsMajor BIT NOT NULL

);

CREATE TABLE pr2.Classroom (

ClassroomID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, Building INTEGER NOT NULL REFERENCES pr2.Buildings(BuildingID)

, RoomNumber VARCHAR(5) NOT NULL

, MaximumSeating INTEGER NOT NULL

, ProjectorID INTEGER NOT NULL REFERENCES pr2.ProjectorInfo(ProjectorID)

, WhiteBoardCount INTEGER NOT NULL DEFAULT 0

, OtherAV VARCHAR(200)

, CONSTRAINT Seating CHECK (MaximumSeating>=0)

, CONSTRAINT BoardCount CHECK (WhiteBoardCount>=0)

);

CREATE TABLE pr2.CourseCatalogue (

CourseCode VARCHAR(20) NOT NULL

, CourseNumber INTEGER NOT NULL

, CourseTitle VARCHAR(100) NOT NULL

, CourseDescription VARCHAR(1000),

PRIMARY KEY (CourseCode,CourseNumber)

);

CREATE TABLE pr2.CourseSchedule (

CourseScheduleID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, CourseCode VARCHAR(20) NOT NULL

, SemesterID INTEGER NOT NULL REFERENCES pr2.SemesterInfo(SemesterID)

, NumberOfSeats INTEGER NOT NULL

, CourseNumber INTEGER NOT NULL

, Location INTEGER REFERENCES pr2.Classroom(ClassroomID)

, CONSTRAINT Seatchk CHECK (NumberOfSeats>=0.00),

FOREIGN KEY (CourseCode,CourseNumber) REFERENCES pr2.CourseCatalogue(CourseCode, CourseNumber)

);

CREATE TABLE pr2.Prerequisites (

ParentCode VARCHAR(20) NOT NULL

, ParentNumber INTEGER NOT NULL

, ChildCode VARCHAR(20) NOT NULL

, ChildNumber INTEGER NOT NULL

, PRIMARY KEY (ParentCode,ParentNumber,ChildCode,ChildNumber),

FOREIGN KEY (ParentCode, ParentNumber) REFERENCES pr2.CourseCatalogue(CourseCode, CourseNumber),

FOREIGN KEY (ChildCode,ChildNumber) REFERENCES pr2.CourseCatalogue(CourseCode, CourseNumber)

);

CREATE TABLE pr2.JobInformation (

JobID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, JobDescription VARCHAR(200) NOT NULL

, JobRequirements VARCHAR(1000)

, MinimumPay DECIMAL(8,2) NOT NULL DEFAULT 0.00

, MaximumPay DECIMAL(8,2) NOT NULL

, UnionJob BIT

, CONSTRAINT Paychk1 CHECK (MinimumPay>=0.00)

, CONSTRAINT Paychk2 CHECK (MaximumPay>=0.00)

, CONSTRAINT Paychk5 CHECK (MinimumPay<MaximumPay)

);

CREATE TABLE pr2.Benefits (

BenefitID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, BenefitCost DECIMAL(8,2) NOT NULL

, BenefitSelection INTEGER NOT NULL REFERENCES pr2.BenefitSelection(BenefitSelectionID)

, BenifitDescription VARCHAR(100)

, CONSTRAINT Costchk1 CHECK (BenefitCost>=0.00)

);

CREATE TABLE pr2.EmployeeInfo

(

EmployeeID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, PersonID INTEGER NOT NULL REFERENCES pr2.People(PersonID)

, IsActive BIT NOT NULL

, YearlyPay DECIMAL(8,2) NOT NULL

, HealthBenefits INTEGER NOT NULL REFERENCES pr2.Benefits(BenefitID)

, VisionBenefits INTEGER NOT NULL REFERENCES pr2.Benefits(BenefitID)

, DentalBenefits INTEGER NOT NULL REFERENCES pr2.Benefits(BenefitID)

, JobInformation integer NOT NULL REFERENCES pr2.JobInformation(JobID)

, CONSTRAINT Paychk3 CHECK (YearlyPay>=0.00)

);

CREATE TABLE pr2.TeachingAssignment (

EmployeeID INTEGER NOT NULL REFERENCES pr2.EmployeeInfo(EmployeeID)

, CourseScheduleID INTEGER NOT NULL REFERENCES pr2.CourseSchedule(CourseScheduleID)

, PRIMARY KEY (EmployeeID,CourseScheduleID)

);

CREATE TABLE pr2.CourseEnrollment (

EnrollmentID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

, CourseID INTEGER NOT NULL REFERENCES pr2.CourseSchedule(CourseScheduleID)

, StudentID INTEGER NOT NULL REFERENCES pr2.StudentInfo(StudentID)

, StatusID INTEGER NOT NULL REFERENCES pr2.StudentGradingStatus(StudentStatusID)

, GradeID INTEGER REFERENCES pr2.Grades(GradeID)

);

CREATE TABLE pr2.CourseDailySchedule (

CourseID INTEGER NOT NULL REFERENCES pr2.CourseSchedule(CourseScheduleID)

, DayOfWeek INTEGER NOT NULL REFERENCES pr2.DayOfWeek(DayOfWeekID)

, StartTime TIME NOT NULL

, EndTime TIME NOT NULL

, DailyID INTEGER NOT NULL PRIMARY KEY IDENTITY(1,1)

);

*3. Data Loading:*

*(Insertion of Data into tables created)*

INSERT INTO pr2.Grades(Grade)

VALUES

('A'),

('A-'),

('B+'),

('B'),

('B-'),

('C+'),

('C'),

('D'),

('E'),

('F');

--SELECT \* FROM pr2.Grades;

INSERT INTO pr2.Buildings(BuildingName)

VALUES

('SIGMA Building'),

('ALPHA Building'),

('BETA Building'),

('GAMMA Building'),

('DELTA Building'),

('THETA Building'),

('PI Building');

--SELECT \* FROM pr2.Buildings;

INSERT INTO pr2.StudentStatus(StudentStatus)

VALUES

('Undergraduate'),

('Federal Graduate'),

('NonFederal Graduate'),

('Graduated'),

('Non Matriculated');

--SELECT \* FROM pr2.StudentStatus;

INSERT INTO pr2.College(CollegeName)

VALUES

('Arizona College of Engineering'),

('Massachusetts Inst of Technology'),

('University of California-Berkeley'),

('Syracuse College of Engineering'),

('University of California-San Diego'),

('University of Southern California'),

('WASHINGTON School of Information Studies'),

('Northwestern University');

--SELECT \* FROM pr2.College;

INSERT INTO pr2.Addresses(Street1, Street2, City, State, Zip)

VALUES

('437', 'Columbus Avenue Apt-6', 'Syracuse', 'NY', 13210),

('527', 'Wescott', 'Syracuse', 'NY', 13210),

('312', 'Lanchester Avenue', 'Syracuse', 'NY', 13210),

('989', 'South Campus', 'Syracuse', 'NY', 13210),

('110', 'Downtown', 'Syracuse', 'NY', 13210),

('109', 'Euclid Avenue', 'Syracuse', 'NY', 13210),

('210', 'Baratheon Avenue', 'DALLAS', 'TX', 75201),

('201', 'Westcott Street', 'Syracuse', 'NY', 13210),

('171', 'columbus Avenue Apt-7', 'Syracuse', 'NY', 13210);

--SELECT \* FROM pr2.Addresses;

INSERT INTO pr2.BenefitSelection(BenefitSelection)

VALUES

('Induvidual'),

('Family'),

('Employer'),

('HMO'),

('Short Term'),

('ObamaCare');

--SELECT \* FROM pr2.BenefitSelection;

INSERT INTO pr2.DayOfWeek(Text)

VALUES

('Sunday'),

('Monday'),

('Tuesday'),

('Wednesday'),

('Thursday'),

('Friday'),

('Saturday');

--SELECT \* FROM pr2.DayOfWeek;

INSERT INTO pr2.SemesterText(SemesterText)

VALUES

('Fall'),

('Spring'),

('Summer'),

('Winter');

--SELECT \* FROM pr2.SemesterText;

INSERT INTO pr2.JobInformation (JobDescription, JobRequirements, MinimumPay, MaximumPay,UnionJob)

VALUES

('Assistant Professor', NULL, 50000, 200000, 1),

('Senior Professor', NULL, 100000, 400000, 0),

('Office Instructor', NULL, 10000, 200000, 1),

('Dean', NULL, 300000, 900000, 0),

('Administrator', NULL, 75000, 250000, 1),

('Teaching Assistant', NULL, 90000, 100000, 1),

('Hostel Warden', NULL, 50000, 150000, 1);

--SELECT \* FROM pr2.JobInformation;

INSERT INTO pr2.People(PersonID, NTID, FirstName, LastName, Password, DOB, SSN, HomeAddress,LocalAddress, IsActive)

VALUES

('528789524', 'Kbanger', 'Karthik', 'Bangera', 'happy@125', '1990-09-13', NULL, 1,8,1),

('672854896', 'Stendu', 'Sachin', 'Tendulkar', NULL, '1998-09-4', '867567843',2,7,1),

('236584366', 'Ysingh', 'Yuvraj', 'Singh', NULL, '1995-9-1', NULL,9,NULL,0),

('982546678', 'Rdravid', 'Rahul', 'Dravid', 'GoliHodi', '1992-01-19', '454534897',8,NULL,1),

('452727892', 'Kpeters', 'Kevin', 'Peterson', NULL, '1998-02-26','234575345',6,7,1),

('928789255', 'Pphanth', 'Pink ', 'Phanther', 'L0pEr007', '1997-0811','989877679',4,7,1),

('526745533', 'Scold', 'Stone', 'Cold', 'Dashu00\*','1996-09-18','415152678',1,6,0),

('245386598', 'Ahathw', 'Anny ', 'Hathway', 'ChotaBheem', '1994-09-21',NULL,6,NULL,0),

('562784256', 'Jcutler', 'Jay', 'Cutler', NULL, '1995-03-15','786734721',5,6,1),

('812653629', 'Ynaveen', 'Yash', 'Naveen', 'Jackie!@', '1994-08-20', '100531336',8,4,1),

('787825614', 'Ureddy', 'Umesh', 'Reddy', 'santa019', '1996-09-13', NULL, 5,8,0),

('787825189', 'Hvenkat', 'Hucha', 'Venkat', 'ramya69', '1990-05-13', '156425876', 5,4,1),

('787876245', 'Ajames', 'Andrew', 'James', 'Money$01', '1996-05-24', '526742561', 2,1,1);

--SELECT \* FROM pr2.People;

INSERT INTO pr2.SemesterInfo(Semester, Year, FirstDay, LastDay)

VALUES

(1, 2015, '2015-08-22', '2015-12-16'),

(2, 2015, '2015-01-18', '2015-06-15'),

(3, 2016, '2016-08-21', '2016-12-20'),

(4, 2016, '2016-01-18', '2016-06-15'),

(1, 2016, '2016-08-22', '2016-12-16'),

(2, 2016, '2016-01-18', '2016-06-15'),

(3, 2015, '2015-08-21', '2016-01-20'),

(4, 2015, '2015-01-18', '2015-06-15');

--SELECT \* FROM pr2.SemesterInfo;

INSERT INTO pr2.CourseDailySchedule(CourseID, DayOfWeek, StartTime, EndTime)

VALUES

(11, 2, '01:15:00', '03:15:00'),

(10, 4, '02:45:00', '04:45:00'),

(9, 6, '04:35:00', '05:30:00'),

(8, 3, '12:00:00', '01:00:00'),

(5, 5, '06:15:00', '07:00:00'),

(6, 2, '02:20:00', '03:30:00'),

(7, 4, '01:35:00', '03:00:00');

--SELECT \* FROM pr2.CourseDailySchedule;

INSERT INTO pr2.CourseSchedule(CourseCode,SemesterID,NumberOfSeats,CourseNumber,Location)

VALUES

('DBMS',2,34,551,1),

('ECE',3,29,801,2),

('SMA',6,53,681,3),

('MCE',5,69,689,4),

('ADS',3,26,674,5),

('ACE',4,100,877,6),

('SE',3,60,684,7),

('IST',3,50,678,4);

--SELECT \* FROM pr2.CourseSchedule;

INSERT INTO pr2.ProjectorInfo(ProjectorText)

VALUES

('Laser Projector'),

('3D Projector'),

('LCD Projector'),

('DLP Projector'),

('LED Projector'),

('Smart-Board');

--SELECT \* FROM pr2.ProjectorInfo;

INSERT INTO pr2.Classroom

(Building, RoomNumber, MaximumSeating, ProjectorID, WhiteBoardCount, OtherAV)

VALUES

(7,'201',100,1,2,NULL),

(3,'211',80,3,3,NULL),

(6,'300',40,4,2,NULL),

(2,'388',50,2,3,NULL),

(1,'115',300,5,3,NULL),

(4,'213',50,6,1,NULL),

(5,'123',60,3,1,NULL);

--SELECT \* FROM pr2.Classroom;

INSERT INTO pr2.CourseCatalogue(CourseCode, CourseNumber, CourseTitle, CourseDescription)

VALUES

('SMA', 681, 'Software Modelling Analysis','Core Subject'),

('SE', 684,'Software Engineering', NULL),

('DBMS',551, 'Database Management System', NULL),

('ECE',801, 'Embedded System', 'Introduction'),

('IST',678, 'Project Management', 'Introduction'),

('ADS',674, 'Advanced Data Structures', 'Core Subject'),

('MCE', 689, 'Mechanical Analogy', 'Core Subject'),

('ACE',877, 'Analog Electronics', 'Advanced');

--SELECT \* FROM pr2.CourseCatalogue;

INSERT INTO pr2.StudentInfo(PersonID, StudentStatusID)

VALUES

('452727892',8),

('812653629',9),

('236584366',10),

('982546678',11),

('562784256',12),

('526745533',8),

('245386598',NULL);

--SELECT \* FROM pr2.StudentInfo;

INSERT INTO pr2.AreaOfStudy(StudyTitle,CollegeID)

VALUES

('Computer Science', 2),

('Computer Engineering', 8),

('Chemical Engineering', 7),

('Mechanic Engineering', 2),

('Civil Engineering', 1),

('Information School', 4),

('Adevertising', 5),

('Law School', 6),

('Physical STUDIES', 3),

('Medical Engineering', 6);

--SELECT \* FROM pr2.AreaOfStudy;

INSERT INTO pr2.StudentAreaOfStudy(StudentID, AreaID, IsMajor)

VALUES

(1,12,1),

(7,13,0),

(2,14,1),

(6,15,0),

(5,16,0),

(3,17,1),

(4,18,0),

(3,19,1),

(7,20,1),

(6,21,0);

--SELECT \* FROM pr2.StudentAreaOfStudy;

INSERT INTO pr2.Benefits(BenefitCost, BenefitSelection, BenefitDescription)

VALUES

(2000,5, NULL),

(1500,6, 'No Dental Checkups'),

(3000, 7, 'No Ear Checkups'),

(4300,8, NULL),

(1200,9, 'Eye care is not covered'),

(1800, 10, Null);

--SELECT \* FROM pr2.Benefits;

INSERT INTO pr2.EmployeeInfo(PersonID, YearlyPay, HealthBenefits, VisionBenefits, DentalBenefits, JobInformation)

VALUES

('928789255',100000,1,2,3,1),

('787825614',90000,2,3,1,5),

('528789524',78000,3,2,6,7),

('787825189',56000, 5,2,4,3),

('672854896',200000,2,3,5,4),

('787876245',80000,3,2,5,2);

--SELECT \* FROM pr2.EmployeeInfo;

INSERT INTO pr2.StudentGradingStatus(StudentStatus)

VALUES

('Graded'),

('Ungraded'),

('Withheld');

--SELECT \* FROM pr2.StudentGradingStatus;

INSERT INTO pr2.CourseEnrollment(CourseID, StudentID, StatusID, GradeID)

VALUES

(7,2,1,4),

(6,3,2,3),

(8,1,3,7),

(11,5,2,2),

(10,6,1,1),

(7,4,1,3),

(5,7,2,5);

--SELECT \* FROM pr2.CourseEnrollment;

INSERT INTO pr2.Prerequisites(ParentCode, ParentNumber, ChildCode, ChildNumber)

VALUES

('ACE', 877,'SE',684),

('MCE', 689,'DBMS',551),

('SMA', 681,'IST',678),

('ECE', 801,'SMA',681),

('ECE', 801,'ADS',674),

('ACE', 877,'IST',678);

--SELECT \* FROM pr2.Prerequisites;

INSERT INTO pr2.TeachingAssignment(EmployeeID, CourseScheduleID)

VALUES

(1,12),

(3,9),

(5,5),

(2,8),

(4,10),

(6,7);

--SELECT \* FROM pr2.TeachingAssignment;

*4. Views:*

(Created 4 Views)

**View that contains some Details Regarding the Courses, which includes the Course Title, Semester, year, First Day, Last Day, Day of week, Start Time and the End Time of Classes**

CREATE VIEW pr2.DetailsOfCourseSchedule (CourseTitle, Semester, Year, FirstDay,LastDay, Text, StartTime, EndTime) AS

SELECT cc.CourseTitle, s.SemesterText, si.Year, si.FirstDay, si.LastDay, d.Text,

cds.StartTime, cds.EndTime

FROM Pr2.CourseSchedule cs

INNER JOIN pr2.CourseCatalogue cc

ON cs.CourseCode=cc.CourseCode

INNER JOIN Pr2.SemesterInfo si

ON cs.SemesterID=si.SemesterID

INNER JOIN Pr2.SemesterText s

ON s.SemesterTextID=si.Semester

INNER JOIN Pr2.CourseDailySchedule cds

ON cds.CourseID=cs.CourseScheduleID

INNER JOIN Pr2.DayOfWeek d

ON d.DayOfWeekID=cds.DayOfWeek;

--SELECT \* FROM pr2.DetailsOfCourseSchedule;

**View that contains some of the details of Students like StudentID, FirstName, LastName and the Grade of students**

CREATE VIEW pr2.DetailsOfEnrolledStudents (StudentID,FirstName,LastName,Grade) AS

SELECT s.StudentID, p.FirstName, p.LastName, g.Grade

FROM pr2.StudentInfo s

INNER JOIN pr2.CourseEnrollment e

ON s.StudentId=e.StudentId

INNER JOIN pr2.People p

ON p.PersonID=s.PersonID

INNER JOIN pr2.Grades g

ON g.GradeID=e.GradeID;

--SELECT \* FROM pr2.DetailsOfEnrolledStudents;

**View that contains some of the Details of Faculty like the Faculty First Name, Last Name, Course Title, Job Description and the Yearly pay of Faculty**

CREATE VIEW Pr2.DetailsOfFaculty (FacultyFirstName, FacultyLastName,CourseTitle, JobDescription,AnnualPay) AS

SELECT p.FirstName,p.LastName,ci.CourseTitle,j.JobDescription,e.YearlyPay

FROM Pr2.CourseSchedule cs

INNER JOIN pr2.CourseCatalogue ci

ON cs.CourseCode=ci.CourseCode

INNER JOIN Pr2.TeachingAssignment t

ON t.CourseScheduleID=cs.CourseScheduleID

INNER JOIN Pr2.EmployeeInfo e

ON t.EmployeeID=e.EmployeeID

INNER JOIN Pr2.PEOPLE p

ON e.PersonID=p.PersonID

INNER JOIN Pr2.JobInformation j

ON j.JobID=e.JobInformation;

--SELECT \* FROM pr2.DetailsOfFaculty;

**View that contains the details of Specialization of students. Here the students First name, Last name, Course name, College and the Student status is being displayed**

CREATE VIEW pr2.StudentSpecializationDetails (StudentFirstName, StudentLastName,

StudyTitle, College, StudentStatus) AS

SELECT p.FirstName, p.LastName, sz.StudyTitle, ci.CollegeName, sf.StudentStatus

FROM pr2.StudentInfo s

INNER JOIN pr2.People p

ON s.PersonID=p.PersonID

INNER JOIN pr2.StudentAreaOfStudy ss

ON ss.StudentID=s.StudentID

INNER JOIN pr2.AreaOfStudy sz

ON sz.AreaOfStudyID=ss.AreaID

INNER JOIN pr2.College ci

ON ci.CollegeID=sz.CollegeID

INNER JOIN pr2.StudentStatus sf

ON sf.StudentStatusID=s.StudentStatusID;

*5. Functions:*

**(3 Functions)**

**This is the Functions which displays the average of Yearly Pay for all the Employees**

CREATE FUNCTION pr2.AveragePaymentForEmployees()

RETURNS DECIMAL(8,2) AS

BEGIN

DECLARE @AveragePay AS DECIMAL(8,2)

DECLARE @Length AS INT

SET @Length=0

DECLARE @TempValue AS DECIMAL(8,2)

SET @TempValue=0

DECLARE AveragePaymentForEmployees CURSOR FOR

SELECT YearlyPay FROM pr2.EmployeeInfo;

OPEN AveragePaymentForEmployees;

FETCH NEXT FROM AveragePaymentForEmployees INTO @AveragePay

WHILE @@FETCH\_STATUS=0

BEGIN

SELECT @TempValue=@TempValue+@AveragePay;

SELECT @Length=@Length+1

FETCH NEXT FROM AveragePaymentForEmployees INTO @AveragePay

END

CLOSE AveragePaymentForEmployees

DEALLOCATE AveragePaymentForEmployees

SELECT @AveragePay = (@TempValue /@Length)

RETURN @AveragePay

END

--SELECT pr2.AveragePaymentForEmployees() as AveragePaymentForEmployees;

**This is the Function which diplays the total count of students that each faculty is teaching.**

CREATE FUNCTION pr2.CountOfStudents()

RETURNS @Return TABLE(FacultyName VARCHAR(100), NumberOfStudents INT)

BEGIN

INSERT INTO @Return

SELECT c.FacultyName, COUNT(\*)

FROM pr2.TeachingAssignment ta INNER JOIN pr2.CourseSchedule cs

ON ta.CourseScheduleID=cs.CourseScheduleID

INNER JOIN pr2.CourseEnrollment ce

ON ce.CourseID=cs.CourseScheduleID

INNER JOIN (

SELECT e.EmployeeID,p.FirstName+ ' ' +p.LastName AS FacultyName

FROM pr2.EmployeeInfo e INNER JOIN pr2.People p

ON p.PersonID=e.PersonID

) AS c

ON c.EmployeeID=ta.EmployeeID

GROUP BY c.FacultyName

RETURN

END;

--SELECT \* FROM pr2.CountOfStudents();

**The below Function Displays the total number of students present in each college**

CREATE FUNCTION pr2.NumberOfStudentsPerCollege()

RETURNS @Return TABLE(CollegeName VARCHAR(100), NumberOfStudents INT)

BEGIN

INSERT INTO @Return

SELECT c.CollegeName, COUNT(\*)

FROM pr2.StudentInfo s INNER JOIN pr2.StudentAreaOfStudy sa

ON sa.StudentID=s.StudentID

INNER JOIN (

SELECT aos.AreaOfStudyID, co.CollegeName

FROM pr2.AreaOfStudy aos INNER JOIN pr2.College co

ON co.CollegeID=aos.CollegeID

) AS c ON sa.AreaID=c.AreaOfStudyID

GROUP BY c.CollegeName

RETURN

END;

--SELECT \* FROM pr2.NumberOfStudentsPerCollege();

*6. Procedures:*

(Created 1 Procedure)

**This is the procedure created to increase the yearly pay of the Employees by 10% of their Income. And if the Current pay of the employee is greater than the Entered value i.e Threshold pay, then the Employee Income increase by 30 %**

CREATE PROCEDURE [pr2].[IncreaseInYearlyPay] (@ThresholdPay AS DECIMAL(8,2)) AS

DECLARE @CurrentPay DECIMAL(8,2)

DECLARE @Employee INTEGER

DECLARE CurrentPay CURSOR FOR

SELECT e.YearlyPay,e.EmployeeID

FROM pr2.EmployeeInfo e;

OPEN CurrentPay;

FETCH NEXT FROM CurrentPay INTO @CurrentPay,@Employee

WHILE @@FETCH\_STATUS=0

BEGIN

IF (@CurrentPay>@ThresholdPay)

BEGIN

Update pr2.EmployeeInfo SET YearlyPay=(1.1\*@CurrentPay) WHERE EmployeeID=@Employee;

END

ELSE

BEGIN

Update pr2.EmployeeInfo SET YearlyPay=(1.3\*@CurrentPay) WHERE EmployeeID=@Employee;

END

FETCH NEXT FROM CurrentPay INTO @CurrentPay,@Employee

END

CLOSE CurrentPay;

DEALLOCATE CurrentPay;

RETURN

END;

*To Check Intial Income of Employees*

select \* from pr2.EmployeeInfo;

*To Execute the Procedure*

exec pr2.Payincrease '56000';

*To check the final income of Employees*

select \* from pr2.EmployeeInfo;