WESTMINSTER INTERNATIONAL UNIVERSITY IN TASHKENT

COURSEWORK SUBMISSION FORM

STUDENT USE		STAFF USE	
Module Name	Database Systems Development	First Marker's (acts as signature)	
Module Code	5BUIS009C	Second Marker's (acts as signature)	
Lecturer Name	Dmitriy Pochitaev	Agreed Mark	
UoW Student IDs		**For Registrar's office use only (hard copy submission)**	
WIUT Student IDs	00012860		
Deadline Date	10-Dec-2021		
Assignment Type	☐ Group ☑ Individual		
Word Count			

SUBMISSION INSTRUCTIONS

COURSEWORKS *must* be submitted in *both* HARD COPY (to the Registrar's Office) *and* ELECTRONIC unless instructed otherwise.
For hardcopy submission instructions refer to:
http://intranet.wiut.uz/Shared%20Documents/Forms/AllItems.aspx - Coursework hard copy submission instructions.doc
For online submission instructions refer to:
http://intranet.wiut.uz/Shared%20Documents/Forms/AllItems.aspx - Coursework online submission instructions.doc

MARKERS FEEDBACK (Continued on the next page)

WESTMINSTER INTERNATIONAL UNIVERSITY IN TASHKENT

Contents

Case Description

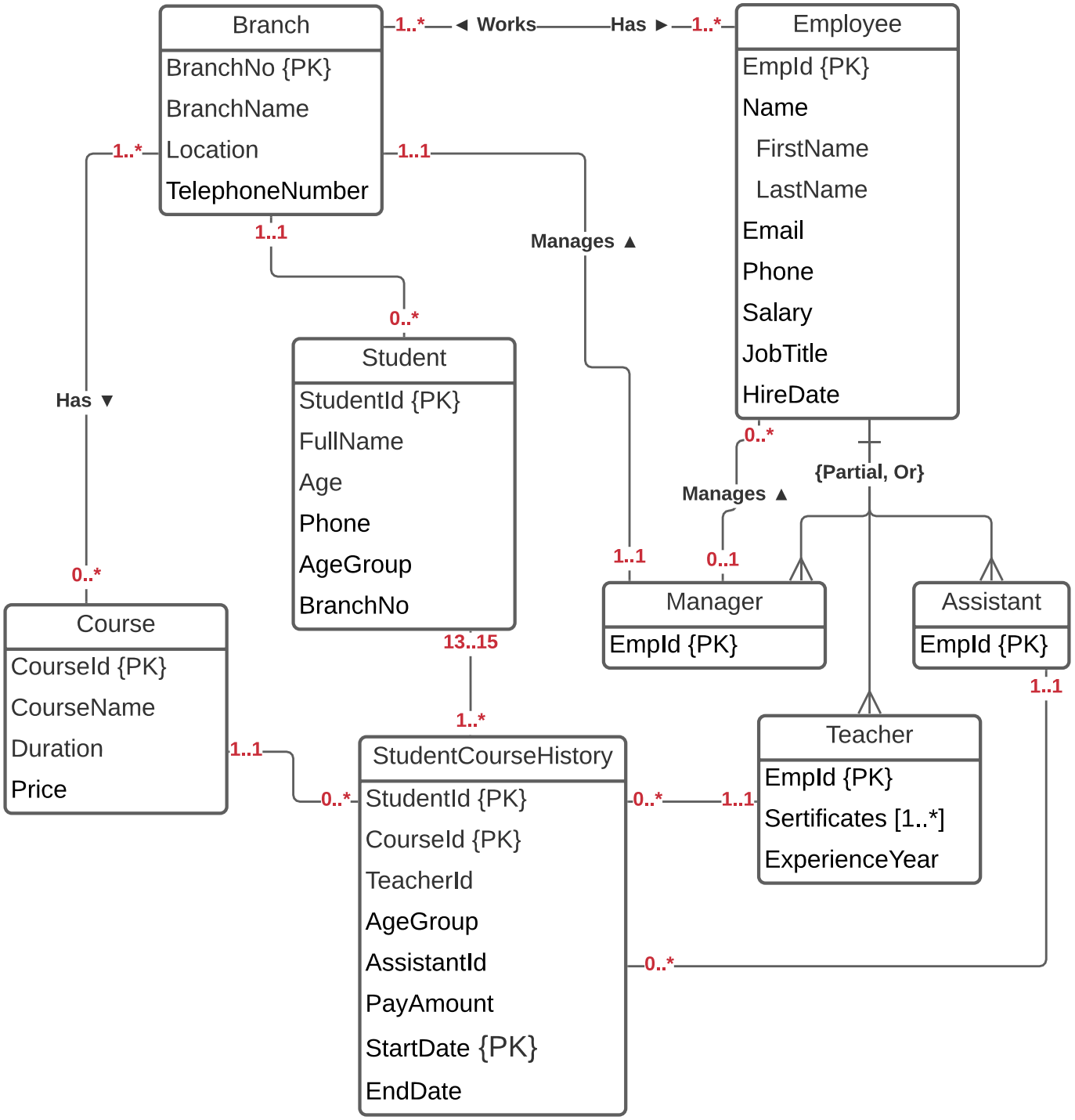
Inter-nation School

Inter nation one of the popular education centers in Tashkent. Company's main service is English education with a considerable number of students. Company offers 3 main types of courses, with each having different levels of teaching programs and levels. Company has 4 branches and operates only in Tashkent city. As it is an education system it is certain that company should have a big and structured database in order to store all records. There are different types of students in terms of age, and education level, and employees in different areas of operation like teachers, managers, call operators, etc. To track all these data a well-structured relation database should be present.

Requirements

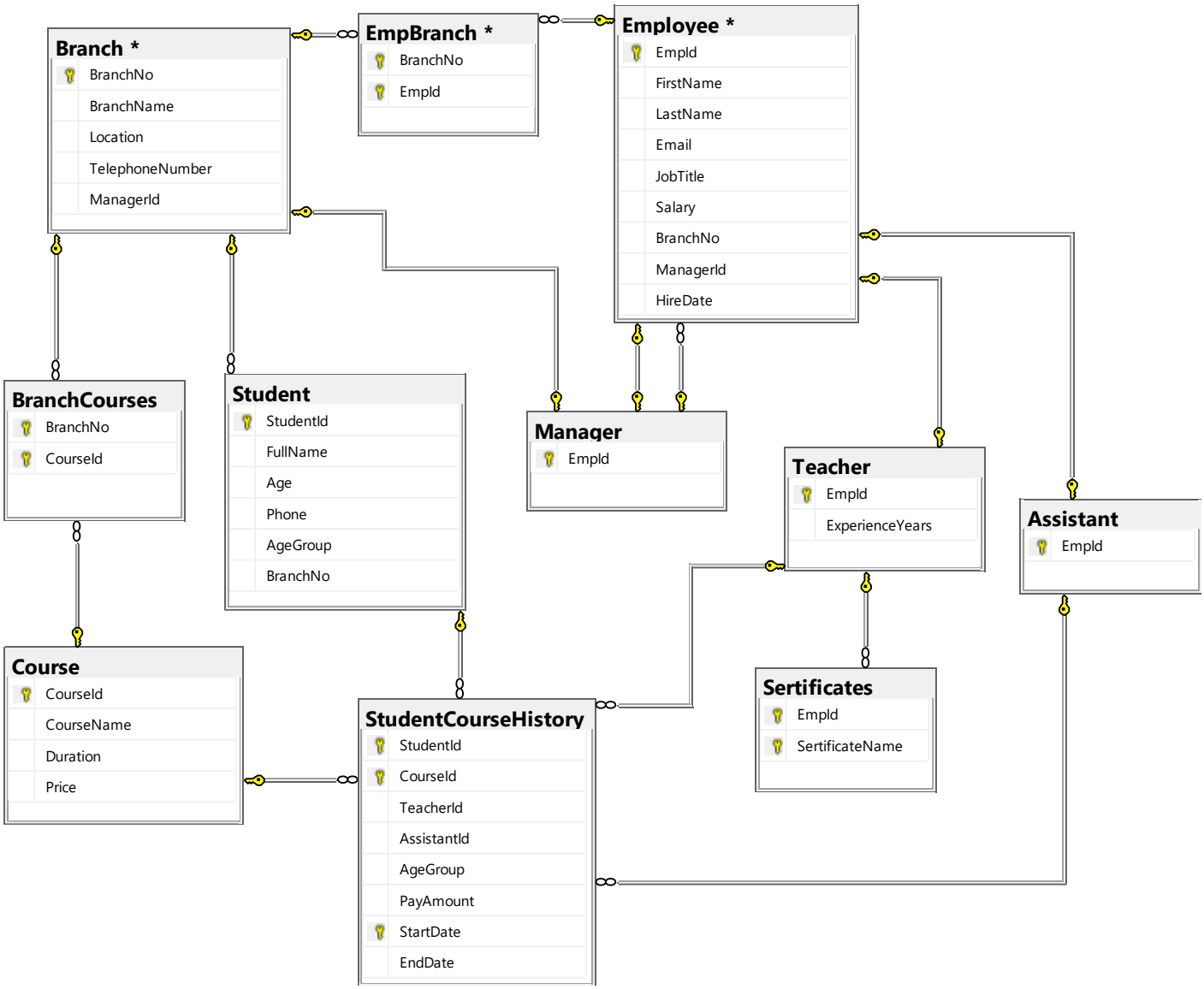
Collecting some information and experience we can implement Conceptual Data Model to properly build logical representation. Starting from top to bottom, there are four branches, and each brand has one main manager and 1-2 intermediate managers. There are three main types of courses, but not every branch has these all three courses. One employee can work in more than one branch and every employee is supervised by one manager. Student can take only one course at the same time while teacher can teach more than one course. Database should show all courses taught, students taken this course, starting and ending date, and teacher that has assigned to the course. There should be an intermediate table that shows all offered courses in each branch.

Enhanced ER diagram



EER diagram

Relational model



Relational model

Explanations: We have more tables in Relational model than ER diagram in order to properly model company's database.

Employee superclass and subclass has constraint as partial and or, because there are other staff working in company who are not either Manager, Teacher or Assistant. And employee can have only one position.

Branch is managed by one manager (mandatory participation) and manager can control only one branch, that is why Manager and Branch has one-to-one relationship. However, each employee has its own manager, this is the reason why every Emp. row has manager id. This will generate less NULLs, in case of very top managers who have no manager.

Multi-valued attribute Certificate has its own table where employee id and certificate name are composite keys so that there will not be duplicate tuples.

To show which branch offers which course intermediate table is present.

Each course history record can contain only 14±1 students. Teacher is not directly connected to the course, because we assume the teacher may teach different course. Every student's taken course is stored in completely another table. In this table all student id, course id and start date are primary keys. This means that there would be one possible class student has taken on specific date. But it will cause data redundancy, because some attributes are repeated which will be fixed in next section.

Normalization

Table Branch: 3NF

BranchNo -> BranchName, Location, TelephoneNumber, MangerId

Course: 3NF

CourseId -> CourseName, Duration, Price

Table Employee: 3NF

EmployeeId -> FirstName, LastName, Email, JobTitle, Salary, BranchNo, ManagerId, HireDate

---

Table Student: Violates 3NF

StudentId -> FullName, Age, Phone, AgeGroup, BranchNo

Age -> AgeGroup : non-primary-key attribute is transitively dependent on the primary key.

Original table

StudentId{PK}	*FullName*	*Age*	*Phone*	*BranchNo*	*AgeGroup*

After normalization

StudentId{PK}	*FullName*	*Age*	*Phone*	*BranchNo*

Age {PK}	*AgeGroup*

Table StudentCourseHistory: Violates 1NF – Data redundancy

We should create another table called Class

Original table

StudentId{PK}	*CourseId{PK}*	*TeacherId{PK}*	*StartDate{PK}*	*AssistantId*	*PayAmount*	*StartDate{PK}*	*EndDate*

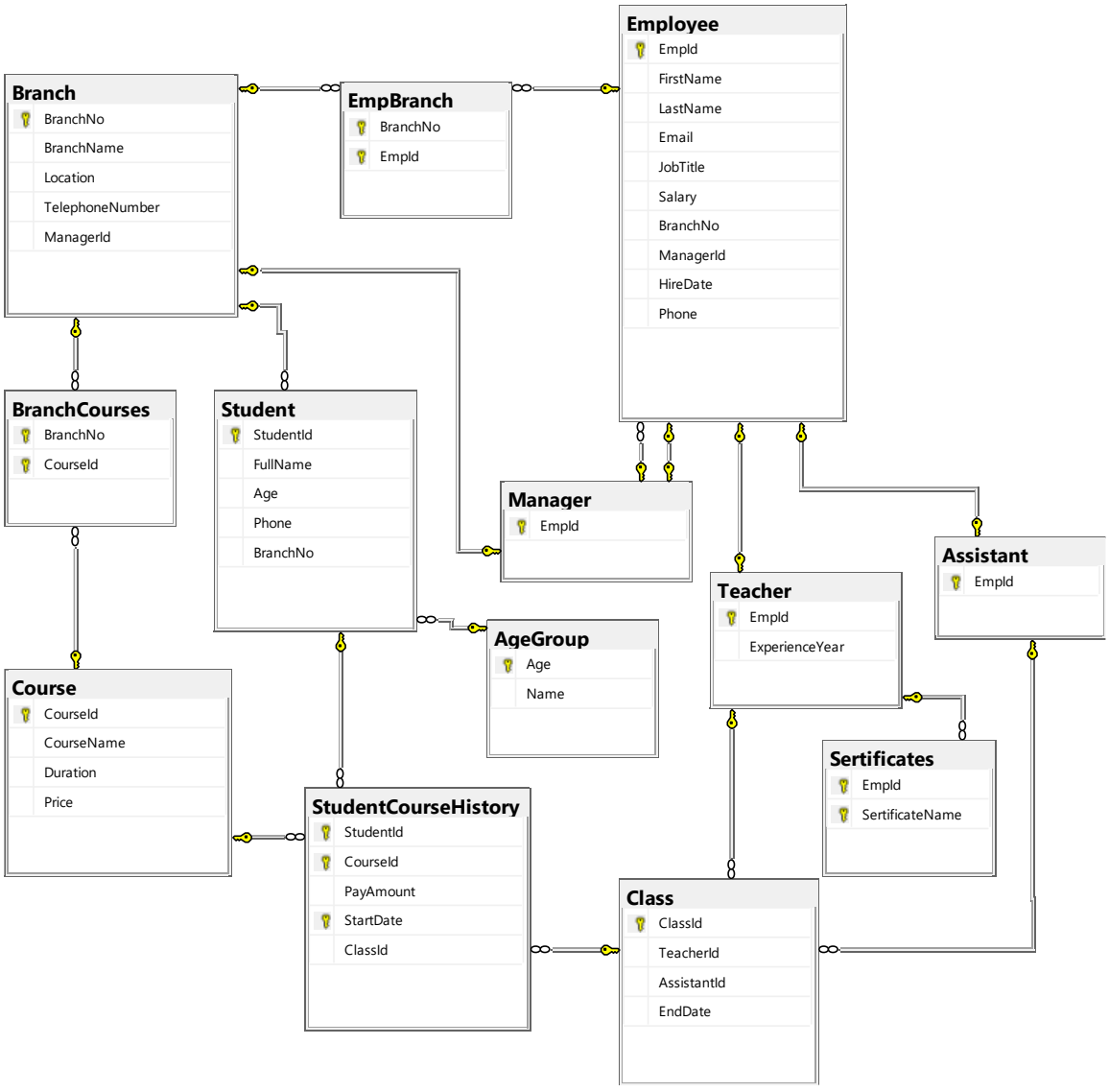
After normalization

StudentId{PK}	*CourseId{PK}*	*PayAmount*	*StartDate{PK}*	*ClassId*

ClassId{PK}	*TeacherId*	*AssistantId*	*EndDate*

All other tables are in 3NF

This is the last output



Appendix

```
USE master
GO
DROP DATABASE cw12860
GO
CREATE DATABASE cw12860
GO
USE cw12860
GO
CREATE TABLE Course(
	CourseId int IDENTITY NOT NULL,
	CourseName nvarchar(200) NOT NULL,
	Duration int NULL,
	Price decimal(10, 2) NULL,
	CONSTRAINT PK_Course_CourseId PRIMARY KEY(CourseId)
)
GO
CREATE TABLE Employee(
	EmpId int IDENTITY NOT NULL,
	FirstName nvarchar(200) NOT NULL,
	LastName nvarchar(200) NOT NULL,
	Email varchar(200) NULL,
	JobTitle nvarchar(200) NULL,
	Salary decimal(10, 2) NULL,
	BranchNo int NULL,
	ManagerId int NULL,
	HireDate date NULL,
	Phone varchar(50) NULL,
	CONSTRAINT PK_Employee_EmpId PRIMARY KEY(EmpId),
	CONSTRAINT UQ_Employee_Email UNIQUE(Email),
	CONSTRAINT UQ_Employee_Phone UNIQUE(Phone)
)
GO
CREATE TABLE Manager(
	EmpId int NOT NULL,
	CONSTRAINT PK_Manager_EmpId PRIMARY KEY(EmpId)
)
GO
CREATE TABLE Assistant(
	EmpId int NOT NULL,
	CONSTRAINT PK_Assistant_EmpId PRIMARY KEY(EmpId),
)
GO
CREATE TABLE Teacher(
	EmpId int NOT NULL,
	ExperienceYear int NULL,
	CONSTRAINT PK_Teacher_EmpId PRIMARY KEY(EmpId),
)
GO
CREATE TABLE Branch(
	BranchNo int IDENTITY NOT NULL,
	BranchName nvarchar(200) NULL,
	Location nvarchar(200) NULL,
	TelephoneNumber varchar(50) NULL,
	ManagerId int NOT NULL,
	CONSTRAINT PK_Branch_BranchNo PRIMARY KEY(BranchNo),
	CONSTRAINT UQ_Branch_ManagerId UNIQUE(ManagerId),
)
GO
CREATE TABLE EmpBranch(
	BranchNo int NOT NULL,
	EmpId int NOT NULL,
	CONSTRAINT PK_EmpBranch_BranchNo_EmpId PRIMARY KEY(BranchNo, EmpId),
)
GO
CREATE TABLE BranchCourses(
	BranchNo int NOT NULL,
```

```
	CourseId int NOT NULL,
	CONSTRAINT PK_BranchCourses_BranchNo_CourseId PRIMARY KEY(BranchNo, CourseId),
)
GO
CREATE TABLE AgeGroup(
	Age int IDENTITY NOT NULL,
	Name nvarchar(200) NULL,
	CONSTRAINT PK_AgeGroup_Age PRIMARY KEY(Age)
)
GO
CREATE TABLE Student(
	StudentId int IDENTITY NOT NULL,
	FullName nvarchar(200) NOT NULL,
	Age int NULL,
	Phone varchar(50) NULL,
	BranchNo int NOT NULL,
	CONSTRAINT PK_Student_StudentId PRIMARY KEY(StudentId),
	CONSTRAINT CK_Student_Age CHECK(Age>=7)
)
GO
CREATE TABLE Class(
	ClassId int IDENTITY NOT NULL,
	TeacherId int NOT NULL,
	AssistantId int NOT NULL,
	EndDate date NULL,
	CONSTRAINT PK_Class_ClassId PRIMARY KEY(ClassId),
)
GO
CREATE TABLE StudentCourseHistory(
	StudentId int NOT NULL,
	CourseId int NOT NULL,
	PayAmount decimal(10, 2) NULL,
	StartDate date NOT NULL,
	ClassId int NOT NULL,
	CONSTRAINT PK_StudentCourseHistory_StudentId_CourseId_StartDate PRIMARY KEY(StudentId,
CourseId, StartDate),
	CONSTRAINT FK_StudentCourseHistory_ClassId FOREIGN KEY(ClassId) REFERENCES
Class(ClassId),
)
GO
CREATE TABLE Sertificates(
	EmpId int NOT NULL,
	SertificateName varchar(200) NOT NULL,
	CONSTRAINT PK_Sertificates_EmpId_SertificateName PRIMARY KEY(EmpId, SertificateName),
)
GO
ALTER TABLE Employee ADD CONSTRAINT FK_Employee_ManagerId FOREIGN KEY(ManagerId) REFERENCES
Manager(EmpId)
GO


-- INSERTING DATA --
USE cw12860
GO
INSERT INTO [Employee]([FirstName],[LastName],[Email],[JobTitle],[Salary],[HireDate],[Phone])
     VALUES ('Mark','Stark','mark@example.com','manager',2000,'07-07-2020','998993217777')
GO
INSERT INTO [Manager]([EmpId]) VALUES(1)
GO
INSERT INTO
[Employee]([FirstName],[LastName],[Email],[JobTitle],[Salary],[BranchNo],[ManagerId],[HireDat
e],[Phone])
     VALUES ('Tony','Harris','tony@example.com','teacher',1500,1,1,'07-08-
2020','998993216666')
GO
INSERT INTO [Teacher]([EmpId]) VALUES(2)
GO
```

```
INSERT INTO
[Employee]([FirstName],[LastName],[Email],[JobTitle],[Salary],[BranchNo],[ManagerId],[HireDat
e],[Phone])
     VALUES ('Kamala','Harris','kamala@example.com','assistant',800,1,1,'11-11-
2018','998991230000')
GO
INSERT INTO [dbo].[Assistant]([EmpId]) VALUES(3)
GO
INSERT INTO [Branch] ([BranchName],[Location],[TelephoneNumber],[ManagerId])
     VALUES ('Amir Temur Branch','Amir Temur, 1','99711231233',1)
GO
INSERT INTO
[Employee]([FirstName],[LastName],[Email],[JobTitle],[Salary],[BranchNo],[ManagerId],[HireDat
e],[Phone])
     VALUES ('John','Doe','john@example.com','operator',1000,1,1,'11-11-2021','998991231212')
GO
INSERT INTO [Student]([FullName],[Age],[Phone],[BranchNo])
     VALUES ('Harry Potter', 15, '998333453333', 1)
GO
INSERT INTO [Student]([FullName],[Age],[Phone],[BranchNo])
     VALUES ('Tom Nason', 14, '998333451111', 1)
GO
INSERT INTO [Course]([CourseName],[Duration],[Price])
     VALUES ('IELTS',5,500.00)
GO
INSERT INTO [Course]([CourseName],[Duration],[Price])
     VALUES ('Beginner English',3,300.00)
GO
INSERT INTO [Class]([TeacherId],[AssistantId],[EndDate])
     VALUES(1,3,'01-05-2022')
GO
INSERT INTO [StudentCourseHistory]([StudentId],[CourseId],[PayAmount],[StartDate],[ClassId])
     VALUES (1,1,500.00, '01-01-2022',1)
GO


-- DELETE --
USE cw12860
GO
DROP TABLE [Employee]
GO
DROP TABLE [Manager]
GO
DROP TABLE [Teacher]
GO
DROP TABLE [Assistant]
GO
DROP TABLE [Branch]
GO
DROP TABLE [Course]
GO
DROP TABLE [Student]
GO
DROP TABLE [AgeGroup]
GO
DROP TABLE [BranchCourses]
GO
ALTER TABLE StudentCourseHistory DROP CONSTRAINT FK_StudentCourseHistory_ClassId
GO
DROP TABLE [Class]
GO
DROP TABLE [EmpBranch]
GO
DROP TABLE [Sertificates]
GO
DROP TABLE [StudentCourseHistory]
```

References

- Inter-nation.uz. (2018). [online] Available at: https://www.inter-nation.uz/ [Accessed 10 Dec. 2021].