Part 1 Design

Austin Beller

IST 659

Project Narrative

The point of this database is to store numerical data which can be used later as a tool. The point of storing this data is to allow users to make appropriate financial decisions when it comes to different careers and the pursuit of those careers. It will do this by allowing individuals to predict the amount of time it will take to pay off loans based on predicted percent of salary. This will enhance and empower the users to make financially sound decision. This will also give us the future ability to monitor and track salary and tuition relationship and give us a better idea of the job market.

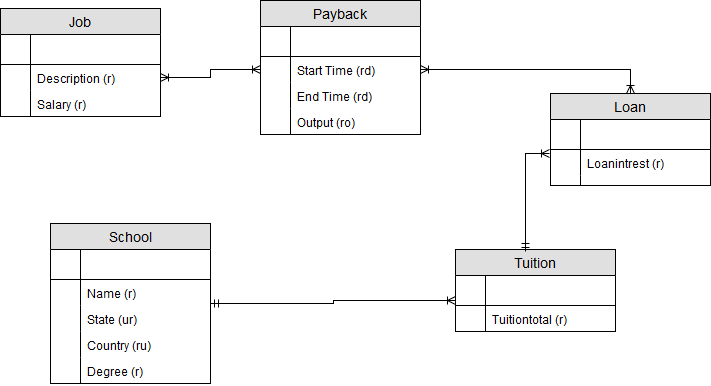
Database Objectives

1. To create a database which stores Tuition, Salaries, and Job Titles to various degrees
2. To create a database which can accurately calculate the amount of time it takes to pay back loans
3. To have a database which is useful.
4. To give individuals a true cost analysis of chosen career path.
5. To compare different salaries and tuition rates across the country.

Data Dictionary

|  |  |  |
| --- | --- | --- |
| Entity | Attribute | Properties |
|  |  |  |
| Tuition | Tuition Total | Required decimal |
| Salary | Salary | Required decimal |
|  |  |  |
| School | Name | Required Full Name |
|  | State | Required Unique |
|  | Country | Required Unique |
|  | Degree | Required Name |
| Job Title | Name | Required Name |
|  | Job Description | Not required characters |
|  |  |  |
| Loans | Loan interest | Required Decimal |
|  |  |  |
| Payback Time | Start Date | Required Date/time |
|  | End Date | Required Date/time |
|  | Output | Required output |

Entity Diagram



Logical model rational

When we create the logical model, we are adding more tables and creating primary keys for our tables to make it easier to integrate into SQL.

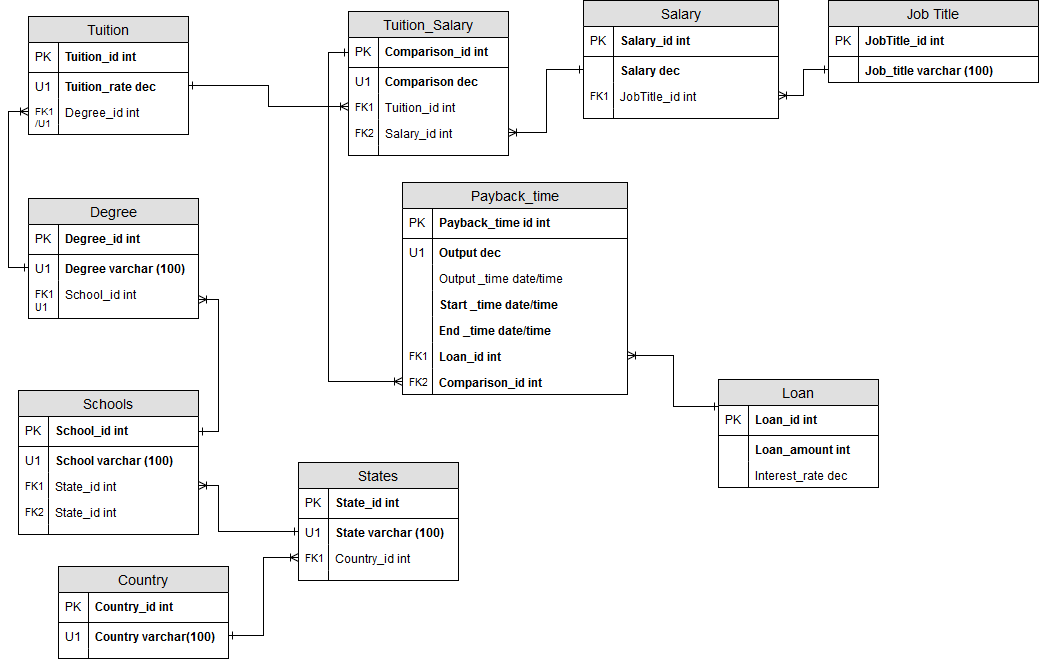
First after listing all of the tables and attributes it was found that many of the attributes listed needed to be given their own table and primary keys. We will also need to eliminate transitive dependencies and functional dependencies.

In this case with our School table we can create tables in order to properly store each attribute without any dependencies. This will maintain the integrity of each category and will also allow the user to select from a dropdown list.

Once all relationships and tables were created it was noted that by making each table attribute unique. It would have made it difficult for two schools who have the same degree programs to be entered in as this would create an error in the database. In order to maintain the integrity of the database we made the foreign keys unique.

This process was done for all the tables represented below as required.

Logical model



Normal Form

The diagram of the logical model is already in normal form. We could normalize further but for the scope of this course 3NF is all that is required.

Part 2: Initialization

--Austin Beller IST 659

--Part 2

DROP TABLE Payback\_time

DROP TABLE Tuition\_Salary

DROP TABLE Tuition

DROP TABLE Loan

DROP TABLE Salary

DROP TABLE JobTitle

DROP TABLE Degree

DROP TABLE Schools

DROP TABLE States

DROP TABLE Country

CREATE TABLE Country (

Country\_id int not null identity

,Country varchar(100)

,CONSTRAINT PK\_Country PRIMARY KEY (Country\_id)

,CONSTRAINT U1\_Country UNIQUE (Country)

)

CREATE TABLE States (

State\_id int not null identity

,States varchar(100)

,Country\_id int not null

,CONSTRAINT PK\_State PRIMARY KEY (State\_id)

,CONSTRAINT U1\_State UNIQUE (States)

,CONSTRAINT FK1\_Country\_id FOREIGN KEY (Country\_id) REFERENCES Country(Country\_id)

)

CREATE TABLE Schools (

School\_id int not null identity

,School varchar (100)

,State\_id int

,CONSTRAINT PK\_School PRIMARY KEY (School\_id)

,CONSTRAINT U1\_School UNIQUE (School)

,CONSTRAINT FK\_State\_id FOREIGN KEY (State\_id) REFERENCES States(State\_id)

)

CREATE TABLE Degree (

Degree\_id int not null identity

,Degree varchar(100)

,School\_id int

,CONSTRAINT PK1\_Degree PRIMARY KEY (Degree\_id)

,CONSTRAINT U1\_Degree UNIQUE (Degree, School\_id)

,CONSTRAINT FK1\_Degree FOREIGN KEY (School\_id) REFERENCES Schools(School\_id)

)

CREATE TABLE Tuition (

Tuition\_id int not null identity

,Tuition\_rate dec

,Degree\_id int

,CONSTRAINT PK1\_Tuition PRIMARY KEY (Tuition\_id)

,CONSTRAINT U1\_Tuition UNIQUE (Tuition\_rate,Degree\_id)

,CONSTRAINT FK1\_Tuition FOREIGN KEY (Degree\_id) REFERENCES Degree(Degree\_id)

)

CREATE TABLE JobTitle(

JobTitle\_id int not null identity

,Job\_Title varchar(100)

,CONSTRAINT PK1\_JobTitle PRIMARY KEY (JobTitle\_id)

)

CREATE TABLE Salary (

Salary\_id int not null identity

,Salary money

,JobTitle\_id int

,CONSTRAINT PK1\_Salary PRIMARY KEY (Salary\_id)

,CONSTRAINT FK1\_Salary FOREIGN KEY (JobTitle\_id) REFERENCES JobTitle(JobTitle\_id)

)

CREATE TABLE Loan (

Loan\_id int not null identity

,Loan\_amount money not null

,Interest\_rate money

,Outputs money

,CONSTRAINT PK1\_Loan PRIMARY KEY (Loan\_id)

)

CREATE TABLE Tuition\_Salary (

Comparison\_id int not null identity

,Comparison dec not null

,Tuition\_id int

,Salary\_id int

,CONSTRAINT PK1\_Tuition\_Salary PRIMARY KEY (Comparison\_id)

,CONSTRAINT U1\_Tuition\_Salary UNIQUE (Comparison)

,CONSTRAINT FK1\_Tuition\_Salary FOREIGN KEY (Tuition\_id) REFERENCES Tuition(Tuition\_id)

,CONSTRAINT FK2\_Tuition\_Salary FOREIGN KEY (Salary\_id) REFERENCES Salary(Salary\_id)

)

CREATE TABLE Payback\_time (

Payback\_time\_id int not null identity

,Outputs dec not null

,Outputs\_time datetime

,Start\_time datetime not null

,End\_time datetime not null

,Loan\_id int not null

,Comparison\_id int not null

,CONSTRAINT PK1\_Payback\_Time PRIMARY KEY (Payback\_time\_id)

,CONSTRAINT U1\_Payback\_Time UNIQUE (Outputs)

,CONSTRAINT FK1\_Payback\_Time FOREIGN KEY (Loan\_id) REFERENCES Loan(Loan\_id)

,CONSTRAINT FK2\_Payback\_Time FOREIGN KEY (Comparison\_id) REFERENCES Tuition\_Salary(Comparison\_id)

)

Go

CREATE FUNCTION dbo.Intrest (@loanamount money, @interest money)

RETURNS int AS

Begin

DECLARE @returnValue int

SET @returnValue = (@loanamount)\*(@interest)

RETURN @returnValue

End

GO

SELECT dbo.Intrest ('3,000', '0.03')

INSERT INTO Loan (Outputs)

Go

CREATE PROCEDURE Getoutput

(@loanamount MONEY, @interestrate MONEY)

AS

BEGIN

INSERT INTO Loan (Outputs, Loan\_amount, Interest\_rate)

SELECT Loan\_id, Loan\_amount, dbo.Intrest (Loan\_amount, @interestrate)

FROM Loan

WHERE Loan\_amount = @loanamount

RETURN @@identity

END

EXEC Getoutput '30', '0.03'

Select \* from Loan

GO

ALTER FUNCTION dbo.HowLong (@paybacktimeid int)

RETURNS int AS

BEGIN

DECLARE @returnValue int

SELECT @returnValue = DATEDIFF (n, Start\_time, End\_time) FROM Payback\_time

Where Payback\_time.Outputs\_time = @paybacktimeid

RETURN @returnValue

End

GO

SELECT dbo.HowLong ('1')

GO

Create Procedure insertdate (@paybackid int)

BEGIN

Declare @returnValue int

SELECT @returnValue = Payback\_time\_id FROM Payback\_time WHERE Payback\_time\_id = @paybackid

UPDATE Payback\_time SET Outputs\_time=DATEDIFF (n, Start\_time, End\_time) Where Payback\_time\_id=@paybackid

Return @@identity

END

We inserted some sample values into our database

INSERT INTO Country (Country)

VALUES ('USA')

INSERT INTO States (States, Country\_id)

VALUES ('New York', '1')

INSERT INTO Schools (School)

VALUES ('Syracuse Univeristy')

INSERT INTO Degree (Degree)

VALUES ('Data Science')

INSERT INTO Tuition (Tuition\_rate)

VALUES ('60000')

INSERT INTO Loan (Loan\_amount, Interest\_rate)

VALUES ('3000','.03')

INSERT INTO JobTitle (Job\_Title)

VALUES ('Data Scientist')

INSERT INTO Salary (Salary)

VALUES ('80000')

We attempted to create functions to do the math

CREATE FUNCTION dbo.Intrest (@loanamount money, @interest money)

RETURNS int AS

Begin

DECLARE @returnValue int

SET @returnValue = (@loanamount)\*(@interest)

RETURN @returnValue

End

GO

CREATE PROCEDURE Getoutput

(@loanamount MONEY, @interestrate MONEY)

AS

BEGIN

INSERT INTO Loan (Outputs, Loan\_amount, Interest\_rate)

SELECT Loan\_id, Loan\_amount, dbo.Intrest (Loan\_amount, @interestrate)

FROM Loan

WHERE Loan\_amount = @loanamount

RETURN @@identity

END

ANSWERS TO DATA QUESTIONS

SELECT

Country.Country

,States.States

FROM STATES

JOIN Country ON Country.Country\_id = States.Country\_id

Select

Degree.Degree

,Schools.School

FROM Schools

JOIN Degree ON Degree.Degree\_id = Degree.School\_id

Select

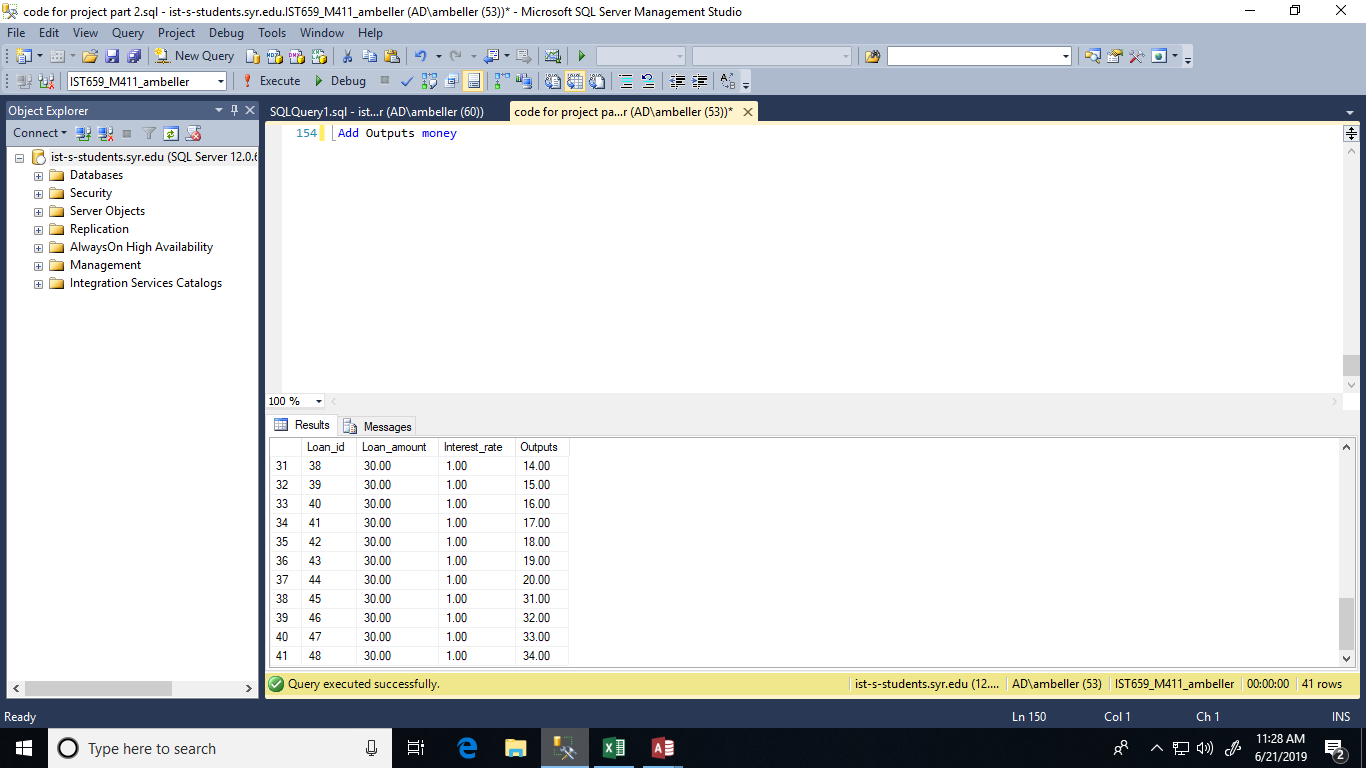
Tuition.Tuition\_rate

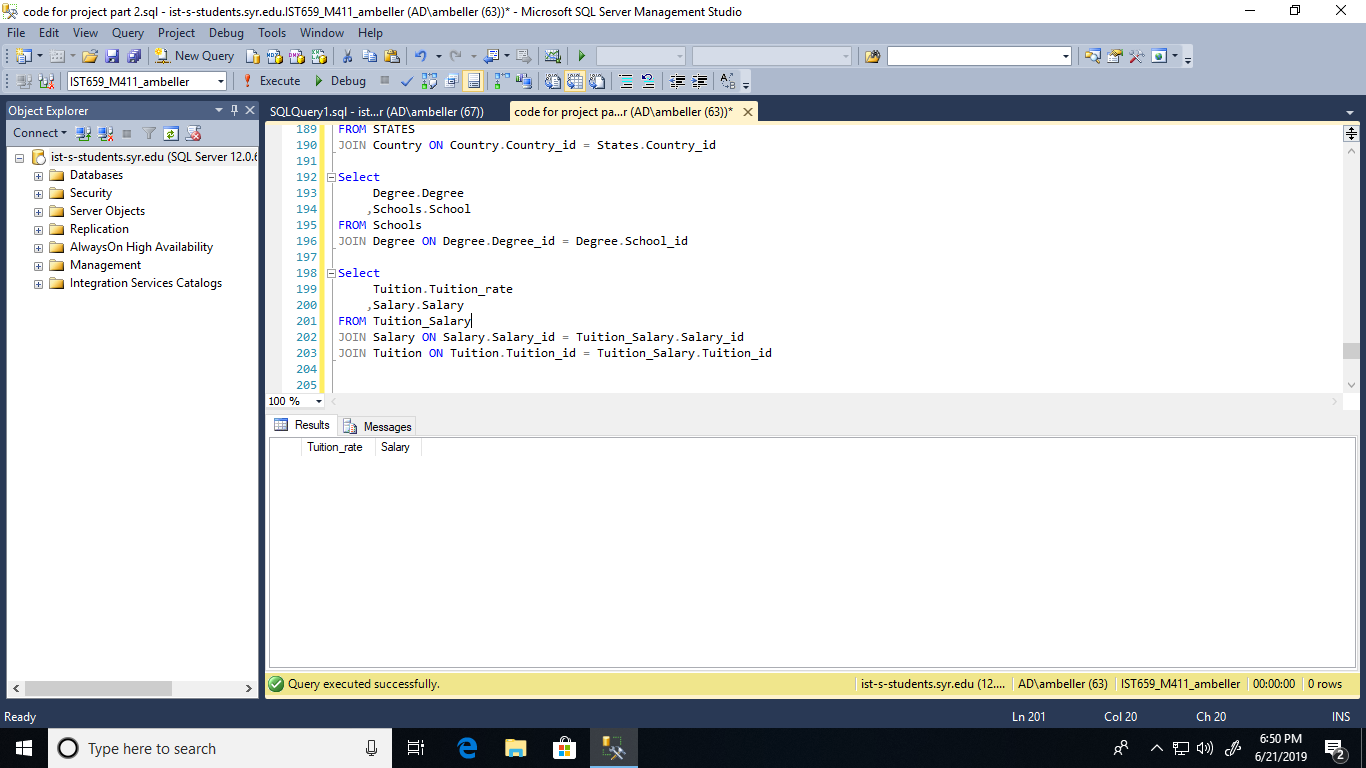
,Salary.Salary

FROM Tuition\_Salary

JOIN Salary ON Salary.Salary\_id = Tuition\_Salary.Salary\_id

JOIN Tuition ON Tuition.Tuition\_id = Tuition\_Salary.Tuition\_id





1. To create a database which stores Tuition, Salaries, and Job Titles to various degrees

We successfully created a database which stored all of these.

1. To create a database which can accurately calculate the amount of time it takes to pay back loans

This was difficult to achieve due to having to account for so many factors. In order to create this, we unsuccessfully attempted to create functions and procedures that would calculate the data which did not go over so well so all calculations were done in excel.

1. To have a database which is useful.

Right now, it is not a useful database since all of the functions and procedures as well as improving the logical design. Once all the discrepancies are worked it absolutely had the potential to be a useful database.

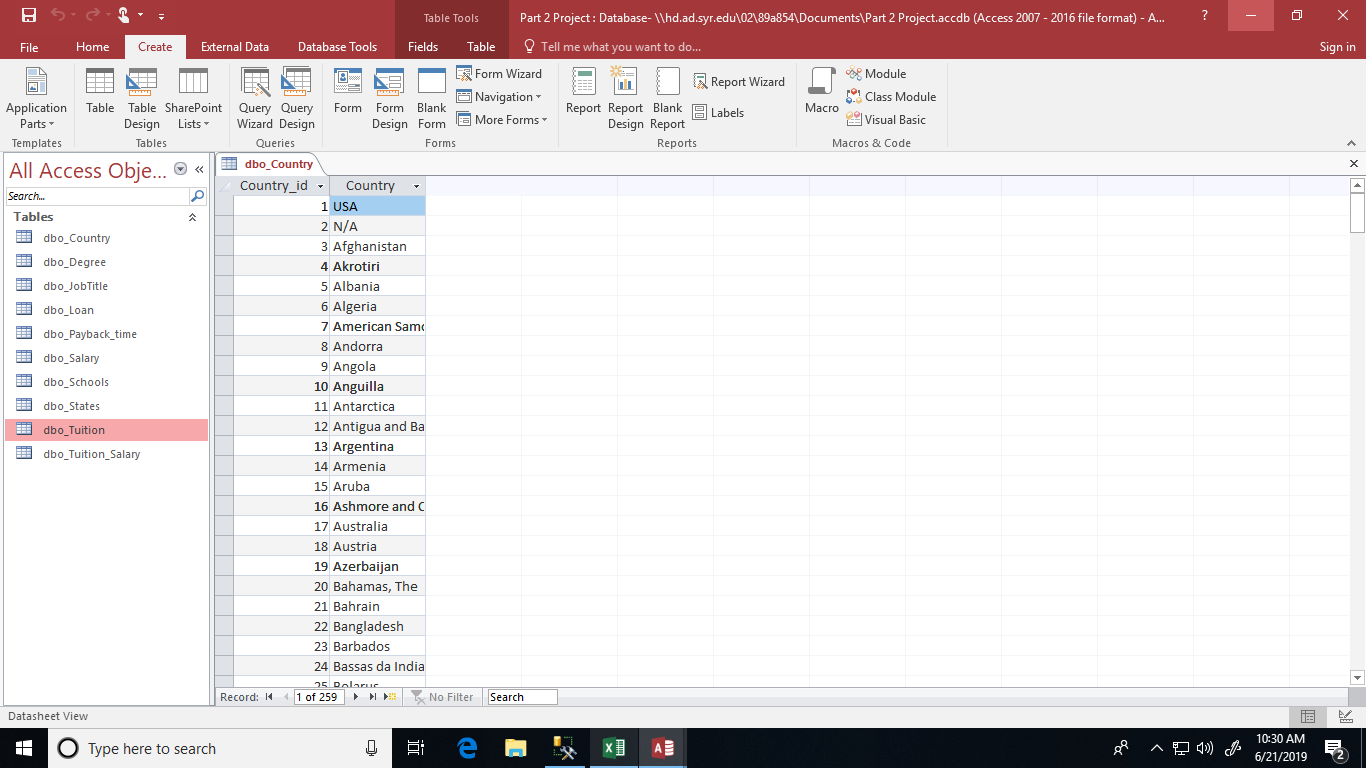
1. To give individuals a true cost analysis of chosen career path.

Given what knowledge we do have now this database is an excellent start. However due to the wide range of factors which would have to be accounted for this is the most accurate simple model we could achieve.

1. To compare different salaries and tuition rates across the country.

Yes, this is a database which could in fact create gather data create a comparison of between tuition and salary rates across the globe.

GUI Prototype



Here we inserted most of the countries via a excel spreadsheet

Reflection-

There are many ways in which this database can be improved upon. One of the ways it can be improved is better design. The scope of the project was very ambitious and still requires a lot of work in order to work out all of the design flaws in the system. We could design more efficient designs and functions to as well as other functions in order to complete the whole database so information inserted flows in a logical order. We also still require a lot of data to be imputed into the system by users. We would have to create views since this is a design for a website for different information on different pages. Many of these things fall out of the scope of the class in terms of time.

SUMMARY-

There were some things that needed to be changed for example we added an output column to the Loans table so that there was an output to the function we created. We also likely needed to work on the logical model as the joins were not working properly. In summary this needs more work.