**ITI EXAMINATION**

**SYSTEM**

****

**Introduction**

**This Project Provides :**

**Examination System**

Constructing an automated system that can perform online exams and build SQL database for ITI system

With The Following Contents:

* Database
* Stored Procedures
* Data Warehouse
* Reports And Dashboards
* Desktop Application

Team Members (Menofia Branch Round 2 (2023/2024))

* Abdelaziz Ragab Abdelaziz Abdelsalam
* Abdelrahman Mohammed Aboulyazeid
* Abdelrahman Mohammed Rabie
* Mohammed Tarek Mohammed Hassan
* Youssef Mohammed Fathy Abdelrahman

Contents

I. Database Design .................................................................................................................................... 4

II. Stored Procedures …............................................................................................................................. 13

III. Data Warehouse ……........................................................................................................................... 19

IV. Extraxt Transform Load (ETL) Process ................................................................................................. 26

V. Report And Dashboard ......................................................................................................................... 32

SSRS Report .............................................................................................................................................. 32

POWER BI Dashboard ............................................................................................................................... 40

VI. Desktop Application ............................................................................................................................ 50

Application Interface ............................................................................................................................... 59

1. **Database Design**

**Overview:**

**The ITI EXAMINATION Database** Constructs an automated system that can perform online exams and manages comprehensive data about departments, instructors, students, courses ,topics ,question ,answer , exams and grades.

The database structure ensures proper organization and relationships between various entities.

**Requirements (Entities) :**

**Departments**

* Each department is uniquely identified by Dep\_ID.
* It has a name, represented by Dep\_Name , Branch , Description ,Capacity and Manager.

**Courses**

* Departments offer many courses.
* Courses are characterized by Crs\_ID and Program\_Name.
* Each Course includes a list of Topics.
* A course is exclusively offered by one or Many Department.

**Students**

* Students are enrolled in a specific Department.
* Student details include ST\_ID, first name, last name, gender, Birthdate, Age

Email, Passward , Intake , Address and City .

* Student are Supervised by one Team Leader Student.

**Instructors**

* Instructors Work in a one specific Department .
* Instructor details include INS\_ID, first name, last name, gender, Salary, Age, Email, OutCopmany , HiringDate , HiringType , Address and City .

**Exams**

* Students can have many exams in many courses .
* Exam details E\_ID, Date, Duration and Exam\_Grade.

**Topics**

* Topic details T\_ID and T\_name .

**Questions**

* Questions details for a specific course are Q\_ID and Q\_name Q\_Type.

**Answers**

* Every question has 4 answers and model answer .

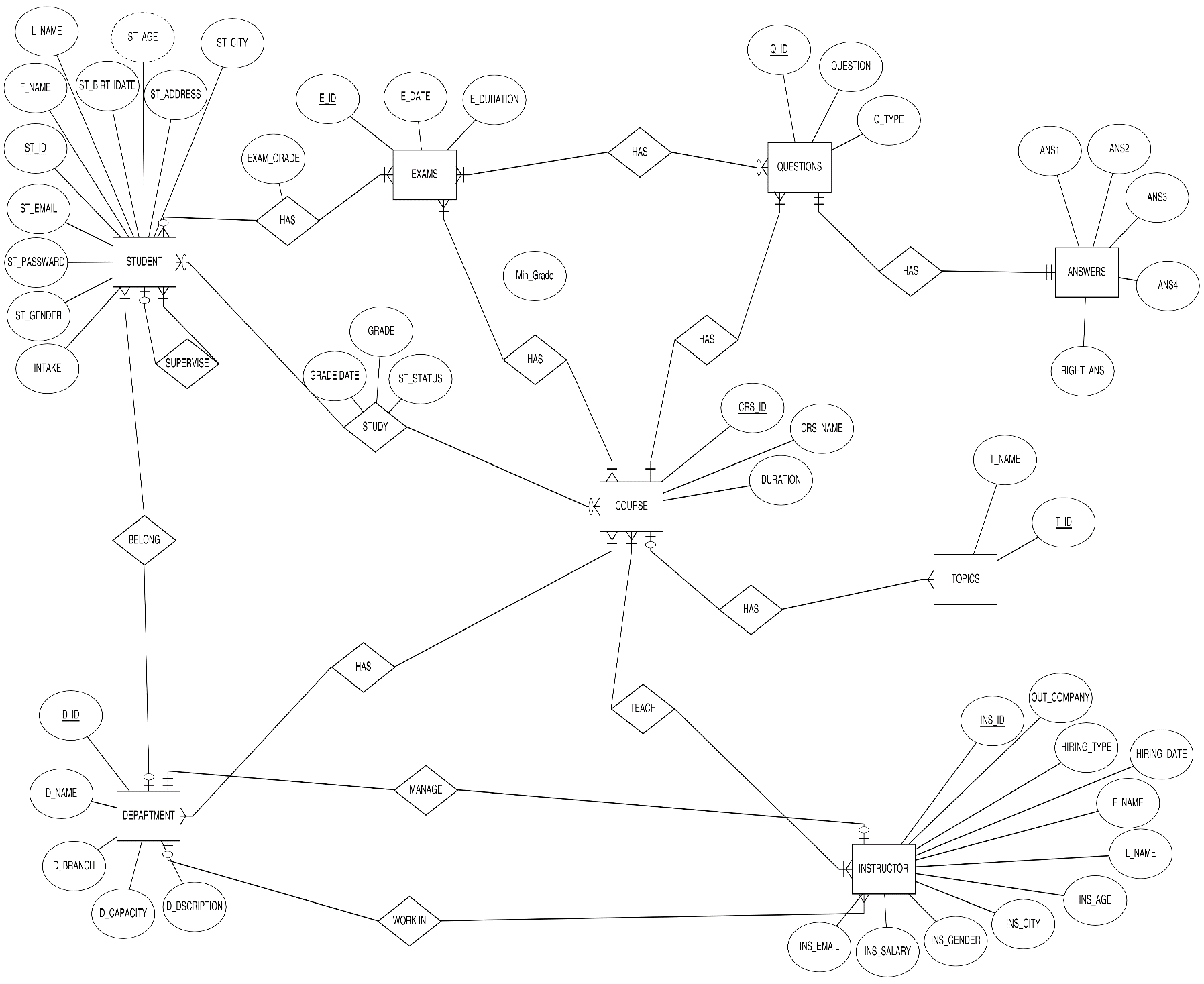
**Relationships:**

* Student must belong to one department and many courses.
* Student may have one or many exams in many courses and may have many exams in the same course.
* Course must have many question and may have many topics.
* Question must have answers.
* Department must be managed by one instructor .
* Instructors must work in one department.
* Student must be supervised by one team leader student .

Grade , Grade\_Date and ST\_Status are attributes in student course relationship .

Exam\_Grade is an attribute in student exam relationship .

* **The ER-Diagram:**

****

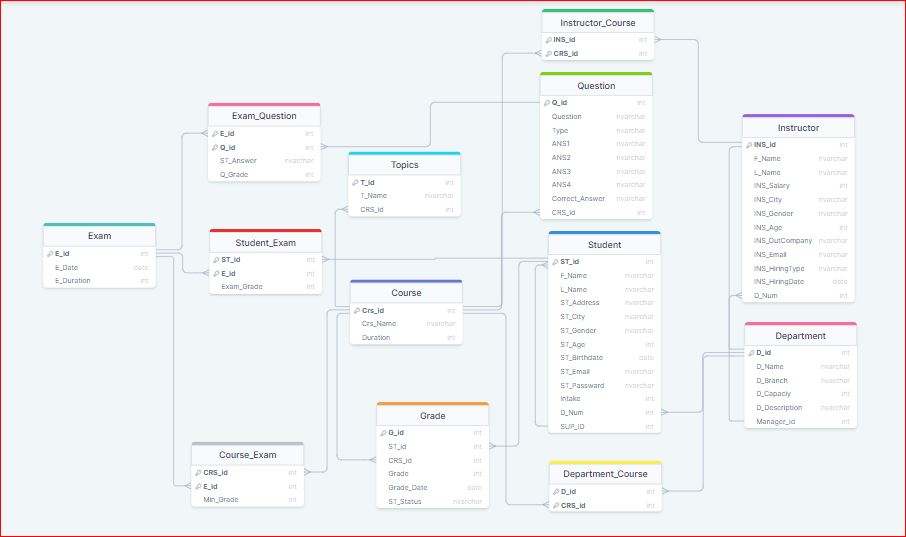
* **Mapping and Normalization:**

**Main Tables (Entities)**

* Departments ( D\_ID , D\_Name , D\_Branch , D\_Capaciy , D\_Description , Manager\_id )
* Courses ( CRS\_ID , CRS\_Name , Duration )
* Students ( ST\_ID , F\_Name , L\_Name , ST\_Address , ST\_City ,ST\_Gender , ST\_Age , ST\_Birthdate , ST\_Email , ST\_Passward , Intake , D\_Num , SUP\_ID )
* Instructors ( INS\_ID , F\_Name , L\_Name , INS\_Salary , INS\_City , INS\_Gender , INS\_Age , INS\_OutCompany , INS\_Email , INS\_HiringType , INS\_HiringDate , D\_Num )
* Exams ( E\_ID , E\_Date , E\_Duration )
* Topics ( T\_ID , T\_Name , CRS\_ID )
* Questions ( Q\_ID , Q\_Name , Q\_Type , Answer 1 , Answer 2 , Answer 3 , Answer 4 , Correct\_Answer , CRS\_ID )

**Secondary Tables (ManyToMany)**

* Grade ( G\_ID , ST\_ID , CRS\_ID , Grade , ST\_Status , Grade\_Date )
* Instructor\_Courses ( ST\_ID , CRS\_ID )
* Department\_Courses ( D\_ID , CRS\_ID )
* Student\_Exam ( ST\_ID , E\_ID , Exam\_Grade )
* Course\_Exam ( CRS\_ID , E\_ID , Min\_Grade )
* Exam\_Question ( E\_ID , Q\_ID , ST\_Answer , Q\_Grade )
* **Mapping-Diagram:**



* **SQL Implementation :**

**Creation of Tables and constraints:**

--Creating Tables Of ITI Relational Database Project

--Department Table

create table Department

(D\_ID int ,

D\_name nvarchar(50),

D\_Branch nvarchar(50) default 'Menofia',

D\_Description nvarchar(50) ,

D\_Capacity int default 25,

D\_ManagerID int ,

constraint D1 primary key(D\_ID))

--Student Table

create table Student

(ST\_ID int ,

F\_name varchar(20),

L\_name varchar(20),

ST\_Address varchar(100),

ST\_City varchar(20) default 'cairo',

ST\_Birthdate date ,

ST\_Age as year(getdate())-year(ST\_Birthdate),

ST\_Gender varchar(1),

ST\_Email varchar(100),

ST\_Password int ,

D\_num int,

SUP\_ID int,

constraint S1 primary key(ST\_ID),

constraint S2 check(ST\_Gender='M' or ST\_Gender='F'),

constraint S3 foreign key(D\_num) references Department (D\_ID)

on delete set null on update cascade )

alter table Student add constraint S4 foreign key(SUP\_ID) references Student (ST\_ID)

--Instructor Table

create table Instructor

(INS\_ID int ,

F\_name varchar(20),

L\_name varchar(20),

INS\_Address varchar(100),

INS\_City varchar(20) default 'cairo',

INS\_Birthdate date ,

INS\_Age as year(getdate())-year(INS\_Birthdate),

INS\_Gender varchar(1),

INS\_Email varchar(100),

INS\_Salary int ,

INS\_HiringType varchar(20),

INS\_Hiringdate date ,

INS\_OutCompany varchar(50),

D\_num int,

constraint IN1 primary key(INS\_ID),

constraint IN2 check(INS\_HiringType='FULL' or INS\_HiringType='PART'),

constraint IN3 check(INS\_Gender='M' or INS\_Gender='F'),

constraint IN4 foreign key(D\_num) references Department (D\_ID)

on delete set null on update cascade )

Alter table Department add constraint D2 foreign key(D\_ManagerID) references Instructor (INS\_ID)

--Course Table

create table Course

(Crs\_ID int ,

Crs\_name varchar(20),

Crs\_Duration int ,

constraint CO1 primary key(Crs\_ID))

--Topic Table

create table Topic

(T\_ID int ,

T\_name varchar(20),

Crs\_ID int,

constraint T1 primary key(T\_ID),

constraint T2 foreign key(Crs\_ID) references Course (Crs\_ID)

on delete set null on update cascade )

--Qusetion Table

create table Question

(Q\_ID int ,

Q\_name varchar(1000),

Q\_Type varchar(10) ,

ANS\_1 varchar(100) ,

ANS\_2 varchar(100) ,

ANS\_3 varchar(100) ,

ANS\_4 varchar(100) ,

ANS\_Right varchar(100) ,

Crs\_ID int ,

constraint Q1 primary key(Q\_ID),

constraint Q2 check(Q\_Type ='MCQ' or Q\_Type='T/F'),

constraint Q3 foreign key(Crs\_ID) references Course (Crs\_ID)

on delete set null on update cascade)

--Exam Table

create table Exam

(E\_ID int identity(1,1),

E\_Date date,

E\_Duration int ,

constraint E1 primary key(E\_ID))

--Creating Tables From Many To Many Relationships

--Grade Table

create table Grade

(Grade\_id int ,

ST\_ID int ,

Crs\_ID int ,

Grade int ,

Grade\_Date date ,

ST\_Status varchar(10),

constraint G1 primary key(Grade\_id),

constraint G2 check(ST\_Status='FAIL' or ST\_Status='GOOD' or ST\_Status='Very Good' or ST\_Status='Excellent'),

constraint G3 foreign key(ST\_ID) references Student (ST\_ID) on update cascade ,

constraint G4 foreign key(Crs\_ID) references Course (Crs\_ID) on update cascade )

--Instructor\_Course Table

create table Instructor\_Course

(INS\_ID int ,

Crs\_ID int ,

constraint IC1 primary key(INS\_ID,Crs\_ID),

constraint IC2 foreign key(INS\_ID) references Instructor (INS\_ID) on update cascade ,

constraint IC3 foreign key(Crs\_ID) references Course (Crs\_ID) on update cascade )

--Department\_Course Table

create table Department\_Course

(D\_ID int ,

Crs\_ID int ,

constraint DC1 primary key(D\_ID,Crs\_ID),

constraint DC2 foreign key(D\_ID) references Department (D\_ID) on update cascade,

constraint DC3 foreign key(Crs\_ID) references Course (Crs\_ID) on update cascade )

--Course\_Exam Table

create table Course\_Exam

( Crs\_ID int ,

E\_ID int ,

Min\_Grade int,

constraint CE1 primary key(Crs\_ID,E\_ID),

constraint CE2 foreign key(Crs\_ID) references Course (Crs\_ID)

on update cascade ,

constraint CE3 foreign key(E\_ID) references Exam (E\_ID)

on update cascade )

--Student\_Exam Table

create table Student\_Exam

( ST\_ID int ,

E\_ID int ,

Exam\_Grade int ,

constraint SE1 primary key(ST\_ID,E\_ID),

constraint SE2 foreign key(ST\_ID) references Student (ST\_ID)

on update cascade ,

constraint SE3 foreign key(E\_ID) references Exam (E\_ID)

on update cascade)

--Exam\_Question Table

create table Exam\_Question

( E\_ID int ,

Q\_ID int ,

ST\_Answer varchar(100),

Q\_Grade int,

constraint EQ1 primary key(E\_ID,Q\_ID),

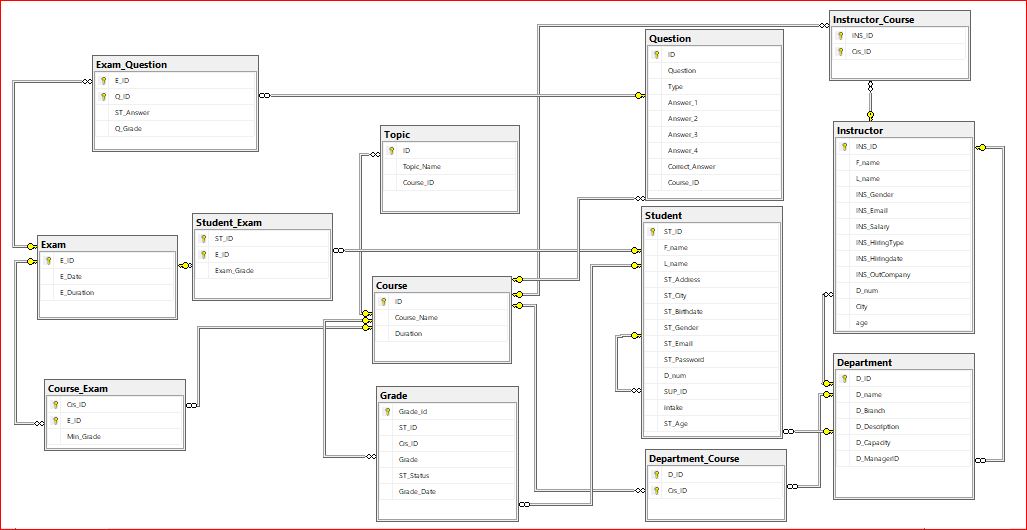
constraint EQ2 check (Q\_Grade=10 or Q\_Grade=0),

constraint EQ3 foreign key(E\_ID) references Exam (E\_ID)

on update cascade ,

constraint EQ4 foreign key(Q\_ID) references Question (Q\_ID)

on update cascade)



1. **Stored Procedures on Database**

**There are four main required stored procedures :**

The first is to generate an exam from existed questions .

The second is to insert answers from student for that questions in database .

The third is to correct that exam and give the grade .

The last one is to update or delete or insert or select from any table .

**Exam Generation :**

It takes the student id and course id then checks if both ids exists then generate an exam contains 10 questions 5 t/f and 5 mcq and inserts data in required tables and finally shows the questions .

create proc Generate\_Exam @Crs\_Id int, @Student\_Id int

as

begin try

if not exists (select \* from Course where ID = @Crs\_Id)

begin

print 'The Course ID You just Entered Does Not Match Any Of Our Records'

end

else

if not exists (select \* from Student where ST\_ID = @Student\_Id)

begin

print 'The Student ID You just Entered Does Not Match Any Of Our Records'

end

else

begin

declare @Exam\_Duration int = 60

insert into Exam(E\_Date, E\_Duration)

values (getdate(), @Exam\_Duration)

declare @Exam\_No int = (select scope\_identity())

insert into Exam\_Question(E\_ID, Q\_ID)

select top(5) @Exam\_No, Q.ID

from Question Q

where Q.Course\_ID = @Crs\_Id and Q.Type = 'TFQ'

order by newid()

insert into Exam\_Question(E\_ID, Q\_ID)

select top(5) @Exam\_No, Q.ID

from Question Q

where Q.Course\_ID = @Crs\_Id and Q.Type = 'MCQ'

order by newid()

insert into Course\_Exam(E\_ID, Crs\_ID)

values (@Exam\_No, @Crs\_Id)

insert into Student\_Exam(E\_ID, ST\_ID)

values (@Exam\_No, @Student\_Id)

select Q.\*

from Question Q, Exam\_Question EQ, Exam E

where Q.ID = EQ.Q\_ID and E.E\_ID = EQ.E\_ID and E.E\_ID = @Exam\_No

end

end try

begin catch

print 'An Error occurred, try again'

end catch

**Exam Answer :**

It takes the student answer and inserts it in Exam\_Question table by knowing the exam id and question id .

CREATE PROCEDURE Exam\_Answer

@Exam\_Id INT,

@Question VARCHAR(100), -- Adjust the size based on your actual data

@Student\_Answer VARCHAR(100) -- Adjust the size based on your actual data

AS

BEGIN

BEGIN TRY

DECLARE @Question\_id INT;

-- Get the ID of the question

SELECT @Question\_id = ID

FROM Question Q

WHERE Question = @Question;

-- Update the student's answer

UPDATE Exam\_Question

SET ST\_Answer = @Student\_Answer

WHERE E\_ID = @Exam\_Id AND Q\_ID = @Question\_id;

END TRY

BEGIN CATCH

-- Log the error or perform any necessary actions for error handling

PRINT 'An error occurred. Please try again with valid input.';

THROW;

END CATCH;

END;

**Exam Correction :**

It takes the student answer from Exam\_Question table and compares it with the right answer from Question table and inserts the grade based on the previous process as well as inserting student status based on his grade .

Create PROCEDURE CorrectExam

@Exam\_Id INT

AS

BEGIN

BEGIN TRY

-- Declare variables for question ID, correct answer, and student's answer

DECLARE @Question\_id INT;

DECLARE @Correct\_Answer VARCHAR(100); -- Adjust the size based on your actual data

DECLARE @Student\_Answer VARCHAR(100); -- Adjust the size based on your actual data

DECLARE @OverallGrade INT = 0;

DECLARE @St\_Id INT;

DECLARE @Crs\_Id INT;

-- Cursor to loop through each question in the exam

DECLARE exam\_cursor CURSOR FOR

SELECT Q\_ID, ST\_Answer

FROM Exam\_Question

WHERE E\_ID = @Exam\_Id;

OPEN exam\_cursor;

FETCH NEXT FROM exam\_cursor INTO @Question\_id, @Student\_Answer;

WHILE @@FETCH\_STATUS = 0

BEGIN

SELECT @Correct\_Answer = Correct\_Answer

FROM Question

WHERE ID = @Question\_id;

-- Update the Q\_Grade column based on whether the student's answer matches the correct answer

UPDATE Exam\_Question

SET Q\_Grade = CASE WHEN @Student\_Answer = @Correct\_Answer THEN 10 ELSE 0 END

WHERE E\_ID = @Exam\_Id AND Q\_ID = @Question\_id;

-- Accumulate the overall grade

SET @OverallGrade = @OverallGrade + CASE WHEN @Student\_Answer = @Correct\_Answer THEN 10 ELSE 0 END;

FETCH NEXT FROM exam\_cursor INTO @Question\_id, @Student\_Answer;

END;

-- Close and deallocate the cursor

CLOSE exam\_cursor;

DEALLOCATE exam\_cursor;

-- Determine status based on overall grade

DECLARE @Status VARCHAR(20);

IF @OverallGrade < 60

SET @Status = 'Fail';

ELSE IF @OverallGrade >= 60 AND @OverallGrade < 80

SET @Status = 'Good';

ELSE IF @OverallGrade >= 80 AND @OverallGrade < 90

SET @Status = 'Very Good';

ELSE

SET @Status = 'Excellent';

-- Get St\_Id for the exam from Student\_Exam table or another relevant table

SELECT @St\_Id = St\_Id

FROM Student\_Exam

WHERE E\_Id = @Exam\_Id;

-- Get Crs\_Id for the exam from Course\_Exam table

SELECT @Crs\_Id = Crs\_Id

FROM Course\_Exam

WHERE E\_Id = @Exam\_Id;

-- Insert/update grade for student and course

IF NOT EXISTS (SELECT \* FROM Student\_Exam WHERE St\_Id = @St\_Id AND E\_Id = @Exam\_Id)

BEGIN

INSERT INTO Student\_Exam(St\_Id, E\_Id, Exam\_Grade)

VALUES (@St\_Id, @Exam\_Id, @OverallGrade);

END

ELSE

BEGIN

UPDATE Student\_Exam

SET Exam\_Grade = @OverallGrade

WHERE St\_Id = @St\_Id AND E\_ID = @Exam\_Id;

END;

END TRY

BEGIN CATCH

print( 'An Error has Occured, Please Enter the Correct Data');

THROW;

END CATCH;

END;

**Tables Manipulations :**

There are many tables which the four main stored procedures created on , as an example we will show the query for only Department table in that documentation and the rest are attached in the main sql file .

create proc View\_Department

@department\_id int

as

if not exists (select \* from Department where D\_ID = @department\_id)

begin

print('The department id you just entered does not match any of our records.')

end

else

begin

select \*

from Department

where D\_ID = @department\_id

end

View\_Department 103

-------------------------------------------

create proc add\_department

@id int,

@name varchar(50) = null,

@branch varchar(50) = null,

@description varchar(max) = null,

@capacity int = null,

@manager\_id int = null

as

if exists (select \* from department where d\_id = @id)

begin

print('Either you did not pass an id for the department or the id already exists.')

end

else

if not exists (select \* from instructor where ins\_id = @manager\_id)

begin

print('Either you did not pass an id for the manager or the id you just entered does not exist.')

end

else

begin

insert into department

values(@id, @name, @branch, @description, @capacity, @manager\_id)

print('The new department data got saved successfully.')

end

add\_department 149, 'Data Science', 'Smart Village', 'Data Science and AI', 20, 1

select \* from Department where D\_ID = 149

-------------------------------------------

create proc update\_department

@id int = null,

@name varchar(50) = null,

@branch varchar(50) = null,

@description varchar(max) = null,

@capacity int = null,

@manager\_id int = null

as

if not exists (select \* from department where d\_id = @id)

begin

print('Either you did not pass an id for the department or the id does not exist.')

end

else

if @manager\_id is not null and not exists (select \* from instructor where ins\_id = @manager\_id)

begin

print('The manager id does not exist.')

end

else

begin

update Department

set D\_name = isnull(@name, D\_name),

D\_Branch = isnull(@branch, D\_Branch),

D\_Description = isnull(@description, D\_Description),

D\_Capacity = isnull(@capacity, D\_Capacity),

D\_ManagerID = isnull(@manager\_id, D\_ManagerID)

where D\_ID = @id

print('Department record got updated successfully.')

end

update\_department @id = 149, @description = 'Information Systems'

select \* from Department where D\_ID = 149

---------------------------------------------

create proc Delete\_Department

@id int

as

if not exists (select \* from department where d\_id = @id)

begin

print('The department id you just entered does not match any of our records.')

end

else

begin

delete from department

where d\_id = @id

print('Department record got deleted successfully.')

end

Delete\_Department 148

select \* from department where d\_id = 148

1. **Dimensional Modeling**

**Data Warehouse Design:**

**Overview:**

**The ITI EXAMINATION Data Warehouse** Considers there are Two Main

Fact Tables (Total Grade ) wich contains measure of total grade in the course

and ( Grade ) wich contains measure of grade in the exam , there are six main Dimensions in the schema ( Date , Student , Department , Course , Exam , Instructor ) , based on previous tables the schema should be a Galaxy schema .

**Requirements (Dimensions) :**

**Department\_DIM**

* Contains data from Department table ( D\_SK , D\_ID , D\_Name , D\_Branch , D\_Capaciy , D\_Description , Manager\_id , Starting\_date , Ending\_date , is\_current , source\_code )

**Instructor\_DIM**

* Contains data from Instructors table ( INS\_SK , INS\_ID , F\_Name ,L\_Name , INS\_Salary , INS\_City , INS\_Gender , INS\_Age , INS\_OutCompany , INS\_Email , INS\_HiringType , INS\_HiringDate , D\_Num , Starting\_date , Ending\_date , is\_current , source\_code )

**Student\_DIM**

* Contains data from Students table ( ST\_SK , INS\_ID , F\_Name ,L\_Name , ST\_Address , ST\_City ,ST\_Gender , ST\_Age , ST\_Birthdate , ST\_Email , ST\_Passward , Intake , D\_Num , SUP\_ID , Starting\_date , Ending\_date , is\_current , source\_code )

**Course\_DIM**

* Contains data from Course , Topics , Questions tables ( CRS\_SK ,

CRS\_ID , CRS\_Name , Duration , T\_ID , T\_Name , Q\_ID , Q\_Name , Q\_Type , Answer 1 , Answer 2 , Answer 3 , Answer 4 , Correct\_Answer , Starting\_date , Ending\_date , is\_current , source\_code )

**Exam\_DIM**

* Contains data from Exam , Student\_Exam , Course\_Exam , Exam\_Question tables ( E\_SK , E\_ID , E\_Date , E\_Duration , ST\_ID ,CRS \_ID ,Min\_Grade Q\_ID , ST\_Answer , Q\_Grade , Starting\_date , Ending\_date , is\_current , source\_code )

**Date\_DIM**

* Contains date from 1/1/2020 to 1/1/2030 with primary key ( Date\_SK )

**Requirements (Hierarchies) :**

**Instructor**\_**Courses**  ( ST\_ID , CRS\_ID )

**Department**\_**Courses**  ( D\_ID , CRS\_ID )

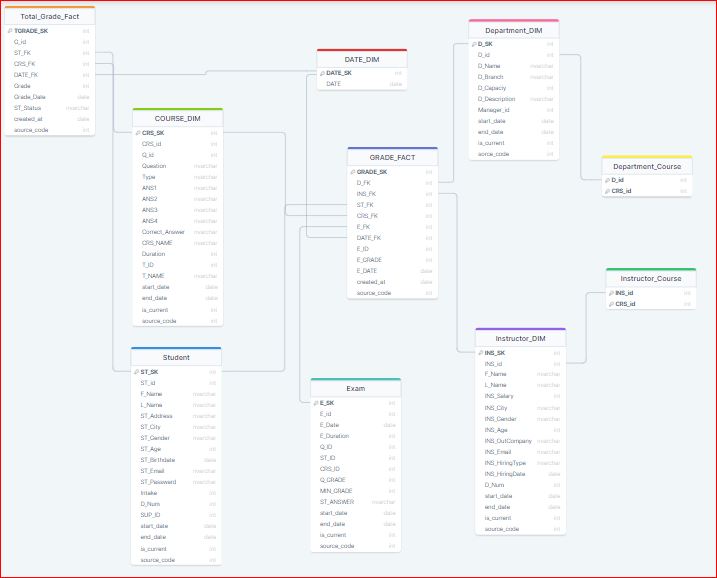
**Requirements (Fact Tables) :**

**Grade\_Fact**

* Contains Exam\_Grade measure and foreign keys from (Department , Student , Course . Exam , Instructor , Date ) Dimensional tables with primary key ( Grade\_SK )

**Total\_Grade\_Fact**

* Contains Grade measure and foreign keys from ( Student , Course , Date ) Dimensional tables with primary key ( Total\_Grade\_SK )
* **DWH-Diagram(Galaxy Schema):**



* **SQL Implementation :**

**Creation of Tables and constraints:**

-- LAST LOAD TABLE

CREATE TABLE [dbo].[LAST\_LOAD]

(

[id] [int] ,

[last\_details] [varchar] (20) ,

[last\_date] [datetime] ,

[last\_id] [int] )

-- DIMENSIONAL TABLES

CREATE TABLE [dbo].[STUDENT\_DIM]

(

[STUDENT\_SK] [int] identity (1,1) PRIMARY KEY ,

[ST\_ID\_BK] [int] NOT NULL,

[F\_NAME] [varchar](20) NULL,

[L\_NAME] [varchar](20) NULL,

[ST\_ADDRESS] [varchar](100) NULL,

[ST\_CITY] [varchar](20) NULL,

[ST\_BIRTHDATE] [date] NULL,

[ST\_AGE] [int] NULL,

[ST\_GENDER] [varchar](1) NULL,

[ST\_EMAIL] [varchar](100) NULL,

[ST\_PASSWARD] [int] NOT NULL,

[D\_NUM] [int] NULL,

[SUP\_ID] [int] NULL,

[starting\_date] [datetime] not null default (getdate()),

[ending\_date] [datetime] null ,

[is\_current] [int] not null default (1),

[Source\_Code] [int] NOT NULL )

CREATE TABLE [dbo].[COURSE\_DIM]

(

[COURSE\_SK] [int] identity (1,1) PRIMARY KEY ,

[CRS\_ID\_BK] [int] NOT NULL,

[CRS\_NAME] [varchar](20) NULL,

[CRS\_DURATION] [INT] NULL,

[TOPIC\_ID\_BK] [int] NOT NULL,

[TOPIC\_NAME] [varchar](20) NULL,

[TOPIC\_DURATION] [INT] NULL,

[QUESTION\_ID\_BK] [int] NOT NULL,

[QUESTION\_NAME] [varchar](1000) NULL,

[QUESTION\_TYPE] [varchar](10) NULL,

[ANS1] [varchar](100) NULL,

[ANS2] [varchar](100) NULL,

[ANS3] [varchar](100) NULL,

[ANS4] [varchar](100) NULL,

[ANS\_RIGHT] [varchar](100) NULL,

[starting\_date] [datetime] not null default (getdate()),

[ending\_date] [datetime] null ,

[is\_current] [int] not null default (1),

[Source\_Code] [int] NOT NULL )

CREATE TABLE [dbo].[INSTRUCTOR\_DIM]

(

[INSTRUCTOR\_SK] [int] identity (1,1) PRIMARY KEY ,

[INS\_ID\_BK] [int] NOT NULL unique ,

[F\_NAME] [varchar](20) NULL,

[L\_NAME] [varchar](20) NULL,

[INS\_ADDRESS] [varchar](100) NULL,

[INS\_CITY] [varchar](20) NULL,

[INS\_BIRTHDATE] [date] NULL,

[INS\_AGE] [int] NULL,

[INS\_GENDER] [varchar](1) NULL,

[INS\_EMAIL] [varchar](100) NULL,

[INS\_SALARY] [int] NOT NULL,

[INS\_HIRING\_TYPE] [varchar](20) NULL,

[INS\_HIRING\_DATE] [date] NULL,

[INS\_OUT\_COMPANY] [varchar](50) NULL,

[D\_NUM] [int] NULL,

[starting\_date] [datetime] not null default (getdate()),

[ending\_date] [datetime] null ,

[is\_current] [int] not null default (1),

[Source\_Code] [int] NOT NULL )

CREATE TABLE [dbo].[DEPARTMENT\_DIM]

(

[DEPARTMENT\_SK] [int] identity (1,1) PRIMARY KEY ,

[D\_ID\_BK] [int] NOT NULL unique ,

[D\_NAME] [varchar](50) NULL,

[D\_BRANCH] [varchar](50) NULL,

[D\_DESCRIPTION] [varchar](50) NULL,

[D\_CAPACITY] [INT] NULL,

[D\_MANAGER\_ID] [INT] NULL,

[starting\_date] [datetime] not null default (getdate()),

[ending\_date] [datetime] null ,

[is\_current] [int] not null default (1),

[Source\_Code] [int] NOT NULL )

CREATE TABLE [dbo].[EXAM\_DIM]

(

[EXAM\_SK] [int] identity (1,1) PRIMARY KEY ,

[E\_ID\_BK] [int] NOT NULL,

[E\_DATE] [DATE] NULL,

[E\_DURATION] [INT] NULL,

[CRS\_ID\_BK] [int] NOT NULL,

[ST\_ID\_BK] [INT] NULL,

[MIN\_GRADE] [INT] NULL,

[QUESTION\_ID\_BK] [int] NOT NULL,

[ST\_ANSWER] [varchar](100) NULL,

[QUESTION\_GRADE] [INT] NULL,

[starting\_date] [datetime] not null default (getdate()),

[ending\_date] [datetime] null ,

[is\_current] [int] not null default (1),

[Source\_Code] [int] NOT NULL )

-- HIERARCHY TABLES

CREATE TABLE [dbo].[INSTRUCTOR\_COURSES]

(

[INS\_ID\_BK] [int] NOT NULL ,

[CRS\_ID\_BK] [int] NOT NULL ,

constraint IC1 primary key( INS\_ID\_BK , Crs\_ID\_BK ))

CREATE TABLE [dbo].[DEPARTMENT\_COURSES]

(

[D\_ID\_BK] [int] NOT NULL ,

[CRS\_ID\_BK] [int] NOT NULL ,

constraint DC1 primary key( D\_ID\_BK , Crs\_ID\_BK ))

-- FACT TABLES

CREATE TABLE [dbo].[GRADE\_FACT]

(

[GRADE\_FACT\_SK] [int] identity (1,1) PRIMARY KEY ,

[STUDENT\_SK\_FK] [int] null ,

[COURSE\_SK\_FK] [int] null,

[INSTRUCTOR\_SK\_FK] [int] null,

[DEPARTMENT\_SK\_FK] [int] null ,

[Date\_SK\_FK] [int] null ,

[EXAM\_SK\_FK] [INT] NULL,

[EXAM\_ID] [INT] NULL,

[EXAM\_GRADE] [int] NOT NULL,

[EXAM\_Date] [DATE] NOT NULL,

[Created\_at] [Datetime] not null default (getdate()),

[Source\_Code] [int] NOT NULL )

CREATE TABLE [dbo].[Total\_GRADE\_FACT]

(

[GRADE\_FACT\_SK] [int] identity (1,1) PRIMARY KEY ,

[STUDENT\_SK\_FK] [int] null ,

[COURSE\_SK\_FK] [int] null,

[Date\_SK\_FK] [int] null ,

[Grade\_ID] [INT] NULL,

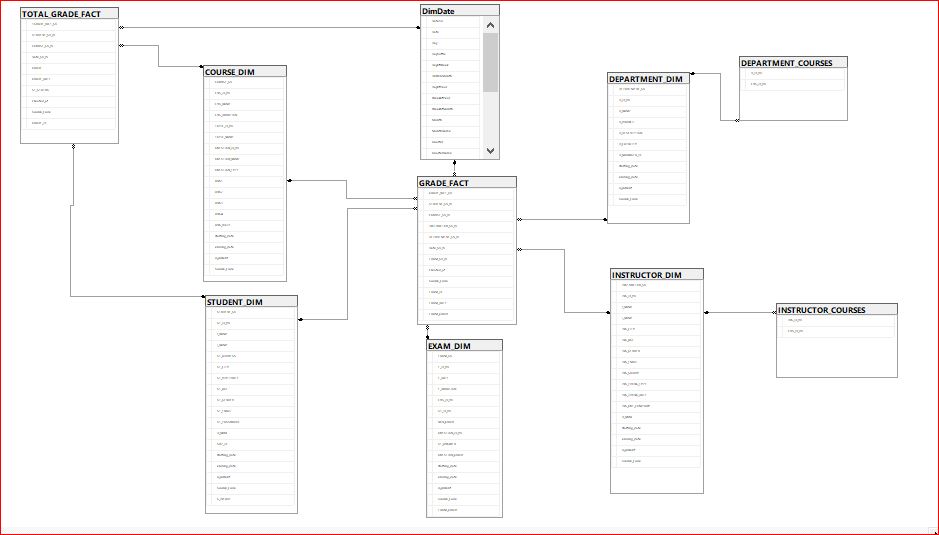
[GRADE] [int] NOT NULL,

[GRADE\_Date] [DATE] NOT NULL,

[ST\_STATUS] [VARCHAR](10) NOT NULL,

[Created\_at] [Datetime] not null default (getdate()),

[Source\_Code] [int] NOT NULL )

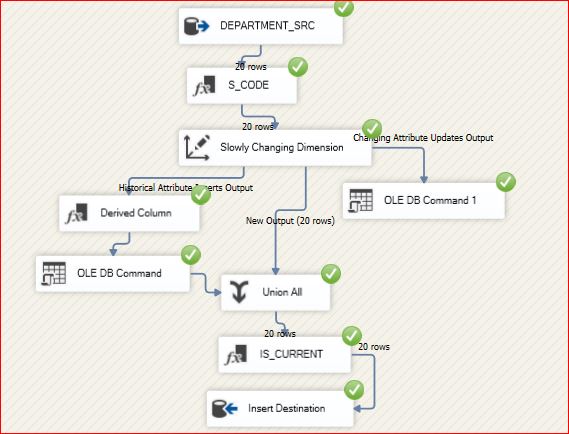


1. **Extract Transform Load (ETL) Process**

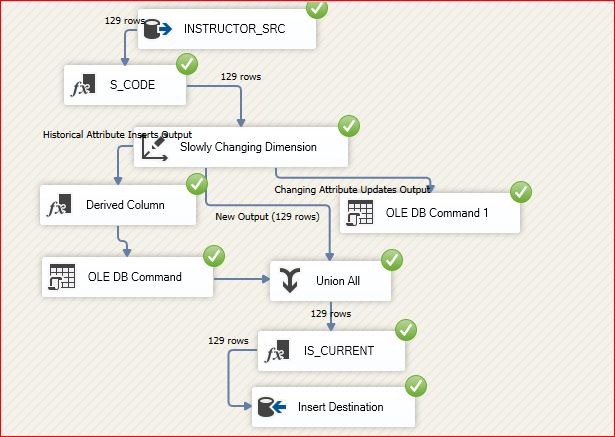
**The ETL PROCESS** is done by SQL SERVER INTEGRATION SERVICES (SSIS) PROGRAM .

**Requirements (Dimensions) :**

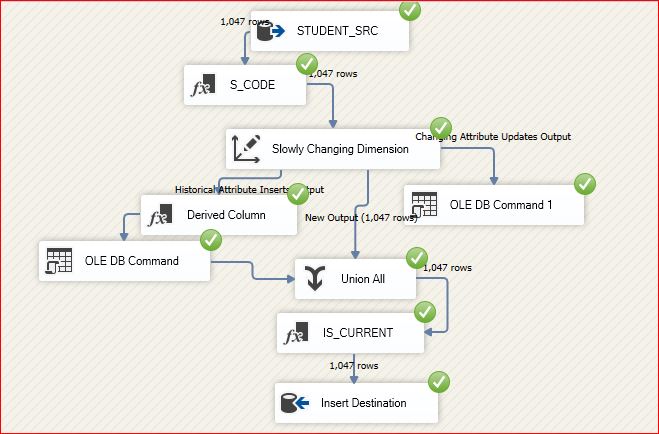
**Department\_DIM**

****

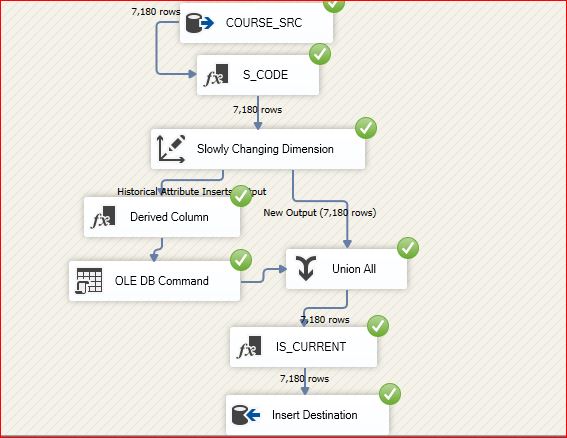
**Instructor\_DIM**



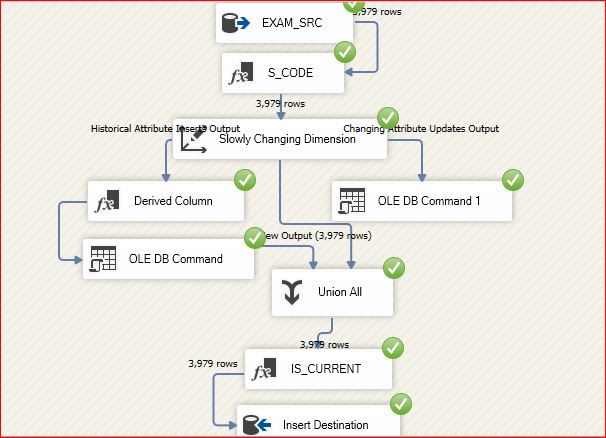
**Student\_DIM**

****

**Course\_DIM**

****

**Exam\_DIM**

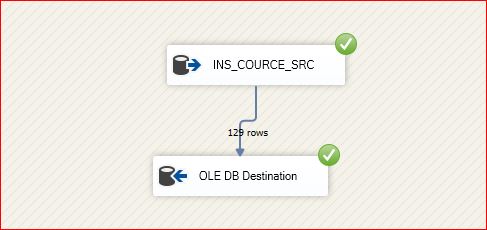


**Date\_DIM**

* Contains date from 1/1/2020 to 1/1/2030 with primary key ( Date\_SK )

**Requirements (Hierarchies) :**

**Instructor**\_**Courses**

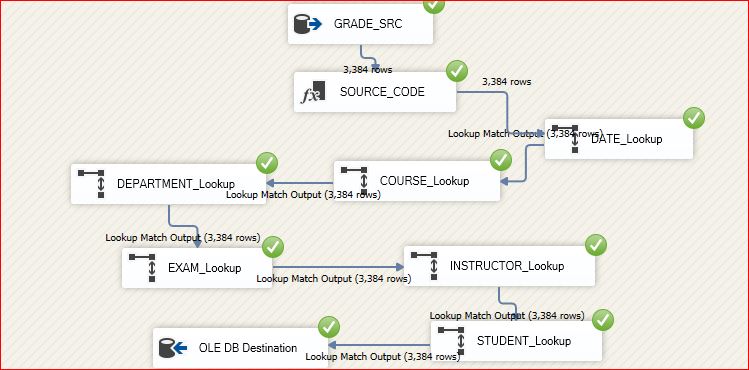


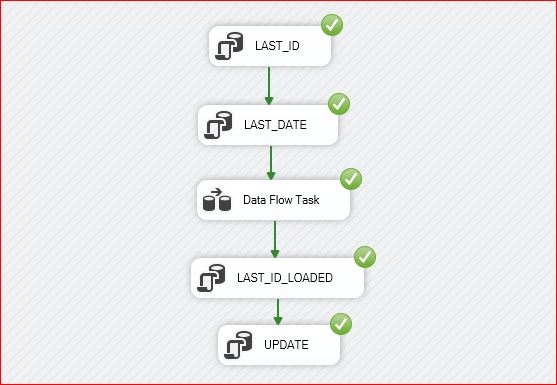
**Department**\_**Courses**



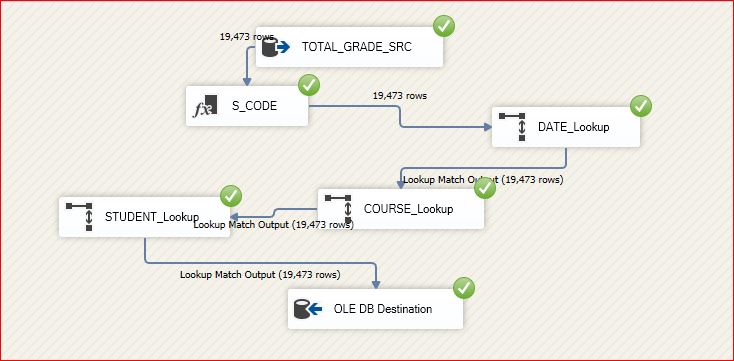
**Requirements (Fact Tables) :**

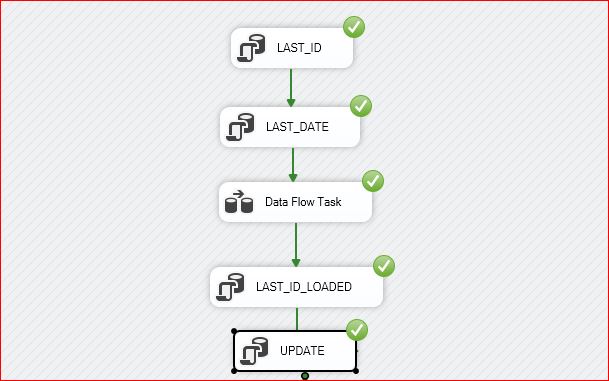
**Grade\_Fact**





**Total\_Grade\_Fact**

****

****

1. **Report And Dashboards**

* SSRS REPORTS

Report that takes exam number and returns the Questions in it and chocies [freeform report]

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  |  |  |  | |
|  | |  |  |  |  |  | |
|  | | |  | | --- | | Power BI Test | | |  |  |
|  | |  | |  | | --- | | Student Name : Abdelhak Boukortt | | |  |
|  | |  |  |  | |
|  | |  |  |  |  |  | |
|  |  | | | | | | | |  |
|  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  | |  | | --- | | 1) | | |  | |  | | --- | | What is the purpose of the "Power Query Editor" in Power BI? | | | |  | | |  |  |  |  |  |  |  |  | | |  |  | |  | | --- | | a) | | | | |  | | --- | | Designing custom visuals | |  |  | | |  |  |  |  |  |  |  |  | | |  |  | |  | | --- | | b) | | | | |  | | --- | | Writing DAX expressions | |  |  | | |  |  |  |  |  |  |  |  | | |  |  | |  | | --- | | c) | | | | |  | | --- | | Data preparation and transformation | |  |  | | |  |  |  |  |  |  |  |  | | |  |  | |  | | --- | | d) | | | | |  | | --- | | Configuring report themes | |  |  | | |  |  |  |  |  |  |  |  | | | | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  | |  | |  | | --- | | 2) | | |  | |  | | --- | | In Power BI, what does the term "Query Folding" refer to? | | | |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | a) | | | | |  | | --- | | A technique for folding visuals in reports | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | b) | | | | |  | | --- | | A method for optimizing data transformation by pushing some operations back to the data source | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | c) | | | | |  | | --- | | A feature for grouping data in tables | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | d) | | | | |  | | --- | | An advanced filter option in visualizations | |  |  | |  |  |  |  |  |  |  |  | | | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  | |  | |  | | --- | | 3) | | |  | |  | | --- | | How does the "Power BI Dataflows" feature differ from traditional Power Query transformations in Power BI Desktop? | | | |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | a) | | | | |  | | --- | | Dataflows support real-time data connectivity | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | b) | | | | |  | | --- | | Dataflows are reusable, shareable entities that enable data transformation at scale | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | c) | | | | |  | | --- | | Power Query is only available in the Power BI Service | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | d) | | | | |  | | --- | | Power Query can only be used for data loading | |  |  | |  |  |  |  |  |  |  |  | | | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  | |  | |  | | --- | | 7) | | |  | |  | | --- | | What is the purpose of the "Smart Narratives" feature in Power BI? | | | |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | a) | | | | |  | | --- | | Designing custom visuals | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | b) | | | | |  | | --- | | Creating calculated columns | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | c) | | | | |  | | --- | | Configuring data refresh intervals | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | d) | | | | |  | | --- | | Generating natural language descriptions of data insights | |  |  | |  |  |  |  |  |  |  |  | | | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  | |  | |  | | --- | | 10) | | |  | |  | | --- | | Which visualization type in Power BI is suitable for displaying hierarchical data with collapsible and expandable nodes? | | | |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | a) | | | | |  | | --- | | Pie Chart | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | b) | | | | |  | | --- | | Treemap | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | c) | | | | |  | | --- | | Funnel Chart | |  |  | |  |  |  |  |  |  |  |  | |  |  | |  | | --- | | d) | | | | |  | | --- | | Scatter Plot | |  |  | |  |  |  |  |  |  |  |  | | | | | | | |  |
|  |  | | | | | | | |  |
|  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  | |  | |  | | --- | | 75) | |  | |  | | --- | | DirectQuery enables users to import data into Power BI for faster performance. | | | | | |  | | |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | | --- | | [True] | |  | |  | | --- | | [False] | |  |  | |  |  |  |  |  |  |  |  |  | | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  | |  | |  | | --- | | 79) | |  | |  | | --- | | Power BI allows users to create custom measures using only DAX functions. | | | | | |  | |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | | --- | | [True] | |  | |  | | --- | | [False] | |  |  | |  |  |  |  |  |  |  |  |  | | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  | |  | |  | | --- | | 78) | |  | |  | | --- | | Power BI Embedded is a feature that allows developers to integrate Power BI reports and dashboards into custom applications. | | | | | |  | |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | | --- | | [True] | |  | |  | | --- | | [False] | |  |  | |  |  |  |  |  |  |  |  |  | | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  | |  | |  | | --- | | 77) | |  | |  | | --- | | The "Data Profiling" feature in Power BI helps users identify and fix data quality issues in their datasets. | | | | | |  | |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | | --- | | [True] | |  | |  | | --- | | [False] | |  |  | |  |  |  |  |  |  |  |  |  | | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  | |  | |  | | --- | | 74) | |  | |  | | --- | | The "Relative Date Slicer" in Power BI allows users to filter data dynamically based on the current date. | | | | | |  | |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | | --- | | [True] | |  | |  | | --- | | [False] | |  |  | |  |  |  |  |  |  |  |  |  | | | | | | | | | |  |
|  | * Report that takes exam number and the student ID then returns   the Questions in this exam with the student answers.   |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  |  |  |  | |  |  |  |  |  | |  | |  | | --- | | **Exam\_Answer** | |  |  | |  |  |  |  | |  |  |  |  |  | | | | | | | | |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Qestion ID** | **QUESTION** | **ST ANSWER** | **GRADE** |
| Student ID : 4001 | | Student Name : Abdelhak Boukortt |  |  |
|  | 75 | DirectQuery enables users to import data into Power BI for faster performance. | [False] | 10 |
|  | 3 | How does the "Power BI Dataflows" feature differ from traditional Power Query transformations in Power BI Desktop? | "Dataflows are reusable, shareable entities that enable data transformation at scale" | 10 |
|  | 2 | In Power BI, what does the term "Query Folding" refer to? | A method for optimizing data transformation by pushing some operations back | 10 |
|  | 79 | Power BI allows users to create custom measures using only DAX functions. | [True] | 10 |
|  | 78 | Power BI Embedded is a feature that allows developers to integrate Power BI reports and dashboards into custom applications. | [True] | 10 |
|  | 77 | The "Data Profiling" feature in Power BI helps users identify and fix data quality issues in their datasets. | [True] | 10 |
|  | 74 | The "Relative Date Slicer" in Power BI allows users to filter data dynamically based on the current date. | [True] | 10 |
|  | 1 | What is the purpose of the "Power Query Editor" in Power BI? | Data preparation and transformation | 10 |
|  | 7 | What is the purpose of the "Smart Narratives" feature in Power BI? | Generating natural language descriptions of data insights | 10 |
|  | 10 | Which visualization type in Power BI is suitable for displaying hierarchical data with collapsible and expandable nodes? | Treemap | 10 |

* Report that takes the student ID and returns the grades of the student in all courses. %

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | |  |  |  |  |
|  | |  |  |  |  |
|  | | |  | | --- | | Student Grades | |  |  |
|  | |  |  |  |
|  |  | | | | | |  |
|  | |  |  |  | | --- | --- | --- | |  |  |  | | **Student Name : Younes Al-Naas** | | | | **Data Visualization (Smart Village)** | | | |  |  |  | | **Course Name** | **GRADE** | **STATUS** | | Data Warehousing & BI Concepts | 73 | Good | | Data storytelling & Data visualization | 78 | V.Good | | Introduction to Oracle SQL and PL/SQL | 76 | V.Good | | Oracle Advanced PL/SQL | 86 | Excellent | | Analytical SQL | 87 | Excellent | | Python | 62 | Fair | | Database Fundamentals | 60 | Fair | | Data Warehouse | 71 | Good | | Big Data | 73 | Good | | Operating Systems Fundamentals | 72 | Good | | Software Testing Fundamentals | 93 | Excellent | | Computer Network Fundamentals | 63 | Fair | | Communication Skills | 66 | Good | | Interviewing Skills | 95 | Excellent | | Presentation Skills | 90 | Excellent | | Freelancing | 79 | V.Good | | | | | | |  |

* Report that returns the students information according to Department No parameter.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  | |  | | --- | | **Students Information** | |  |  |
|  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Department Name** | | **Power BI Developer** | **Branch** | | **Menofia** | | |
| **ID** | **NAME** | **ADDRESS** | **CITY** | **BIRTHDATE** | **AGE** | **GENDER** | **INTAKE** |
| 4001 | Abdelhak Boukortt | 8 El Shorouk City | Zagazig | 10/5/1997 | 27 | M | 44 |
| 4021 | Abdulrahman Al-Farsi | 50 Minya Street | Minya | 1/25/1998 | 26 | M | 44 |
| 4041 | Ahmad Al-Halabi | 35. El Nahr St. | Sohag | 4/28/1999 | 25 | M | 44 |
| 4061 | Aicha Ben Dhia | 22 Abbassiya Road | Cairo | 8/8/2000 | 24 | F | 44 |
| 4081 | Ali Al-Jundi | 79. El Wefaq St. | Sohag | 3/19/1998 | 26 | M | 44 |
| 4101 | Amina Al-Mulla | 16. El Bahr St. | Asyut | 4/9/1999 | 25 | F | 44 |
| 4121 | Amir Chaouch | 84. El Azhar Street | Sohag | 7/19/1997 | 27 | M | 44 |
| 4141 | Asma El-Mansuri | 38 Tanta Road | Monofia | 8/12/1999 | 25 | F | 44 |
| 4161 | Bilal Al-Fitouri | 31 Quesna Road | Monofia | 7/20/1999 | 25 | M | 44 |
| 4181 | Dina Al-Awad | 46. El Mohandessin Street | Sohag | 5/25/1998 | 26 | F | 44 |
| 4201 | Fatemah Al-Attiyah | 310 Sohag Boulevard | Sohag | 8/27/1999 | 25 | F | 44 |
| 4221 | Fatima Al-Harbi | 98. El Obour City | Tanta | 10/23/1999 | 25 | F | 44 |
| 4241 | Habiba Yahya | 7 Al Tahrir Square | Damietta | 11/7/1997 | 27 | F | 44 |
| 4261 | Hamza Al-Khalaf | 150 Giza Avenue | Giza | 3/10/1997 | 27 | M | 44 |
| 4281 | Hassan Al-Masri | 83. El Koshary St. | Beni Suef | 3/23/1998 | 26 | M | 44 |
| 4301 | Hichem Krichene | 12 Borg El Arab Road | Alexandria | 5/29/1999 | 25 | M | 44 |
| 4321 | Huda Abdullah | 21 Shibin El Kom Road | Monofia | 7/17/1999 | 25 | F | 44 |
| 4341 | Ines Zwai | 31 El Horreya Street | Port Said | 11/26/1998 | 26 | F | 44 |
| 4361 | Karim Hassan | 10 Al Tahrir St | Cairo | 7/14/1998 | 26 | M | 44 |
| 4381 | Khaled Salih | 24 El Jazeera Street | Assiut | 12/9/1998 | 26 | F | 44 |
| 4401 | Khaled Al-Bassam | 385 Port Said Lane | Port Said | 1/12/2000 | 24 | M | 44 |
| 4421 | Laila Al-Dossari | 26. El Nasr Street | Luxor | 11/20/1999 | 25 | F | 44 |
| 4441 | Layla Al-Mansoori | 50. El Amal St. | Sohag | 6/13/1998 | 26 | F | 44 |
| 4461 | Lina Saleh | 5 Nile Corniche | Port Said | 11/10/1995 | 29 | F | 44 |
| 4481 | Mahmoud Al-Hassan | 89. El Kowther St. | Sohag | 3/29/1998 | 26 | M | 44 |
| 4501 | Mehdi Farid | 1 El Agouza Street | Mahalla El Kubra | 5/27/2000 | 24 | M | 44 |
| 4521 | Mohamed Kridi | 15 Talaat Harb Street | Monofia | 7/10/1999 | 25 | M | 44 |
| 4541 | Mona Al-Khatib | 48. El Nada St. | Beni Suef | 2/16/1998 | 26 | F | 43 |
| 4561 | Mustafa Al-Mansoori | 57. El Masry St. | Assiut | 2/25/1998 | 26 | M | 43 |
| 4581 | Nadia Al-Mulla | 95 Banha Street | Banha | 5/19/1997 | 27 | F | 43 |
| 4601 | Nahla Al-Qubaisi | 500 Suez Lane | Suez | 3/24/1999 | 25 | F | 43 |
| 4621 | Nawel Shukri | 20 Kaser Street | Giza | 11/14/1998 | 26 | F | 43 |
| 4641 | Nour Al-Fadil | 72. El Agouza Street | Mahalla El Kubra | 6/20/1998 | 26 | F | 43 |
| 4661 | Noureddine Saif al-Islam | 13 El Kaser Street | Assiut | 11/7/1998 | 26 | M | 43 |
| 4681 | Omar Al-Masri | 20 Luxor Road | Luxor | 2/12/1997 | 27 | M | 43 |
| 4701 | Rahaf Al-Khatib | 14. Al Haram Street | Kafr El Sheikh | 4/23/1998 | 26 | F | 43 |
| 4721 | Rana Al-Masri | 88. El Shorouk City | Zagazig | 7/6/1998 | 26 | F | 43 |
| 4741 | Rashid Al-Hashemi | 33. El Gomhoria Square | Assiut | 11/27/1999 | 25 | M | 43 |
| 4761 | Rim Ismail | 20 El Nasr City | Cairo | 6/3/2000 | 24 | F | 43 |
| 4781 | Roula Al-Zaid | 47. El Nakhil Street | Asyut | 12/11/1999 | 25 | F | 43 |
| 4801 | Sabrina Mehenni | 12 Al Haram Street | Kafr El Sheikh | 9/24/1997 | 27 | F | 43 |
| 4821 | Safia Al-Mutlaq | 22. Al Tahrir Square | Damietta | 5/1/1998 | 26 | F | 43 |
| 4841 | Said Al-Zentani | 46 Shebin El Koum Square | Monofia | 8/5/1999 | 25 | M | 43 |
| 4861 | Sami Shebani | 19 El Tahrir Street | Giza | 11/13/1998 | 26 | M | 43 |
| 4881 | Sara Ahmed | 49 Nasr City Road | Cairo | 7/13/2000 | 24 | F | 43 |
| 4901 | Sarah Al-Jarrah | 10. Abdel Hamid Badawy Street | Al Minya | 11/4/1999 | 25 | F | 43 |
| 4921 | Sultan Al-Jumaily | 31. El Hegaz Street | Fayoum | 11/25/1999 | 25 | M | 43 |
| 4941 | Tariq Al-Dawood | 455 New Cairo Corniche | New Cairo | 3/15/1999 | 25 | M | 43 |
| 4961 | Walid Trabelsi | 9 El Salam Street | Port Said | 11/19/1998 | 26 | M | 43 |
| 4981 | Yasmin Al-Khalil | 38. El Maadi Road | Banha | 5/17/1998 | 26 | F | 43 |
| 5001 | Younes Othman | 55 El Obour City | Tanta | 11/2/1997 | 27 | M | 43 |
| 5021 | Youssef Abdelrahman | 12 El Mohandessin Street | Sohag | 10/17/1997 | 27 | M | 43 |
| 5041 | Ziad Al-Asiri | 190 Ismailia Street | Ismailia | 3/18/1997 | 27 | M | 43 |

* Report that takes the instructor ID and returns the name of the courses that he teaches and the number of student per course.

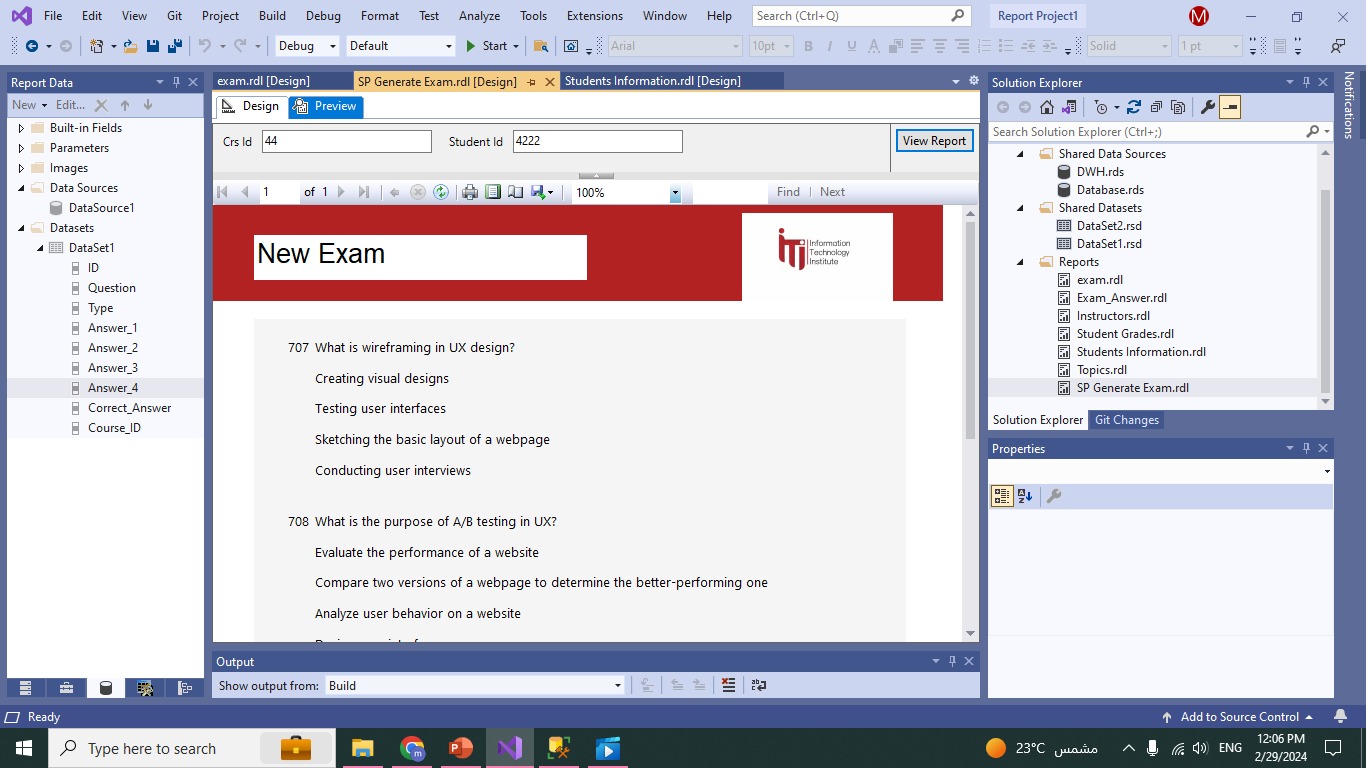
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  | |  | | --- | | **Instructors** | |  |  |
|  |  |  |  |
|  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  | |  |  |  |  | | --- | --- | --- | --- | | **ID** | **NAME** | **Course** | **Student Number** | | 1 | Ahmed Ali | Power BI | 210 | |  |
|  |  |  |

* Report that takes course ID and returns its topics

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  | |  | | --- | | **Topics** | |  |  |
|  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **Course** | **TOPIC ID** | **TOPIC NAME** |
| Computer Network Fundamentals |  |  |
|  | 211 | Introduction to Computer Networks |
|  | 212 | Network Models and Layered Architectures |
|  | 213 | OSI Model and TCP/IP Protocol Suite |
|  | 214 | Ethernet and LAN Technologies |
|  | 215 | Wireless Networking |
|  | 216 | Network Devices and Infrastructure |
|  | 217 | IP Addressing and Subnetting |
|  | 218 | Routing and Switching |
|  | 219 | Network Security and Cryptography |
|  | 220 | Network Management and Performance Optimization |

****

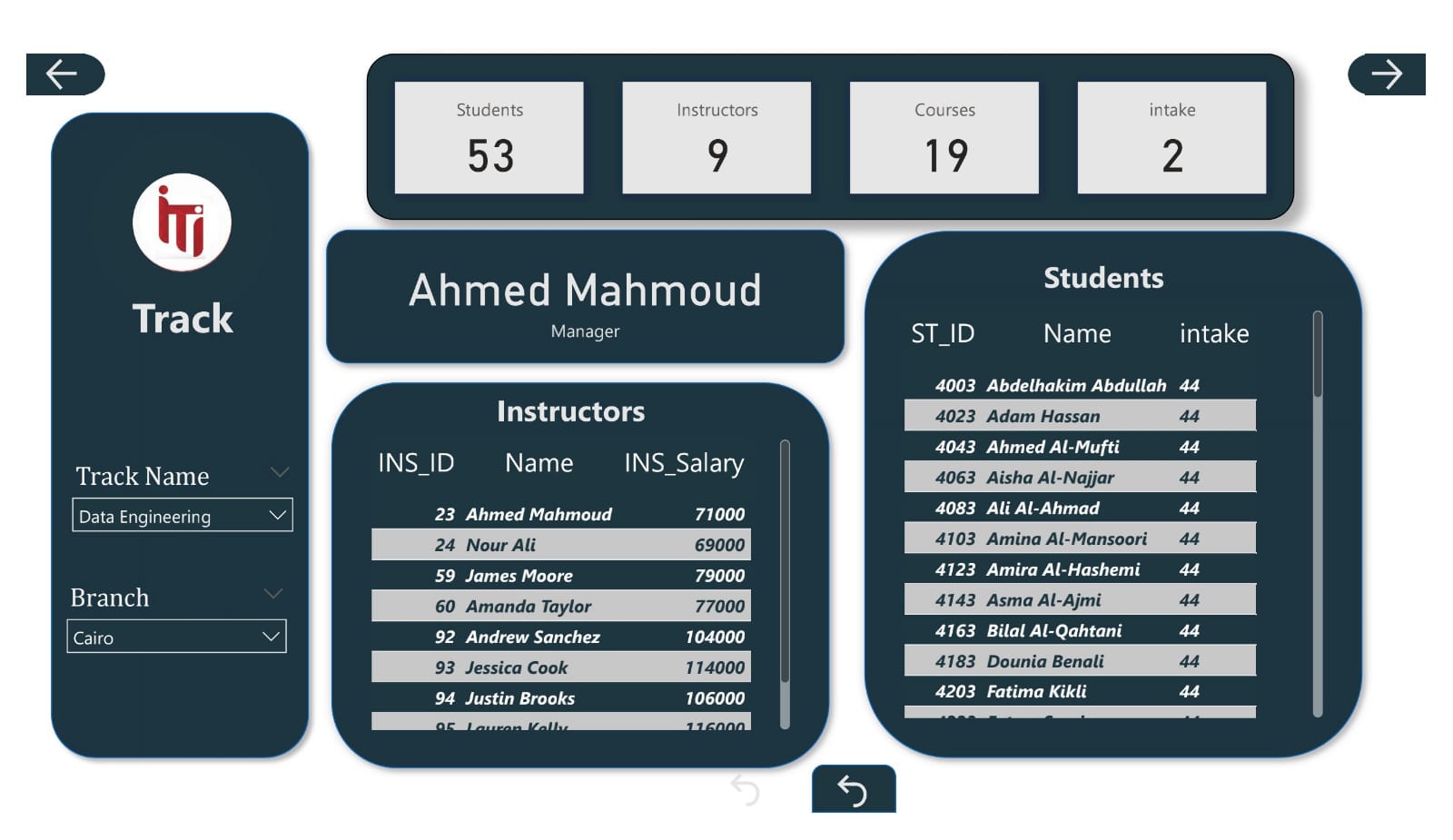
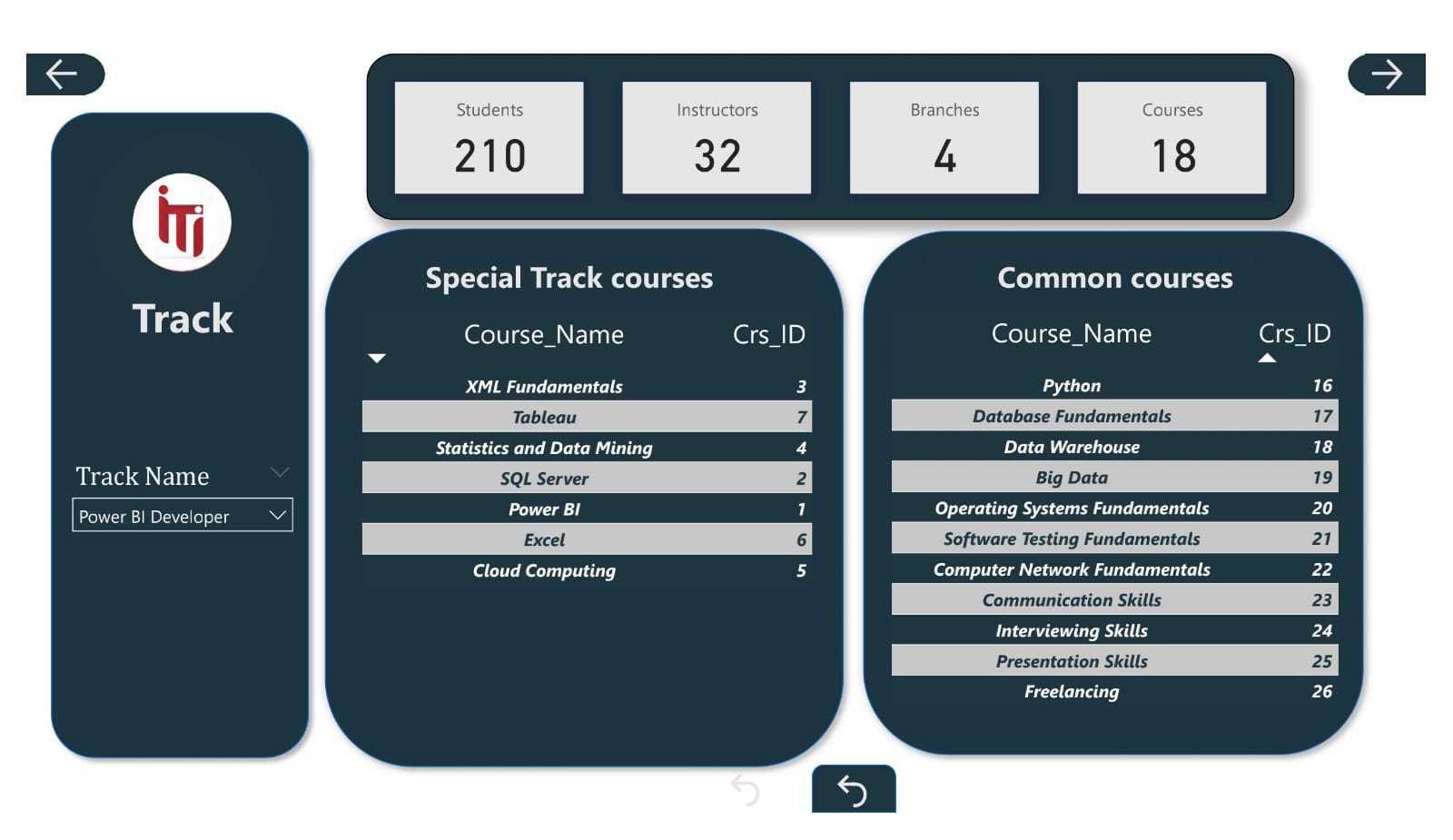
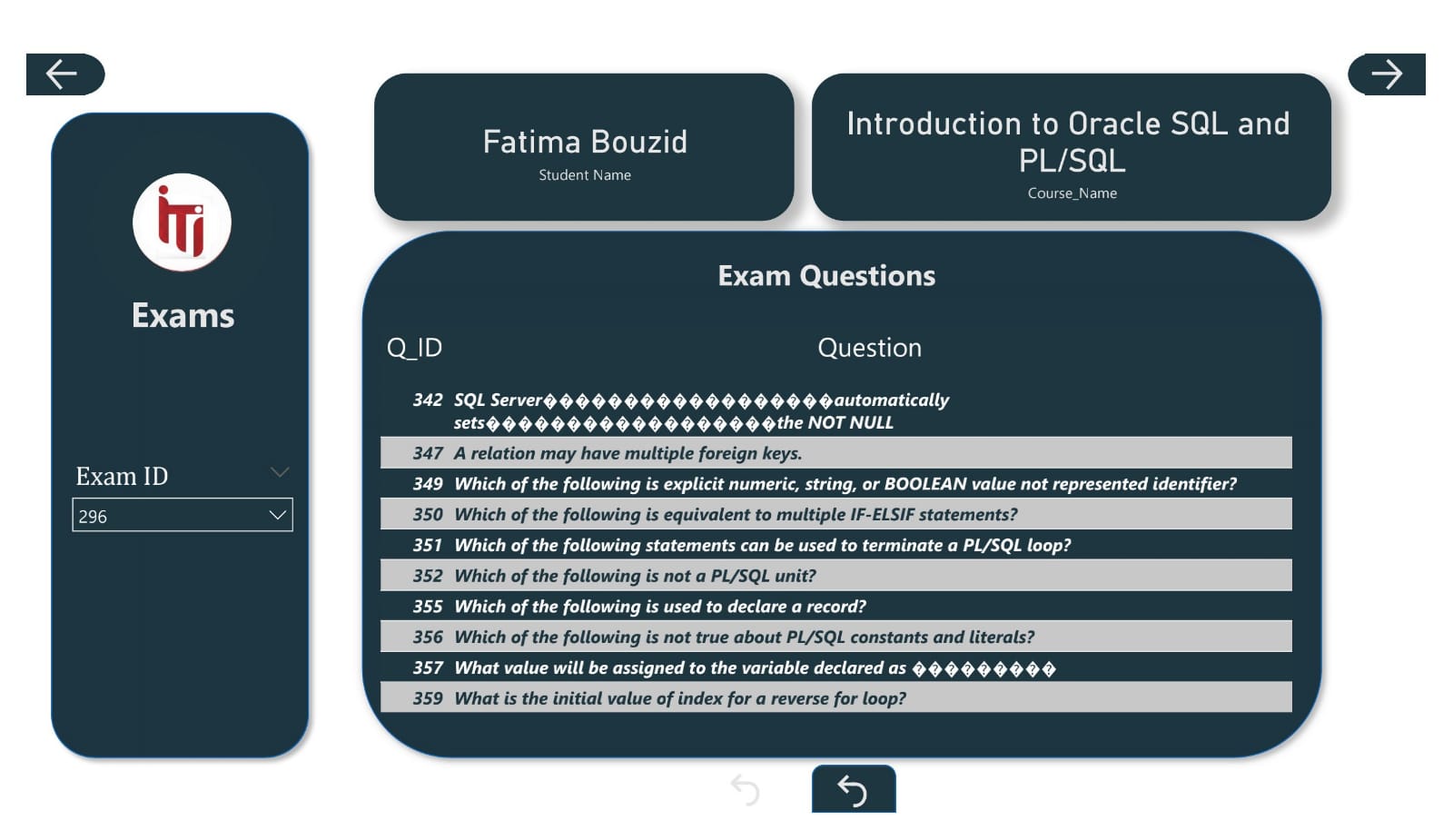
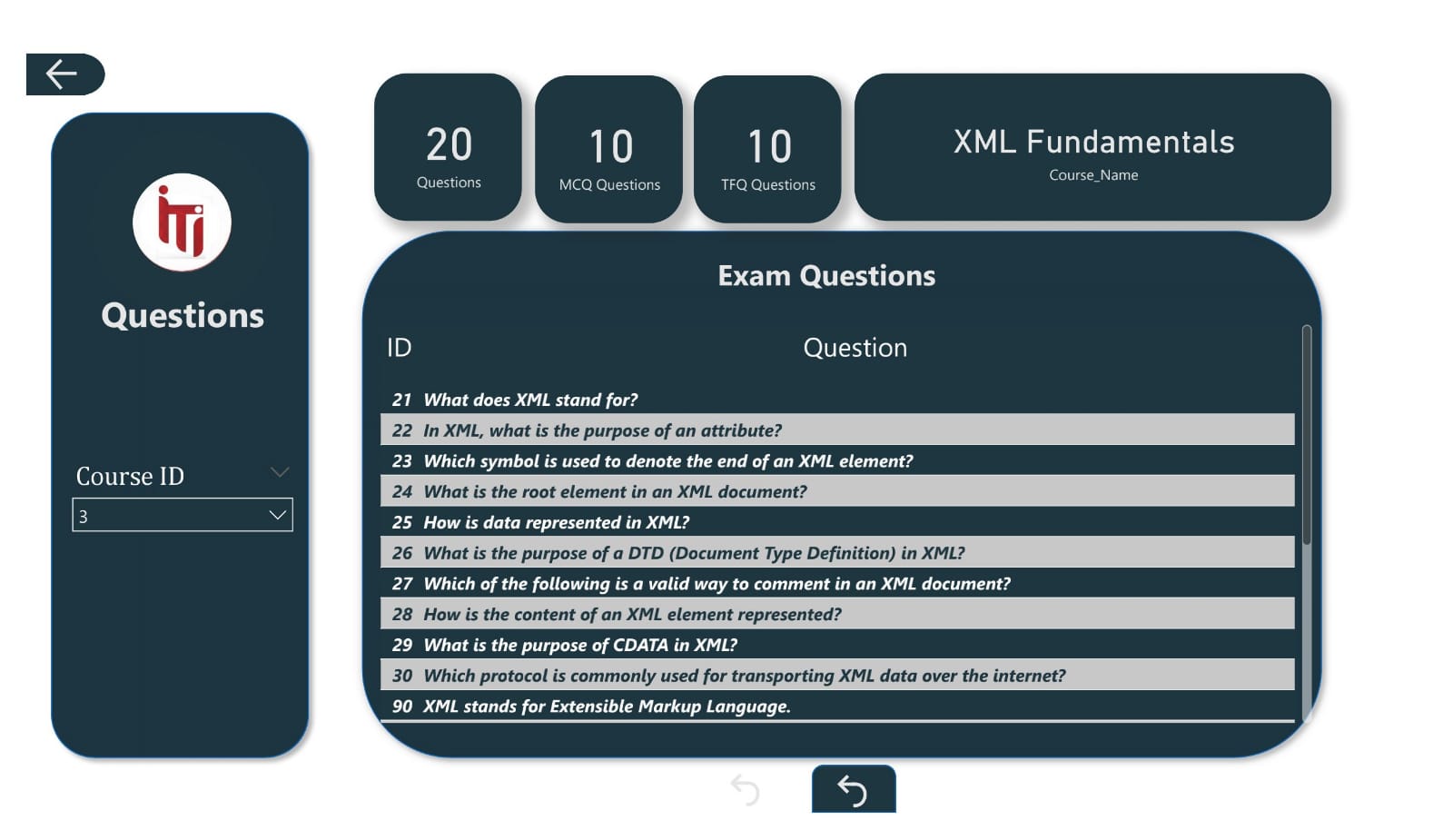
**POWER BI DASHBOARDS**

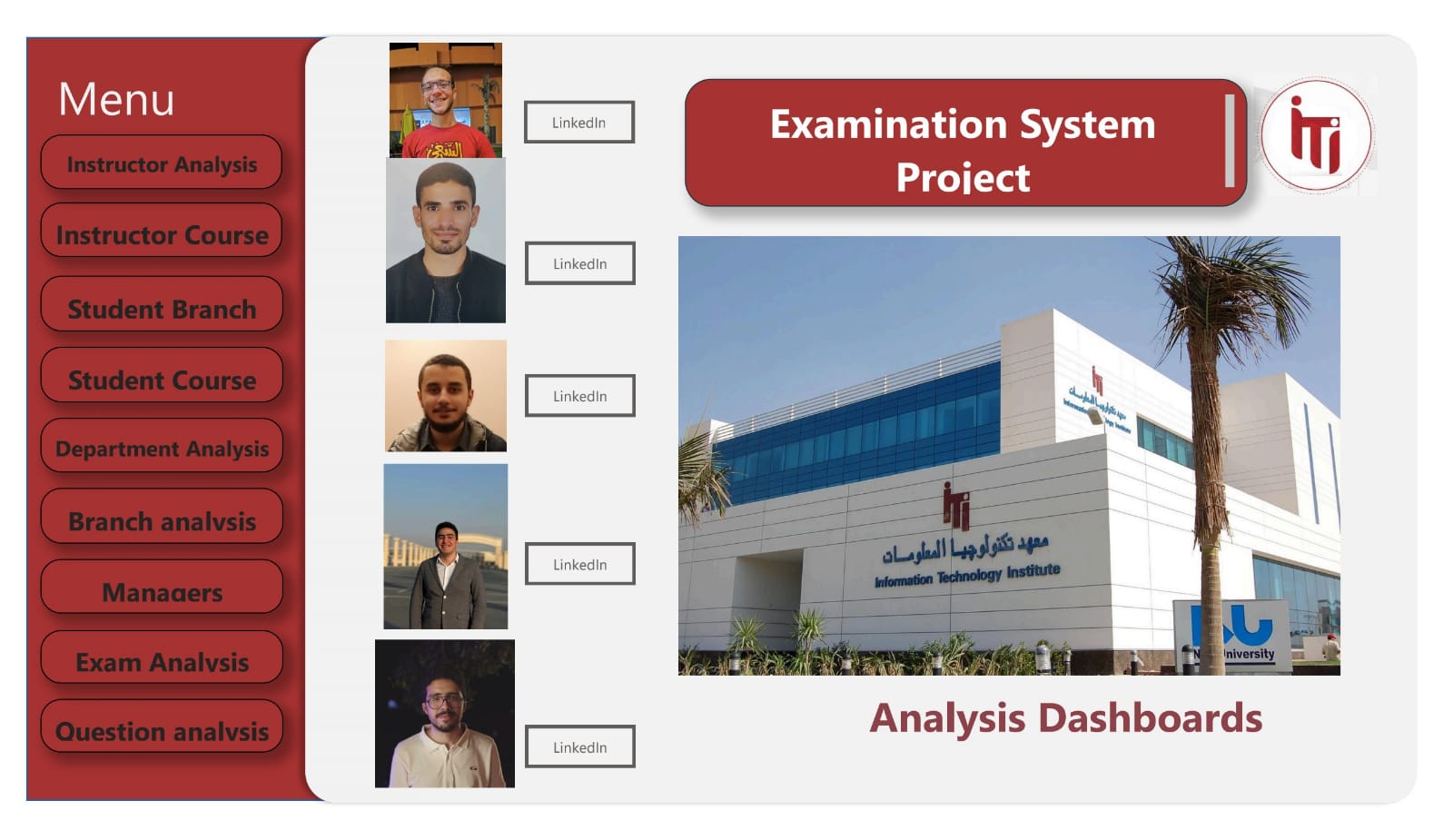
****

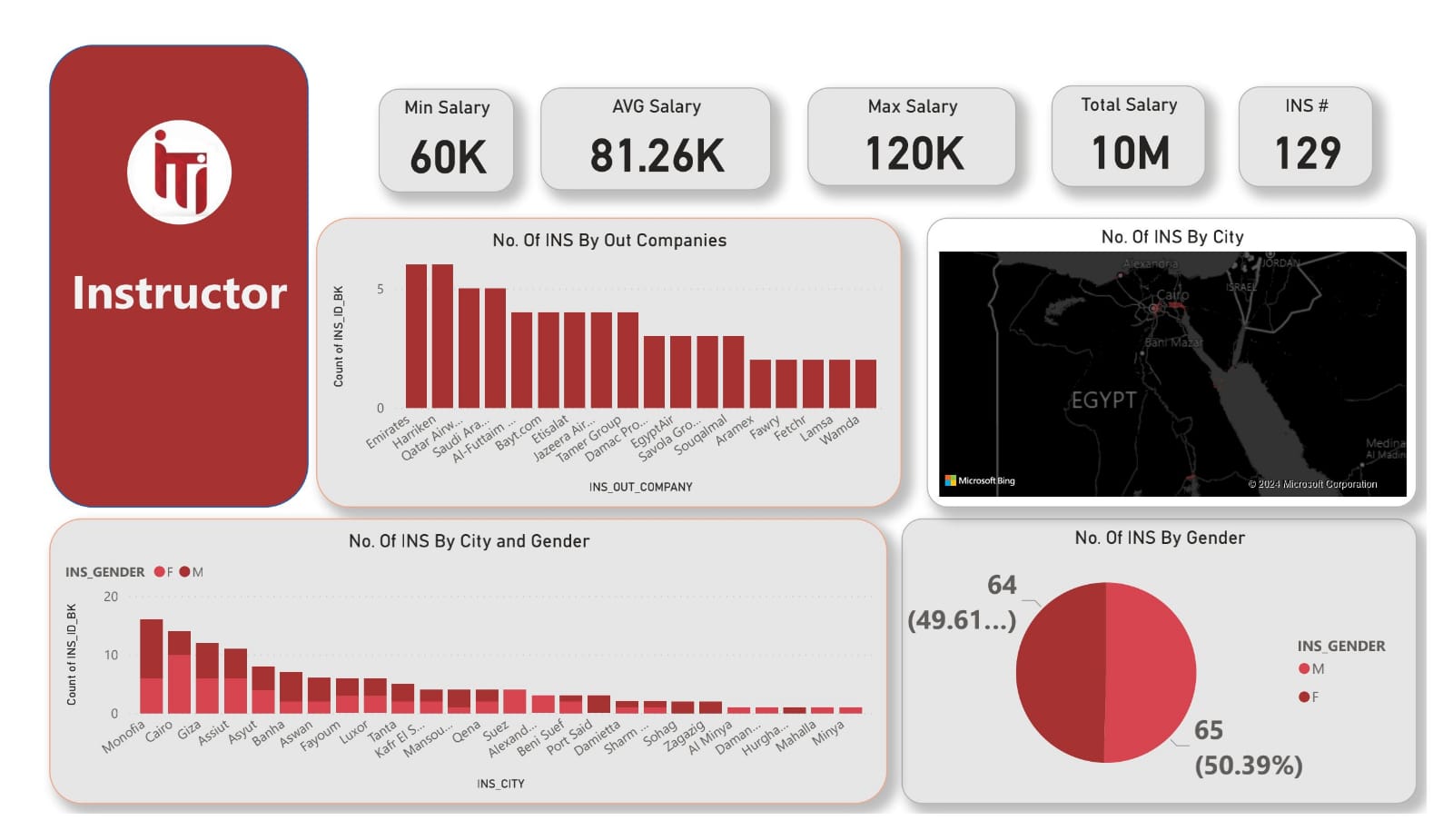
****

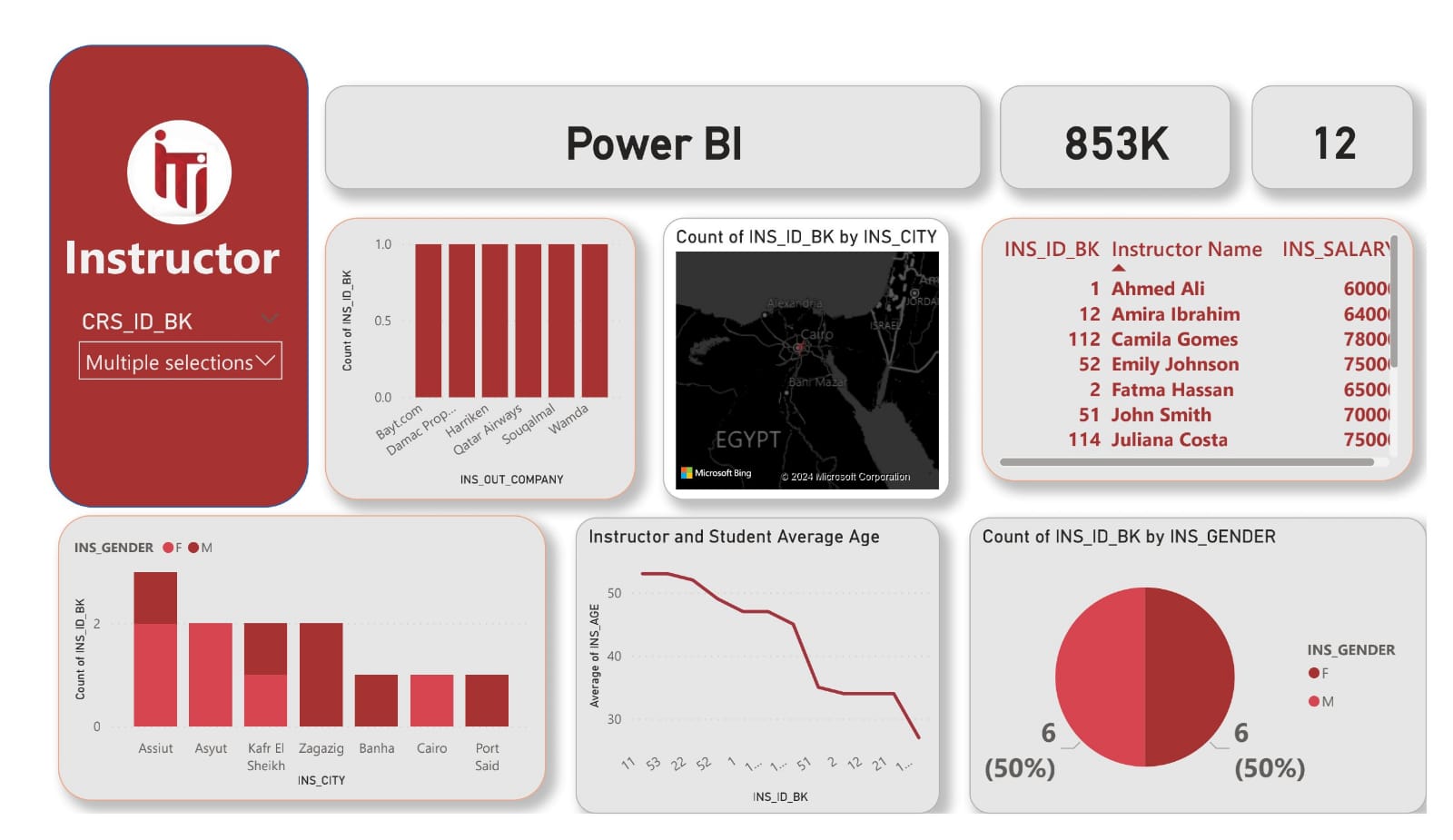
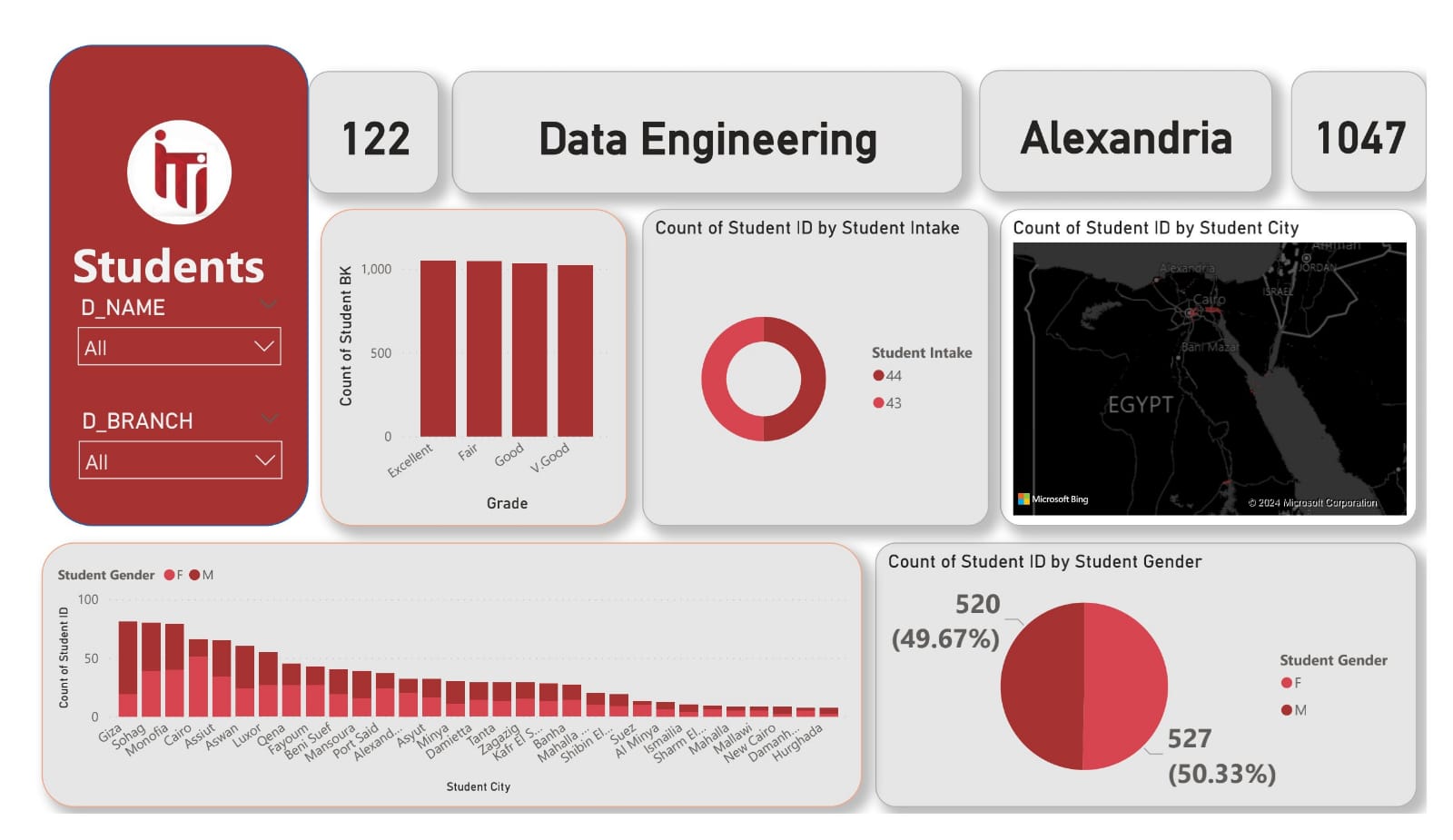
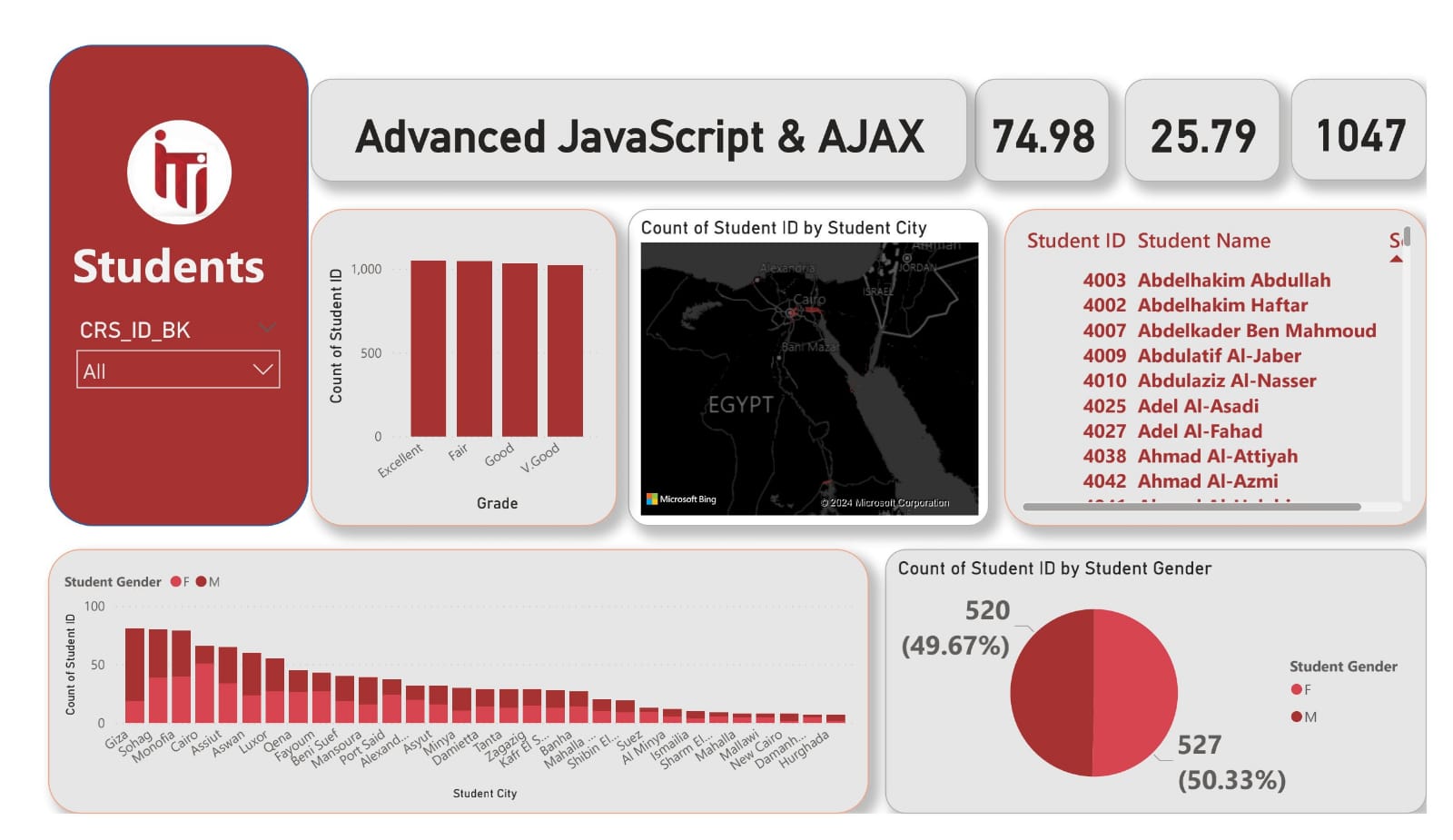
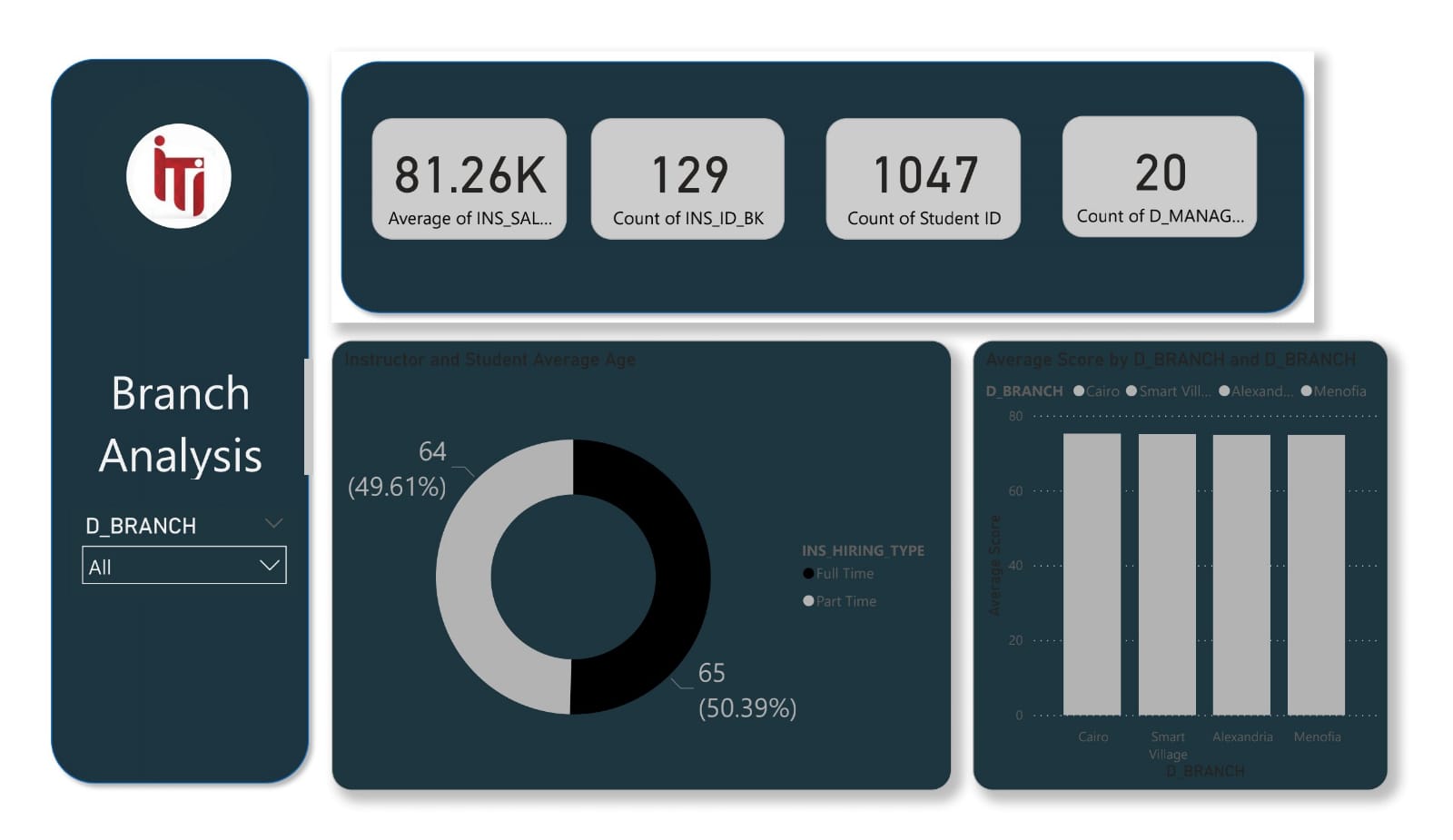
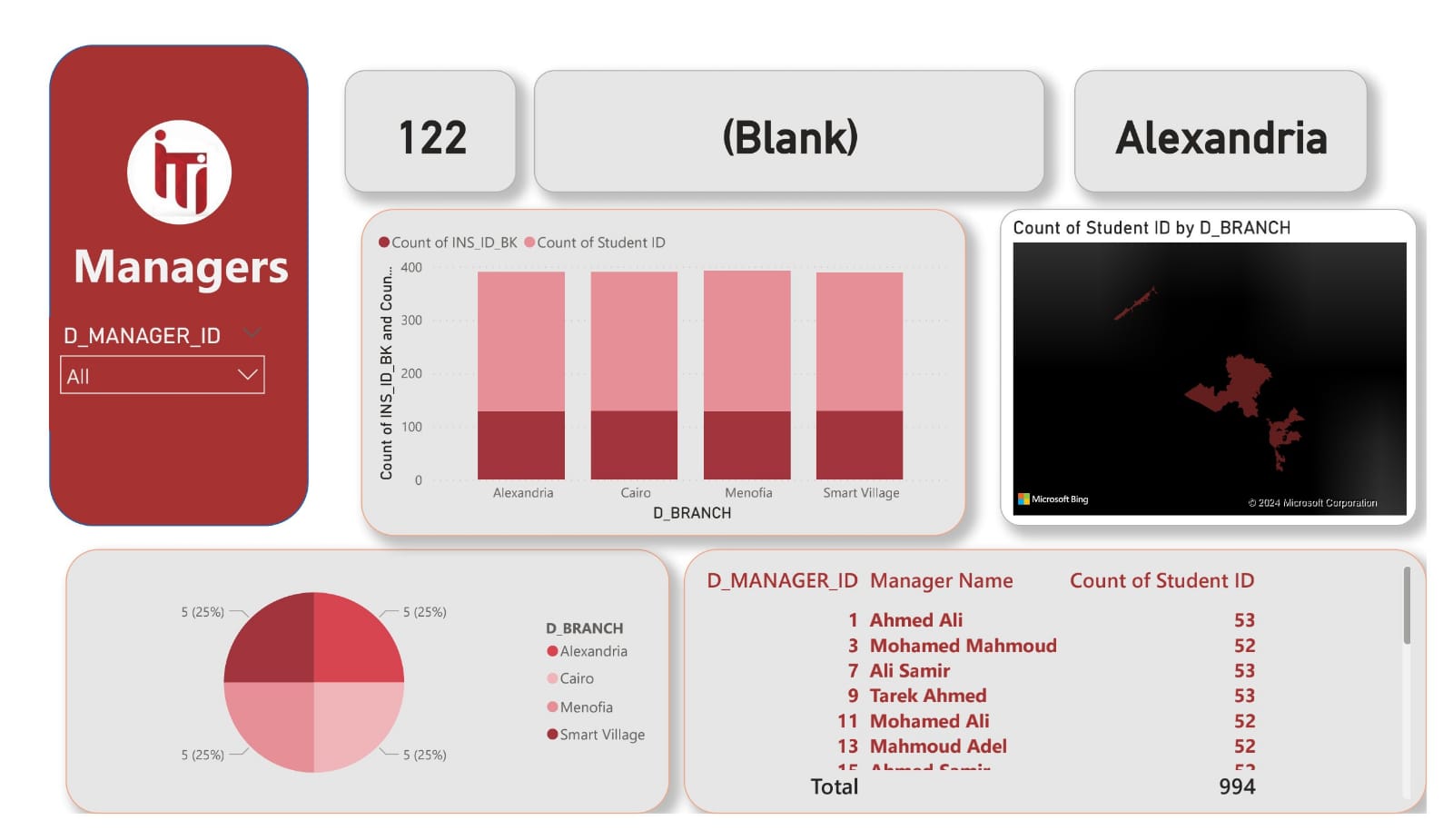
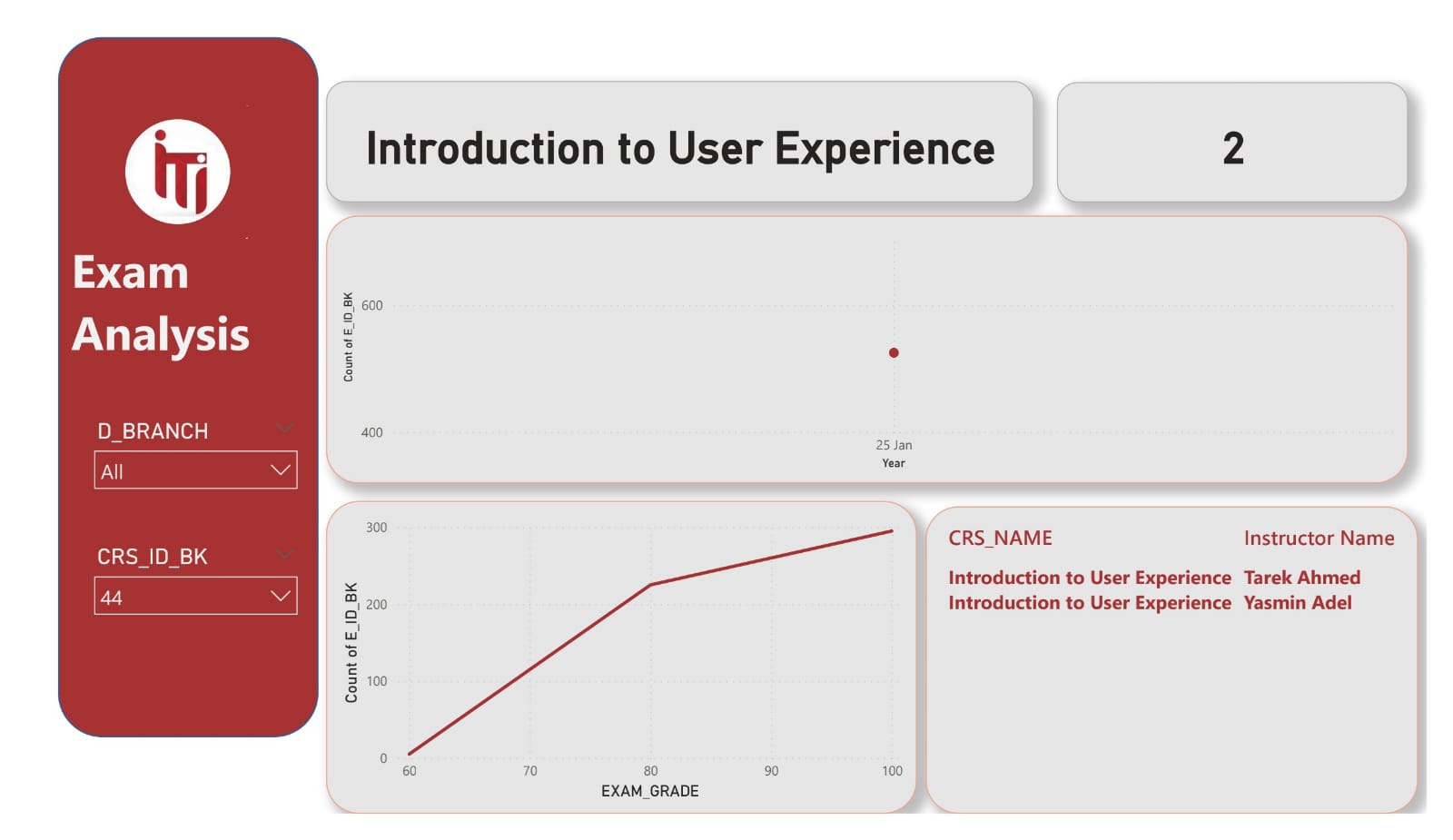
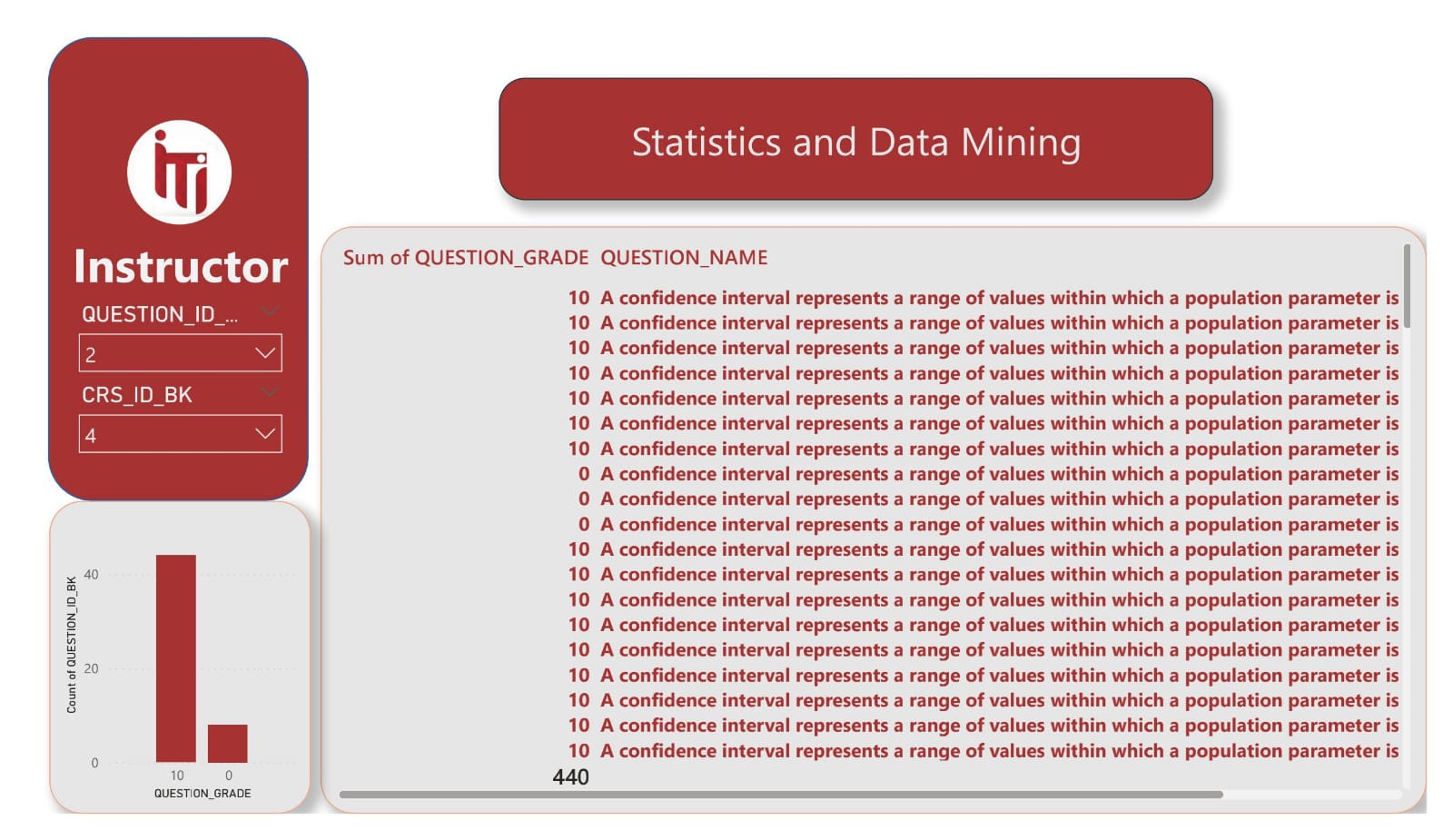
****

****

****

****

****

****

1. **Desktop Application**

The Application based on the Database generates an exam with 5 questions and corrects them then shows the exam grade , the student can log in by his email and password then he can insert the course id which he has the exam on .

The Application code was written in python as following .

import tkinter

from tkinter import ttk

from tkinter import \*

from tkinter import messagebox

import pyodbc

import tkinter as tk

from tkinter import StringVar

index=0

correct=0

def login():

    connection = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};'

                            'Server=DESKTOP-DP443HJ;'

                            'Database=ITI Final;'

                            'Trusted\_Connection=yes;')

# Creating a cursor object

    cursor = connection.cursor()

    code = ("SELECT \* FROM student WHERE ST\_Email = ? AND ST\_PassWord = ?")

    cursor.execute(code , ((EntEmail.get()), Entpass.get()))

    reuslt = cursor.fetchall()

    connection.commit()

    connection.close()

    if reuslt:

        messagebox.showinfo('','Login worked Well ')

        frm.destroy()

        Course\_frm = tk.Tk()

        Course\_frm.geometry('500x500')

        Course\_frm.iconbitmap('G:\ITI\BI-Data\Final Project\iti.ico')

        Course\_frm.title( 'Course Page ')

        Ent\_Crs\_llb =ttk.Label(Course\_frm, text='Course\_ID',foreground='#57a1f8',background='white', font=('Arail',24,'bold'))

        Ent\_Crs\_llb.pack()

        Ent\_Crs=IntVar()

        Ent\_Crs=ttk.Entry(Course\_frm,width=25,foreground='black' ,background= 'White' , font=('Arail',11),textvariable=Ent\_Crs)

        Ent\_Crs.pack()

        Ent\_Crs.get()

        def Exam\_Crs():

                root = tk.Tk()

                root.geometry('500x500')

                root.iconbitmap('G:\ITI\BI-Data\Final Project\iti.ico')

                root.title( 'Exam Page ')

                #QUE1()

                Entque1=ttk.Entry(root,width=70,foreground='black',background= 'White' , font=('Arail',15))

                Answer\_1\_Ent1 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_2\_Ent1 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_3\_Ent1 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_4\_Ent1 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Correct\_Answer\_Ent1 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                connection = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};'

                                                'Server=DESKTOP-DP443HJ;'

                                                'Database=ITI Final;'

                                                'Trusted\_Connection=yes;')

                mycursor = connection.cursor()

                Q1=("select Top 1  Question   , Answer\_1 , Answer\_2 , Answer\_3 ,Answer\_4  ,Correct\_Answer  FROM dbo.Question ORDER BY NEWID()")

                mycursor.execute(Q1)

                records = mycursor.fetchall()

                for  i ,  (Question , Answer\_1 , Answer\_2, Answer\_3  , Answer\_4 , Correct\_Answer) in enumerate(records ,start=1):

                    Entque1.insert("", (Question))

                    Answer\_1\_Ent1.insert("",str(Answer\_1))

                    Answer\_2\_Ent1.insert("",str(Answer\_2) )

                    Answer\_3\_Ent1.insert("",str(Answer\_3) )

                    Answer\_4\_Ent1.insert("",str(Answer\_4) )

                    Correct\_Answer\_Ent1.insert("",str(Correct\_Answer) )

                    connection.commit()

                #Que2

                Entque2=ttk.Entry(root,width=70,foreground='black',background= 'White' , font=('Arail',15))

                Answer\_1\_Ent2 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_2\_Ent2 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_3\_Ent2 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_4\_Ent2 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Correct\_Answer\_Ent2 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                connection = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};'

                                                'Server=DESKTOP-DP443HJ;'

                                                'Database=ITI Final;'

                                                'Trusted\_Connection=yes;')

                mycursor = connection.cursor()

                Q2=(("select  Top 1  Question  , Answer\_1 , Answer\_2 , Answer\_3 ,Answer\_4  ,Correct\_Answer  FROM dbo.Question ORDER BY NEWID()"))

                mycursor.execute(Q2)

                records2 = mycursor.fetchall()

                for  i ,  (Question2 , Answer\_1\_2 , Answer\_2\_2, Answer\_3\_2  , Answer\_4\_2 , Correct\_Answer\_2) in enumerate(records2 ,start=1):

                    Entque2.insert("", Question2)

                    Answer\_1\_Ent2.insert("",str(Answer\_1\_2))

                    Answer\_2\_Ent2.insert("",str(Answer\_2\_2) )

                    Answer\_3\_Ent2.insert("",str(Answer\_3\_2) )

                    Answer\_4\_Ent2.insert("",str(Answer\_4\_2) )

                    Correct\_Answer\_Ent2.insert("",str(Correct\_Answer\_2) )

                connection.commit()

                connection.close()

                #Que3

                Entque3=ttk.Entry(root,width=70,foreground='black',background= 'White' , font=('Arail',15))

                Answer\_1\_Ent3 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_2\_Ent3 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_3\_Ent3 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_4\_Ent3 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Correct\_Answer\_Ent3 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                connection = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};'

                                                'Server=DESKTOP-DP443HJ;'

                                                'Database=ITI Final;'

                                                'Trusted\_Connection=yes;')

                mycursor = connection.cursor()

                Q3=(("select Top 1 Question  , Answer\_1 , Answer\_2 , Answer\_3 ,Answer\_4  ,Correct\_Answer  FROM dbo.Question ORDER BY NEWID()"))

                mycursor.execute(Q3)

                records3 = mycursor.fetchall()

                for  i ,  (Question3 , Answer\_1\_3 , Answer\_2\_3, Answer\_3\_3  , Answer\_4\_3 , Correct\_Answer\_3) in enumerate(records3 ,start=1):

                    Entque3.insert("", Question3)

                    Answer\_1\_Ent3.insert("",str(Answer\_1\_3))

                    Answer\_2\_Ent3.insert("",str(Answer\_2\_3) )

                    Answer\_3\_Ent3.insert("",str(Answer\_3\_3) )

                    Answer\_4\_Ent3.insert("",str(Answer\_4\_3) )

                    Correct\_Answer\_Ent3.insert("",str(Correct\_Answer\_3) )

                connection.commit()

                connection.close()

                #Que4

                Entque4=ttk.Entry(root,width=70,foreground='black',background= 'White' , font=('Arail',15))

                Answer\_1\_Ent4 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_2\_Ent4 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_3\_Ent4 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_4\_Ent4 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Correct\_Answer\_Ent4 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                connection = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};'

                                                'Server=DESKTOP-DP443HJ;'

                                                'Database=ITI Final;'

                                                'Trusted\_Connection=yes;')

                mycursor = connection.cursor()

                Q4=(("select Top 1 Question  , Answer\_1 , Answer\_2 , Answer\_3 ,Answer\_4  ,Correct\_Answer  FROM dbo.Question ORDER BY NEWID()"))

                mycursor.execute(Q4)

                records4 = mycursor.fetchall()

                for  i ,  (Question4 , Answer\_1\_4 , Answer\_2\_4, Answer\_3\_4  , Answer\_4\_4 , Correct\_Answer\_4) in enumerate(records4 ,start=1):

                    Entque4.insert("", Question4)

                    Answer\_1\_Ent4.insert("",str(Answer\_1\_4))

                    Answer\_2\_Ent4.insert("",str(Answer\_2\_4) )

                    Answer\_3\_Ent4.insert("",str(Answer\_3\_4) )

                    Answer\_4\_Ent4.insert("",str(Answer\_4\_4) )

                    Correct\_Answer\_Ent4.insert("",str(Correct\_Answer\_4) )

                connection.commit()

                connection.close()

                #Que5

                Entque5=ttk.Entry(root,width=70,foreground='black',background= 'White' , font=('Arail',15))

                Answer\_1\_Ent5 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_2\_Ent5 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_3\_Ent5 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Answer\_4\_Ent5 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                Correct\_Answer\_Ent5 = ttk.Entry(root, width=50,foreground='black' ,background= 'White' ,font=('Arail', 15))

                connection = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};'

                                                'Server=DESKTOP-DP443HJ;'

                                                'Database=ITI Final;'

                                                'Trusted\_Connection=yes;')

                mycursor = connection.cursor()

                Q5=(("select Top 1 Question  , Answer\_1 , Answer\_2 , Answer\_3 ,Answer\_4  ,Correct\_Answer  FROM dbo.Question ORDER BY NEWID()"))

                mycursor.execute(Q5)

                records5 = mycursor.fetchall()

                for  i ,  (Question5 , Answer\_1\_5 , Answer\_2\_5, Answer\_3\_5  , Answer\_4\_5 , Correct\_Answer\_5) in enumerate(records5 ,start=1):

                    Entque5.insert("", Question5)

                    Answer\_1\_Ent5.insert("",str(Answer\_1\_5))

                    Answer\_2\_Ent5.insert("",str(Answer\_2\_5) )

                    Answer\_3\_Ent5.insert("",str(Answer\_3\_5) )

                    Answer\_4\_Ent5.insert("",str(Answer\_4\_5) )

                    Correct\_Answer\_Ent5.insert("",str(Correct\_Answer\_5) )

                connection.commit()

                connection.close()

                #ADD More Questions if you want

                #Create Questions for the students

                questions = [Question,Question2,Question3,Question4,Question5]

                options = [[str(Answer\_1),str(Answer\_2),str(Answer\_3),str(Answer\_4),str(Correct\_Answer)],

                        [str(Answer\_1\_2),str(Answer\_2\_2),str(Answer\_3\_2),str(Answer\_4\_2),Correct\_Answer\_2],

                        [str(Answer\_1\_3),str(Answer\_2\_3),str(Answer\_3\_3),str(Answer\_4\_3),Correct\_Answer\_3],

                        [str(Answer\_1\_4),str(Answer\_2\_4),str(Answer\_3\_4),str(Answer\_4\_4),Correct\_Answer\_4],

                        [str(Answer\_1\_5),str(Answer\_2\_5),str(Answer\_3\_5),str(Answer\_4\_5),Correct\_Answer\_5]]

                frame = tk.Frame(root, padx=10, pady=10,bg='#fff')

                question\_label = tk.Label(frame,height=5, width=50,bg='grey',fg="#fff",

                                        font=('Verdana', 20),wraplength=500)

                v1 = StringVar(frame)

                v2 = StringVar(frame)

                v3 = StringVar(frame)

                v4 = StringVar(frame)

                option1 = tk.Radiobutton(frame, bg="#fff", variable=v1, font=('Verdana', 20),

                                        command = lambda : checkAnswer(option1))

                option2 = tk.Radiobutton(frame, bg="#fff", variable=v2, font=('Verdana', 20),

                                        command = lambda : checkAnswer(option2))

                option3 = tk.Radiobutton(frame, bg="#fff", variable=v3, font=('Verdana', 20),

                                        command = lambda : checkAnswer(option3))

                option4 = tk.Radiobutton(frame, bg="#fff", variable=v4, font=('Verdana', 20),

                                        command = lambda : checkAnswer(option4))

                button\_next = tk.Button(frame, text='Next',bg='Orange', font=('Verdana', 20),

                                        command = lambda : displayNextQuestion())

                frame.pack(fill="both", expand="true")

                question\_label.grid(row=0, column=0)

                option1.grid(sticky= 'W', row=1, column=0)

                option2.grid(sticky= 'W', row=2, column=0)

                option3.grid(sticky= 'W', row=3, column=0)

                option4.grid(sticky= 'W', row=4, column=0)

                button\_next.grid(row=6, column=0)

                global index

                global correct

                # create a function to disable radiobuttons

                def disableButtons(state):

                    option1['state'] = state

                    option2['state'] = state

                    option3['state'] = state

                    option4['state'] = state

                # create a function to check the selected answer

                def checkAnswer(radio):

                    global correct, index

                    # the 4th item is the correct answer

                    # we will check the user selected answer with the 4th item

                    if radio['text'] == options[index][4]:

                        correct +=1

                    index +=1

                    disableButtons('disable')

                # create a function to display the next question

                def displayNextQuestion():

                    global index, correct

                    if button\_next['text'] == 'Restart The Quiz':

                        correct = 0

                        index = 0

                        question\_label['bg'] = 'grey'

                        button\_next['text'] = 'Next'

                    if index == len(options):

                        question\_label['text'] = str(correct) + " / " + str(len(options))

                        button\_next['text'] = 'Restart The Quiz'

                        connection = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};'

                                                'Server=DESKTOP-DP443HJ;'

                                                'Database=ITI Final;'

                                                'Trusted\_Connection=yes;')

                    # Creating a cursor object

                        cursor = connection.cursor()

                        Update = ("UPDATE grade SET Grade = ?  FROM grade JOIN Student ON grade.ST\_ID = Student.ST\_ID WHERE Student.ST\_Email = ? and [Crs\_ID] = ? "  )

                        cursor.execute(Update , ((correct/len(options))\*100 ), Email , Ent\_Crs.get() )

                        connection.commit()

                        connection.close()

                        if correct >= len(options)/2:

                            question\_label['bg'] = 'green'

                        else:

                                question\_label['bg'] = 'red'

                    else:

                            question\_label['text'] = questions[index]

                            disableButtons('normal')

                            opts = options[index]

                            option1['text'] = opts[0]

                            option2['text'] = opts[1]

                            option3['text'] = opts[2]

                            option4['text'] = opts[3]

                            v1.set(opts[0])

                            v2.set(opts[1])

                            v3.set(opts[2])

                            v4.set(opts[3])

                            if index == len(options) - 1:

                                button\_next['text'] = 'Check the Results'

                displayNextQuestion()

                root.mainloop()

        bt\_Crs=Button(Course\_frm , text = 'Enter' , command=lambda: [Exam\_Crs()]).pack(pady=10)

        Course\_frm.mainloop

        return True

    else:

        messagebox.showinfo('','Please Enter a vaild Email or Password')

        return False

frm=tkinter.Tk()

frm.title(' Exam Login')

frm.geometry('925x500+300+300')

frm.iconbitmap('G:\ITI\BI-Data\Final Project\iti.ico')

frm.config(background= '#FFF')

img=PhotoImage(file='download 2.png')

Label(frm,image=img,bg='white').place(x=50,y=50)

fram=Frame(frm,width=350,height=350,bg='white')

fram.place(x=480,y=70)

llb =ttk.Label(fram, text='Sign In',foreground='#57a1f8',background='white', font=('Arail',24,'bold'))

llb.place(x=100,y=5)

Svemail=StringVar()

EntEmail=ttk.Entry(fram,width=25,foreground='black' ,background= 'White' , font=('Arail',11),textvariable=Svemail)

EntEmail.place(x=30,y=80)

EntEmail.insert(0,'Abdelhak.Boukortt73@gmail.com')

Frame(fram,width=295,height=2,background='black').place(x=25,y=107)

Entpass=ttk.Entry(fram,width=25,foreground='black' , background= 'White' , font=('Arail',11))#,textvariable=SvPass )

Entpass.place(x=30,y=150)

Entpass.insert(0,'7019729139')

Email=EntEmail.get()

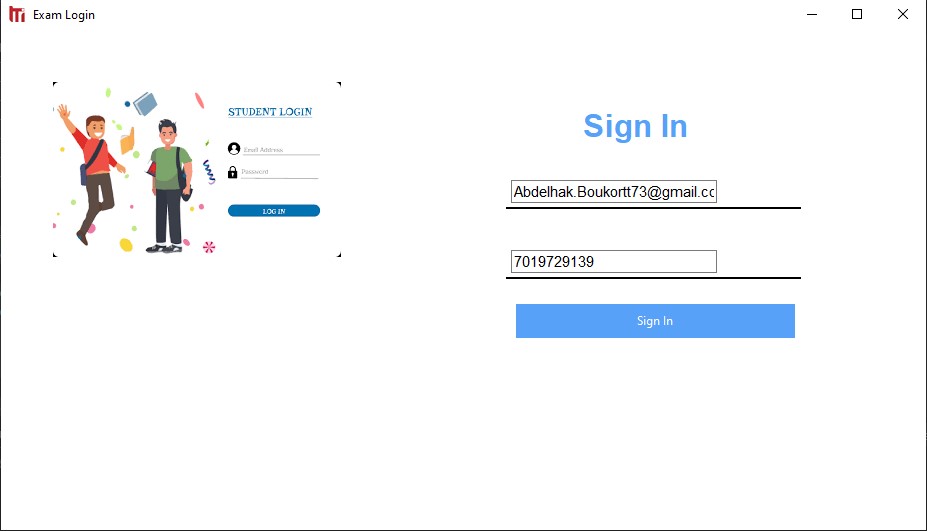
Pass=Entpass.get()

Frame(fram,width=295,height=2,background='black').place(x=25,y=177)

Button(fram,width=39,pady=7,background='#57a1f8',foreground='White',text='Sign In',border=0,command=lambda: [login()]).place(x=35,y=204)

frm.mainloop()

**Application Interface**

****

