

**INFORMATION TECHNOLOGY INSTITUTE**

**ITI Power bi developer track**

Examination System

Feb 2024



**Made by: -**

* Shimaa Abdelaal
* Rahma Tarek
* Esraa Emad
* Marwa Samir
* **Introduction:**

The project’s purpose is to design automated system that can perform online exams.

* **Database design:**

**Entities identifications: -**

1. **Students:**

Represents data about students, like **st\_id, st\_fname, st\_lname, st\_age, st\_address, st\_email.**

1. **Departments:**

Represents data about departments, like **dept\_id, dept\_name, dept\_location, mgr\_hiredate.**

1. **Instructors:**

Represents data about instructors, like **ins\_id, ins\_name, salary.**

1. **Courses:**

Represents data about courses, like **crs\_id, crs\_name, crs\_duration.**

1. **Topics:**

Represents data about topics, like **top\_id, top\_name.**

1. **Skills:**

Represents data about skills, like **sk\_id, sk\_name.**

1. **Projects:**

Represents data about projects, like **pro\_id, pro\_name, start\_date, end\_date.**

1. **Company:**

Represents data about company, like **company\_id, company\_name, location.**

1. **Position:**

Represents data about positions, like **position\_id, position\_name, salary.**

1. **Training:**

Represents data about training, like **training\_id, training\_name, start\_date, end\_date.**

1. **Freelancing:**

Represents data about freelancing, like **freelancing\_id, freelancing\_name.**

1. **Certificates:**

Represents data about certificates, like **cre\_id, cre\_name, issuer, issue\_date.**

1. **Exams:**

Represents data about exams, like **exam\_id, exam\_name, exam\_date.**

1. **Exam\_questions:**

Represents data about exam\_questions, like **question\_id, question\_text, question\_type.**

1. **Question\_option:**

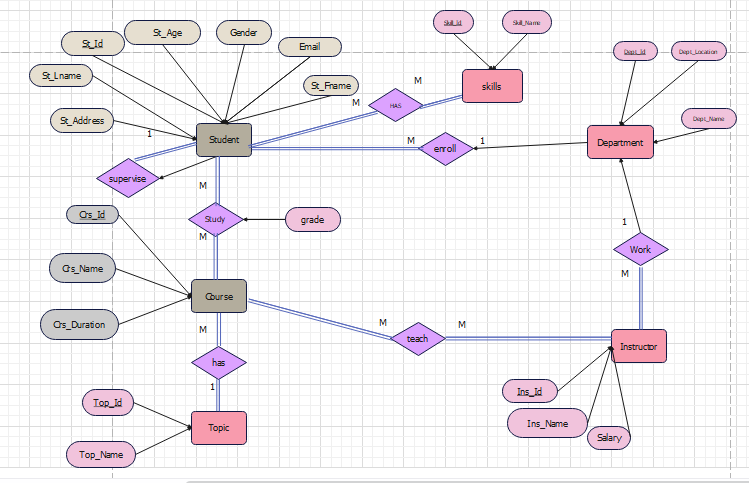
Represents data about question\_options, like **option\_id, option\_text, is\_currect.**

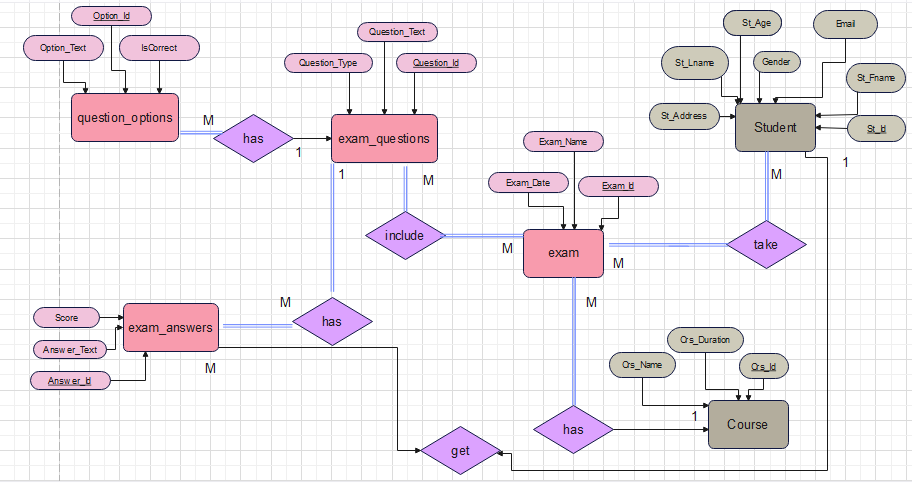
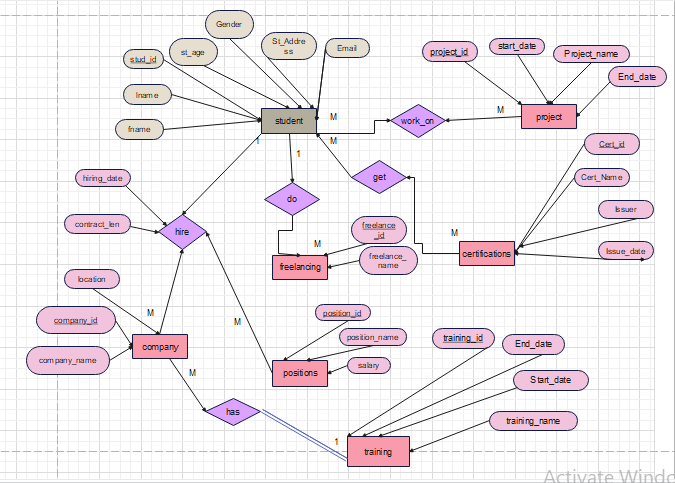
1. **Exam\_answers:**

Represents data about exam\_answers, like **answer\_id, answer\_text, score.**

**ERD\_diagram: -**

This diagram explains the relationships, the cardinality ratio and participations between entities.

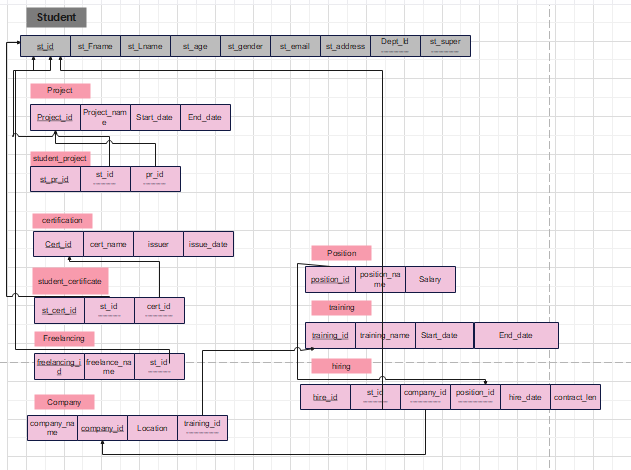
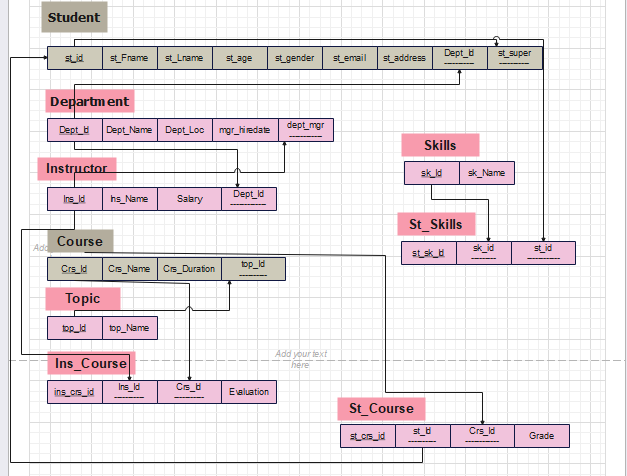


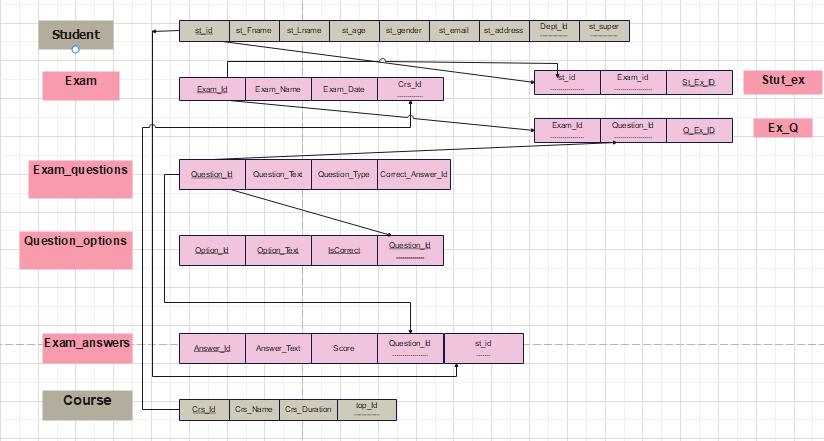


1. Each student must enroll in one department and each department may have many students.
2. Each instructor must work in many departments and each department may have many instructors.
3. Each student must study many courses and each course must being studied by many students with grade.
4. Each instructor must teach many courses and each course must being taught with many instructors.
5. Each course must have 1 topic but each topic must have many courses.
6. Each student must be under supervised by 1 supervisor but supervisor may supervise many students.
7. Each student must have many skills and each skill must be in many students.
8. Each student works on many projects and each project must have many students.
9. Each student may get many certificates and each certificate may be owned with many students.
10. Each student may do many freelancing jobs and each freelancing job may be done by many students.
11. Each company hiring with many positions with hiring date and contract length and many students may apply.
12. Each company may have 1 training program for every position and each training program must be in many companies.
13. Each student must take many exams and each exam must be taken with many students.
14. Each exam must be for 1 course and each course may have many exams.
15. Each exam must have many questions and each question must be in many exams.
16. Each exam may have many options and each option must be in 1 question.
17. Each question must have many answers and each answer must have 1 question.
18. Each student may have many answers but each answer to 1 student.

**Mapping: -**

convert the conceptual design to logical design, and represent the relationships.





**Physical Model:**

The creation of the tables using SQL queries.

1. **Student:**

CREATE TABLE [dbo].[Student](

[St\_Id] int IDENTITY(1,1) NOT NULL,

[St\_Fname] varchar(100),

[St\_Lname] varchar(100),

[St\_Address] varchar(100),

[St\_Age] int,

[Email] varchar(100),

[Dept\_Id] int,

[St\_super] int,

[Gender] varchar(50),

CONSTRAINT [FK\_Student\_Department] FOREIGN KEY([Dept\_Id]) REFERENCES [dbo].[Department] ([Dept\_Id]),

CONSTRAINT [FK\_Student\_Student] FOREIGN KEY([St\_super]) REFERENCES [dbo].[Student] ([St\_Id]))

1. **Department:**

CREATE TABLE [dbo].[Department](

[Dept\_Id] int IDENTITY(1,1) NOT NULL,

[Dept\_Name] varchar(100),

[Dept\_Location] varchar(100),

[Dept\_Manager] int,

[Manager\_hiredate] date,

CONSTRAINT [FK\_Department\_Instructor] FOREIGN KEY([Dept\_Manager]) REFERENCES [dbo].[Instructor] ([Ins\_Id]))

1. **Instructor:**

CREATE TABLE [dbo].[Instructor](

[Ins\_Id] int IDENTITY(1,1) NOT NULL,

[Ins\_Name] varchar(100),

[Salary] int,

[Dept\_Id] int,

CONSTRAINT [FK\_Instructor\_Department] FOREIGN KEY([Dept\_Id]) REFERENCES [dbo].[Department] ([Dept\_Id]))

1. **Course:**

CREATE TABLE [dbo].[Course](

[Crs\_Id] int IDENTITY(1,1) NOT NULL,

[Crs\_Name] varchar(100),

[Crs\_Duration] int,

[Top\_Id] int,

CONSTRAINT [FK\_Course\_Topic] FOREIGN KEY([Top\_Id]) REFERENCES [dbo].[Topic] ([Top\_Id]))

1. **Topic:**

CREATE TABLE [dbo].[Topic](

[Top\_Id] int IDENTITY(1,1) NOT NULL,

[Top\_Name] varchar(100))

1. **Skills:**

CREATE TABLE [dbo].[Skills](

[Skill\_Id] int IDENTITY(1,1) NOT NULL,

[Skill\_Name] varchar(100))

1. **Student\_skills:**

CREATE TABLE [dbo].[Student\_skills](

[Sk\_St\_id] int IDENTITY(1,1) NOT NULL,

[Sk\_Id] int,

[St\_Id] int,

CONSTRAINT [FK\_Student\_skills\_Skills] FOREIGN KEY([Sk\_Id]) REFERENCES [dbo].[Skills] ([Skill\_Id]),

CONSTRAINT [FK\_Student\_skills\_Student] FOREIGN KEY([St\_Id]) REFERENCES [dbo].[Student] ([St\_Id]))

1. **Student\_course:**

CREATE TABLE [dbo].[Student\_course](

[Crs\_Id] int,

[St\_Id] int,

[Grade] int,

[crs\_st\_id] int IDENTITY(1,1) NOT NULL,

CONSTRAINT [FK\_Student\_course\_Course] FOREIGN KEY([Crs\_Id]) REFERENCES [dbo].[Course] ([Crs\_Id]),

CONSTRAINT [FK\_Student\_course\_Student] FOREIGN KEY([St\_Id]) REFERENCES [dbo].[Student] ([St\_Id]))

1. **Instructor\_course:**

CREATE TABLE [dbo].[Instructor\_course](

[Ins\_Crs\_ID] int IDENTITY(1,1) NOT NULL,

[Ins\_Id] int,

[Crs\_Id] int,

[Evaluation] varchar(100) NULL,

CONSTRAINT [FK\_Instructor\_course\_Instructor] FOREIGN KEY([Ins\_Id]) REFERENCES [dbo].[Instructor] ([Ins\_Id]),

CONSTRAINT [FK\_Instructor\_course\_Course] FOREIGN KEY([Crs\_Id]) REFERENCES [dbo].[Course] ([Crs\_Id]))

1. **Certificates:**

CREATE TABLE [dbo].[Certificates](

[Cert\_Id] int IDENTITY(1,1) NOT NULL,

[Cert\_Name] varchar(100),

[Issuer] varchar(100),

[Issue\_Date] date)

1. **Student\_Certificates:**

CREATE TABLE [dbo].[Student\_Certificates](

[St\_Cer\_ID] int IDENTITY(1,1) NOT NULL,

[St\_Id] int,

[Cert\_Id] int,

CONSTRAINT [FK\_Student\_Certificates\_Student] FOREIGN KEY([St\_Id]) REFERENCES [dbo].[Student] ([St\_Id]),

CONSTRAINT [FK\_Student\_Certificates\_Certificates] FOREIGN KEY([Cert\_Id]) REFERENCES [dbo].[Certificates] ([Cert\_Id]))

1. **Freelancing:**

CREATE TABLE [dbo].[Freelancing](

[Freelance\_Id] int IDENTITY(1,1) NOT NULL,

[Freelance\_Name] varchar(100) NULL,

[St\_Id] int NULL,

CONSTRAINT [FK\_Freelancing\_Student] FOREIGN KEY([St\_Id]) REFERENCES [dbo].[Student] ([St\_Id]))

1. **Companies:**

CREATE TABLE [dbo].[Companies](

[Company\_Id] int IDENTITY(1,1) NOT NULL,

[Company\_Name] varchar(100),

[Location] varchar(100),

[Training\_ID] int,

CONSTRAINT [FK\_Companies\_Training] FOREIGN KEY([Training\_ID]) REFERENCES [dbo].[Training] ([Train\_ID]))

1. **Positions:**

CREATE TABLE [dbo].[Positions](

[Position\_Id] int IDENTITY(1,1) NOT NULL,

[Position\_Name] varchar(100),

[Salary] int)

1. **Hiring:**

CREATE TABLE [dbo].[Hiring](

[Hiring\_Id] int IDENTITY(1,1) NOT NULL,

[Company\_Id] int,

[Position\_Id] int,

[St\_Id] int,

[Hire\_Date] date,

[ContractLength] int,

CONSTRAINT [FK\_Hiring\_Companies] FOREIGN KEY([Company\_Id]) REFERENCES [dbo].[Companies] ([Company\_Id]),

CONSTRAINT [FK\_Hiring\_Positions] FOREIGN KEY([Position\_Id]) REFERENCES [dbo].[Positions] ([Position\_Id]),

CONSTRAINT [FK\_Hiring\_Student] FOREIGN KEY([St\_Id]) REFERENCES [dbo].[Student] ([St\_Id]))

1. **Training:**

CREATE TABLE [dbo].[Training](

[Train\_ID] int IDENTITY(1,1) NOT NULL,

[Train\_Name] varchar(100),

[Start\_Date] date,

[End\_Date] date)

1. **Projects:**

CREATE TABLE [dbo].[Projects](

[Project\_Id] int IDENTITY(1,1) NOT NULL,

[Project\_Name] varchar(100),

[Start\_Date] date,

[End\_Date] date)

1. **Student\_Project:**

CREATE TABLE [dbo].[Student\_Project](

[St\_Pro\_Id] int IDENTITY(1,1) NOT NULL,

[St\_Id] int,

[Pro\_Id] int,

CONSTRAINT [FK\_Student\_Project\_Project] FOREIGN KEY([Pro\_Id]) REFERENCES [dbo].[Projects] ([Project\_Id]),

CONSTRAINT [FK\_Student\_Project\_Student] FOREIGN KEY([St\_Id]) REFERENCES [dbo].[Student] ([St\_Id]))

1. **Exam:**

CREATE TABLE [dbo].[Exam](

[Exam\_Id] int IDENTITY(1,1) NOT NULL,

[Exam\_Name] varchar](100),

[Exam\_Date] date,

[Crs\_Id] int,

CONSTRAINT [FK\_Exam\_Course] FOREIGN KEY(Crs\_Id]) REFERENCES [dbo].[Course] ([Crs\_Id]))

1. **St\_Exam:**

CREATE TABLE [dbo].[St\_Exam](

[St\_Ex\_Id] int IDENTITY(1,1) NOT NULL,

[St\_Id] int,

[Ex\_Id] int,

CONSTRAINT [FK\_Student\_Exam\_Student] FOREIGN KEY([St\_Id]) REFERENCES [dbo].[Student] ([St\_Id]),

CONSTRAINT [FK\_Student\_Exam\_Exam] FOREIGN KEY([Ex\_Id]) REFERENCES [dbo].[Exam] ([Exam\_Id]))

1. **Exam\_answers:**

CREATE TABLE [dbo].[Exam\_answers](

[Answer\_Id] int IDENTITY(1,1) NOT NULL,

[Answer\_Text] varchar(300),

[Question\_Id] int,

[Score] int,

[St\_Id] int,

CONSTRAINT [FK\_Exam\_answers\_Exam\_questions] FOREIGN KEY([Question\_Id]) REFERENCES [dbo].[Exam\_questions] ([Question\_Id]),

CONSTRAINT [FK\_Exam\_answers\_Student] FOREIGN KEY([St\_Id]) REFERENCES [dbo].[Student] ([St\_Id]))

1. **Exam\_questions:**

CREATE TABLE [dbo].[Exam\_questions](

[Question\_Id] int IDENTITY(1,1) NOT NULL,

[Question\_Text] varchar(200),

[Question\_Type] varchar(50),

[Correct\_Answer\_Id] int,

[Category\_Name] varchar(100),

CONSTRAINT [FK\_Exam\_questions\_Exam\_answers] FOREIGN KEY([Correct\_Answer\_Id]) REFERENCES [dbo].[Exam\_answers] ([Answer\_Id]))

1. **Ques\_Exam:**

CREATE TABLE [dbo].[Ques\_Exam](

[Q\_Ex\_Id] int IDENTITY(1,1) NOT NULL,

[Ex\_Id] int,

[Q\_Id] int,

CONSTRAINT [FK\_Ques\_Exam\_Exam] FOREIGN KEY([Ex\_Id]) REFERENCES [dbo].[Exam] ([Exam\_Id]),

CONSTRAINT [FK\_Ques\_Exam\_Exam\_questions] FOREIGN KEY([Q\_Id]) REFERENCES [dbo].[Exam\_questions] ([Question\_Id]))

1. **Ques\_Exam:**

CREATE TABLE [dbo].[Question\_options](

[Option\_Id] int IDENTITY(1,1) NOT NULL,

[Option\_Text] varchar(max),

[IsCorrect] int,

[Question\_Id] int,

[Category\_Name] varchar(100),

CONSTRAINT [FK\_Question\_options\_Exam\_questions] FOREIGN KEY([Question\_Id]) REFERENCES [dbo].[Exam\_questions] ([Question\_Id]))

**Reports using SSRS:**

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

CREATE PROCEDURE StudentInfo

@DeptId INT

AS

BEGIN

SET NOCOUNT ON;

SELECT S.St\_Id, S.St\_Fname, S.St\_Lname, S.St\_Address, S.St\_Age, S.Email, S.Gender

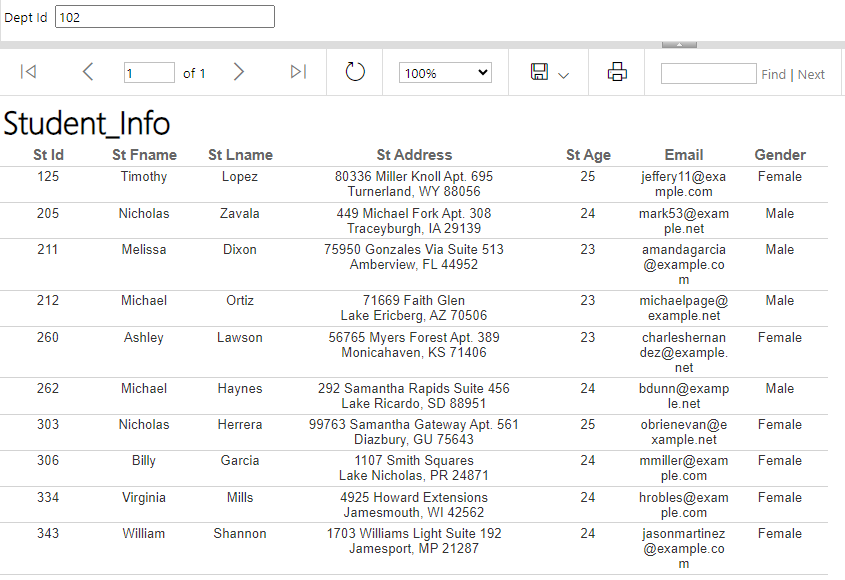
FROM Student S

INNER JOIN Department D ON S.Dept\_Id = D.Dept\_Id

WHERE D.Dept\_Id = @DeptId;

END;

StudentInfo @DeptId = 102



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

CREATE PROCEDURE GetStudentGrades

@StudentID INT

AS

BEGIN

SET NOCOUNT ON;

SELECT

C.Crs\_Name AS CourseName,

SC.Grade

FROM

Student\_course SC

INNER JOIN

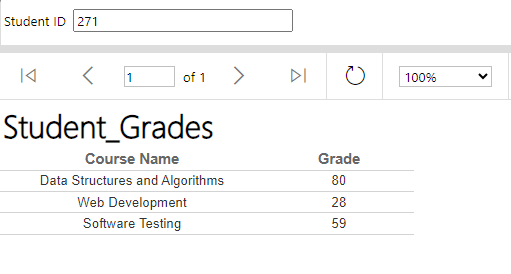
Course C ON SC.Crs\_Id = C.Crs\_Id

WHERE

SC.St\_Id = @StudentID;

END;

EXEC GetStudentGrades @StudentID = 271;



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

CREATE PROCEDURE GetrCoursesAndStudentsNum

@InstructorID INT

AS

BEGIN

SET NOCOUNT ON;

SELECT

C.Crs\_Name AS CourseName,

COUNT(SC.St\_Id) AS NumberOfStudents

FROM

Instructor\_course IC

INNER JOIN

Course C ON IC.Crs\_Id = C.Crs\_Id

LEFT JOIN

Student\_course SC ON IC.Crs\_Id = SC.Crs\_Id

WHERE

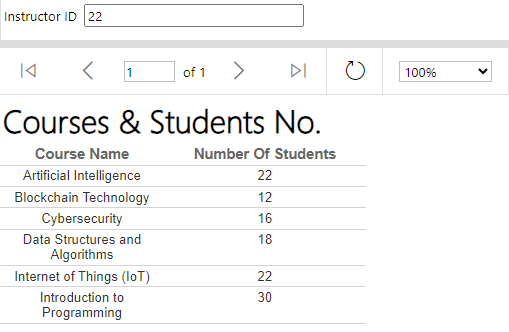
IC.Ins\_Id = @InstructorID

GROUP BY

C.Crs\_Name;

END;

GetrCoursesAndStudentsNum @InstructorID=22



1. Report that takes course ID and returns its topics

CREATE PROCEDURE GetCourseTopics

@CourseID INT

AS

BEGIN

SET NOCOUNT ON;

SELECT

T.Top\_Name AS TopicName

FROM

Course C

INNER JOIN

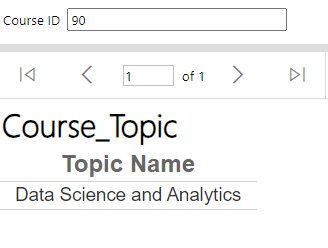
Topic T ON C.Top\_Id = T.Top\_Id

WHERE

C.Crs\_Id = @CourseID;

END;

EXEC GetCourseTopics @CourseID = 90;



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

create PROCEDURE GetExamQuestionsAndChoices

@ExamNumber INT

AS

BEGIN

SET NOCOUNT ON;

WITH CTE AS (

SELECT

CASE

WHEN ROW\_NUMBER() OVER (PARTITION BY EQ.Question\_Text ORDER BY (SELECT NULL)) = 1 THEN EQ.Question\_Text

ELSE ''

END AS Question\_Text,

QO.Option\_Text,

QO.IsCorrect

FROM

Exam E

INNER JOIN

Ques\_Exam QE ON E.Exam\_Id = QE.Ex\_Id

INNER JOIN

Exam\_questions EQ ON QE.Q\_Id = EQ.Question\_Id

LEFT JOIN

Question\_options QO ON EQ.Question\_Id = QO.Question\_Id

WHERE

E.Exam\_Id = @ExamNumber

)

SELECT

Question\_Text,

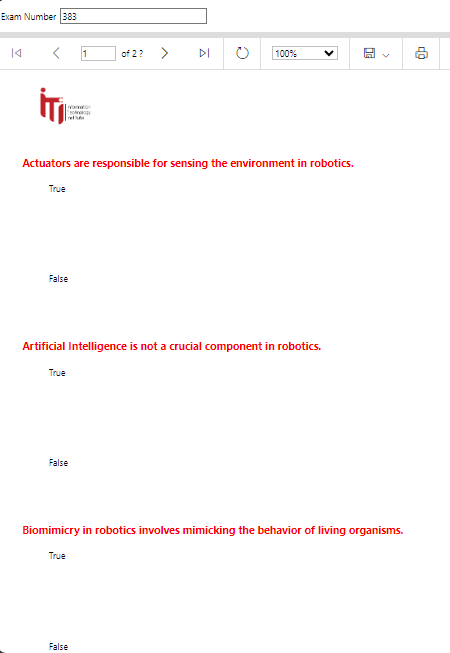
Option\_Text,

IsCorrect

FROM CTE;

END;

EXEC GetExamQuestionsAndChoices @ExamNumber =383;



1. Report that takes exam number and the student ID then returns the Questions in this exam with the student answers.

CREATE PROCEDURE GetExamQuestionsAndAnswers

@ExamId INT,

@StudentId INT

AS

BEGIN

SET NOCOUNT ON;

SELECT EQ.Question\_Text,

EA.Answer\_Text AS Student\_Answer,

EA.Score

FROM Exam E

INNER JOIN Ques\_Exam QE ON E.Exam\_Id = QE.Ex\_Id

INNER JOIN Exam\_questions EQ ON QE.Q\_Id = EQ.Question\_Id

LEFT JOIN Exam\_answers EA ON EQ.Question\_Id = EA.Question\_Id

LEFT JOIN Student S ON EA.St\_Id = S.St\_Id

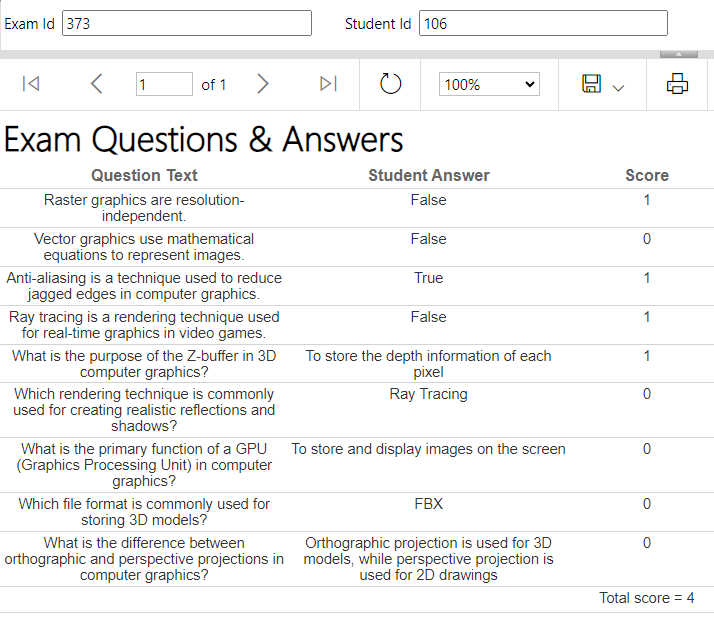
LEFT JOIN St\_Exam SE ON E.Exam\_Id = SE.Ex\_Id AND S.St\_Id = SE.St\_Id

WHERE E.Exam\_Id = @ExamId

AND S.St\_Id = @StudentId;

END;

EXEC GetExamQuestionsAndAnswers @ExamId = 373, @StudentId = 106;



**Data warehousing:**

Creation of dimension and fact tables.

1. **Dim\_Exam:**

This table holds information about exams. It includes details such as exam ID (both surrogate and business key), exam name, date, and related question and answer details.

CREATE TABLE [dbo].[Dim\_Exam](

[exam\_id\_sk] int IDENTITY(1,1) NOT NULL,

[exam\_id\_bk] int,

[exam\_name] varchar(100),

[exam\_date] date,

[question\_id\_bk] int,

[question\_text] varchar(200),

[question\_type] varchar(50),

[correct\_answer\_id] int,

[answer\_id\_bk] int,

[answer\_text] varchar(300),

[score] int,

[option\_id\_bk] int,

[option\_text] varchar(300),

[is\_correct] int,

[start\_date] datetime,

[end\_date] datetime,

[is\_current] tinyint,

CONSTRAINT [FK\_Dim\_Exam\_Fact\_table] FOREIGN KEY([exam\_id\_sk]) REFERENCES [dbo].[Fact\_table] ([exam\_id\_fk]))

1. **Dim\_Hiring:**

This table stores data related to hiring processes. It includes hiring ID (surrogate and business key), hire date, contract length, company details, training information, position details, salary, and temporal information.

CREATE TABLE [dbo].[Dim\_Hiring](

[hiring\_id\_sk] int IDENTITY(1,1) NOT NULL,

[hiring\_id\_bk] int,

[hire\_date] date,

[contract\_length] int,

[company\_id\_bk] int,

[company\_name] varchar(100),

[location] varchar(100),

[training\_id\_bk] int,

[training\_name] varchar(100),

[training\_startdate] date,

[training\_enddate] date,

[position\_id\_bk] int,

[position\_name] varchar(100),

[salary] int,

[start\_date] datetime,

[end\_date] datetime,

[is\_current] tinyint,

CONSTRAINT [FK\_Dim\_Hiring\_Fact\_table] FOREIGN KEY ([hiring\_id\_sk]) REFERENCES [dbo].[Fact\_table] ([hiring\_id\_fk]))

1. **Dim\_Student:**

This table contains information about students. It includes student ID (surrogate and business key), name, address, age, email, department details, certification information, freelancing details, skill and project information, temporal data, etc.

CREATE TABLE [dbo].[Dim\_Student](

[st\_id\_sk] int IDENTITY(1,1) NOT NULL,

[st\_id\_bk] int,

[st\_fname] varchar(100),

[st\_lname] varchar(100),

[st\_address] varchar(100),

[st\_age] int,

[email] varchar(100),

[st\_super\_bk] int,

[gender] varchar(50),

[dept\_id\_bk] int,

[dept\_name] varchar(100),

[dept\_location] varchar(100),

[dept\_mgr] int,

[mgr\_hiredate] date,

[cert\_id\_bk] int,

[cert\_name] varchar(100),

[issuer] varchar(100),

[issue\_date] date,

[freelancing\_id\_bk] int,

[freelancing\_name] varchar(100),

[skill\_id] int,

[skill\_name] varchar(100),

[project\_id] int,

[project\_name] varchar(100),

[pro\_startdate] date,

[pro\_enddate] date,

[start\_date] datetime,

[end\_date] datetime,

[is\_current] tinyint,

CONSTRAINT [FK\_Dim\_Student\_Fact\_table] FOREIGN KEY ([st\_id\_sk]) REFERENCES [dbo].[Fact\_table] ([student\_id\_fk]))

1. **DimDate:**

This table is a time dimension table. It includes various attributes related to dates such as day, month, quarter, year, and some additional attributes like holiday text.

CREATE TABLE [dbo].[DimDate](

[DateSK] int NOT NULL,

[Date] date NOT NULL,

[Day] char(2) NOT NULL,

[DaySuffix] varchar(4) NOT NULL,

[DayOfWeek] varchar(9) NOT NULL,

[DOWInMonth] tinyint NOT NULL,

[DayOfYear] int NOT NULL,

[WeekOfYear] tinyint NOT NULL,

[WeekOfMonth] tinyint NOT NULL,

[Month] char(2) NOT NULL,

[MonthName] varchar(9) NOT NULL,

[Quarter] tinyint NOT NULL,

[QuarterName] varchar(6) NOT NULL,

[Year] char(4) NOT NULL,

[StandardDate] varchar(10) NULL,

[HolidayText] varchar(50) NULL,

CONSTRAINT [FK\_DimDate\_Fact\_table] FOREIGN KEY ([DateSK]) REFERENCES [dbo].[Fact\_table] ([date\_id\_fk]))

1. **Fact\_table:**

This table represents a fact table that likely holds the central metrics or measurements of interest. It includes foreign keys to the dimension tables (Dim\_Exam, Dim\_Hiring, Dim\_Student, DimDate) and additional attributes like course details, instructor details, salary, grade, and creation timestamp.

CREATE TABLE [dbo].[Fact\_table](

[fact\_id\_sk] int IDENTITY(1,1) NOT NULL,

[hiring\_id\_fk] int,

[exam\_id\_fk] int,

[student\_id\_fk] int,

[date\_id\_fk] int,

[crs\_id\_bk] int,

[crs\_name] varchar(100),

[crs\_duration] int,

[topic\_id\_bk] int,

[topic\_name] varchar(100),

[ins\_id\_bk] int,

[ins\_name] varchar(100),

[salary] int,

[grade] int,

[created\_at] datetime,

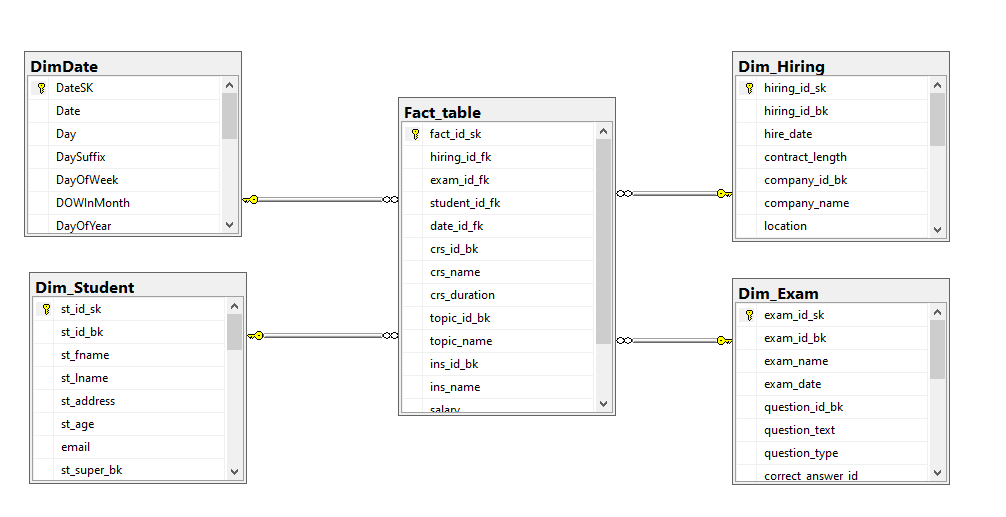
CONSTRAINT [FK\_Fact\_table\_Dim\_Exam] FOREIGN KEY ([exam\_id\_fk]) REFERENCES [dbo].[Dim\_Exam] ([exam\_id\_sk]),

CONSTRAINT [FK\_Fact\_table\_Dim\_Hiring] FOREIGN KEY ([hiring\_id\_fk]) REFERENCES [dbo].[Dim\_Hiring] ([hiring\_id\_sk]),

CONSTRAINT [FK\_Fact\_table\_Dim\_Student] FOREIGN KEY ([student\_id\_fk]) REFERENCES [dbo].[Dim\_Student] ([st\_id\_sk]),

CONSTRAINT [FK\_Fact\_table\_DimDate] FOREIGN KEY ([date\_id\_fk]) REFERENCES [dbo].[DimDate] ([DateSK]))

**DWH\_Diagram:**



**ETL using SSIS:**

1. **Dim\_Exam:**

select e.Exam\_Id, e.Exam\_Name, e.Exam\_Date, eq.Question\_Id, eq.Question\_Text, eq.Question\_Type, eq.Correct\_Answer\_Id,

qo.Option\_Id, qo.Option\_Text, qo.IsCorrect, ea.Answer\_Id, ea.Answer\_Text, ea.Score

from exam e left join Ques\_Exam qe

on e.Exam\_Id = qe.Ex\_Id

left join Exam\_questions eq

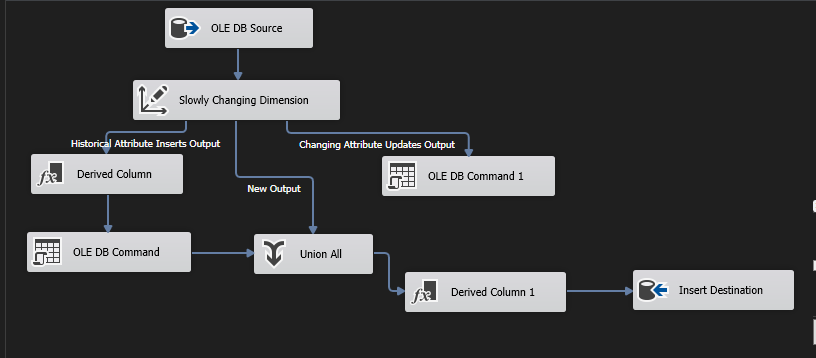
on qe.Q\_Id = eq.Question\_Id

left join Question\_options qo

on qo.Question\_Id = eq.Question\_Id

left join Exam\_answers ea

on ea.Question\_Id = eq.Question\_Id



1. **Dim\_Hiring:**

select c.Company\_Id, c.Company\_Name, c.Location, t.Train\_ID, t.Train\_Name, t.Start\_Date,

t.End\_Date, h.Hiring\_Id, h.Hire\_Date, h.ContractLength, p.Position\_Id, p.Position\_Name, p.Salary

from Companies c left join Training t

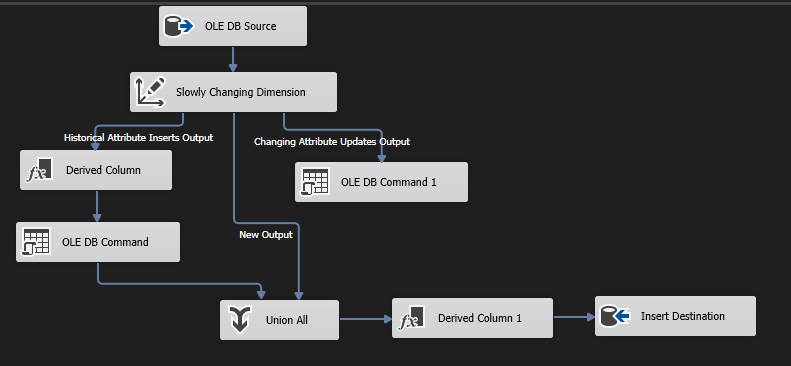
on t.Train\_ID = c.Training\_ID

left join Hiring h

on h.Company\_Id = c.Company\_Id

left join Positions p

on p.Position\_Id = h.Position\_Id

****

1. **Dim\_Student:**

select s.St\_Id, s.St\_Fname, s.St\_Lname, s.St\_Address, s.Gender, s.St\_Age,s.Email, s.St\_super, c.Cert\_Id, c.Cert\_Name, c.Issue\_Date, c.Issuer,

d.Dept\_Id, d.Dept\_Name, d.Dept\_Location, d.Dept\_Manager, d.Manager\_hiredate, sk.Skill\_Id, sk.Skill\_Name, f.Freelance\_Id, f.Freelance\_Name,

p.Project\_Id, p.Project\_Name, p.Start\_Date, p.End\_Date

from student s left join Student\_Certificates sc

on s.St\_Id = sc.St\_Id

left join Certificates c

on c.Cert\_Id = sc.Cert\_Id

left join Department d

on d.Dept\_Id = s.Dept\_Id

left join Student\_skills ss

on ss.St\_Id = s.St\_Id

left join Skills sk

on sk.Skill\_Id = ss.Sk\_Id

left join Freelancing f

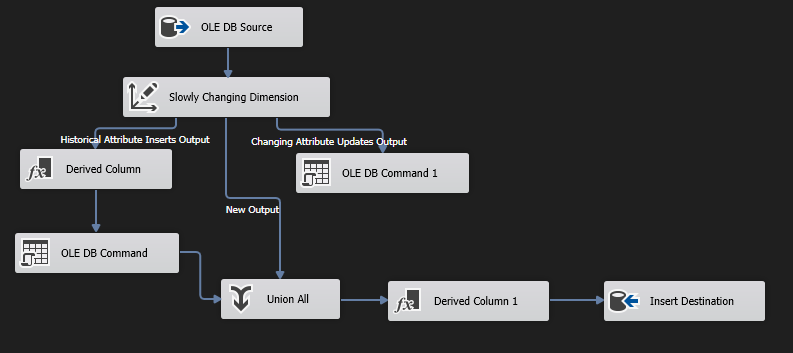
on f.St\_Id = s.St\_Id

left join Student\_Project sp

on sp.St\_Id = s.St\_Id

left join Projects p

on sp.Pro\_Id = p.Project\_Id



1. **Fact\_table:**

select s.St\_Id, h.Hiring\_Id, e.Exam\_Id, c.Crs\_Id, c.Crs\_Name, c.Crs\_Duration, c.Top\_Id, t.Top\_Name, i.Ins\_Id,

i.Ins\_Name, i.Salary, sc.Grade, d.Manager\_hiredate, ce.Issue\_Date, e.Exam\_Date, h.Hire\_Date

from Student s inner join Student\_course sc

on s.St\_Id = sc.St\_Id

inner join Course c

on c.Crs\_Id = sc.Crs\_Id

inner join Instructor\_course ic

on ic.Crs\_Id = c.Crs\_Id

inner join Instructor i

on i.Ins\_Id = ic.Ins\_Id

inner join Topic t

on t.Top\_Id = c.Top\_Id

inner join Department d

on s.Dept\_Id = d.Dept\_Id

inner join Hiring h

on h.St\_Id = s.St\_Id

inner join Exam e

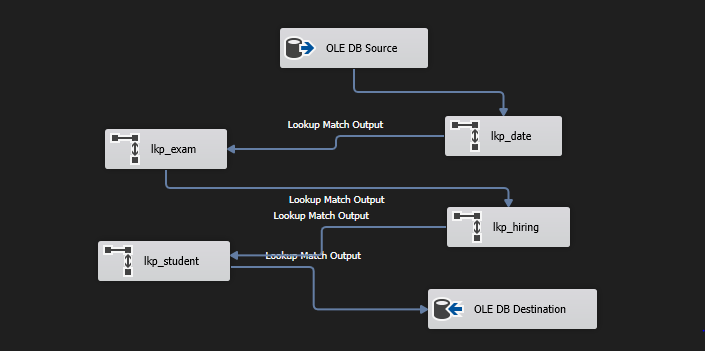
on e.Crs\_Id = c.Crs\_Id

inner join Student\_Certificates ss

on ss.St\_Id = s.St\_Id

inner join Certificates ce

on ce.Cert\_Id = ss.Cert\_Id



**Select stored procedures:**

1. **Certificates\_table:**

create proc GetCertificateData as

select \* from Certificates

1. **Companies\_table:**

create proc GetCompaniesData as

select \* from Companies

1. **Course\_table:**

create proc GetcourseData as

select \* from Course

1. **Department\_table:**

create proc GetDepartmentData as

select \* from Department

1. **Exam\_table:**

create proc GetExamData as

select \* from Exam

1. **Exam\_answers\_table:**

create proc GetExamAswersData as

select \* from Exam\_answers

1. **Exam\_questions\_table:**

create proc GetExamQuestionData as

select \* from Exam\_questions

1. **Freelancing\_table:**

create proc GetFreelacingData as

select \* from Freelancing

1. **Hiring \_table:**

create proc GetExamHringData as

select \* from Hiring

1. **Instructor\_table:**

create proc GetInstructorData as

select \* from Instructor

1. **Instructor\_course\_table:**

create proc GetInstructorCourseData as

select \* from Instructor\_course

1. **Positions\_table:**

create proc GetPositionData as

select \* from Positions

1. **Projects\_table:**

create proc GetProjectsData as

select \* from Projects

1. **Ques\_Exam\_table:**

create proc GetQues\_ExamData as

select \* from Ques\_Exam

1. **Question\_options\_table:**

create proc GetQuestion\_optionsData as

select \* from Question\_options

1. **Skills\_table:**

create proc GetSkillsData as

select \* from Skills

1. **St\_Exam\_table:**

create proc GetStExamData as

select \* from St\_Exam

1. **Student\_table:**

create proc GetStudentData as

select \* from Student

1. **Student\_Certificates\_table:**

create proc GetStudentCertificatesData as

select \* from Student\_Certificates

1. **Student\_course\_table:**

create proc GetStudentCourseData as

select \* from Student\_course

1. **Student\_Project\_table:**

create proc GetStudenProjectstData as

select \* from Student\_Project

1. **Student\_skills\_table:**

create proc GetStudentSkillsData as

select \* from Student\_skills

1. **Topic\_table:**

create proc GetTopicData as

select \* from Topic

1. **Training\_table:**

create proc GettraningData as

select \* from Training

**Insert stored procedures:**

1. **Certificates\_table:**

CREATE PROCEDURE InsertCertificate

@Cert\_Name VARCHAR(100),

@Issuer VARCHAR(100),

@Issue\_Date DATE

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Certificates] ([Cert\_Name], [Issuer], [Issue\_Date])

VALUES (@Cert\_Name, @Issuer, @Issue\_Date);

SELECT SCOPE\_IDENTITY() AS Cert\_Id;

END;

1. **Companies\_table:**

create PROCEDURE InsertCompany

@Company\_Name VARCHAR(100),

@Location VARCHAR(100),

@Training\_ID INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Companies] ([Company\_Name], [Location],[Training\_ID])

VALUES (@Company\_Name, @Location, @Training\_ID);

END;

1. **Course\_table:**

create PROCEDURE InsertCourse

@Crs\_Name VARCHAR(100),

@Crs\_Duration INT,

@Top\_Id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Course] ([Crs\_Name], [Crs\_Duration], [Top\_Id])

VALUES (@Crs\_Name, @Crs\_Duration, @Top\_Id);

END;

1. **Department\_table:**

CREATE PROCEDURE InsertDepartment

@Dept\_Name VARCHAR(100),

@Dept\_Location VARCHAR(100),

@Dept\_Manager INT,

@Manager\_hiredate DATE

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Department] ([Dept\_Name], [Dept\_Location], [Dept\_Manager], [Manager\_hiredate])

VALUES (@Dept\_Name, @Dept\_Location, @Dept\_Manager, @Manager\_hiredate);

END;

1. **Exam\_table:**

CREATE PROCEDURE InsertExam

@Exam\_Name VARCHAR(100),

@Exam\_Date DATE,

@Crs\_Id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Exam] ([Exam\_Name], [Exam\_Date], [Crs\_Id])

VALUES (@Exam\_Name, @Exam\_Date, @Crs\_Id);

END;

1. **Exam\_answers\_table:**

Create PROCEDURE InsertExamAnswer

@Answer\_Text VARCHAR(100),

@Question\_Id INT,

@Score INT,

@st\_id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Exam\_answers] ([Answer\_Text],[Question\_Id] ,[Score],[St\_Id])

VALUES (@Answer\_Text,@Question\_Id, @Score, @st\_id );

END;

1. **Exam\_questions\_table:**

CREATE PROCEDURE InsertExamQuestion

@Question\_Text VARCHAR(100),

@Question\_Type VARCHAR(50),

@Correct\_ans int,

@Category varchar(100)

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Exam\_questions] ( [Question\_Text], [Question\_Type],[Correct\_Answer\_Id],[Category\_Name])

VALUES ( @Question\_Text, @Question\_Type,@Correct\_ans ,@Category);

END;

1. **Freelancing\_table:**

CREATE PROCEDURE InsertFreelancing

@Freelance\_Name VARCHAR(100),

@St\_Id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Freelancing] ([Freelance\_Name],[St\_Id])

VALUES (@Freelance\_Name,@St\_Id);

END;

1. **Hiring \_table:**

CREATE PROCEDURE InsertHiring

@Company\_Id INT,

@Position\_Id INT,

@St\_Id INT,

@Hire\_Date DATE,

@ContractLength INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Hiring] ([Company\_Id], [Position\_Id], [St\_Id], [Hire\_Date], [ContractLength])

VALUES (@Company\_Id, @Position\_Id, @St\_Id, @Hire\_Date, @ContractLength);

END;

1. **Instructor\_table:**

CREATE PROCEDURE InsertInstructor

@Ins\_Name VARCHAR(100),

@Salary INT,

@Dept\_Id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Instructor] ([Ins\_Name], [Salary], [Dept\_Id])

VALUES (@Ins\_Name, @Salary, @Dept\_Id);

END;

1. **Instructor\_course\_table:**

CREATE PROCEDURE InsertInstructorCourse

@Ins\_Id INT,

@Crs\_Id INT,

@Evaluation VARCHAR(100)

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Instructor\_course] ([Ins\_Id], [Crs\_Id], [Evaluation])

VALUES (@Ins\_Id, @Crs\_Id, @Evaluation);

END;

1. **Positions\_table:**

CREATE PROCEDURE InsertPosition

@Position\_Name VARCHAR(100),

@Salary INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Positions] ([Position\_Name], [Salary])

VALUES (@Position\_Name, @Salary);

END;

1. **Projects\_table:**

CREATE PROCEDURE InsertProject

@Project\_Name VARCHAR(100),

@Start\_Date DATE,

@End\_Date DATE

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Projects] ([Project\_Name], [Start\_Date], [End\_Date])

VALUES (@Project\_Name, @Start\_Date, @End\_Date);

END;

1. **Ques\_Exam\_table:**

CREATE PROCEDURE InsertQuesExam

@Ex\_Id INT,

@Q\_Id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Ques\_Exam] ([Ex\_Id], [Q\_Id])

VALUES (@Ex\_Id, @Q\_Id);

END;

1. **Question\_options\_table:**

CREATE PROCEDURE InsertQuestionOption

@Option\_Text VARCHAR(10),

@IsCorrect INT,

@Question\_Id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Question\_options] ([Option\_Text], [IsCorrect], [Question\_Id])

VALUES (@Option\_Text, @IsCorrect, @Question\_Id);

END;

1. **Skills\_table:**

CREATE PROCEDURE InsertSkill

@Skill\_Name VARCHAR(100)

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Skills] ([Skill\_Name])

VALUES (@Skill\_Name);

END;

1. **St\_Exam\_table:**

CREATE PROCEDURE InsertStExam

@St\_Id INT,

@Ex\_Id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[St\_Exam] ([St\_Id], [Ex\_Id])

VALUES (@St\_Id, @Ex\_Id);

END;

1. **Student\_table:**

CREATE PROCEDURE InsertStudent

@St\_Fname VARCHAR(100),

@St\_Lname VARCHAR(100),

@St\_Address VARCHAR(100),

@St\_Age INT,

@Email VARCHAR(100),

@Dept\_Id INT,

@St\_super INT,

@Gender VARCHAR(50)

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Student] ([St\_Fname], [St\_Lname], [St\_Address], [St\_Age], [Email], [Dept\_Id], [St\_super], [Gender])

VALUES (@St\_Fname, @St\_Lname, @St\_Address, @St\_Age, @Email, @Dept\_Id, @St\_super, @Gender);

END;

1. **Student\_Certificates\_table:**

CREATE PROCEDURE InsertStudentCertificate

@St\_Id INT,

@Cert\_Id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Student\_Certificates] ([St\_Id], [Cert\_Id])

VALUES (@St\_Id, @Cert\_Id);

END;

1. **Student\_course\_table:**

CREATE PROCEDURE InsertStudentCourse

@Crs\_Id INT,

@St\_Id INT,

@Grade INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Student\_course] ([Crs\_Id], [St\_Id], [Grade])

VALUES (@Crs\_Id, @St\_Id, @Grade);

END;

1. **Student\_Project\_table:**

CREATE PROCEDURE InsertStudentProject

@St\_Id INT,

@Pro\_Id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Student\_Project] ([St\_Id], [Pro\_Id])

VALUES (@St\_Id, @Pro\_Id);

END;

1. **Student\_skills\_table:**

CREATE PROCEDURE InsertStudentSkill

@Sk\_Id INT,

@St\_Id INT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Student\_skills] ([Sk\_Id], [St\_Id])

VALUES (@Sk\_Id, @St\_Id);

END;

1. **Topic\_table:**

CREATE PROCEDURE InsertTopic

@Top\_Name VARCHAR(100)

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Topic] ([Top\_Name])

VALUES (@Top\_Name);

END;

1. **Training\_table:**

CREATE PROCEDURE InsertTraining

@Train\_Name VARCHAR(100),

@Start\_Date DATE,

@End\_Date DATE

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO [dbo].[Training] ([Train\_Name], [Start\_Date], [End\_Date])

VALUES (@Train\_Name, @Start\_Date, @End\_Date);

END;

**Update stored procedures:**

1. **Certificates\_table:**

CREATE PROCEDURE UpdateCertificate

@CertID INT,

@CertName varchar(100),

@IssueDate DATE,

@Issuer varchar(100)

AS

BEGIN

UPDATE certificates

SET

cert\_name = @CertName,

issue\_date = @IssueDate,

issuer = @Issuer

WHERE

cert\_id = @CertID;

END;

1. **Companies\_table:**

create PROCEDURE UpdateCompany

@Company\_Id int,

@Company\_Name varchar(100),

@Location varchar(100)

AS

BEGIN

UPDATE companies

SET

Company\_Name= @Company\_Name,

Location = @Location

WHERE

Company\_Id = @Company\_Id;

END;

1. **Course\_table:**

CREATE PROCEDURE UpdateCourse

@Crs\_Id INT,

@Crs\_Name VARCHAR(100),

@Crs\_Duration INT

AS

BEGIN

UPDATE Course

SET

Crs\_Name = @Crs\_Name,

Crs\_Duration = @Crs\_Duration

WHERE

Crs\_Id = @Crs\_Id;

END;

1. **Department\_table:**

CREATE PROCEDURE UpdateDepartment

@Dept\_Id INT,

@Dept\_Name VARCHAR(100),

@Dept\_Location VARCHAR(100),

@Dept\_Manager INT,

@Manager\_hiredate DATE

AS

BEGIN

UPDATE Department

SET

Dept\_Name = @Dept\_Name,

Dept\_Location = @Dept\_Location,

Dept\_Manager = @Dept\_Manager,

Manager\_hiredate = @Manager\_hiredate

WHERE

Dept\_Id = @Dept\_Id;

END;

1. **Exam\_table:**

CREATE PROCEDURE UpdateExam

@Exam\_Id INT,

@Exam\_Name VARCHAR(100),

@Exam\_Date DATE

AS

BEGIN

UPDATE Exam

SET

Exam\_Name = @Exam\_Name,

Exam\_Date = @Exam\_Date

WHERE

Exam\_Id = @Exam\_Id;

END;

1. **Freelancing\_table:**

CREATE PROCEDURE UpdateFreelancing

@Freelance\_Id INT,

@Freelance\_Name VARCHAR(100),

@St\_Id INT

AS

BEGIN

UPDATE Freelancing

SET

Freelance\_Name = @Freelance\_Name,

St\_Id = @St\_Id

WHERE

Freelance\_Id = @Freelance\_Id;

END;

1. **Hiring \_table:**

CREATE PROCEDURE UpdateHiring

@Hiring\_Id INT,

@Position\_Id INT,

@St\_Id INT,

@Hire\_Date DATE

AS

BEGIN

UPDATE Hiring

SET

Position\_Id = @Position\_Id,

St\_Id = @St\_Id,

Hire\_Date = @Hire\_Date

WHERE

Hiring\_Id = @Hiring\_Id;

END;

1. **Instructor\_table:**

CREATE PROCEDURE UpdateInstructor

@Ins\_Id INT,

@Ins\_Name VARCHAR(100),

@Salary INT

AS

BEGIN

UPDATE Instructor

SET

Ins\_Name = @Ins\_Name,

Salary = @Salary

WHERE

Ins\_Id = @Ins\_Id;

END;

1. **Positions\_table:**

CREATE PROCEDURE UpdatePosition

@Position\_Id INT,

@Position\_Name VARCHAR(100),

@Salary INT

AS

BEGIN

UPDATE Positions

SET

Position\_Name = @Position\_Name,

Salary = @Salary

WHERE

Position\_Id = @Position\_Id;

END;

1. **Projects\_table:**

CREATE PROCEDURE UpdateProject

@Project\_Id INT,

@Project\_Name VARCHAR(100),

@Start\_Date DATE,

@End\_Date DATE

AS

BEGIN

UPDATE Projects

SET

Project\_Name = @Project\_Name,

Start\_Date = @Start\_Date,

End\_Date = @End\_Date

WHERE

Project\_Id = @Project\_Id;

END;

1. **Skills\_table:**

CREATE PROCEDURE UpdateSkills

@Skill\_Id INT,

@Skill\_Name VARCHAR(100)

AS

BEGIN

UPDATE Skills

SET Skill\_Name = @Skill\_Name

WHERE Skill\_Id = @Skill\_Id;

END;

1. **Student\_table:**

CREATE PROCEDURE UpdateStudent

@St\_Id INT,

@St\_Lname VARCHAR(100),

@St\_Address VARCHAR(100),

@St\_Age INT,

@Email VARCHAR(100),

@Dept\_Id INT,

@Gender VARCHAR(50)

AS

BEGIN

UPDATE Student

SET

St\_Lname = @St\_Lname,

St\_Address = @St\_Address,

St\_Age = @St\_Age,

Email = @Email,

Dept\_Id = @Dept\_Id,

Gender = @Gender

WHERE St\_Id = @St\_Id;

END;

1. **Topic\_table:**

CREATE PROCEDURE UpdateTopic

@Top\_Id INT,

@Top\_Name VARCHAR(100)

AS

BEGIN

UPDATE Topic

SET

Top\_Name = @Top\_Name

WHERE Top\_Id = @Top\_Id;

END;

1. **Training\_table:**

CREATE PROCEDURE UpdateTraining

@Train\_ID INT,

@Train\_Name VARCHAR(100),

@Start\_Date DATE,

@End\_Date DATE

AS

BEGIN

UPDATE Training

SET

Train\_Name = @Train\_Name,

Start\_Date = @Start\_Date,

End\_Date = @End\_Date

WHERE Train\_ID = @Train\_ID;

END;

**Delete stored procedures:**

1. **Certificates\_table:**

CREATE PROCEDURE DeleteCertificate

@CertificateId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Student\_Certificates table

DELETE FROM dbo.Student\_Certificates WHERE Cert\_Id = @CertificateId;

-- Delete records from Certificates table

DELETE FROM dbo.Certificates WHERE Cert\_Id = @CertificateId;

END;

1. **Companies\_table:**

CREATE PROCEDURE DeleteCompany

@CompanyId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Hiring table

DELETE FROM dbo.Hiring WHERE Company\_Id = @CompanyId;

-- Delete records from Companies table

DELETE FROM dbo.Companies WHERE Company\_Id = @CompanyId;

END

1. **Course\_table:**

CREATE PROCEDURE DeleteCourse

@CourseId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Instructor\_course table

DELETE FROM dbo.Instructor\_course WHERE Crs\_Id = @CourseId;

-- Delete records from Student\_course table

DELETE FROM dbo.Student\_course WHERE Crs\_Id = @CourseId;

-- Delete records from Exam table

DELETE FROM dbo.Exam WHERE Crs\_Id = @CourseId;

-- Delete records from Course table

DELETE FROM dbo.Course WHERE Crs\_Id = @CourseId;

END

1. **Department\_table:**

CREATE PROCEDURE DeleteDepartment

@DepartmentId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Instructor table

DELETE FROM dbo.Instructor WHERE Dept\_Id = @DepartmentId;

-- Delete records from Student table

DELETE FROM dbo.Student WHERE Dept\_Id = @DepartmentId;

-- Delete records from Department table

DELETE FROM dbo.Department WHERE Dept\_Id = @DepartmentId;

END

1. **Exam\_table:**

CREATE PROCEDURE DeleteExam

@ExamId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Ques\_Exam table

DELETE FROM dbo.Ques\_Exam WHERE Ex\_Id = @ExamId;

-- Delete records from Exam table

DELETE FROM dbo.Exam WHERE Exam\_Id = @ExamId;

END

1. **Exam\_questions\_table:**

CREATE PROCEDURE DeleteExamQuestions

@QuestionId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Question\_options table

DELETE FROM dbo.Question\_options WHERE Question\_Id = @QuestionId;

-- Delete records from Ques\_Exam table

DELETE FROM dbo.Ques\_Exam WHERE Q\_Id = @QuestionId;

-- Delete records from Exam\_questions table

DELETE FROM dbo.Exam\_questions WHERE Question\_Id = @QuestionId;

END

1. **Freelancing\_table:**

CREATE PROCEDURE DeleteFreelancing

@FreelanceId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Freelancing table

DELETE FROM dbo.Freelancing WHERE Freelance\_Id = @FreelanceId;

END

1. **Instructor\_table:**

CREATE PROCEDURE DeleteInstructor

@InstructorId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Instructor\_course table

DELETE FROM dbo.Instructor\_course WHERE Ins\_Id = @InstructorId;

-- Delete records from Instructor table

DELETE FROM dbo.Instructor WHERE Ins\_Id = @InstructorId;

END

1. **Positions\_table:**

CREATE PROCEDURE DeletePosition

@PositionId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Hiring table

DELETE FROM dbo.Hiring WHERE Position\_Id = @PositionId;

-- Delete records from Instructor\_course table

DELETE FROM dbo.Instructor\_course WHERE Evaluation = 'Position' AND Crs\_Id = @PositionId;

-- Delete records from Positions table

DELETE FROM dbo.Positions WHERE Position\_Id = @PositionId;

END

1. **Projects\_table:**

CREATE PROCEDURE DeleteProject

@ProjectId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Student\_Project table

DELETE FROM dbo.Student\_Project WHERE Pro\_Id = @ProjectId;

-- Delete records from Projects table

DELETE FROM dbo.Projects WHERE Project\_Id = @ProjectId;

END

1. **Question\_options\_table:**

CREATE PROCEDURE DeleteQuestionAndOptions

@QuestionId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Question\_options table

DELETE FROM dbo.Question\_options WHERE Question\_Id = @QuestionId;

-- Delete records from Ques\_Exam table

DELETE FROM dbo.Ques\_Exam WHERE Q\_Id = @QuestionId;

-- Delete records from Exam\_answers table

DELETE FROM dbo.Exam\_answers WHERE Question\_Id = @QuestionId;

-- Delete records from Exam\_questions table

DELETE FROM dbo.Exam\_questions WHERE Question\_Id = @QuestionId;

END

1. **Skills\_table:**

CREATE PROCEDURE DeleteSkill

@SkillId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Student\_skills table

DELETE FROM dbo.Student\_skills WHERE Sk\_Id = @SkillId;

-- Delete records from Instructor\_course table

DELETE FROM dbo.Instructor\_course WHERE Evaluation = 'Skill' AND Crs\_Id = @SkillId;

-- Delete records from Skills table

DELETE FROM dbo.Skills WHERE Skill\_Id = @SkillId;

END

1. **Student\_table:**

create PROCEDURE DeleteStudent

@StudentId INT

AS

BEGIN

SET NOCOUNT ON;

-- Remove self-references

UPDATE dbo.Student

SET St\_super = NULL

WHERE St\_super = @StudentId;

-- Delete records from St\_Exam table

DELETE FROM dbo.St\_Exam WHERE St\_Id = @StudentId;

-- Delete records from Student\_course table

DELETE FROM dbo.Student\_course WHERE St\_Id = @StudentId;

-- Delete records from Hiring table

DELETE FROM dbo.Hiring WHERE St\_Id = @StudentId;

-- Delete records from Freelancing table

DELETE FROM dbo.Freelancing WHERE St\_Id = @StudentId;

-- Delete records from Student\_Project table

DELETE FROM dbo.Student\_Project WHERE St\_Id = @StudentId;

-- Delete records from Student\_skills table

DELETE FROM dbo.Student\_skills WHERE St\_Id = @StudentId;

-- Delete records from Student\_Certificates table

DELETE FROM dbo.Student\_Certificates WHERE St\_Id = @StudentId;

-- Delete records from Student table

DELETE FROM dbo.Student WHERE St\_Id = @StudentId;

END

1. **Topic\_table:**

CREATE PROCEDURE DeleteTopic

@TopicId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Course table

DELETE FROM dbo.Course WHERE Top\_Id = @TopicId;

-- Delete records from Topic table

DELETE FROM dbo.Topic WHERE Top\_Id = @TopicId;

END

1. **Training\_table:**

CREATE PROCEDURE DeleteTraining

@TrainingId INT

AS

BEGIN

SET NOCOUNT ON;

-- Delete records from Companies table

DELETE FROM dbo.Companies WHERE Training\_ID = @TrainingId;

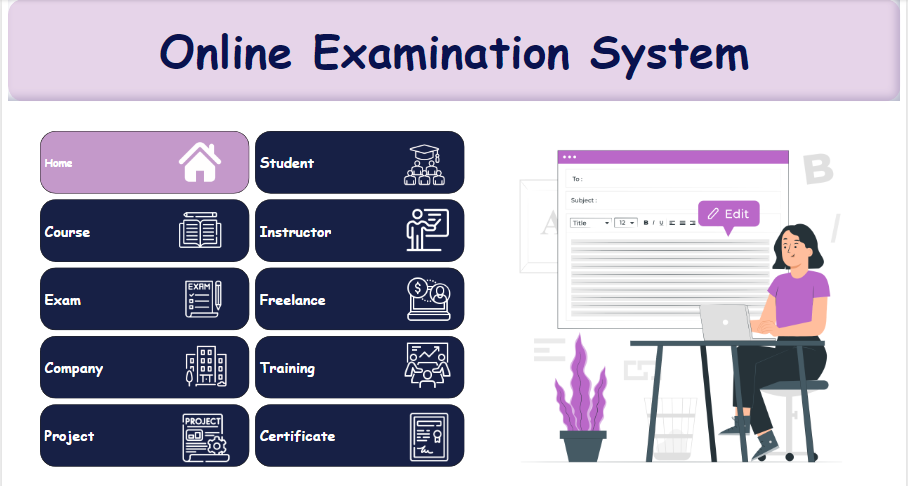
-- Delete records from Training table

DELETE FROM dbo.Training WHERE Train\_ID = @TrainingId;

END

**Power bi Dashboards:**

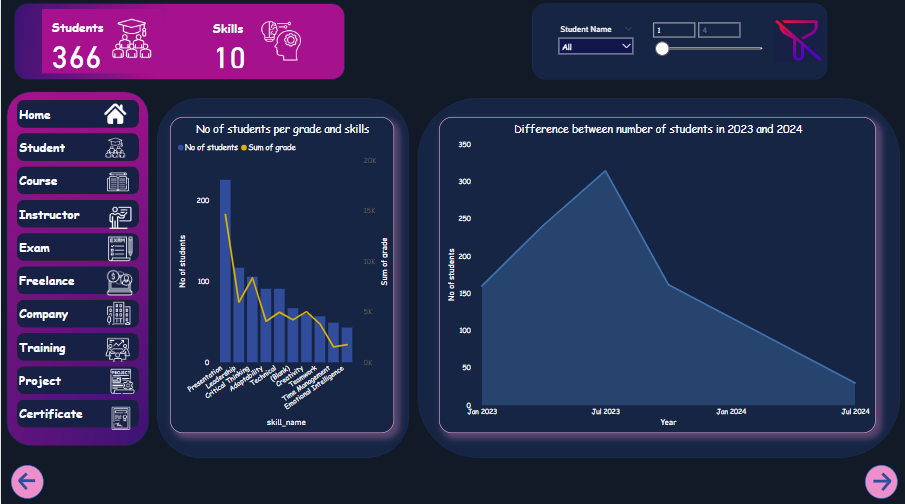
1. **Overview:**

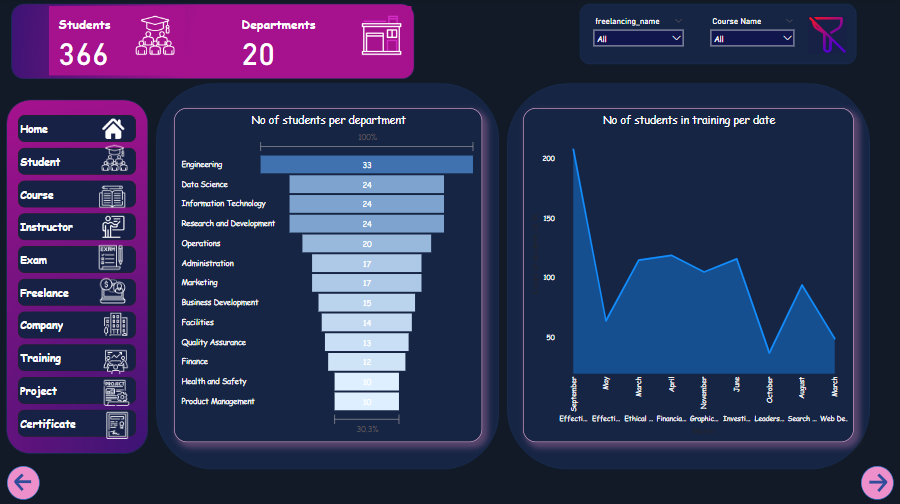


1. **About Student:**

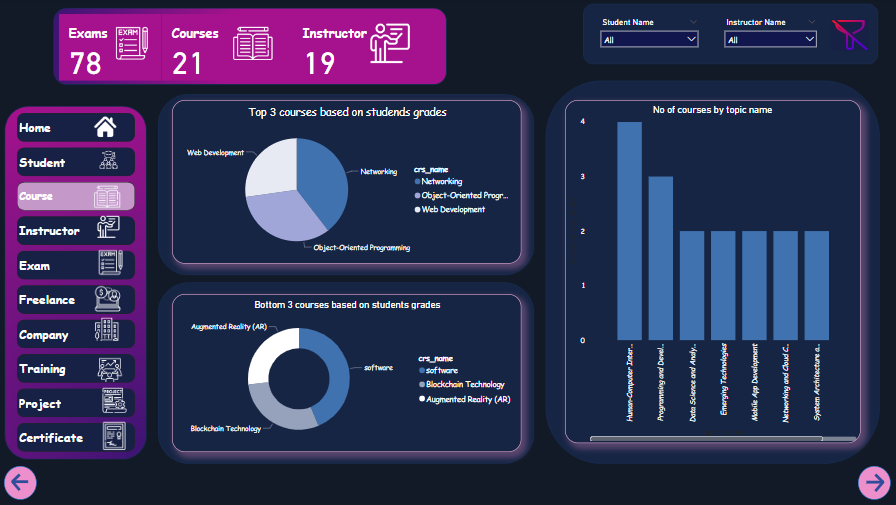




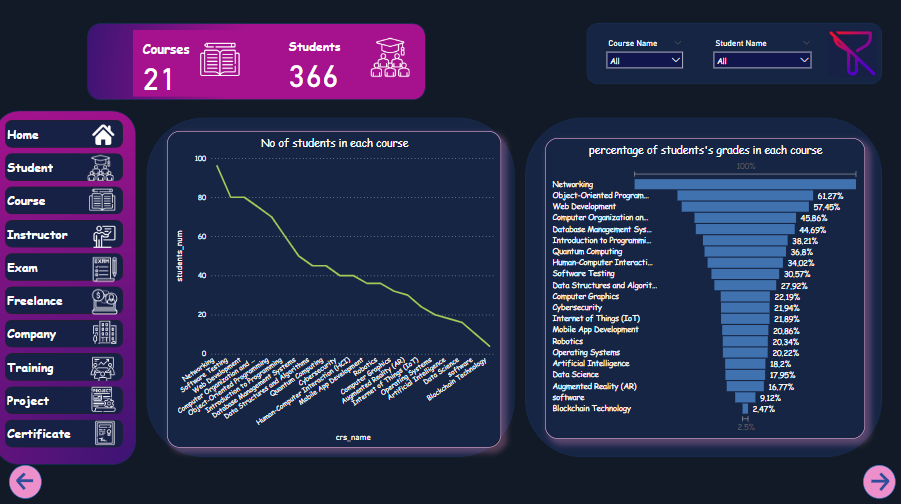


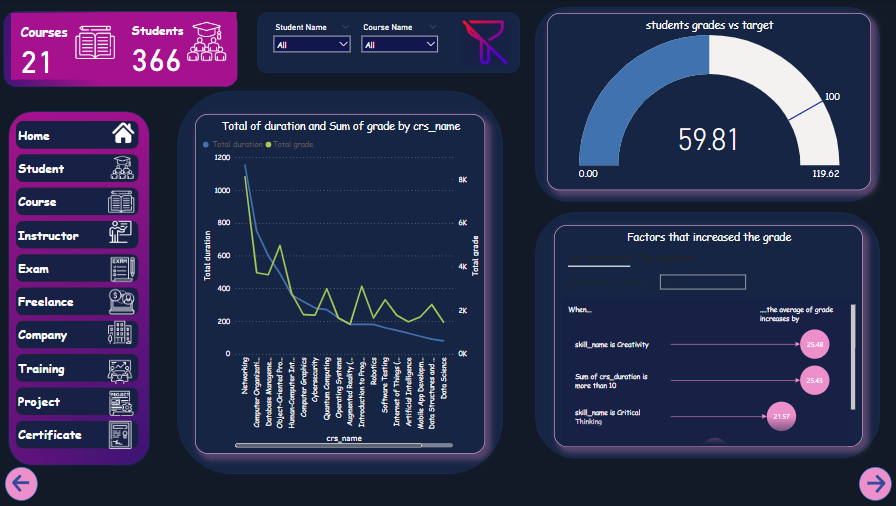


1. **About Courses:**

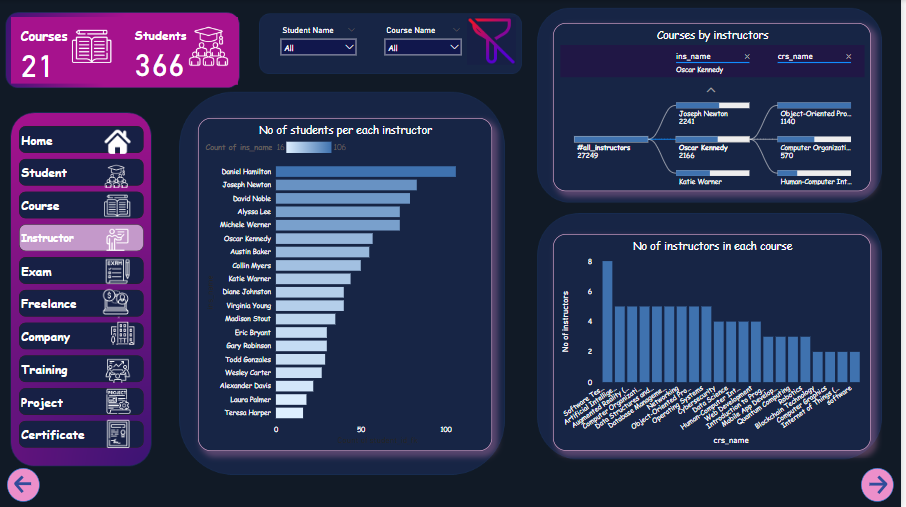
****

****

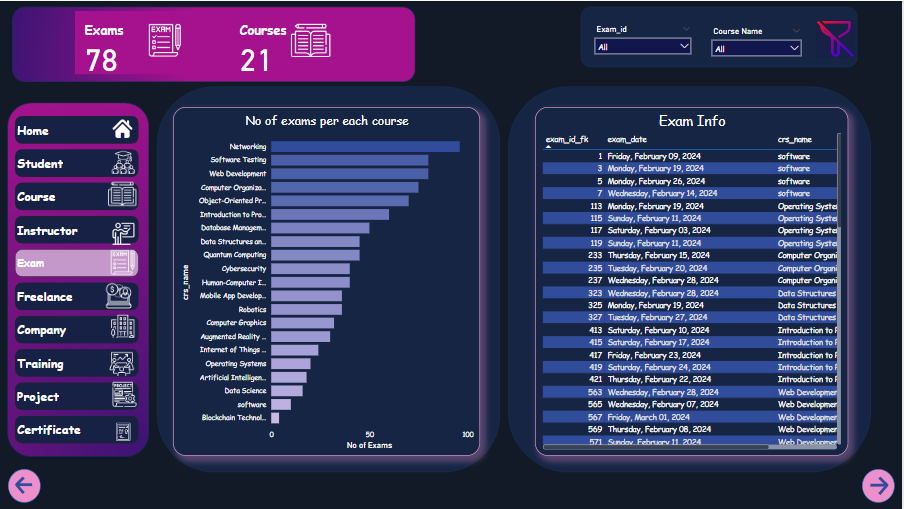
****

****

1. **About Instructors:**

****

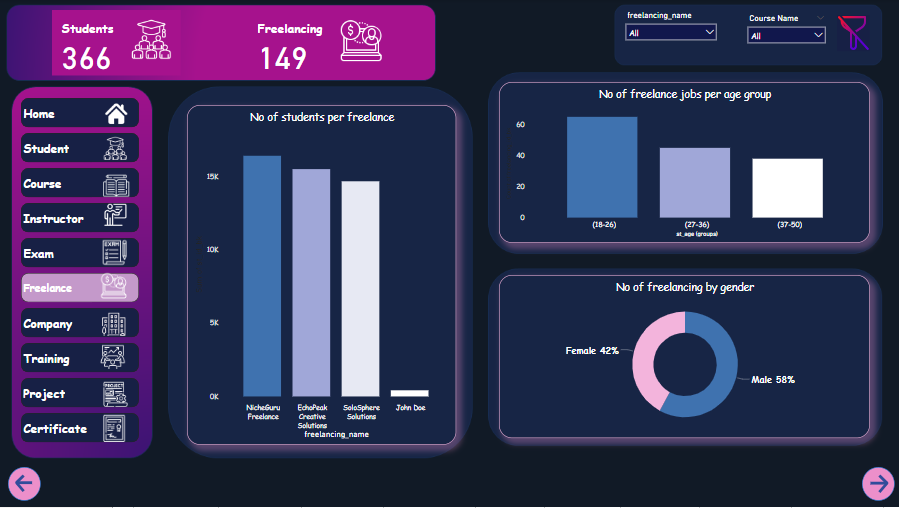
1. **About Exams:**



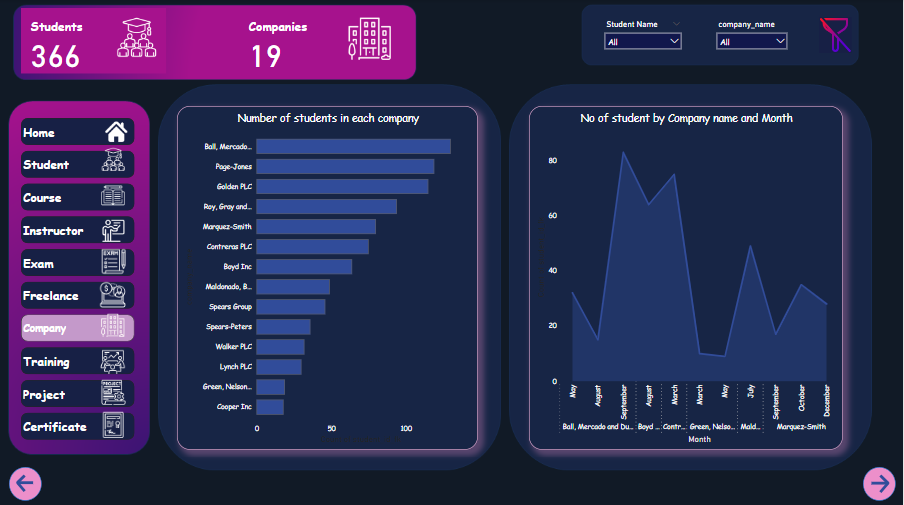


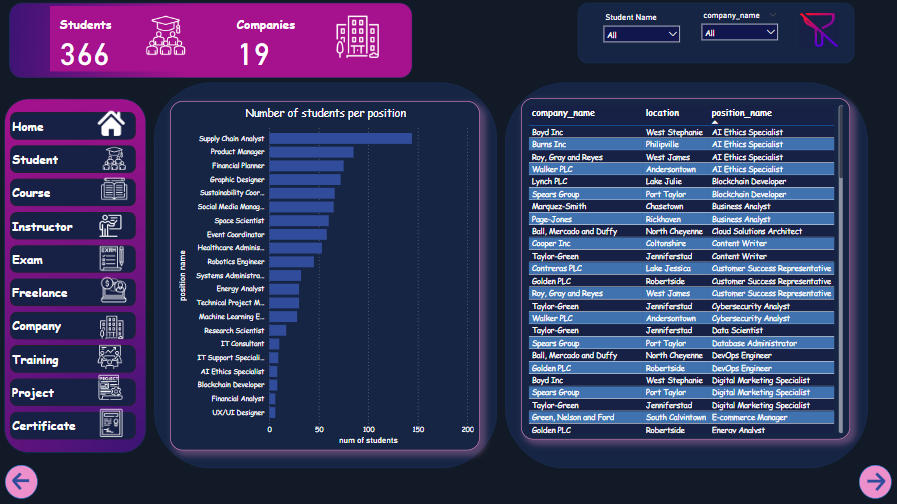


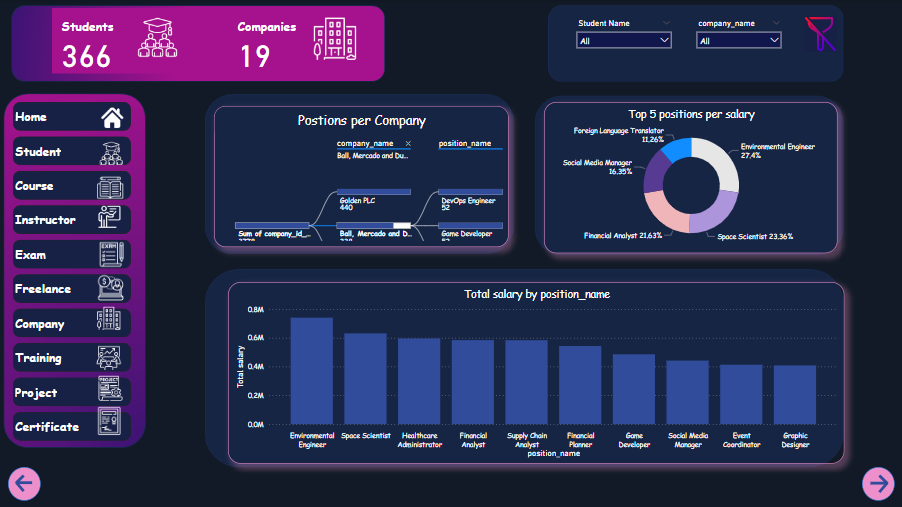
1. **About Freelancing:**

****

1. **About Companies:**

****

****

****

****

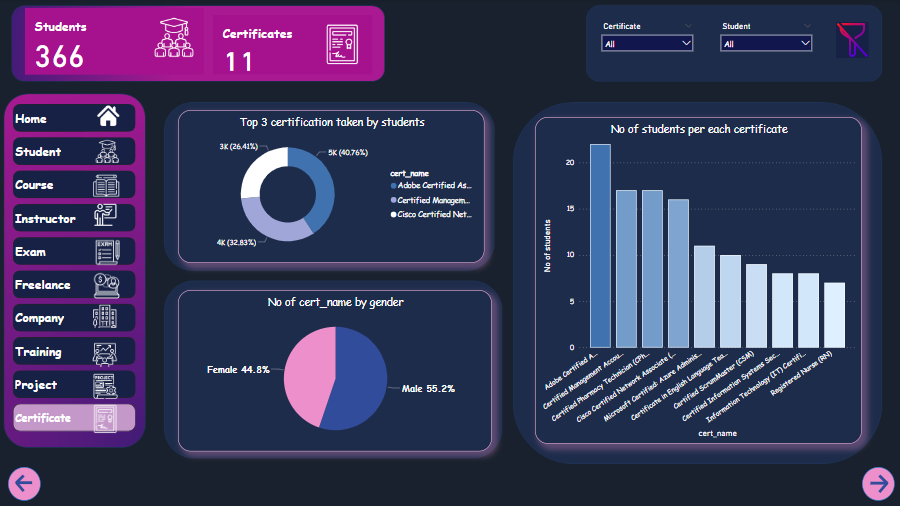
1. **About Training:**

****

1. **About Projects:**

****

1. **About Certificates:**

****

**Web Application:**

1. **Stored procedure for Exam generation:**

create PROCEDURE GenerateBubbleSheet

@ExamId INT

AS

BEGIN

SET NOCOUNT ON;

-- Declare variables

DECLARE @ExamName VARCHAR(100);

-- Get exam name

SELECT @ExamName = Exam\_Name

FROM Exam

WHERE Exam\_Id = @ExamId;

-- Print exam name

PRINT 'Exam: ' + @ExamName;

-- Declare a temporary table to store the results

CREATE TABLE #BubbleSheet (

QuestionText VARCHAR(200),

OptionText VARCHAR(MAX)

);

-- Insert questions and options into the temporary table

INSERT INTO #BubbleSheet (QuestionText, OptionText)

SELECT EQ.Question\_Text, QO.Option\_Text

FROM Exam\_questions EQ

INNER JOIN Ques\_Exam QE ON EQ.Question\_Id = QE.Q\_Id

INNER JOIN Question\_options QO ON EQ.Question\_Id = QO.Question\_Id

WHERE QE.Ex\_Id = @ExamId;

-- Select the results from the temporary table

SELECT \* FROM #BubbleSheet;

-- Drop the temporary table

DROP TABLE #BubbleSheet;

END

1. **Stored procedure for Exam Answers:**

CREATE PROCEDURE InsertExamAnswer

@Answer\_Text VARCHAR(300),

@Question\_Id INT,

@St\_Id INT

AS

BEGIN

DECLARE @Score INT;

-- Check if the provided answer matches a correct option for the given question

SELECT @Score = CASE WHEN EXISTS (

SELECT 1

FROM Question\_options

WHERE Question\_Id = @Question\_Id

AND Option\_Text = @Answer\_Text

AND IsCorrect = 1

)

THEN 1

ELSE 0

END;

INSERT INTO Exam\_answers (Answer\_Text, Question\_Id, Score, St\_Id)

VALUES (@Answer\_Text, @Question\_Id, @Score, @St\_Id);

END;

DECLARE @Answer\_Text VARCHAR(300) = 'False';

DECLARE @Question\_Id INT = 140; -- Provide the appropriate Question\_Id

DECLARE @St\_Id INT = 108; -- Provide the Student\_Id

1. **Stored procedure for Exam corrections:**

create PROCEDURE CompareAnswers

@Question\_Id INT,

@Selected\_Answer VARCHAR(MAX)

AS

BEGIN

DECLARE @Score INT = 0;

-- Check if the selected answer is correct for the given question

SELECT

CASE WHEN EXISTS (

SELECT 1

FROM Question\_options

WHERE Question\_Id = @Question\_Id

AND Option\_Text = @Selected\_Answer

AND IsCorrect = 1

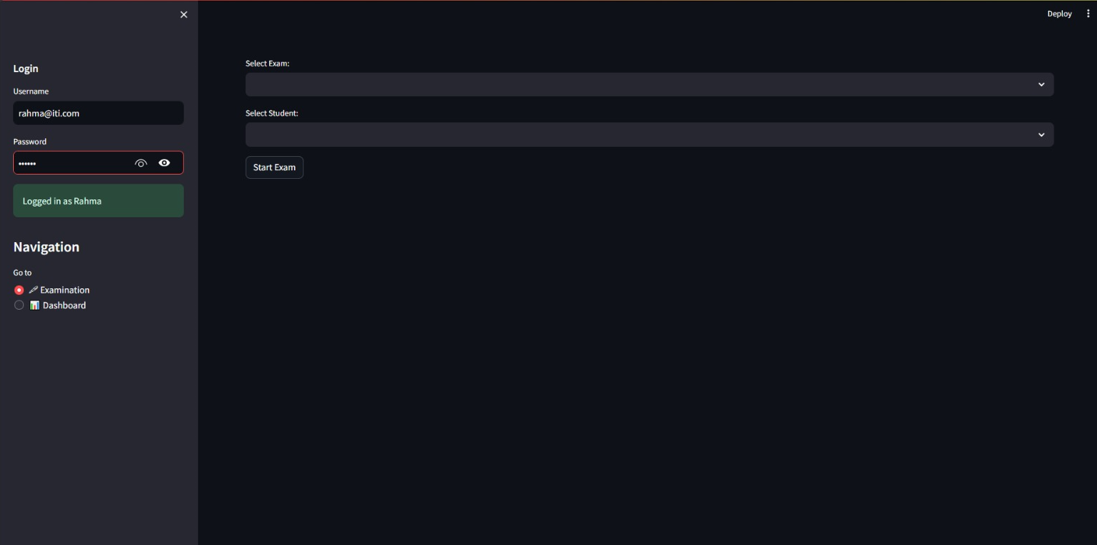
)

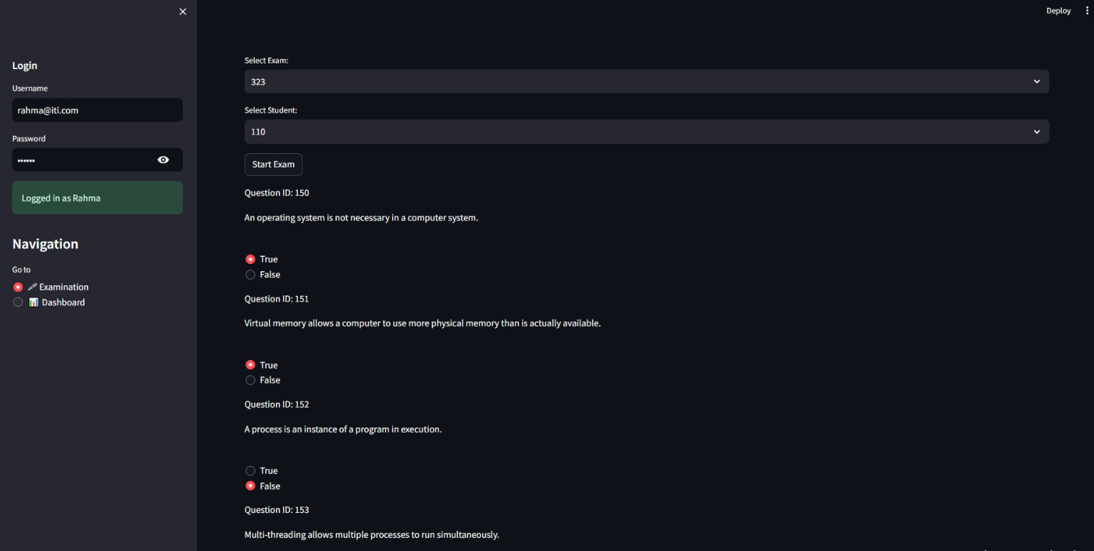
THEN 1

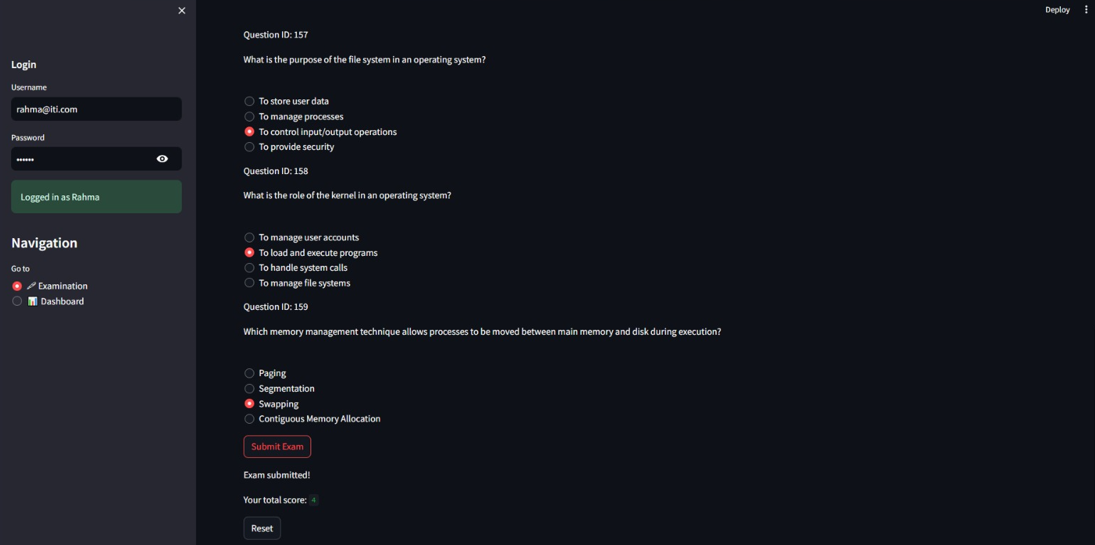
ELSE 0

END AS Score;

END;

****

****

****