Exam System Project

Data Dictionary

2023-01-17

Table of contents

Legend

- Primary key
- Primary key disabled
- User-defined primary key
- Unique key
- Unique key disabled
- User-defined unique key
- Active trigger
- Disabled trigger
- Many to one relationship
- User-defined many to one relationship
- One to many relationship
- User-defined one to many relationship
- Many to many relationship
- User-defined many to many relationship
- One to one relationship
- User-defined one to one relationship
- Input
- Output
- Input/Output
- Uses dependency
- User-defined uses dependency
- Used by dependency
- User-defined used by dependency

1. Cover

Exam system Project

Presented to : ENG Rami

Created by :

1-Mohamed Abdelfattah

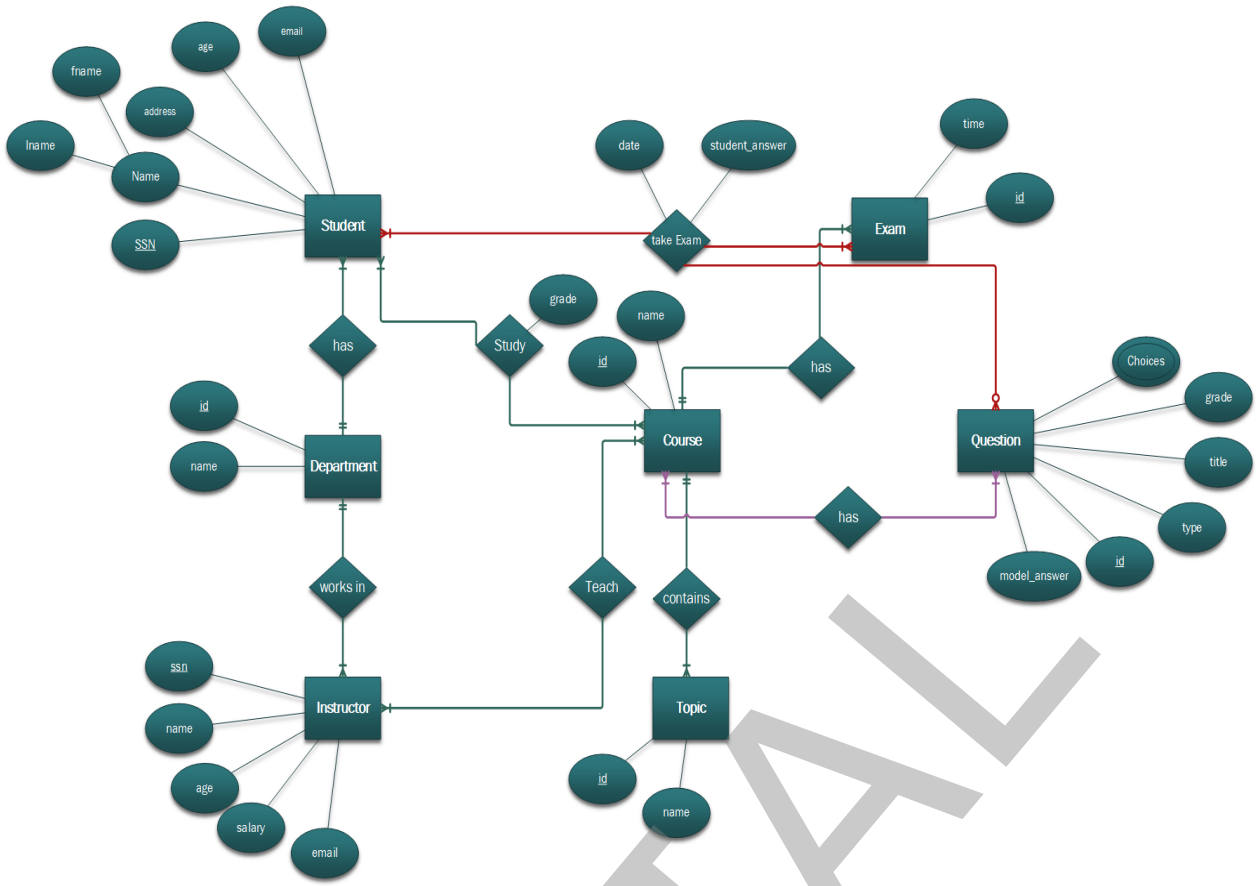
2-Heba Allah Ahmed

3-Asmaa Abdeen

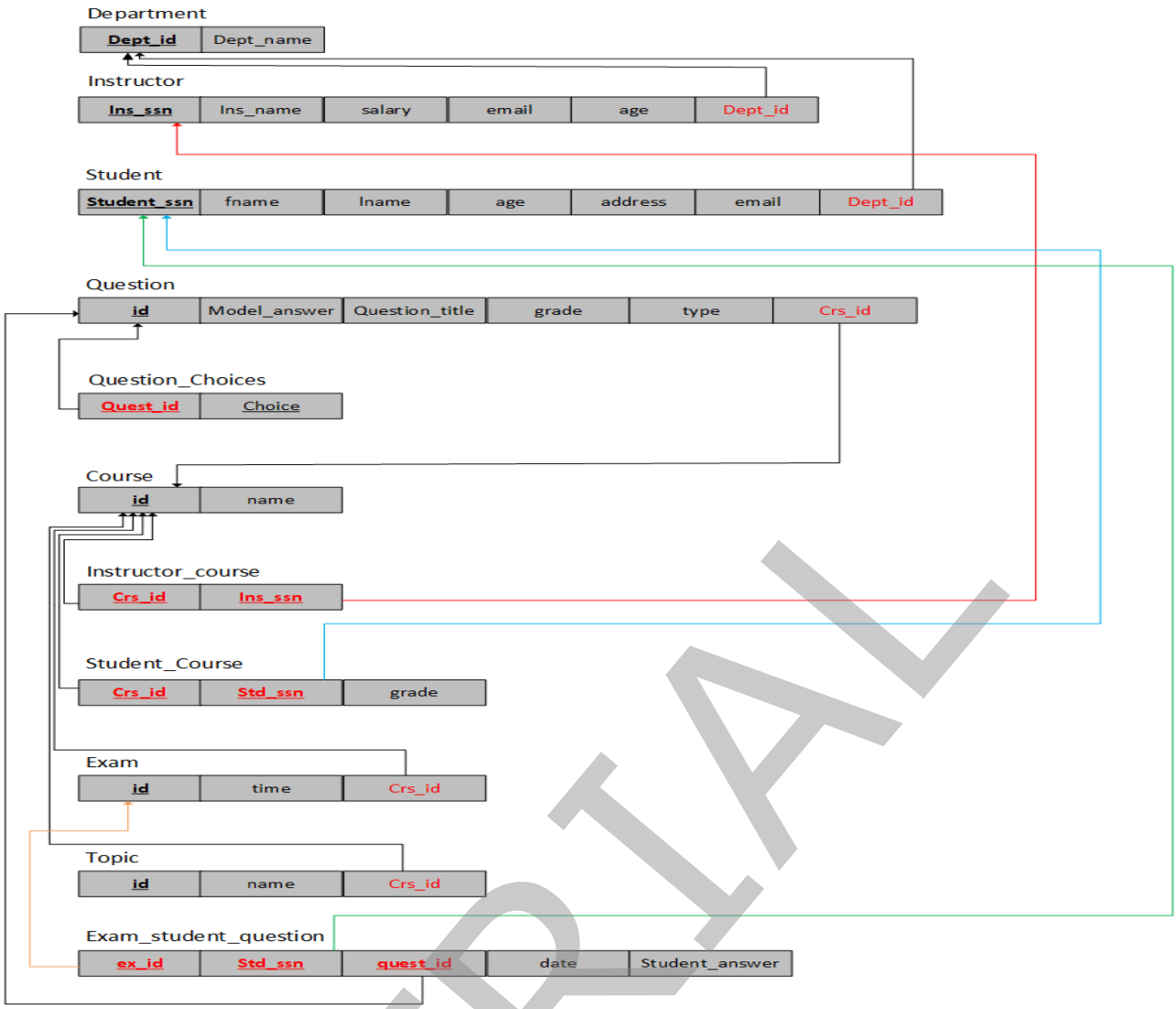
4-Maha Yehia

5-Ahmed Taha

2. ERD


email
age
fname
address
lname
Name
SSN
Student
date
student_answer
take Exam
time
id
Exam
grade
name
Study
id
has
has
Course
id
name
Department
Choices
grade
Question
title
has
type
id
model_answer
works in
Teach
contains
ssn
name
Instructor
Topic
age
salary
email
id
name

3. Mapping



4. Other

4.1. Tables

4.1.1. Table: dbo.Course

Columns

		Name	Data type	Description / Attributes
		id	int	Identity / Auto increment
		name	nvarchar(70)	

Linked from

	Table	Join	Title / Name / Description
	dbo.Exam	**dbo.Course**id = dbo.Examcrs_id	FK_Exam_Course
	dbo.Instructor_course	**dbo.Course**id = dbo.Instructor_coursecrs_id	FK_Instructor_course_Course
	dbo.Question	**dbo.Course**id = dbo.Questioncrs_id	FK_Question_Course
	dbo.Student_course	**dbo.Course**id = dbo.Student_coursecrs_id	FK_Student_course_Course2
	dbo.Topic	**dbo.Course**id = dbo.Topiccrs_id	FK_Topic_Course

Unique keys

	Columns	Name / Description
	id	PK_Course

Used By

Name
dbo.Course
dbo.Exam
dbo.Instructor_course
dbo.Question
dbo.Student_course
dbo.Topic

4.1.2. Table: dbo.Department

Columns

		Name	Data type	Description / Attributes
		dept_id	int	Identity / Auto increment
		dept_name	nvarchar(70)	Nullable

Linked from

	Table	Join	Title / Name / Description
	dbo.Instructor	dbo.Departmentdept_id = dbo.Instructordept_id	FK_Instructor_Department
	dbo.Student	dbo.Departmentdept_id = dbo.Studentdept_id	FK_Student_Department

Unique keys

	Columns	Name / Description
	dept_id	PK_Department

Used By

Name
dbo.Department
dbo.Instructor
dbo.Student

4.1.3. Table: dbo.Exam

Columns

		Name	Data type	Description / Attributes
		id	int	**Identity / Auto increment**
		time	int	
		crs_id	int	**References**: dbo.Course

Links to

	Table	Join	Title / Name / Description
	dbo.Course	**dbo.Exam**crs_id = dbo.Courseid	FK_Exam_Course

Linked from

	Table	Join	Title / Name / Description
	dbo.Exam_std_quest	**dbo.Exam**id = dbo.Exam_std_questex_id	FK_Exam_std_quest_Exam

Unique keys

	Columns	Name / Description
	id	PK_Exam

Uses

Name
dbo.Exam
dbo.Course

Used By

Name
dbo.Exam
dbo.Exam_std_quest

4.1.4. Table: dbo.Exam_std_quest

Columns

Name	Data type	Description / Attributes
ex_id	int	**References**: dbo.Exam
quest_id	int	**References**: dbo.Question
std_ssn	int	**References**: dbo.Student
date	datetime	
std_answer	nvarchar(20)	**Nullable**

Links to

Table	Join	Title / Name / Description
dbo.Exam	**dbo.Exam_std_quest**ex_id = dbo.Examid	FK_Exam_std_quest_Exam
dbo.Question	**dbo.Exam_std_quest**quest_id = dbo.Questionid	FK_Exam_std_quest_Question
dbo.Student	**dbo.Exam_std_quest**std_ssn = dbo.Studentssn	FK_Exam_std_quest_Student

Unique keys

Columns	Name / Description
ex_id, quest_id, std_ssn	PK_Exam_std_quest_1

Uses

Name
dbo.Exam_std_quest
dbo.Exam
dbo.Question
dbo.Student

4.1.5. Table: dbo.Instructor

Columns

		Name	Data type	Description / Attributes
		ins_ssn	int	
		ins_name	nvarchar(70)	
		salary	float	
		email	nvarchar(70)	**Nullable**
		age	int	**Nullable**
		dept_id	int	**Nullable** **References**: dbo.Department

Links to

	Table	Join	Title / Name / Description
	dbo.Department	**dbo.Instructor**dept_id = dbo.Departmentdept_id	FK_Instructor_Department

Linked from

	Table	Join	Title / Name / Description
	dbo.Instructor_course	**dbo.Instructor**ins_ssn = dbo.Instructor_courseins_id	FK_Instructor_course_Instructor

Unique keys

	Columns	Name / Description
	ins_ssn	PK_Instructor

Uses

Name
dbo.Instructor
dbo.Department

Used By

Name
dbo.Instructor
dbo.Instructor_course

4.1.6. Table: dbo.Instructor_course

Columns

Name	Data type	Description / Attributes
crs_id	int	**References**: dbo.Course
ins_id	int	**References**: dbo.Instructor

Links to

Table	Join	Title / Name / Description
dbo.Course	**dbo.Instructor_course**crs_id = dbo.Courseid	FK_Instructor_course_Course
dbo.Instructor	**dbo.Instructor_course**ins_id = dbo.Instructorins_ssn	FK_Instructor_course_Instructor

Unique keys

Columns	Name / Description
crs_id, ins_id	PK_Instructor_course

Uses

Name
dbo.Instructor_course
dbo.Course
dbo.Instructor

4.1.7. Table: dbo.Question

Columns

Name	Data type	Description / Attributes
id	int	**Identity / Auto increment**
model_ans	nvarchar(MAX)	
title	nvarchar(MAX)	
grade	int	
type	nvarchar(50)	
crs_id	int	**References**: dbo.Course

Links to

Table	Join	Title / Name / Description
dbo.Course	**dbo.Question**crs_id = dbo.Courseid	FK_Question_Course

Linked from

Table	Join	Title / Name / Description
dbo.Exam_std_quest	**dbo.Question**id = dbo.Exam_std_questquest_id	FK_Exam_std_quest_Question
dbo.Question_Choices	**dbo.Question**id = dbo.Question_Choicesques_id	FK_Question_Choices_Question

Unique keys

Columns	Name / Description
id	PK_Question

Triggers

Name	When	Description
tr_delete	Instead Of Delete	

```
create trigger tr_delete
on question
instead of delete
as select 'it is not allowed to delete'
```

Uses

Name
dbo.Question
dbo.Course

Used By

Name
dbo.Question
dbo.Exam_std_quest
dbo.Question_Choices

4.1.8. Table: dbo.Question_Choices

Columns

Name	Data type	Description / Attributes
ques_id	int	**References**: dbo.Question
choice	nvarchar(200)	

Links to

Table	Join	Title / Name / Description
dbo.Question	**dbo.Question_Choices**ques_id = dbo.Questionid	FK_Question_Choices_Question

Unique keys

Columns	Name / Description
ques_id, choice	PK_Question_Choices

Uses

Name
dbo.Question_Choices
dbo.Question

4.1.9. Table: dbo.Student

Columns

		Name	Data type	Description / Attributes
		ssn	int	
		fname	nvarchar(70)	
		lname	nvarchar(70)	**Nullable**
		age	int	**Nullable**
		address	nvarchar(90)	**Nullable**
		email	nvarchar(70)	
		dept_id	int	**Nullable** **References**: dbo.Department

Links to

	Table	Join	Title / Name / Description
	dbo.Department	**dbo.Student**dept_id = dbo.Departmentdept_id	FK_Student_Department

Linked from

	Table	Join	Title / Name / Description
	dbo.Exam_std_quest	**dbo.Student**ssn = dbo.Exam_std_queststd_ssn	FK_Exam_std_quest_Student
	dbo.Student_course	**dbo.Student**ssn = dbo.Student_coursestd_ssn	FK_Student_course_Student

Unique keys

	Columns	Name / Description
	ssn	PK_Student

Uses

Name
dbo.Student
dbo.Department

Used By

Name
dbo.Student
dbo.Exam_std_quest
dbo.Student_course

4.1.10. Table: dbo.Student_course

Columns

		Name	Data type	Description / Attributes
		crs_id	int	**References**: dbo.Course
		std_ssn	int	**References**: dbo.Student
		grade	float	**Nullable**

Links to

	Table	Join	Title / Name / Description
	dbo.Course	**dbo.Student_course**crs_id = dbo.Courseid	FK_Student_course_Course2
	dbo.Student	**dbo.Student_course**std_ssn = dbo.Studentssn	FK_Student_course_Student

Unique keys

	Columns	Name / Description
	crs_id, std_ssn	PK_Student_course

Uses

Name
dbo.Student_course
dbo.Course
dbo.Student

4.1.11. Table: dbo.Topic

Columns

		Name	Data type	Description / Attributes
		id	int	**Identity / Auto increment**
		name	nvarchar(50)	
		crs_id	int	**References**: dbo.Course

Links to

	Table	Join	Title / Name / Description
	dbo.Course	**dbo.Topic**crs_id = dbo.Courseid	FK_Topic_Course

Unique keys

	Columns	Name / Description
	id	PK_Topic

Uses

Name
dbo.Topic
dbo.Course

4.2. Procedures

4.2.1. Procedure: dbo.delete_course

Input/Output

	Name	Data type	Description
→@	id	int	

Script

```
--delete proc
create proc delete_course @id int
as
if exists(select id from course where id=@id)
begin---
delete from Question_Choices where ques_id in (select id from Question where crs_id = @id)
delete from Exam_std_quest where quest_id in (select id from question where crs_id=@id)
delete from Question where crs_id in (select id from course where id=@id)
delete from Instructor_course where crs_id =@id
delete from Student_course where crs_id=@id
delete from Exam where crs_id =@id
delete from Topic where crs_id=@id
delete  from Course where id=@id
end---
else select 'course not found'
```

4.2.2. Procedure: dbo.delete_department

Input/Output

	Name	Data type	Description
→@	id	int	

Script

```
---delete
create proc delete_department @id int
as
if exists(select dept_id from Department where dept_id=@id)
begin
update Instructor set dept_id=NULL where dept_id=@id
update Student set dept_id=NULL where dept_id=@id
delete  from Department where dept_id=@id
end
else select 'department not found'
```

4.2.3. Procedure: dbo.delete_exam

Input/Output

	Name	Data type	Description
→@	id	int	

Script

```
create procedure delete_exam @id int
as
delete from Exam_std_quest
where ex_id=@id
delete from exam
where id=@id

-------------------------------------------------------
```

4.2.4. Procedure: dbo.delete_exStdQuest

Input/Output

	Name	Data type	Description
→@	ex_id	int	
→@	quest_id	int	
→@	std_ssn	int	

Script

```
--delete from exam_std_quest
create proc delete_exStdQuest @ex_id int =0,@quest_id int =0,@std_ssn int =0
as
if(@ex_id<>0 and @quest_id=0 and @std_ssn=0)
            delete from Exam_std_quest where ex_id=@ex_id
else if (@ex_id=0 and @quest_id<>0 and @std_ssn=0)
            delete from Exam_std_quest where quest_id=@quest_id
else if (@ex_id=0 and @quest_id=0 and @std_ssn<>0)
            delete from Exam_std_quest where std_ssn=@std_ssn
else if (@ex_id<>0 and @quest_id<>0 and @std_ssn<>0)
            delete from Exam_std_quest where quest_id=@quest_id and ex_id=@ex_id and std_ssn=@std_ssn
else
            select 'you must enter either examId or StudentSSN or Question ID or ALL'
```

4.2.5. Procedure: dbo.delete_instructor

Input/Output

	Name	Data type	Description
→@	id	int	

Script

```
--delete ins by id
create proc delete_instructor @id int
as
if exists(select ins_ssn from Instructor where ins_ssn = @id)
begin
delete from Instructor where ins_ssn = @id
delete from Instructor_course where ins_id = @id
end
else select 'this id does not exist'
```

4.2.6. Procedure: dbo.delete_instructor_Course

Input/Output

	Name	Data type	Description
→@	ins_id	int	
→@	crs_id	int	

Script

```
--delete instructor_course
create proc delete_instructor_Course @ins_id int = 0,@crs_id int =0
as
if(@ins_id<>0 and @crs_id =0)
delete from Instructor_course where ins_id=@ins_id
else if (@ins_id =0 and @crs_id <>0)
           delete from Instructor_course where crs_id=@crs_id
else if(@ins_id<>0 and @crs_id<>0)
           delete from Instructor_course where ins_id=@ins_id and crs_id=@crs_id
```

4.2.7. Procedure: dbo.delete_question

Input/Output

	Name	Data type	Description
→@	qustion_id	int	

Script

```
create proc delete_question @qustion_id int
as
delete from Question_Choices where ques_id =@qustion_id
delete from Exam_std_quest where quest_id=@qustion_id
delete from Question
where id=@qustion_id


--delete based on id from question-choices table
```

4.2.8. Procedure: dbo.delete_question_choices

Input/Output

	Name	Data type	Description
→@	qustion_id	int	

Script

```
create proc delete_question_choices @qustion_id int
as
delete from Question_Choices where ques_id=@qustion_id


--update question table
```

4.2.9. Procedure: dbo.delete_student

Input/Output

	Name	Data type	Description
→@	id	int	

Script

```
--delete student by id
create proc delete_student @id int
as
if exists(select ssn from Student where ssn = @id)
begin
delete from Student where ssn = @id
delete from Exam_std_quest where std_ssn = @id
delete from Student_course where std_ssn = @id
end
else select 'this id does not exist'
```

4.2.10. Procedure: dbo.delete_student_course

Input/Output

	Name	Data type	Description
→@	id	int	
→@	ssn	int	

Script

```
-------------------------------------------
create procedure delete_student_course @id int=-1,@ssn int=-1
as
if @id!=-1 and @ssn!=-1
delete from Student_course
where crs_id=@id and std_ssn=@ssn
else if  @id!=-1 and @ssn=-1
delete from Student_course
where crs_id=@id
else if @id=-1 and @ssn!=-1
delete from Student_course
where std_ssn=@ssn
```

4.2.11. Procedure: dbo.delete_topic

Input/Output

	Name	Data type	Description
→@	id	int	

Script

```
---delete topic
create proc delete_topic @id int
as
delete from Topic where id=@id
```

4.2.12. Procedure: dbo.insert_course

Input/Output

	Name	Data type	Description
→@	name	nvarchar(20)	

Script

```
---insert
create proc insert_course  @name nvarchar(20)
as
insert into Course values(@name)
```

4.2.13. Procedure: dbo.insert_department

Input/Output

	Name	Data type	Description
→@	name	nvarchar(20)	

Script

```
---insert
create proc insert_department  @name nvarchar(20)
as
insert into Department values(@name)
```

4.2.14. Procedure: dbo.insert_exam

Input/Output

	Name	Data type	Description
→@	x	int	
→@	y	int	

Script

```
create procedure insert_exam @x int,@y int
as
begin try
insert into exam
values (@x,@y)
end try
begin catch
select 'error in foreign key'
end catch

------------------------------------
```

4.2.15. Procedure: dbo.insert_exStdQuest

Input/Output

	Name	Data type	Description
→@	ex_id	int	
→@	quest_id	int	
→@	std_ssn	int	
→@	date	date	
→@	std_answer	nvarchar(100)	

Script

```
--insert into exam_std_quest
create proc insert_exStdQuest @ex_id int , @quest_id int ,@std_ssn int , @date date , @std_answer nvarchar(100)
as
begin try
insert into Exam_std_quest values(@ex_id,@quest_id,@std_ssn,@date,@std_answer)
end try
begin catch
select 'Please enter valid data'
end catch
```

4.2.16. Procedure: dbo.insert_instructor

Input/Output

	Name	Data type	Description
→@	ssn	int	
→@	name	nvarchar(70)	
→@	sal	float	
→@	mail	nvarchar(70)	
→@	age	int	
→@	did	int	

Script

```
--insert new instructor
create proc insert_instructor @ssn int,@name nvarchar(70),@sal float,@mail nvarchar(70)=null,@age int=null,@did int
as
begin try
insert into Instructor values(@ssn,@name,@sal,@mail,@age,@did)
end try
begin catch
--select Error_message()
select 'an error happened while inserting'
end catch
```

4.2.17. Procedure: dbo.insert_instructor_Course

Input/Output

	Name	Data type	Description
→@	ins_id	int	
→@	crs_id	int	

Script

```
--insert into Instructor_Course
create proc insert_instructor_Course @ins_id int ,@crs_id int
as
begin try
insert into Instructor_course values(@crs_id,@ins_id)
end try
begin catch
select 'please enter valid data'
end catch
```

4.2.18. Procedure: dbo.insert_question

Input/Output

	Name	Data type	Description
→@	model_answer	nvarchar(90)	
→@	title	nvarchar(MAX)	
→@	grade	int	
→@	type	nvarchar(MAX)	
→@	course_id	int	

Script

```
create proc insert_question @model_answer nvarchar(90),@title nvarchar(max),@grade int,@type nvarchar(max),@course_id int
as
begin try
insert into Question
values(@model_answer,@title,@grade,@type,@course_id)
select 'data inserted successfully'
end try
begin catch
select 'an error happend from insert_question proc'
end catch


--insert all values into question-choices table
```

4.2.19. Procedure: dbo.insert_question_choices

Input/Output

	Name	Data type	Description
→@	question_id	int	
→@	choice	nvarchar(90)	

Script

```
create proc insert_question_choices @question_id int,@choice nvarchar(90)
as
begin try
insert into Question_Choices
values(@question_id,@choice)
end try
begin catch
select 'an error happend from insert_question_choices proc [this id not exist]'
end catch


--delete based on id from question table
```

4.2.20. Procedure: dbo.insert_student

Input/Output

	Name	Data type	Description
→@	ssn	int	
→@	fname	nvarchar(70)	
→@	lname	nvarchar(70)	
→@	age	int	
→@	add	nvarchar(90)	
→@	mail	nvarchar(70)	
→@	did	int	

Script

```
--insert new student
create proc insert_student @ssn int,@fname nvarchar(70),@lname nvarchar(70)=null,@age int=null , @add nvarchar(90)=null,@mail
nvarchar(70),@did int
as
begin try
insert into Student values(@ssn,@fname,@lname,@age,@add,@mail,@did)
end try
begin catch
select 'an error happened while inserting'
end catch
```

4.2.21. Procedure: dbo.insert_student_course

Input/Output

	Name	Data type	Description
→@	id	int	
→@	ssn	int	

Script

```
create procedure insert_student_course @id int,@ssn int
as
begin try
insert into student_course
values (@id,@ssn,null)
end try
begin catch
select 'error in insert'
end catch
```

4.2.22. Procedure: dbo.insert_topic

Input/Output

	Name	Data type	Description
→@	name	nvarchar(20)	
→@	crs_id	int	

Script

```
---insert new topic
create proc insert_topic @name nvarchar(20),@crs_id int
as
begin try
insert into Topic values(@name,@crs_id)
end try
begin catch
select 'an error happened while inserting in topic'
end catch
```

4.2.23. Procedure: dbo.select_allquestion_choices

Input/Output

	Name	Data type	Description
→@	id	int	

Script

```
create proc select_allquestion_choices @id int=0
as
if @id !=0
select ques_id as [QuestionId],choice as [Choices]  from Question_Choices
where ques_id=@id
else
select ques_id as [QuestionId],choice as [Choices]  from Question_Choices


--insert all values into question table
```

4.2.24. Procedure: dbo.select_course

Input/Output

	Name	Data type	Description
→@	id	int	

Script

```
-----------------------------------------------------------------------------
---course

create proc select_course  @id int =-1
as
if(@id!=-1)
select id as [course id] , name as [course name]  from Course where id=@id
else
select id as [course id] , name as [course name]  from Course
```

4.2.25. Procedure: dbo.select_department

Input/Output

	Name	Data type	Description
→@	id	int	
→@	name	nvarchar(50)	

Script

```
-----------------------------------------------------------------------------------------------
---department
create proc select_department @id int =-1 , @name nvarchar(50)=' '
as
if @id=-1 and @name!=' '
begin
if exists (select dept_name from Department where dept_name=@name )
select dept_id as [dept id] , dept_name as [dept name] from Department where dept_name=@name
else select 'department not found'
end
else if @id!=-1 and @name=' '
begin
if exists (select dept_id from Department where dept_id=@id )
select dept_id as [dept id] , dept_name as [dept name] from Department where dept_id=@id
else select 'department not found'
end
else
select dept_id as [dept id] , dept_name as [dept name] from Department
```

4.2.26. Procedure: dbo.select_exam

Input/Output

	Name	Data type	Description
→@	x	int	

Script

```
-----------------------------------------------------
--mohamed stored proc

create procedure select_exam @x int=-1
as
if @x!=-1
           select e.id as exam_id,time as exam_time,crs_id as crs_time,name as crs_name
           from exam e,Course c
           where e.crs_id=c.id and e.id=@x
else
select e.id as exam_id,time as exam_time,crs_id as crs_time,name as crs_name
from exam e,Course c
where e.crs_id=c.id
 ----------------------------------
```

4.2.27. Procedure: dbo.select_exStdQuest

Input/Output

	Name	Data type	Description
→@	ex_id	int	
→@	std_id	int	
→@	quest_id	int	

Script

```
--select from exam_std_quest
create proc select_exStdQuest @ex_id int=0 , @std_id int =0 , @quest_id int =0
as
if (@ex_id<>0 and @std_id <>0 and @quest_id<>0)
            select  esq.ex_id examID , esq.std_ssn studentSSN,CONCAT(s.fname,' ',s.lname) StudentName, esq.quest_id
QuestionID,q.title QuestionHeader,
            esq.date as ExamDate, esq.std_answer StudentAnswer
            from Exam_std_quest esq inner join Student s
            on esq.std_ssn=s.ssn inner join Question q
            on esq.quest_id=q.id inner join Exam e
            on esq.ex_id=e.id
            where esq.ex_id=@ex_id and esq.std_ssn=@std_id and esq.quest_id=@quest_id
            else
            select  esq.ex_id examID , esq.std_ssn studentSSN,CONCAT(s.fname,' ',s.lname) StudentName, esq.quest_id
QuestionID,q.title QuestionHeader,
            esq.date as ExamDate, esq.std_answer StudentAnswer
            from Exam_std_quest esq inner join Student s
            on esq.std_ssn=s.ssn inner join Question q
            on esq.quest_id=q.id inner join Exam e
            on esq.ex_id=e.id
```

4.2.28. Procedure: dbo.select_instructor

Input/Output

	Name	Data type	Description
→@	ssn	int	
→@	name	nvarchar(70)	

Script

```
-----------------------------------------------------------------------------
-----------------------------------------------------------------------------


--select instructor (all || by fname || by ssn)
create proc select_instructor @ssn int = -1,@name nvarchar(70)=' '
as
if @ssn = -1 and @name !=' '
begin
if exists(select ins_name from Instructor where ins_name = @name)
select ins_ssn SSN,ins_name [name],salary Ins_Salary,email Ins_Email,age Ins_Age,i.dept_id [department id],dept_name
[department name]
from Instructor i,Department d where d.dept_id=i.dept_id and ins_name = @name
else select 'no matching instructor'
end
else if  @ssn != -1 and @name =' '
begin
if exists(select ins_ssn from Instructor where ins_ssn = @ssn)
select ins_ssn SSN,ins_name [name],salary Ins_Salary,email Ins_Email,age Ins_Age,i.dept_id [department id],dept_name
[department name]
from Instructor i,Department d where d.dept_id=i.dept_id and ins_ssn = @ssn
else select 'no matching instructor'
end
else select ins_ssn SSN,ins_name [name],salary Ins_Salary,email Ins_Email,age Ins_Age,i.dept_id [department id],dept_name
[department name]
from Instructor i,Department d where d.dept_id=i.dept_id
```

4.2.29. Procedure: dbo.select_instructor_Course

Input/Output

	Name	Data type	Description
→@	ins_id	int	
→@	crs_id	int	

Script

```
--*--------------------------------
--taha stored proc


--select from InstructorCourse
create proc select_instructor_Course @ins_id int=0 ,@crs_id int =0
as
if(@ins_id=0 and @crs_id<>0)
           select crs_id as courseID ,c.name as courseName,ins_id as instructorID ,i.ins_name InstructorName   from
Instructor_course ic
           inner join Course c on ic.crs_id=c.id  inner join Instructor i on i.ins_ssn=ic.ins_id
           where ic.crs_id=@crs_id
else if(@crs_id=0 and @ins_id<>0)
           select crs_id as courseID ,c.name as courseName,ins_id as instructorID ,i.ins_name InstructorName   from
Instructor_course ic
           inner join Course c on ic.crs_id=c.id  inner join Instructor i on i.ins_ssn=ic.ins_id
           where ic.ins_id=@ins_id
else if(@ins_id=0 and @crs_id=0)
           select crs_id as courseID ,c.name as courseName,ins_id as instructorID ,i.ins_name InstructorName   from
Instructor_course ic
           inner join Course c on ic.crs_id=c.id  inner join Instructor i on i.ins_ssn=ic.ins_id
else
           select crs_id as courseID ,c.name as courseName,ins_id as instructorID ,i.ins_name InstructorName   from
Instructor_course ic
           inner join Course c on ic.crs_id=c.id  inner join Instructor i on i.ins_ssn=ic.ins_id
           where ic.crs_id=@crs_id and ic.ins_id=@ins_id
```

4.2.30. Procedure: dbo.select_question

Input/Output

	Name	Data type	Description
→@	qustion_id	int	

Script

```
---------------------------------
--heba stored and trigger
--select all from question table
create proc select_question @qustion_id int=0
as
if @qustion_id !=0
select q.id as [QuestionID],q.model_ans as [Model Answer],q.title as [Question Title], q.grade as [Question Grade]
,q.type as [Question Type], q.crs_id as [Course Id],c.name as [Course Name]
from Question q inner join Course c on q.crs_id=c.id
where q.id= @qustion_id
else
select q.id as [QuestionID],q.model_ans as [Model Answer],q.title as [Question Title], q.grade as [Question Grade]
,q.type as [Question Type], q.crs_id as [Course Id],c.name as [Course Name]
from Question q inner join Course c on q.crs_id=c.id
--select all from question-choices table
```

4.2.31. Procedure: dbo.select_student

Input/Output

	Name	Data type	Description
→@	ssn	int	
→@	fname	nvarchar(70)	

Script

```
--asmaa stored proc
--select Student (all || by fname || by ssn)
create proc select_student @ssn int = -1,@fname nvarchar(70)=' '
as
if @ssn = -1 and @fname !=' '
begin
if exists(select fname from Student where fname = @fname)
select ssn Std_ssn,fname [first name],lname[last name],age std_age,address std_address,email std_Email,s.dept_id [department
id],dept_name [department name]
from Student s, Department d where s.dept_id=d.dept_id and fname = @fname
else select 'no matching student'
end
else if  @ssn != -1 and @fname =' '
begin
if exists(select ssn from Student where ssn = @ssn)
select ssn Std_ssn,fname [first name],lname[last name],age std_age,address std_address,email std_Email,s.dept_id [department
id],dept_name [department name]
from Student s, Department d where s.dept_id=d.dept_id and ssn = @ssn
else select 'no matching student'
end
else select ssn Std_ssn,fname [first name],lname[last name],age std_age,address std_address,email std_Email,s.dept_id
[department id],dept_name [department name]
from Student s, Department d where s.dept_id=d.dept_id
```

4.2.32. Procedure: dbo.select_student_course

Input/Output

	Name	Data type	Description
→@	id	int	
→@	ssn	int	

Script

```
-------------------------------
create procedure select_student_course @id int=-1,@ssn int=-1
as
if @id!=-1 and @ssn!=-1
select crs_id as crs_id,std_ssn as stu_ssn ,ssn as student_ssn,fname as f_name,c.id as crs_id,c.name as crs_name
from Student_course sc inner join Student s
on s.ssn=sc.std_ssn inner join Course c
on sc.crs_id=c.id
where sc.crs_id=@id and sc.std_ssn=@ssn

else if @id!=-1 and @ssn=-1
select crs_id as crs_id,std_ssn as stu_ssn ,ssn as student_ssn,fname as f_name,c.id as crs_id,c.name as crs_name
from Student_course sc inner join Student s
on s.ssn=sc.std_ssn inner join Course c
on sc.crs_id=c.id
where sc.crs_id=@id

else if @id=-1 and @ssn!=-1
select crs_id as crs_id,std_ssn as stu_ssn ,ssn as student_ssn,fname as f_name,c.id as crs_id,c.name as crs_name
from Student_course sc inner join Student s
on s.ssn=sc.std_ssn inner join Course c
on sc.crs_id=c.id
where sc.std_ssn=@ssn
else
select crs_id as crs_id,std_ssn as stu_ssn ,ssn as student_ssn,fname as f_name,c.id as crs_id,c.name as crs_name
from Student_course sc inner join Student s
on s.ssn=sc.std_ssn inner join Course c
on sc.crs_id=c.id

------------------------------------
```

4.2.33. Procedure: dbo.select_topic

Input/Output

	Name	Data type	Description
→@	id	int	
→@	name	nvarchar(20)	

Script

```
-----------------------------------------------------------------------------------------
--maha stored proc

---select topic
create proc select_topic  @id int =-1 , @name nvarchar(20) =' '
as
if @id =-1 and @name !=' '
begin
IF EXISTS(select name from Topic where name=@name )
select t.id as [topic id] , t.name as [topic name],t.crs_id as [Course Id],c.name as [Course Name] from Topic t inner join
course c
on t.crs_id=c.id
where t.name=@name
else select 'topic not found' as [message]
end
else if @id !=-1 and @name =' '
begin
IF EXISTS(select id from Topic where id=@id)
select t.id as [topic id] , t.name as [topic name],t.crs_id as [Course Id],c.name as [Course Name] from Topic t inner join
course c
on t.crs_id=c.id
where t.id=@id
else select 'topic not found' as [message]
end
else
select t.id as [topic id] , t.name as [topic name],t.crs_id as [Course Id],c.name as [Course Name] from Topic t inner join
course c
on t.crs_id=c.id
```

4.2.34. Procedure: dbo.update_course

Input/Output

	Name	Data type	Description
→@	id	int	
→@	name	nvarchar(50)	

Script

```
---update
create proc update_course @id int , @name nvarchar(50)
as
if exists(select id from Course where id=@id)
update Course set name = @name where id=@id
else select 'course not found'
```

4.2.35. Procedure: dbo.update_department

Input/Output

	Name	Data type	Description
→@	id	int	
→@	name	nvarchar(50)	

Script

```
---update
create proc update_department @id int , @name nvarchar(50)
as
if exists(select dept_id from Department where dept_id=@id)
begin
update Department set dept_name = @name where dept_id=@id
end
else select 'department not found'
```

4.2.36. Procedure: dbo.update_exam

Input/Output

	Name	Data type	Description
→@	id	int	
→@	time	int	
→@	crsid	int	

Script

```
create procedure update_exam @id int,@time int=-1,@crsid int=-1
as
if exists(select* from exam where id=@id)
begin
begin try
if @time!=-1
update exam set time=@time where id=@id
if @crsid!=-1
update exam set crs_id=@crsid where id=@id
end try
begin catch
select 'error in foreign key'
end catch
end
else
select 'no matched id'

---------------------------------------
```

4.2.37. Procedure: dbo.update_exStdQuest

Input/Output

	Name	Data type	Description
→@	ex_id	int	
→@	quest_id	int	
→@	std_ssn	int	
→@	date	date	
→@	std_answer	nvarchar(100)	

Script

```
--update exam_std_quest
create proc update_exStdQuest @ex_id int,@quest_id int,@std_ssn int ,@date date ='',@std_answer nvarchar(100)=''
as
begin try
if(@date<>'')
            update Exam_std_quest set date=@date where ex_id=@ex_id and quest_id=@quest_id and std_ssn=@std_ssn
if(@std_answer<>'')
update Exam_std_quest set std_answer=@std_answer where ex_id=@ex_id and quest_id=@quest_id and std_ssn=@std_ssn
end try
begin catch
select 'Please enter valid data'
end catch
```

4.2.38. Procedure: dbo.update_instructor

Input/Output

	Name	Data type	Description
→@	ssn	int	
→@	newssn	int	
→@	name	nvarchar(70)	
→@	age	int	
→@	sal	int	
→@	mail	nvarchar(70)	
→@	did	int	

Script

```
--update instructor data
create proc update_instructor @ssn int,@newssn int =0,@name nvarchar(70)='NA',@age int=0 , @sal int=0,@mail
nvarchar(70)='NA',@did int=0
as
if exists(select ins_ssn from Instructor where ins_ssn = @ssn)
begin
begin try

if @name != 'NA'
update Instructor set ins_name=@name where ins_ssn = @ssn
if @age != 0
update Instructor set age=@age where ins_ssn = @ssn
if @sal != 0
update Instructor set salary=@sal where ins_ssn = @ssn
if @mail != 'NA'
update Instructor set email=@mail where ins_ssn = @ssn
if @did != 0
update Instructor set dept_id=@did where ins_ssn = @ssn
if @newssn != 0
update Instructor set ins_ssn=@newssn where ins_ssn = @ssn
end try
begin catch
select 'error,can not update'
end catch
end
else select 'no matched ssn'
```

4.2.39. Procedure: dbo.update_instructor_Course

Input/Output

	Name	Data type	Description
→@	crs_id	int	
→@	ins_id	int	
→@	ins_newId	int	
→@	crs_newId	int	

Script

```
--update instructor_Course
create proc update_instructor_Course @crs_id int , @ins_id int,@ins_newId int=0 , @crs_newId int=0
as
begin try
if(@ins_newId<>0)
            update Instructor_course set ins_id=@ins_newId where crs_id=@crs_id and ins_id=@ins_id
if(@crs_newId<>0 and @ins_newId=0)
            update Instructor_course set crs_id=@crs_newId where crs_id=@crs_id and ins_id=@ins_id
if(@crs_newId<>0 and @ins_newId<>0)
            update Instructor_course set crs_id=@crs_newId where crs_id=@crs_id and ins_id=@ins_newId
end try
begin catch
select 'please enter valid data'
end catch
```

4.2.40. Procedure: dbo.update_question

Input/Output

	Name	Data type	Description
→@	qustion_id	int	
→@	model_answer	nvarchar(90)	
→@	title	nvarchar(MAX)	
→@	grade	int	
→@	type	nvarchar(MAX)	
→@	course_id	int	

Script

```
create proc update_question @qustion_id int,
@model_answer nvarchar(90)='NA'
,@title nvarchar(max)='NA'
,@grade int=0
,@type nvarchar(max)='NA'
,@course_id int=0
as
if exists(select id from Question where id=@qustion_id)
begin
begin try

if @model_answer != 'NA'
update Question set model_ans=@model_answer where id= @qustion_id
if @title != 'NA'
update Question set title=@title where id = @qustion_id
if @grade != 0
update Question set grade=@grade where id = @qustion_id
if @type != 'NA'
update Question set type=@type where id = @qustion_id
if @course_id != 0
update Question set crs_id=@course_id where id = @qustion_id

end try
begin catch
select 'an error happend from update_question proc'
end catch
end


--update question-choices table
```

4.2.41. Procedure: dbo.update_question_choice

Input/Output

	Name	Data type	Description
→@	qustion_id	int	
→@	choice	nvarchar(90)	
→@	oldchoice	nvarchar(90)	

Script

```
create proc update_question_choice @qustion_id int,@choice nvarchar(90),@oldchoice nvarchar(90)
as
begin try
update Question_Choices set choice=@choice
where ques_id=@qustion_id and choice=@oldchoice
end try
begin catch
select 'an error happend from update_question_choice proc'
end catch
```

4.2.42. Procedure: dbo.update_student

Input/Output

	Name	Data type	Description
→@	ssn	int	
→@	newssn	int	
→@	fname	nvarchar(70)	
→@	lname	nvarchar(70)	
→@	age	int	
→@	add	nvarchar(90)	
→@	mail	nvarchar(70)	
→@	did	int	

Script

```
--update student data  --on update cascade
create proc update_student @ssn int,@newssn int =0,@fname nvarchar(70)='NA',@lname nvarchar(70)='NA',@age int=0 , @add
nvarchar(90)='NA',@mail nvarchar(70)='NA',@did int=0
as
if exists(select ssn from Student where ssn =@ssn)
begin
begin try
if @fname != 'NA'
update Student set fname=@fname where ssn = @ssn
if @lname != 'NA'
update Student set lname=@lname where ssn = @ssn
if @age != 0
update Student set age=@age where ssn = @ssn
if @add != 'NA'
update Student set address=@add where ssn = @ssn
if @mail != 'NA'
update Student set email=@mail where ssn = @ssn
if @did != 0
update Student set dept_id=@did where ssn = @ssn
if @newssn != 0
update Student set ssn=@newssn where ssn = @ssn
end try
begin catch
select 'error,can not update'
end catch
end
else select 'no matched ssn'
```

4.2.43. Procedure: dbo.update_student_course

Input/Output

	Name	Data type	Description
→@	crs_is	int	
→@	std_ssn	int	
→@	crs_newId	int	
→@	std_newId	int	

Script

```
-------------------------------------------
create proc update_student_course @crs_is int , @std_ssn int,@crs_newId int=0 , @std_newId int=0
as
begin try
if(@crs_newId<>0)
            update student_course set crs_id=@crs_newId where crs_id=@crs_is and std_ssn=@std_ssn
if(@std_ssn<>0)
            update student_course set std_ssn=@std_newId where crs_id=@crs_is and std_ssn=@std_ssn
if(@crs_newId<>0 and @std_newId<>0)
            update student_course set crs_id=@crs_newId where crs_id=@crs_is and std_ssn=@std_ssn
end try
begin catch
select 'please enter valid data'
end catch
```

4.2.44. Procedure: dbo.update_topic

Input/Output

	Name	Data type	Description
→@	id	int	
→@	name	nvarchar(50)	
→@	crs_id	int	

Script

```
---update topic
create proc update_topic @id int , @name nvarchar(50)='',@crs_id int=0
as
if exists(select id from Topic where id=@id)
begin
begin try
if(@name<>'')
update topic set name = @name where id=@id
if(@crs_id<>0)
update Topic set crs_id=@crs_id where id=@id
end try
begin catch
select 'an error happened while updating in topic'
end catch
end
else select 'topic not found'
```

4.2.45. Procedure: ExamStored.Exam_Correction

Input/Output

	Name	Data type	Description
→@	exam_id	int	
→@	std_id	int	

Script

```
--
create proc Exam_Correction      @exam_id int , @std_id int
as
declare @Totalgrade decimal(5,1)=0
declare @studentgrade decimal(5,1)=0
declare @percent decimal(5,1)=0
declare c1 cursor
for select std_answer,model_ans,grade from Exam_std_quest esq inner join Question q on
esq.quest_id=q.id
where ex_id=@exam_id and std_ssn=@std_id
for read only
declare @studentAnswer nvarchar(20)
declare @model_answer nvarchar(20)
declare @grade int
open c1
fetch c1 into @studentAnswer,@model_answer,@grade
while @@FETCH_STATUS=0
begin
           if(TRIM(@studentAnswer)=TRIM(@model_answer))
                      begin
                                 set @Totalgrade+=@grade
                                 set @studentgrade+=@grade
                      end
           else
           begin
           set @Totalgrade+=@grade
           end
           set @percent = (@studentgrade/@Totalgrade)*100
fetch c1 into @studentAnswer,@model_answer,@grade
end
select CONCAT(@percent,'%') as StudentGrade
close c1
deallocate c1
declare @crs_id int
select @crs_id=e.crs_id from Exam_std_quest esq inner join Exam e
on e.id = esq.ex_id
where esq.std_ssn=@std_id and esq.ex_id=@exam_id
update Student_course set grade = @percent
where crs_id=@crs_id and std_ssn=@std_id
```

4.2.46. Procedure: ExamStored.Exam_Generation

Input/Output

	Name	Data type	Description
→@	crs_type	nvarchar(50)	
→@	std_ssn	int	
→@	t_fQ	int	
→@	M_Q	int	

Script

```
----------------------------------------
--exam stored proc

create proc Exam_Generation @crs_type nvarchar(50), @std_ssn int , @t_fQ int , @M_Q int
as
if (@t_fQ + @M_Q <>10)
select 'Enter valid number of question'
else
begin
begin try
declare @crs_id int
declare @questions table (quest_id int)
-- create new exam
select @crs_id=id from Course where name=@crs_type
insert into Exam values (60,@crs_id)
-- insert question

insert into @questions
select top(@t_fQ) id from Question
where type='t/f'and crs_id=@crs_id
order by NEWID()
--
insert into @questions
select top(@M_Q) id from Question
where type='mcq'and crs_id=@crs_id
order by NEWID()
-- create new exam
-- generated 10 question
declare @CurrentExam_id int
select top(1)@CurrentExam_id=id from Exam order by id desc
declare c1 cursor
for select * from @questions
for read only
declare @id int
open c1
fetch c1 into @id
while @@FETCH_STATUS=0
begin
insert into Exam_std_quest values (@CurrentExam_id,@id,@std_ssn,GETDATE(),null)
fetch c1 into @id
end
close c1
deallocate c1
select * from Exam_std_quest esq
inner join Question q
on esq.quest_id=q.id
where
ex_id=@CurrentExam_id and std_ssn=@std_ssn
order by quest_id
end try
begin catch
select 'An error has occured'
end catch
end
```

4.2.47. Procedure: ExamStored.ExamAnswers

Input/Output

	Name	Data type	Description
→@	exam_id	int	
→@	std_id	int	
→@	a1	nvarchar(20)	
→@	a2	nvarchar(20)	
→@	a3	nvarchar(20)	
→@	a4	nvarchar(20)	
→@	a5	nvarchar(20)	
→@	a6	nvarchar(20)	
→@	a7	nvarchar(20)	
→@	a8	nvarchar(20)	
→@	a9	nvarchar(20)	
→@	a10	nvarchar(20)	

Script

```
-- procedure to enter answers
create proc ExamAnswers @exam_id int ,@std_id int ,
@a1 nvarchar(20),@a2 nvarchar(20),@a3 nvarchar(20),@a4 nvarchar(20),
@a5 nvarchar(20),@a6 nvarchar(20),@a7 nvarchar(20),@a8 nvarchar(20),
@a9 nvarchar(20),@a10 nvarchar(20)
as
--
if exists(select * from Exam_std_quest where ex_id=@exam_id and std_ssn=@std_id)
begin
begin try
declare @answers_table table(ans nvarchar(20))
insert into @answers_table values(@a1),(@a2),(@a3),(@a4),(@a5),(@a6),(@a7),(@a8),(@a9),(@a10)
declare c1 cursor
for select quest_id from Exam_std_quest where ex_id=@exam_id and std_ssn=@std_id
for read only
declare @question_id int
open c1
fetch c1 into @question_id
declare c2 cursor
for select ans from @answers_table
for read only
declare @ans nvarchar(20)
open c2
fetch c2 into @ans
while @@FETCH_STATUS=0
begin
update Exam_std_quest set std_answer =@ans where ex_id=@exam_id and std_ssn=@std_id and quest_id=@question_id
fetch c1 into @question_id
fetch c2 into @ans
end
close c1
close c2
deallocate c1
deallocate c2
end try
begin catch
select 'An error has occured'
end catch
end
else
select 'enter valid data'

--
```

4.2.48. Procedure: Reports.CourseTopics

Input/Output

	Name	Data type	Description
→@	crs_id	int	

Script

```
create proc CourseTopics @crs_id int
as
select c.name as [course Name],t.name as [topic Name] from Course c inner join Topic t
on c.id=t.crs_id
where c.id=@crs_id
```

4.2.49. Procedure: Reports.freeFormReport

Input/Output

	Name	Data type	Description
→@	ex_id	int	

Script

```
-- this is the free form report
create proc freeFormReport @ex_id int
as
declare @counter int =1
declare c1 cursor
for
select q.title,LEAD(q.title) over (order by id) as titleNext ,qc.choice from Question q inner join Question_Choices qc
on q.id=qc.ques_id inner join Exam_std_quest esq on
esq.quest_id=q.id where esq.ex_id=@ex_id
for read only
declare @currentTitle nvarchar(200),@nextTitle nvarchar(200),@choice nvarchar(200)
declare @showTable table (title nvarchar(200),choice nvarchar(200))
open c1
fetch c1 into @currentTitle,@nextTitle,@choice
while @@FETCH_STATUS=0
begin
declare @title nvarchar(200)= concat(@counter,') ',@currentTitle)
insert into @showTable values (@title,@choice)
if @currentTitle<>@nextTitle
set @counter=@counter+1
fetch c1 into @currentTitle,@nextTitle,@choice
end
close c1
deallocate c1
select * from @showTable
```

4.2.50. Procedure: Reports.InstructorCoursesR3

Input/Output

	Name	Data type	Description
→@	ins_id	int	

Script

```
create proc InstructorCoursesR3 @ins_id int
as
select i.ins_name,c.name,Count(sc.crs_id) as [Student No] from Instructor_course ic inner join Instructor i
on ic.ins_id=i.ins_ssn inner join Student_course sc
on sc.crs_id=ic.crs_id inner join Course c
on c.id=sc.crs_id
where i.ins_ssn=@ins_id
group by i.ins_name,c.name
```

4.2.51. Procedure: Reports.StudentAnswersR6

Input/Output

	Name	Data type	Description
→@	ex_id	int	
→@	std_id	int	

Script

```
create proc StudentAnswersR6  @ex_id int ,@std_id int
as
select esq.std_ssn, q.title,esq.std_answer ,q.model_ans from Exam_std_quest esq inner join Question q
on esq.quest_id=q.id
where esq.ex_id=@ex_id and esq.std_ssn=@std_id
```

4.2.52. Procedure: Reports.StudentGradesR2

Input/Output

	Name	Data type	Description
→@	std_id	int	

Script

```
create proc StudentGradesR2 @std_id int
as
select sc.std_ssn,c.name,sc.grade from Student_course sc inner join Course c
on c.id=sc.crs_id
where sc.std_ssn=@std_id
```

4.2.53. Procedure: Reports.StudentInfoR1

Input/Output

	Name	Data type	Description
→@	dept_no	int	

Script

```
------------------------------
--report stored proc

create proc StudentInfoR1 @dept_no int
as
select s.ssn,s.fname,s.lname,s.email,s.address,s.age,d.dept_name from Student s inner join Department d
on s.dept_id=d.dept_id
where s.dept_id=@dept_no
```