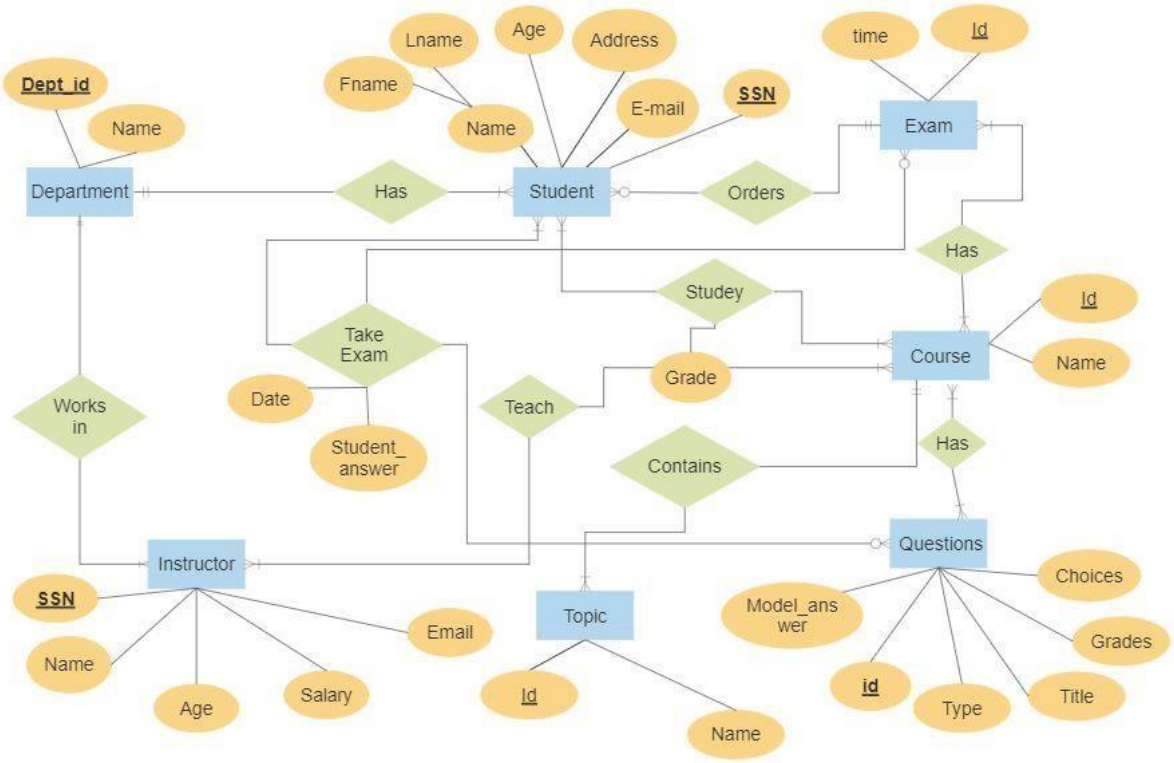
Team members:

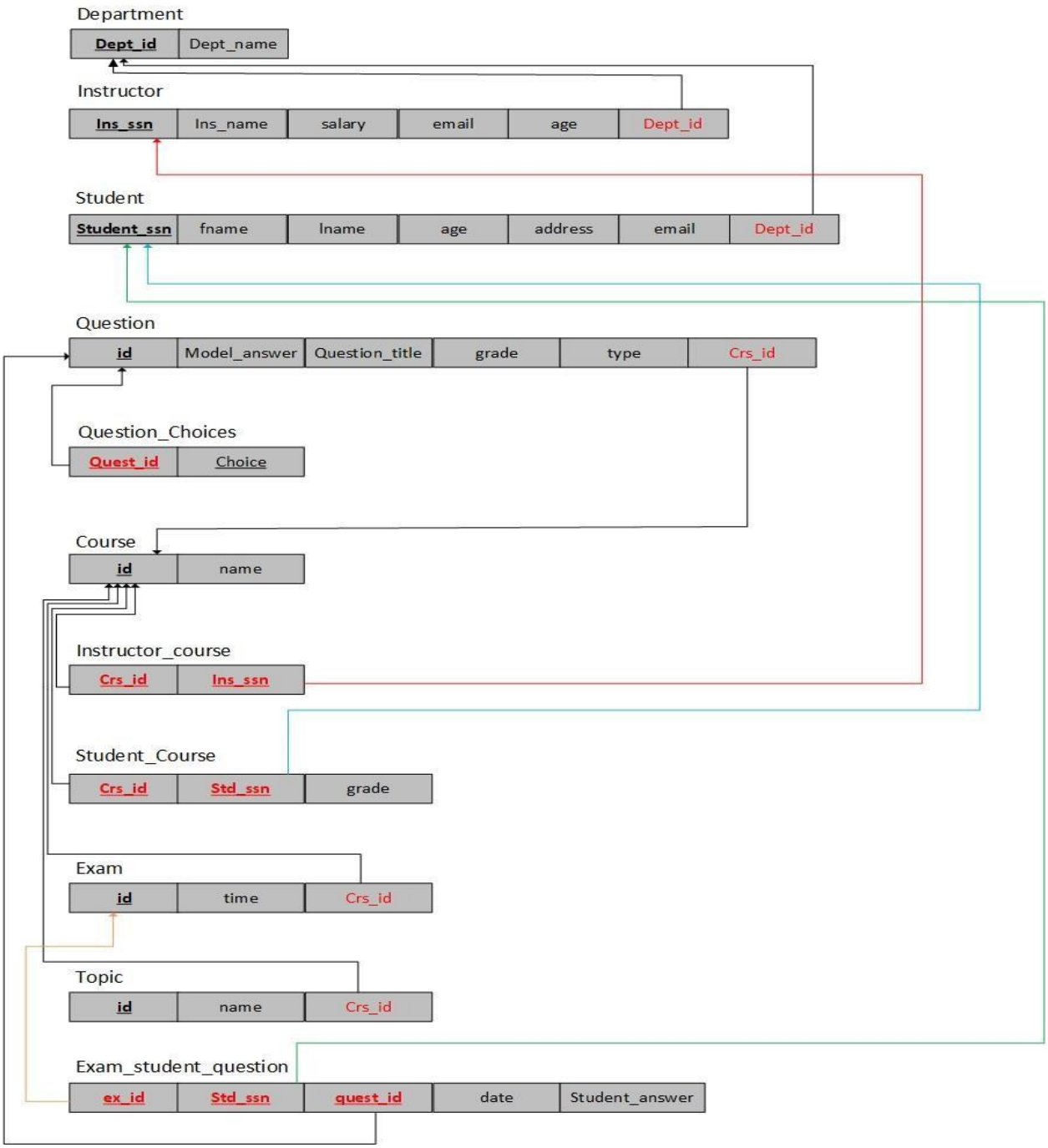
1) Aya Adel
2) Mariam Saad
3) Micheal Ibrahim
4) Mohamd Alaa
5) Yasser Momtaz

Exam System

Project

ERD





Tables

Table Course:

Columns

	Column Name	Data Type	Allow Nulls
	Id	int	☐
	name	varchar(50)	☑
			☐

Dependencies

Dependencies

- Course
  - course_delete
  - course_insert
  - course_select
  - course_update
  - Exam
  - Instructor_course
  - selectQuestion
  - Student_Course
  - Topic

Table Department:

Columns

	Column Name	Data Type	Allow Nulls
	Dept_id	int	☐
	Dept_name	varchar(50)	☐
			☐

Dependencies

Dependencies

- Department
  - delete_dept
  - insert_dept
  - Instructor
  - Student
  - update_dept
  - view_depert

Table exam:

Columns

	Column Name	Data Type	Allow Nulls
	id	int	☐
	duration	int	☐
	Crs_id	int	☐
			☐

Dependencies

Dependencies

- Exam
  - Exam_student_question
  - insert_exam
  - select_exam
  - update_exam

Table exam:

Columns

Column Name	Data Type	Allow Nulls
id	int	☐
duration	int	☐
Crs_id	int	☐
		☐

Dependencies

Dependencies

- Exam
  - Exam_student_question
  - insert_exam
  - select_exam
  - update_exam

Table Exam_std_quest:

Columns

Column Name	Data Type	Allow Nulls
ex_id	int	☐
Std_ssn	int	☐
quest_id	int	☐
date	datetime	☑
Student_answer	varchar(50)	☑

Dependencies

Dependencies

- Exam_student_question
  - delete_exam
  - delete_exstdques
  - Enter_Answers
  - Exam_Correction
  - Exam_Generation
  - insert_exstdques
  - select_exstdques
  - update_exstdques

Table instructor:

Columns

	Column Name	Data Type	Allow Nulls
	Ins_ssn	int	☐
	Ins_name	varchar(50)	☑
	salary	int	☑
	email	varchar(50)	☑
	Bday	date	☑
	Dept_id	int	☑
			☐

Dependencies

Dependencies

- Instructor
  - delete_instr
  - insert_instr
  - Instructor_course
  - update_instr

Table instructor_course:

Columns

Column Name	Data Type	Allow Nulls
Crs_id	int	☐
ins_ssn	int	☐
		☐

Dependencies

Dependencies

- Instructor_course
  - delete_Instructor_Course
  - insert_Instructor_Course
  - select_Instructor_Course
  - update_Instructor_Course

Table Qusetion:

Columns

Table Qusetion_Choices:

Column Name	Data Type	Allow Nulls
id	int	☐
model_ans	varchar(50)	☑
title	varchar(MAX)	☑
grade	int	☑
type	varchar(50)	☑
crs_id	int	☑
		☐

Dependencies

Dependencies

- Question
  - deleteQuestion
  - Exam_student_question
  - insertQuestion
  - Question_Choices
  - selectQuestion
  - updateQuestion

Columns

Column Name	Data Type	Allow Nulls
ques_id	int	☐
choice	nvarchar(180)	☐
		☐

Dependencies

Dependencies

- Question_Choices
  - deleteAllChoices
  - deleteChoice
  - insertChoice
  - selectChoice
  - UpdateChoice

Table Student:

Columns

Column Name	Data Type	Allow Nulls
Student_ssn	int	☐
fname	varchar(90)	☑
lname	varchar(90)	☑
Bday	date	☑
address	varchar(90)	☑
email	varchar(90)	☑
Dept_id	int	☐
		☐

Dependencies

Dependencies

- Student
  - Exam_student_question
  - Student_Course
  - student_delete
  - student_Insert
  - student_select
  - student_Update

Table Student_Course:

Columns

	Column Name	Data Type	Allow Nulls
	Crs_id	int	☐
	Std_ssn	int	☐
	grade	int	☑
			☐

Dependencies

Dependencies

- Student_Course
  - delete_Student_Course
  - Exam_Correction
  - insert_Student_Course
  - select_Student_Course
  - update_Student_Course

Table topic:

Columns

Column Name	Data Type	Allow Nulls
id	int	☐
name	varchar(50)	☑
Crs_id	int	☑
		☐

Dependencies

Dependencies

- Topic
  - delete_topic
  - ins_topic
  - select_topic
  - update_topic

Procedures

Department

```
/*insert into depart procedure*/
create proc insert_dept(@id Integer, @Dept_name varchar(20))
as BEGIN

            INSERT INTO Department
                        (Dept_id,Dept_name
                         )
            VALUES      (@id,@Dept_name )
END  insert_dept 9,
'cloud'


/*update depart procedure*/
alter proc update_dept(@id Integer, @Dept_name varchar(20))
as BEGIN

            UPDATE Department
            SET    Dept_name = @Dept_name
            WHERE  Dept_id = @id
END
```

```
update_dept 10,'network'



/*delete  depart by name or id
procedure*/
 alter proc delete_dept( @Dept_name varchar(20)=NULL ,@id
Integer=NULL) as
BEGIN
/*if you want to delete by id*/      if(@Dept_name
is NULL)

              begin
             DELETE FROM Department
WHERE  Dept_id = @id
 end        else if(@id is
NULL)
              begin
              DELETE FROM Department
        WHERE  Dept_name = @Dept_name
              end

END




    delete_dept 'sim',NULL
/*select  depart procedure*/
  create proc view_depert
as
       begin
     select*from Department
       end
 view_depert
```

	Dept_id	Dept_name
1	1	PD
2	2	AI
3	3	SA
4	4	OS
5	9	cloud

Instructor

```
/*insert into instructor procedure*/
  create proc insert_instr(@id Integer, @name varchar(20),@salary
int,@email
```

```
varchar(50),@bday datetime,@dept_id int)
as BEGIN


            INSERT INTO Instructor

(Ins_ssn,Ins_name,salary,email,Bday,Dept_id)

            VALUES     (@id,@name,@salary,@email,@bday,@dept_id) END
insert_instr 9,'ouf',3000,'@ouf.com','1999-01-01
00:00:00.000',2  /*update instructor procedure*/
   create proc update_instr(@ssn Integer, @name varchar(20),@dept_id
int=NULL,@update_type
varchar(20)) as BEGIN
        if(@update_type='name')
              begin
            UPDATE Instructor
            SET    Ins_name = @name
            WHERE  Ins_ssn = @ssn
              end
            else if(@update_type='department')
              begin
              UPDATE Instructor
            SET    dept_id = @dept_id
            WHERE  Ins_ssn = @ssn
              end



END
     update_instr  9 ,null ,3,
'department'


/*delete  instructor by name or id procedure*/
  create proc delete_instr( @name varchar(20)=NULL ,@ssn
Integer=NULL)
as
BEGIN
/*if you want to delete by id*/      if(@name
is NULL)

              begin
             DELETE FROM Instructor
WHERE  Ins_ssn = @ssn
 end        else if(@ssn is
NULL)       begin
              DELETE FROM Instructor
        WHERE  ins_name =@name
              end
  END
delete_instr 'mahmoudouf',null
/*view instructor data */
create Proc view_instructor( @col
```

```
varchar(20),@cond varchar(20)) as begin
     execute('select '+@col+' from
'+'Instructor'+' where '+@cond) end
execute view_instructor '*','Ins_ssn=2'
```

	Ins_ssn	Ins_name	salary	email	Bday	Dept_id
1	2	karim	3000	karim@yahoo.com	1975-01-01	1

```
create
proc instcrsR3 @ins_ssn 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_ssn=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_ssn group by
i.Ins_name , c.name
----instcrsR3 6 ----only data in 2 and 6
instcrsR3
2
```

	Ins_name	name	Student No
1	karim	DB	10

Student

```
-- insert of student
  create proc student_Insert @ssn int,@fname varchar(50),@lname
varchar(50)=null,@Bday
datetime=null,@address varchar(50)=null,@email varchar(50)=null,@deptid int  as begin
try
       insert into Student values(@ssn,@fname,@lname,@Bday,@address,@email,@deptid)
end try begin catch
       select 'something wrong with
data' end catch  go

-- update of student
  create proc student_Update @ssn int,@newssn int=0,@fname
varchar(50)='',@lname
varchar(50)='',@Bday datetime='',@address varchar(50)='',@email varchar(50)='',@deptid
int=0 as
if exists(select Student_ssn from Student where Student_ssn=@ssn) begin
       begin try
              if @newssn !=0
                     update Student set Student_ssn=@newssn where Student_ssn=@ssn
       else if @fname !=''
                    update Student set fname=@fname where Student_ssn=@ssn
     else if @lname !=''
                    update Student set lname=@lname where Student_ssn=@ssn
     else if @Bday !=''
      update Student set Bday=@Bday where Student_ssn=@ssn     else if @address
!=''
```

```
      update Student set                              address=@address where
Student_ssn=@ssn     else if @email !=''
                   update Student set                 email=@email where
Student_ssn=@ssn                 else if @deptid !=0
                   if exists(select     Dept_id from Department where Dept_id=@deptid)
                    update Student set Dept_id=@deptid
where Student_ssn=@ssn                  else select 'departement is not available'
     end try       begin catch     select 'invalid data'     end
catch
end
else select 'student is not available'
go
-- delete of student   create proc
student_delete @ssn int as
     if exists(select Student_ssn from Student where Student_ssn=@ssn)
     begin
              delete from Student where Student_ssn=@ssn
       end
     else select 'student is not available'
go
 -- select of student
  create proc student_select @ssn int=0,@fname varchar(50)='', @deptid int=0
as  if @ssn=0 and @fname='' and @deptid=0 begin   select s.Student_ssn ,
s.fname +' '+ s.lname as [Full name],s.Bday
[Birthday],s.address ,s.email,s.Dept_id,d.Dept_name
     from Student s,Department d    where d.Dept_id=s.Dept_id end else
if @ssn !=0 and @fname='' and @deptid=0 begin        if exists(select
Student_ssn from Student where Student_ssn=@ssn)     begin
              select s.Student_ssn , s.fname +' '+ s.lname as [Full name],s.Bday
[Birthday],s.address ,s.email,s.Dept_id,d.Dept_name
            from Student s,Department d
            where d.Dept_id=s.Dept_id and s.Student_ssn=@ssn  end
       else select 'Student Not Available'
end
else if @ssn=0 and @deptid=0 and @fname!='' begin    if
exists(select Student_ssn from Student where fname=@fname)
              begin
                     select s.Student_ssn , s.fname +' '+ s.lname as [Full name],s.Bday
[Birthday],s.address ,s.email,s.Dept_id,d.Dept_name
                   from Student s,Department d                where
d.Dept_id=s.Dept_id and s.fname=@fname
              end
       else select 'Student Not Available'
end
else if @ssn=0 and @deptid!=0 and @fname='' begin
       if exists(select Student_ssn from Student where Dept_id=@deptid)
              begin
                     select s.Student_ssn , s.fname +' '+ s.lname as [Full name],s.Bday
[Birthday],s.address ,s.email,s.Dept_id,d.Dept_name
                     from Student s,Department d
                     where d.Dept_id=s.Dept_id and s.Dept_id=@deptid
              end
       else select 'Student Not
```

```
Available' end        go
student_select 0, 'michael'
```

	Student_ssn	Full name	Birthday	address	email	Dept_id	Dept_name
1	2	Michael Ibrahim	1997-03-11	alex	michael@yahoo.com	2	AI

```
create proc StudentinfoR1 @dept_no int as
select s.Student_ssn,s.fname,s.lname,s.email,s.address,s.Bday,d.Dept_name from Student s
inner join Department d  on s.Dept_id=d.Dept_id where s.Dept_id=@dept_no  ----StudentinfoR1
1 Go

StudentinfoR1 2
```

	Student_ssn	fname	lname	email	address	Bday	Dept_name
1	2	Michael	Ibrahim	michael@yahoo.com	alex	1997-03-11	AI
2	4	Mariam	saad	mariam@yahoo.com	cairo	1998-01-01	AI

```
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 go
StudentgradesR2 5
```

	Std_ssn	name	grade
1	5	DB	NULL
2	5	OS	NULL
3	5	SA1	NULL

```
create proc stdansR6  @ex_id int ,@std_id int as
select esq.Std_ssn, q.title,esq.Student_answer ,q.model_ans from Exam_student_question
esq , Question q  where esq.quest_id=q.id
and esq.ex_id=@ex_id and esq.Std_ssn=@std_id
stdansR6
9,1s
```

	Std_ssn	title	Student_answer	model_ans
1	1	Which of the following isn't a level of abstraction?	c	c
2	1	A logical structure of the database.	a	a
3	1	Consider money is transferred from (1)account-A ...	a	c
4	1	Identify the characteristics of transactions	a	d
5	1	means that the data used during the execution of...	a	d
6	1	In a database, data is stored in spreadsheets whi...	b	b
7	1	A database has data and relationships.	c	a
8	1	The purpose of a database is to help people stop ...	b	b
9	1	A database has a built-in capability to create, proc...	d	b
10	1	Prior to 1970, all data was stored in separate files...	b	a

course

course

```
                                                                    -insert
course

create proc course_insert @id int, @name
varchar(50) as
if exists(select Id from Course where Id=@id)
 select 'Course is Already Exists' else
insert into Course values (@id,@name)
 go
--update course
  create proc course_update @id int,@newid int=0, @name varchar(50)=''
as  if exists (select Id  from Course where Id=@id) begin          if
@newid=0 and @name!=''
              update Course set name=@name where Id=@id
       else if @newid!=0 and @name=''
              update Course set Id=@newid where Id=@id
       else if @newid!=0 and @name!=''
            update Course set Id=@newid,name=@name where Id=@id end
else select 'Course is not Avalilable'   go
--delete course
create proc course_delete @id int=0,@name varchar(50)='' as
if @id !=0 and @name='' begin   if exists (select
Id from Course where Id=@id)           delete from
Course where Id=@id       else select 'course is
not available' end
else if @id =0 and @name!='' begin      if exists
(select name from Course where name=@name)
     delete from Course where name=@name       else
select 'course is not available' end
go


--select course
create proc course_select @id int=0,@name varchar(50)=''
as
if @id=0 and @name=''     select
Id,name from Course
else if @id!=0 and @name=''
       select Id,name from Course where Id=@id
else if @id=0 and @name!=''
     select Id,name from Course where name=@name   go
course_select 2
```

	Id	name
1	2	OS

```
create proc course_topicsR4 @id int as
```

```
select c.name as [course name], t.name                as [topic name] from Topic t , Course
c  where t.Crs_id=c.Id and
c.Id=@id
 -----course_topicsR4 1  go
                                  create
                                proc
```

	course name	topic name
1	OS	DeadLock
2	OS	Scheduling

```
examquestR5 @ex_id int  as
 select e.id,q.title,qc.choice  from                  Question q, Question_Choices qc ,
Exam_student_question  esq  ,  Exam  e
where e.id=esq.ex_id and
esq.quest_id=q.id and q.id=qc.ques_id                 and
e.id=@ex_id  go    course_topicsR4
2
```

Question

```
go  create proc selectQuestion @id int
= -1
as  if(@id!=-1) begin
select  q.id , q.title ,q.type , q.model_ans , q.grade , q.crs_id , c.name
from Question q inner join Course c on q.crs_id=c.Id where q.id=@id end else
begin
select  q.id , q.title ,q.type , q.model_ans , q.grade , q.crs_id , c.name
from Question q inner join Course c on q.crs_id=c.Id end
selectQuestion 10
```

	id	title	type	model_ans	grade	crs_id	name
1	10	Identify the characteristics of transactions	C	d	1	1	DB

```
 go
create proc insertQuestion @title nvarchar(max) ,@type nvarchar(2), @model_Ans
nvarchar(2),@crs_ID int, @grade int=1 as begin try insert into Question
values (@title,@type,@model_Ans,@grade,@crs_ID)
end try  begin catch
select 'An Error occured will entring data please enter 5 valid parameters' end
catch

---Delete Question---
--update Relationship diagram "on delete casscade" go  create
proc deleteQuestion @qustion_id int
as begin try  delete from Question where
id=@qustion_id end try begin catch
select 'please enter a valid ID' end
catch
----Update Question---- go
create proc updateQuestion @qustion_id int,
@model_answer nvarchar(50)='NoChange'
```

```
,@title nvarchar(max)='NoChange'
,@grade int=0
,@type nvarchar(50)='NoChange'
,@course_id int=0 as  if exists(select id from
Question where id=@qustion_id)
begin begin try  if
@model_answer != 'NoChange'
update Question set
model_ans=@model_answer where id=                    @qustion_id if
@title != 'NoChange'  update Question set title=@title
where id = @qustion_id if
@grade != 0  update Question set grade=@grade where id =
@qustion_id if
@type != 'NoChange'  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 occured please enter valid parameters '
end catch end else  begin
select 'this questions ID does not exist' end
```

Question_Choices

```
---Select Choice--- go
create proc selectChoice @id int = -1
as  if(@id!=-1) begin
select  q.id , q.title , c.choice , q.model_ans from Question q
inner join Question_Choices c on q.id=c.ques_id where q.id=@id end
else begin
select  q.id , q.title , c.choice , q.model_ans
from Question q inner join Question_Choices c on q.id=c.ques_id
end
selectChoice 10
```

	id	title	choice	model_ans
1	10	Identify the characteristics of transactions	a) Atomicity	d
2	10	Identify the characteristics of transactions	b) Durability	d
3	10	Identify the characteristics of transactions	c) Isolatio	d
4	10	Identify the characteristics of transactions	d) All of the mentioned	d

```
 ---Insert Choice---
go  create proc insertChoice @Q_id int,@choice
nvarchar(90)
as begin try
insert into Question_Choices values(@Q_id,@choice)  select
'Choice inserted Successfully'
end try begin catch
select 'an error occurred [this id not exist]' end
catch

---Update Choice--- go
```

```
create proc UpdateChoice @Q_id                          int,@choice
nvarchar(100),@newChoice nvarchar(100)                  as  if exists (select ques_id from
Question_Choices where ques_id=@Q_id and                choice=@choice)
begin begin try update Question_Choices set choice = @newChoice
where ques_id=@Q_id and choice=@choice
end try begin catch
select 'An Error occurred please enter valid parameters'
end catch end else
select 'This Choice Does not exist please enter valid data'
---Delete Choice --- go
create proc deleteChoice @qustion_id int , @Choice nvarchar(100) as
if exists (select choice from                           Question_Choices where
ques_id=@qustion_id and choice=@Choice)                 begin
delete from Question_Choices
where ques_id=@qustion_id and
choice=@Choice end else
select 'This Choice Does not exist'
---Delete All Choices --- go
create proc deleteAllChoices @qustion_id int  as
if exists (select ques_id from Question_Choices where ques_id=@qustion_id) begin
delete from Question_Choices
where ques_id=@qustion_id
end else
```

Topic

```
              ---- insert into topic---

select 'This Choice Does not exist'
```

```
create proc ins_topic @id int , @name varchar(100), @crs_id int
as  begin try
insert into Topic values(@id,@name,@crs_id)
end try  begin catch
select 'error in insert topic please check if crs id is exsit' end
catch   go


---select topic ----
create proc select_topic @id int= -1, @name varchar(100)=' ' as
if @id=-1 and @name!=' ' ---select with name if it not null
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, Course c where t.Crs_id=c.Id and
t.name=@name
```

```
else select 'topic not found' end
else if @id!=-1 and @name=' '---select              with id if not -1
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, Course c where t.Crs_id=c.Id and
t.id=@id
else select 'topic not found' end   else
-- select with both id and 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, Course c  where
t.Crs_id=c.Id
```

```
select_topic 1
```

	topic_id	topic_name	course_id	course_name
1	1	T_sql	1	DB

```
---update topic-------------
create proc update_topic @id int, @name varchar(100) =' ',@crs_id int =0 as  if
exists(select id from Topic where id=@id)
begin begin try
if(@name!=' ') --update with name update
Topic set name= @name where id=@id
if(@crs_id!=0)--update with crsid 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' ----update_topic 1
,@crs_id=4
----update_topic 1 ,'second'
go
---delete topic----- create proc delete_topic
@id int  as
delete from Topic where id=@id
go
```

Exam

```
----select from exam with id---
create proc select_exam @id int=-1
as  if @id!=-1 begin
if exists(select id from Exam where
id=@id)
select e.id as exam_id, e.time as exam_time,c.name as course_name from Exam e, Course c
where e.Crs_id=c.Id and
e.id=@id
else select 'exam not found'
end else  select 'exam not found'
```

Exam

```
---select_exam 2


----insert into exam--
```

```
create proc insert_exam @id int,                    @duration int=3,@crs_id int
as  begin try
insert into Exam                                    values(@id,@duration,@crs_id)
end try begin catch
select 'an error happened while inserting in exam'
end catch
---insert_exam 3,@crs_id=3  go   -----
update exam----
create procedure update_exam @id int,@time int=-1,@crsid int=-1 as
if exists(select id from Exam where id=@id)
begin begin try if @time!=-1
update Exam set duration=@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' ----
update_exam 1,@crsid=1 go
------delete from exam-------------- create
procedure delete_exam @id int as
delete from Exam_student_question ---delete from child first
where ex_id=@id delete from Exam where id=@id ----
delete_exam 1
Go
```

exam-student-question

exam-student-question

```
 create proc select_exstdques @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 as exam_id, esq.Std_ssn as student_id,CONCAT(s.fname,' ',s.lname)
as
StudentName, esq.quest_id as QuestionID,q.title as QuestionHeader,
esq.date as ExamDate, esq.Student_answer as StudentAnswer    from
Exam_student_question esq , Student s,Exam e , Question q   where
esq.Std_ssn=s.Student_ssn and
     esq.ex_id=e.id and   esq.quest_id=q.id
and
       esq.ex_id=@ex_id
else if (@ex_id=0 and @std_id !=0 and @quest_id=0)
select esq.ex_id as exam_id, esq.Std_ssn as student_id,CONCAT(s.fname,' ',s.lname) as
StudentName, esq.quest_id as QuestionID,q.title as QuestionHeader,
esq.date as ExamDate, esq.Student_answer as StudentAnswer    from
Exam_student_question esq , Student s,Exam e , Question q   where
esq.Std_ssn=s.Student_ssn and   esq.ex_id=e.id and
       esq.quest_id=q.id and esq.Std_ssn=@std_id
else if (@ex_id=0 and @std_id =0 and @quest_id!=0)
select esq.ex_id as exam_id, esq.Std_ssn as student_id,CONCAT(s.fname,' ',s.lname) as
StudentName, esq.quest_id as QuestionID,q.title as QuestionHeader,
esq.date as ExamDate, esq.Student_answer as StudentAnswer    from
Exam_student_question esq , Student s,Exam e , Question q   where
esq.Std_ssn=s.Student_ssn and
```

```
     esq.ex_id=e.id and
     esq.quest_id=q.id and
esq.quest_id=@quest_id
else
         select esq.ex_id as exam_id, esq.Std_ssn as student_id,CONCAT(s.fname,' ',s.lname)
as StudentName, esq.quest_id as QuestionID,q.title as QuestionHeader,
 esq.date as ExamDate, esq.Student_answer as StudentAnswer    from Exam_student_question
esq , Student s,Exam e , Question q   where esq.Std_ssn=s.Student_ssn and
       esq.ex_id=e.id and
       esq.quest_id=q.id
  select_exstdques
3,1,3
```

	exam_id	student_id	StudentName	QuestionID	QuestionHeader	ExamDate	StudentAnswer
1	9	1	Aya adel	2	Which of the following isn't a level of abstraction?	2023-03-18 01:33:14.417	c
2	9	1	Aya adel	4	A logical structure of the database.	2023-03-18 01:33:14.417	a
3	9	1	Aya adel	8	Consider money is transferred from (1)account-A t...	2023-03-18 01:33:14.417	a
4	9	1	Aya adel	10	Identify the characteristics of transactions	2023-03-18 01:33:14.417	a
5	9	1	Aya adel	14	means that the data used during the execution of...	2023-03-18 01:33:14.417	a
6	9	1	Aya adel	19	In a database, data is stored in spreadsheets whi...	2023-03-18 01:33:14.413	b
7	9	1	Aya adel	20	A database has data and relationships.	2023-03-18 01:33:14.387	c
8	9	1	Aya adel	24	The purpose of a database is to help people stop ...	2023-03-18 01:33:14.413	b
9	9	1	Aya adel	28	A database has a built-in capability to create, proc...	2023-03-18 01:33:14.413	d
10	9	1	Aya adel	30	Prior to 1970, all data was stored in separate files...	2023-03-18 01:33:14.413	b
11	10	5	Mohamed Alaa	2	Which of the following isn't a level of abstraction?	2023-03-19 01:26:31.897	NULL
12	10	5	Mohamed Alaa	5	The actual content in the database at a particular ...	2023-03-19 01:26:31.893	NULL
13	10	5	Mohamed Alaa	8	Consider money is transferred from (1)account-A t...	2023-03-19 01:26:31.893	NULL
14	10	5	Mohamed Alaa	10	Identify the characteristics of transactions	2023-03-19 01:26:31.897	NULL
15	10	5	Mohamed Alaa	15	. Locks placed by command are called	2023-03-19 01:26:31.893	NULL
16	10	5	Mohamed Alaa	20	A database has data and relationships.	2023-03-19 01:26:31.893	NULL

```
--insert into exam_std_quest-----------------------------
create proc insert_exstdques @ex_id int,@std_ssn int ,@quest_id int , @std_answer
varchar(10) as begin try
insert into Exam_student_question values(@ex_id,@std_ssn,@quest_id,GETDATE(),@std_answer)
end try begin catch
select 'Please enter valid data' end
catch
---insert_exstdques 2,1,5,'1.1.2015', 'c'

go
-------update date and answer---------------------------------
create proc update_exstdques @ex_id int,@std_ssn int , @quest_id int,@date date
='',@std_answer varchar(10)=''  as
begin
try
if(@date!='') ----change the value of date with the three primary keys
 update Exam_student_question set date=@date where ex_id=@ex_id and
quest_id=@quest_id and Std_ssn=@std_ssn  if(@std_answer!='')----change the value
of std answer with the three primary keys
update Exam_student_question set Student_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
```

```
 -----update_exstdques 2,1,1,'1.2.2018'
 ----------- update_exstdques 2,1,5,
@std_answer='d'  -----update_exstdques              2,1,5 ,' '----
nothing   go   -----------delete ---------------------
 create proc delete_exstdques @ex_id int =0,@std_ssn int =0,@quest_id int =0
as
 if(@ex_id!=0 and @quest_id=0 and @std_ssn=0)----delete with exam id only
delete from Exam_student_question where ex_id=@ex_id
 else if (@ex_id=0 and @quest_id!=0 and @std_ssn=0) ---delete with q id only
delete from Exam_student_question where quest_id=@quest_id
 else if (@ex_id=0 and @quest_id=0 and @std_ssn!=0) ---delete with std id only
delete from Exam_student_question where Std_ssn=@std_ssn
 else if (@ex_id!=0 and @quest_id!=0 and @std_ssn!=0)---delete with the 3 primary
 delete from Exam_student_question 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'  ------
delete_exstdques 2
```

exam generation ,correction and result

```
--Exam Generation
alter proc Exam_Generation @std_ssn int , @crs_name varchar(50), @t_f int , @choice int
as
if (@t_f + @choice !=10)   select
'Enter 10 Questions '  else
begin
     begin try     declare
@crs_id int
              declare @questions table (ques_id int)
```

```
        --create exam
               --insert in Exam table
              select @crs_id=Id from                  Course where
  name=@crs_name            insert into Exam values(60,@crs_id)
       --insert t/f questions
insert into @questions               select
  top(@t_f) id
              from Question              where
  type='T' and crs_id=@crs_id
         order by NEWID()
              --insert choose questions
  insert into @questions           select
  top(@choice) id            from Question
       where type='C' and crs_id=@crs_id
         order by NEWID()
              -- insert in exam_std_questions table     declare
  @current_exam int
               select top(1)                          @current_exam=  id

  from Exam order by id desc                              declare c1 cursor

         for select * from @questions             for read only
   declare @ques_id int
               open c1
               fetch c1 into @ques_id

   while @@FETCH_STATUS=0
                      begin
                             insert into Exam_student_question values
  (@current_exam,@std_ssn,@ques_id,GETDATE(),null)
                             fetch c1 into
  @ques_id                 end           close c1
       deallocate c1
              select *
              from Exam_student_question ex ,Question q              where
  ex.quest_id=q.id and ex.ex_id=@current_exam and ex.Std_ssn=@std_ssn        order
  by ex.quest_id

        end try       begin catch
   select 'error happened'   end catch end
```

```
Exam_Generation  2,'OS',          2,8
```

	ex_id	Std_ssn	quest_id	date	Student_answer	id	model_ans	title	grade	type	crs_id
1	12	2	31	2023-03-20 20:00:59.737	NULL	31	b	When several processes access the same data conc...	1	C	2
2	12	2	34	2023-03-20 20:00:59.737	NULL	34	a	A semaphore is a shared integer variable	1	C	2
3	12	2	35	2023-03-20 20:00:59.737	NULL	35	c	Mutual exclusion can be provided by the	1	C	2
4	12	2	38	2023-03-20 20:00:59.737	NULL	38	a	To enable a process to wait within the monitor	1	C	2
5	12	2	39	2023-03-20 20:00:59.737	NULL	39	c	Concurrent access to shared data may result in	1	C	2
6	12	2	41	2023-03-20 20:00:59.737	NULL	41	b	The segment of code in which the process may chan...	1	C	2
7	12	2	42	2023-03-20 20:00:59.737	NULL	42	d	Which of the following conditions must be satisfied to...	1	C	2
8	12	2	44	2023-03-20 20:00:59.737	NULL	44	a	What are the two atomic operations permissible on s...	1	C	2

```
--Enter Answers
alter proc Enter_Answers @exam_id int ,@std_ssn int ,@a1 varchar(10),@a2 varchar(10),@a3
varchar(10) ,@a4 varchar(10),@a5 varchar(10),@a6 varchar(10),@a7 varchar(10),@a8
varchar(10),@a9 varchar(10),@a10 varchar(10)        as
if exists(select * from Exam_student_question where ex_id=@exam_id and Std_ssn=@std_ssn)
begin
     begin try            declare @ans_table table (answers
varchar(20))     insert into @ans_table
values(@a1),(@a2),(@a3),(@a4),(@a5),(@a6),(@a7),(@a8),(@a9),(@a10)
              declare c1 cursor
  for select quest_id from Exam_student_question where ex_id=@exam_id and
Std_ssn=@std_ssn
              for read only
            declare @question_id int
            open c1
              fetch c1 into @question_id
        declare c2 cursor
               for select * from
@ans_table            for read only
        declare @ans varchar(20)
               open c2
               fetch c2 into @ans
        while @@FETCH_STATUS=0
                      begin
                              update Exam_student_question set Student_answer=@ans where
ex_id=@exam_id and Std_ssn=@std_ssn 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 'error happened'
end catch
end
else select 'Enter Valid data'
```

```
Exam_Generation 2,'DB',5,5
go
Enter_Answers 9,1,'c','a','a','a','a','b','c','b','d','b'
```

	ex_id	Std_ssn	quest_id	date	Student_answer	id	model_ans	title	grade	type	crs_id
1	14	2	2	2023-03-20 20:04:13.077	NULL	2	c	Which of the following isn't a level of abstraction?	1	C	1
2	14	2	3	2023-03-20 20:04:13.077	NULL	3	a	A level that describes how a record is stored.	1	C	1
3	14	2	7	2023-03-20 20:04:13.077	NULL	7	d	A level that describes data stored in a database and ...	1	C	1
4	14	2	8	2023-03-20 20:04:13.077	NULL	8	c	Consider money is transferred from (1)account-A to ...	1	C	1
5	14	2	10	2023-03-20 20:04:13.077	NULL	10	d	Identify the characteristics of transactions	1	C	1
6	14	2	17	2023-03-20 20:04:13.073	NULL	17	a	The relational database model was created by E.F. ...	1	T	1
7	14	2	19	2023-03-20 20:04:13.073	NULL	19	b	In a database, data is stored in spreadsheets which ...	1	T	1
8	14	2	22	2023-03-20 20:04:13.073	NULL	22	b	In an enterprise-class database system, business u...	1	T	1
9	14	2	29	2023-03-20 20:04:13.073	NULL	29	b	Enterprise Resource Planning (ERP) is an example...	1	T	1
10	14	2	30	2023-03-20 20:04:13.073	NULL	30	a	Prior to 1970, all data was stored in separate files, w...	1	T	1

```
--Exam correction
alter proc Exam_Correction @exam_id int, @st_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 ex.Student_answer,q.model_ans,q.grade from Exam_student_question ex,Question q
where ex.ex_id=@exam_id and ex.Std_ssn=@st_id and ex.quest_id=q.id for read only
declare @stdAns varchar(20)
declare @modelAns varchar(20)
declare @grade int open c1
fetch c1 into @stdAns,@modelAns,@grade
while @@FETCH_STATUS=0 begin
       if(TRIM(@stdAns)=TRIM(@modelAns))
              begin
                     set @totalGrade+=@grade
                   set @studentGrade+=@grade     end
       else
     set @totalGrade+=@grade  fetch c1
into @stdAns,@modelAns,@grade end  set
@percent =
(@studentGrade/@totalGrade)*100 select
CONCAT(@percent,'%') as StudentGrade close c1  deallocate
c1 declare @crsId int
select @crsId= e.Crs_id from Exam_student_question ex,Exam e where e.id=ex.ex_id and
ex.ex_id=@exam_id and ex.Std_ssn=@st_id
update Student_Course set grade=@percent where Crs_id=@crsId and Std_ssn=@st_id
Exam_Correction 9,1
```

	StudentGrade
1	40.0%