iti

ExaminationDB@.

Data Dictionary

3/23/2021

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 relation
- ≻— User-defined many to one relation
- —≺ One to many relation
- —≺ User-defined one to many relation
- ≻≺ Many to many relation
- ≻≺ User-defined many to many relation
- — One to one relation
- — User-defined one to one relation
- →@ Input
- @→ Output
- →@→ Input/Output
- ⧉ Uses dependency
- ⧉ User-defined uses dependency
- ⧉ Used by dependency
- ⧉ User-defined used by dependency

ExaminationDB@.

1. Tables

1.1. Table: Choices

Columns

		Name	Data type	Description / Attributes
▤	🔑	QuestID	int	**References**: Question
▤	🔑	QuestType	varchar(20)	**References**: Question
▤	🔑	QuestChoices	varchar(100)	

Links to

	Table	Join	Title / Name / Description
⤚	Question	**Choices**QuestID = QuestionQuestID, **Choices**QuestType = QuestionQuestType	FK__Choices__46E78A0C

Unique keys

		Name / Description
🔑	QuestID, QuestType, QuestChoices	PK__Choices__D7B11F2EDE88E0D0

1.2. Table: Course

Columns

		Name	Data type	Description / Attributes
▤	🔑	CourseID	varchar(5)	
▤		CourseName	varchar(20)	
▤		CourseMarks	int	
▤		CoursePassPercentage	decimal(18, 0)	
▤		MaxNoOfStuds	int	
▤		InstID	int	**References**: Instructor

Links to

	Table	Join	Title / Name / Description
⤚	Instructor	**Course**InstID = InstructorInstID	FK__Course__InstID__300424B4

Linked from

	Table	Join	Title / Name / Description
⤙	Question	**Course**CourseID = QuestionCourseID	FK__Question__Course__35BCFE0A
⤙	StudCourse	**Course**CourseID = StudCourseCourseID	FK__StudCours__Cours__3C69FB99
⤙	Topic	**Course**CourseID = TopicCourseID	FK__Topic__CourseID__38996AB5

Unique keys

		Name / Description
🔑	CourseID	PK__Course__C92D7187C1A4CDDA

1.3. Table: Department

Columns

		Name	Data type	Description / Attributes
▤	🔑	DeptID	varchar(10)	
▤		DeptDesc	varchar(50)	Nullable
▤		DeptName	varchar(30)	

Linked from

	Table	Join	Title / Name / Description
⤙	Instructor	**Department**DeptID = InstructorDeptID	FK__Instructo__DeptI__2D27B809
⤙	Students	**Department**DeptID = StudentsDeptID	FK__Students__DeptID__276EDEB3

Unique keys

		Name / Description
🔑	DeptID	PK__Departme__0148818E12538737

1.4. Table: Exam

Columns

		Name	Data type	Description / Attributes
▤	🔑	ExamID	int	
▤		ExamNoOfQuest	int	
▤		ExamDatetime	datetime	Nullable **Default**: getdate()
▤		ExamDuration	int	Nullable
▤		ExamType	varchar(20)	Nullable

Linked from

	Table	Join	Title / Name / Description
⤙	ExamQuest	**Exam**ExamID = ExamQuestExamID	FK__ExamQuest__ExamI__4316F928
⤙	Participaton	**Exam**ExamID = ParticipatonExamID	FK__Participa__ExamI__49C3F6B7

Unique keys

		Name / Description
🔑	ExamID	PK__Exam__297521A739DABE5E

1.5. Table: ExamQuest

Columns

		Name	Data type	Description / Attributes
▤	🔑	ExamID	int	**References**: Exam
▤	🔑	QuestID	int	**References**: Question
▤	🔑	QuestType	varchar(20)	**References**: Question

Links to

	Table	Join	Title / Name / Description
≻—	Exam	**ExamQuest**ExamID = ExamExamID	FK__ExamQuest__ExamI__4316F928
≻—	Question	**ExamQuest**QuestID = QuestionQuestID, **ExamQuest**QuestType = QuestionQuestType	FK__ExamQuest__440B1D61

Unique keys

		Name / Description
🔑	ExamID, QuestID, QuestType	PK__ExamQues__241882F6B24F49CA

1.6. Table: Instructor

Columns

		Name	Data type	Description / Attributes
▤	🔑	InstID	int	
▤		InstFName	varchar(30)	**Nullable**
▤		InstLName	varchar(30)	**Nullable**
▤	🔑	InstEmail	varchar(30)	
▤		InstPassword	varchar(20)	
▤		DeptID	varchar(10)	**References**: Department

Links to

	Table	Join	Title / Name / Description
≻—	Department	**Instructor**DeptID = DepartmentDeptID	FK__Instructo__DeptI__2D27B809

Linked from

	Table	Join	Title / Name / Description
—≺	Course	**Instructor**InstID = CourseInstID	FK__Course__InstID__300424B4

Unique keys

		Name / Description
🔑	InstID	PK__Instruct__E2A296668ED9A604
🔑	InstEmail	UQ__Instruct__91D7BAB74BE2DE63

1.7. Table: Participaton

Columns

		Name	Data type	Description / Attributes
▤	🔑	PartID	int	
▤	🔑	ExamID	int	**References**: Exam
▤		StudID	int	**Nullable** **References**: Students
▤	🔑	QuestID	int	**References**: Question
▤	🔑	QuestType	varchar(20)	**References**: Question
▤		StudAns	varchar(100)	**Nullable**
▤		StudMarks	int	**Nullable**

Links to

	Table	Join	Title / Name / Description
≻—	Exam	**Participaton**ExamID = ExamExamID	FK__Participa__ExamI__49C3F6B7
≻—	Question	**Participaton**QuestID = QuestionQuestID, **Participaton**QuestType = QuestionQuestType	FK__Participaton__4BAC3F29
≻—	Students	**Participaton**StudID = StudentsStudID	FK__Participa__StudI__4AB81AF0

Unique keys

		Name / Description
🔑	PartID, ExamID, QuestID, QuestType	PK_Participaton_1

1.8. Table: Question

Columns

		Name	Data type	Description / Attributes
▤	🔑	QuestID	int	
▤	🔑	QuestType	varchar(20)	
▤		QuestBody	varchar(200)	
▤		Marks	int	
▤		CourseID	varchar(5)	**References**: Course
▤		CorrectAns	varchar(100)	

Links to

	Table	Join	Title / Name / Description
≻—	Course	**Question**CourseID = CourseCourseID	FK__Question__Course__35BCFE0A

Linked from

	Table	Join	Title / Name / Description
—≺	Choices	**Question**QuestID = ChoicesQuestID, **Question**QuestType = ChoicesQuestType	FK__Choices__46E78A0C

	Table	Join	Title / Name / Description
—<	ExamQuest	**Question**QuestID = ExamQuestQuestID, **Question**QuestType = ExamQuestQuestType	FK__ExamQuest__440B1D61
—<	Participaton	**Question**QuestID = ParticipatonQuestID, **Question**QuestType = ParticipatonQuestType	FK__Participaton__4BAC3F29

Unique keys

		Name / Description
🔑	QuestID, QuestType	PK__Question__D6DA3511E21FC3D7

1.9. Table: StudCourse

Columns

		Name	Data type	Description / Attributes
▤	🔑	StudID	int	**References**: Students
▤	🔑	CourseID	varchar(5)	**References**: Course
▤		CourseMarks	int	**Nullable**

Links to

	Table	Join	Title / Name / Description
>—	Course	**StudCourse**CourseID = CourseCourseID	FK__StudCours__Cours__3C69FB99
>—	Students	**StudCourse**StudID = StudentsStudID	FK__StudCours__StudI__3B75D760

Unique keys

		Name / Description
🔑	StudID, CourseID	PK__StudCour__99527F0776E493E7

1.10. Table: Students

Columns

		Name	Data type	Description / Attributes
▤	🔑	StudID	int	
▤		StudFName	varchar(30)	**Nullable**
▤		StudLName	varchar(30)	**Nullable**
▤	🔑	StudEmail	varchar(30)	
▤		StudPassword	varchar(20)	
▤		StudTotalGrade	varchar(1)	**Nullable**
▤		DeptID	varchar(10)	**References**: Department

Links to

	Table	Join	Title / Name / Description
>—	Department	**Students**DeptID = DepartmentDeptID	FK__Students__DeptID__276EDEB3

Linked from

	Table	Join	Title / Name / Description
—<	Participaton	**Students**StudID = ParticipatonStudID	FK__Participa__StudI__4AB81AF0
—<	StudCourse	**Students**StudID = StudCourseStudID	FK__StudCours__StudI__3B75D760

Unique keys

		Name / Description
🔑	StudID	PK__Students__F5C0A81FB3C8E8B4
🔑	StudEmail	UQ__Students__70045C4B7038629E

1.11. Table: Topic

Columns

		Name	Data type	Description / Attributes
▤	🔑	TopID	int	
▤	🔑	CourseID	varchar(5)	**References**: Course
▤		TopName	varchar(20)	**Nullable**

Links to

	Table	Join	Title / Name / Description
>—	Course	**Topic**CourseID = CourseCourseID	FK__Topic__CourseID__38996AB5

Unique keys

		Name / Description
🔑	TopID, CourseID	PK__Topic__70565AC18E5BDBA0

2. Procedures

2.1. Procedure: CheckPassword

Input/Output

	Name	Data type	Description
→@	useremail	varchar(20)	
→@	password	varchar(20)	

Script

```
CREATE PROCEDURE [dbo].[CheckPassword]
    @useremail VARCHAR(20),
    @password varchar(20)
AS
BEGIN
SET NOCOUNT ON
IF EXISTS(SELECT * FROM Students s WHERE StudEmail = @useremail AND StudPassword = @password)
    SELECT StudID from Students
ELSE
    SELECT 'false' AS UserExists
END
```

2.2. Procedure: getExamQuestions

Input/Output

	Name	Data type	Description
→@	ExamID	int	

Script

```
CREATE proc [dbo].[getExamQuestions] @ExamID int
as
select  Q.QuestID, Q.QuestType ,Q.QuestBody,Q.Marks
           from Participaton P
           inner join Question Q on Q.QuestID = P.QuestID and Q.QuestType = P.QuestType
           where P.ExamID=@ExamID order by Q.QuestID
```

2.3. Procedure: getStudentGrade

Input/Output

	Name	Data type	Description
→@	studentID	int	

Script

```
create proc getStudentGrade @studentID int
as
           select S.StudID, S.StudFName + ' ' + S.StudLName as 'Full Name', C.CourseName , SC.CourseMarks
           from Students S
           inner join StudCourse SC
           on S.StudID= SC.StudID
           inner join Course C
           on SC.CourseID = C.CourseID
           where S.StudID= @studentID
```

2.4. Procedure: getTopics

Input/Output

	Name	Data type	Description
→@	CourseID	varchar(5)	

Script

```
create proc getTopics @CourseID varchar(5)
as
           select T.TopName
           from Topic T
           inner join Course C on C.CourseID= T.CourseID
    where C.CourseID= @CourseID
```

2.5. Procedure: Sp_addChoicesMCQ

Input/Output

	Name	Data type	Description
→@	questID	int	
→@	choice1	varchar(100)	
→@	choice2	varchar(100)	
→@	choice3	varchar(100)	
→@	choice4	varchar(100)	

Script

```
create proc Sp_addChoicesMCQ @questID int , @choice1 varchar(100),@choice2 varchar(100),@choice3 varchar(100),@choice4
varchar(100) as
begin
insert into Choices values(@questID ,'MCQ' ,@choice1),(@questID ,'MCQ',@choice2),
(@questID ,'MCQ',@choice3),(@questID ,'MCQ',@choice4)
end
```

2.6. Procedure: Sp_addchoicesTorF

Input/Output

	Name	Data type	Description
→@	questID	int	

Script

```
CREATE proc Sp_addchoicesTorF @questID int as
begin
insert into Choices values (@questID ,'TorF' ,'True'),(@questID ,'TorF','False')
end
```

2.7. Procedure: Sp_addNewCourse

Input/Output

	Name	Data type	Description
→@	newCourseID	varchar(5)	
→@	newCourseName	varchar(20)	
→@	newCourseMarks	int	
→@	newCoursePassPercentage	decimal(18, 0)	
→@	newMaxNoOfStuds	int	
→@	newInstID	int	

Script

```
create proc Sp_addNewCourse @newCourseID varchar(5),@newCourseName varchar(20) , @newCourseMarks int,
@newCoursePassPercentage decimal(18, 0), @newMaxNoOfStuds int ,@newInstID int as
begin
insert into Course values
(@newCourseID,@newCourseName,@newCourseMarks,@newCoursePassPercentage,@newMaxNoOfStuds,@newInstID)
end
```

2.8. Procedure: Sp_addNewTopic

Input/Output

	Name	Data type	Description
→@	newTopicID	int	
→@	courseID	varchar(5)	
→@	topicName	varchar(20)	

Script

```
CREATE proc Sp_addNewTopic @newTopicID int  , @courseID varchar(5),@topicName varchar(20) as
begin
insert into Topic
values (@newTopicID ,@courseID,@topicName)
end
```

2.9. Procedure: Sp_courseFullMark

Input/Output

	Name	Data type	Description
→@	ExamID	int	

Script

```
CREATE proc Sp_courseFullMark @ExamID int
as
select  sum(Q.Marks) as Marks
           from Participaton P
           inner join Question Q on Q.QuestID = P.QuestID and Q.QuestType = P.QuestType
           where P.ExamID=@ExamID
```

2.10. Procedure: Sp_EnrollCourse

Input/Output

	Name	Data type	Description
→@	courseID	varchar(5)	
→@	studID	int	

Script

```
CREATE proc Sp_EnrollCourse  @courseID varchar(5),@studID int
  as
  begin
  declare @maxNom int
  set @maxNom=
  (select c.MaxNoOfStuds
  from Course c
  where c.CourseID=@courseID)
  declare @count int
  set @count =(
  select COUNT(s.StudID)
  from StudCourse s
  where s.CourseID=@courseID)
   if @maxNom < @count
   select 'reached max nom of students' as ResMessage
   else
   insert into StudCourse (StudID,CourseID)values(@studID ,@courseID)
  end
```

2.11. Procedure: Sp_ExamCorrection

Input/Output

	Name	Data type	Description
→@	StudID	int	
→@	ExamID	int	

Script

```
create proc Sp_ExamCorrection @StudID int ,@ExamID int
as
begin
UPDATE Participaton
SET StudMarks = Q.Marks
FROM Participaton P
INNER JOIN Question Q ON P.QuestID = Q.QuestID and P.QuestType =Q.QuestType
WHERE P.StudAns = Q.CorrectAns and P.StudID =@StudID and P.ExamID=@ExamID
end
```

2.12. Procedure: Sp_examinedStudents

Input/Output

	Name	Data type	Description
→@	ExamID	int	

Script

```
CREATE proc Sp_examinedStudents @ExamID int as
begin
select distinct s.StudFName +' '+s.StudLName as 'Student Full Name'
from Participaton P inner join Students s
on P.StudID =s.StudID
where P.ExamID=@ExamID
end
```

2.13. Procedure: Sp_failed

Script

```
create proc Sp_failed as
  begin
  delete from Students
  where StudTotalGrade='F'
  end
```

2.14. Procedure: Sp_GetChoices

Input/Output

	Name	Data type	Description
→@	QuestID	int	
→@	QuestType	varchar(5)	

Script

```
create proc Sp_GetChoices @QuestID int, @QuestType varchar(5)
as
begin
select Q.QuestBody,
stuff ((select ' - ' + Ch.QuestChoices
                      from Question Q inner join Choices Ch
                      on Q.QuestID = Ch.QuestID and Q.QuestType = Ch.QuestType
                      where Ch.QuestID = @QuestID and Ch.QuestType = @QuestType
                      FOR XML PATH('')), 1, 1, '') [Choices]
from Question Q inner join Choices Ch
on Q.QuestID = Ch.QuestID and Q.QuestType = Ch.QuestType
where Ch.QuestID = @QuestID and Ch.QuestType = @QuestType
GROUP BY Q.QuestBody
ORDER BY 1
end
```

2.15. Procedure: Sp_getCourseDetails

Input/Output

	Name	Data type	Description
→@	courseID	varchar(5)	

Script

```
create proc Sp_getCourseDetails @courseID varchar(5) as
begin
select c.CourseName ,c.CourseMarks ,i.InstFName +' '+i.InstLName as 'instructor'
from Course c inner join Instructor i
on c.InstID=i.InstID
where c.CourseID=@courseID
end
```

2.16. Procedure: Sp_getDetails

Input/Output

	Name	Data type	Description
→@	col	varchar(20)	
→@	t	varchar(50)	

Script

```
create proc Sp_getDetails  (@col varchar(20), @t varchar(50))
as
execute('select '+@col+' from '+@t)
```

2.17. Procedure: Sp_GetEnrolledCourses

Input/Output

	Name	Data type	Description
→@	studID	int	

Script

```
CREATE proc Sp_GetEnrolledCourses @studID int as
begin
select c.CourseName as 'Enrolled Courses'
from StudCourse sc inner join Course c
on sc.CourseID=c.CourseID
where sc.StudID =@studID
end
```

2.18. Procedure: SP_getExamChoices

Input/Output

	Name	Data type	Description
→@	CourseID	varchar(5)	

Script

```
CREATE proc [dbo].[SP_getExamChoices] @CourseID varchar(5)
as
select distinct Q.QuestID, Q.QuestType ,Ch.QuestChoices
           from Participaton P
           inner join Question Q on Q.QuestID = P.QuestID and Q.QuestType = P.QuestType
           inner join Choices Ch on Ch.QuestID = Q.QuestID and Ch.QuestType = Q.QuestType
           where Q.CourseID=@CourseID order by Q.QuestID
```

2.19. Procedure: Sp_GetExamDateAndTime

Input/Output

	Name	Data type	Description
→@	examID	int	

Script

```
CREATE proc Sp_GetExamDateAndTime @examID int as
begin
select distinct E.ExamDatetime as 'Exam Date and Time' , CONCAT( E.ExamDuration , ' Hour') as 'Duration'
from Participaton P inner join Exam E
on P.ExamID=E.ExamID
where P.ExamID=@examID
end
```

2.20. Procedure: Sp_getExamQuestionsAndStudAnswers

Input/Output

	Name	Data type	Description
→@	ExamID	int	
→@	StudID	int	

Script

```
create proc Sp_getExamQuestionsAndStudAnswers @ExamID int, @StudID int
as
           select Q.QuestBody, P.StudAns
           from Participaton P inner join Question Q on P.QuestID = Q.QuestID and P.QuestType = Q.QuestType
           where P.ExamID = @ExamID and P.StudID = @StudID
```

2.21. Procedure: SP_GetInstructorCourseData

Input/Output

	Name	Data type	Description
→@	insID	int	

Script

```
create procedure SP_GetInstructorCourseData @insID int
as
begin
            select C.CourseName,count(*) AS NumberOfStudnt
            from Instructor I
            inner join Course C
            on I.InstID = C.InstID
            inner join StudCourse SC
            on C.CourseID = SC.CourseID where I.InstID=@insID
            group by C.CourseName
end
```

2.22. Procedure: Sp_GetInstructorCourses

Input/Output

	Name	Data type	Description
→@	insID	int	

Script

```
create proc Sp_GetInstructorCourses @insID int
as
begin
declare c1 cursor
for select distinct I.InstFName + ' ' + I.InstLName as 'Instructor Name', C.CourseName
from Instructor I inner join Course C
on I.InstID = C.InstID
where I.InstID = @insID
for read only
declare @Iname varchar(30),@Cname varchar(20),@AllCourses varchar(200)
open c1
fetch c1 into @Iname,@Cname
while @@FETCH_STATUS=0
            begin
            if @AllCourses is null
                        begin
                        set @AllCourses = @Cname
                        fetch c1 into @Iname,@Cname
                        end
            else
                        begin
                        set @AllCourses = CONCAT(@AllCourses,',',@Cname)
                        fetch c1 into @Iname,@Cname
                        end
            end
select @Iname as 'Instructor Name',@AllCourses as 'Course Names'
close c1
deallocate c1
end
```

2.23. Procedure: Sp_GetNumOfStudents

Input/Output

	Name	Data type	Description
→@	insID	int	

Script

```
create proc Sp_GetNumOfStudents @insID int
as
begin
select I.InstFName + ' ' + I.InstLName as 'Instructor Name', COUNT(S.StudID) as 'Number of Students'
from Instructor I inner join Department D
on I.DeptID = D.DeptID
inner join Students S
on D.DeptID = S.DeptID
where I.InstID = @insID
group by I.InstFName, I.InstLName
end
```

2.24. Procedure: Sp_GetStudentData

Script

```
create procedure Sp_GetStudentData
as
begin
            select s.*
            from Students S
            inner join Department D
            on  D.DeptID = S.DeptID
end
```

2.25. Procedure: Sp_GetStudentInformation

Input/Output

	Name	Data type	Description
→@	departnum	varchar(10)	

Script

```
CREATE proc Sp_GetStudentInformation @departnum varchar(10)
  as
  begin
      select s.* from Students s
      inner join Department d
      on  d.DeptID=s.DeptID
      where d.DeptID=@departnum
  end
```

2.26. Procedure: Sp_GetStudentMarksPerCourse

Input/Output

	Name	Data type	Description
→@	studID	int	

Script

```
create proc Sp_GetStudentMarksPerCourse @studID int as
begin
select c.CourseName as 'Enrolled Courses' , sc.CourseMarks as 'Student Mark'
from StudCourse sc inner join Course c
on sc.CourseID=c.CourseID
where sc.StudID =@studID
end
```

2.27. Procedure: Sp_GetStudentTotalGrade

Input/Output

	Name	Data type	Description
→@	StudID	int	

Script

```
create proc Sp_GetStudentTotalGrade @StudID int
as
begin
select S.StudFName + ' ' + S.StudLName, S.StudTotalGrade
from Students S
where S.StudID = @StudID
end
```

2.28. Procedure: Sp_getUserInfo

Input/Output

	Name	Data type	Description
→@	ID	int	

Script

```
create proc Sp_getUserInfo @ID int as
begin
IF EXISTS(SELECT * FROM Instructor WHERE InstID = @ID)
begin
SELECT * FROM Instructor WHERE InstID = @ID
end
else if EXISTS(SELECT * FROM Students WHERE StudID = @ID)
begin
SELECT * FROM Students WHERE StudID = @ID
end
else
select 'ID is not Found' as 'STATUS'
end
```

2.29. Procedure: Sp_InsGenerateExam

Input/Output

	Name	Data type	Description
→@	CourseID	varchar(5)	
→@	ExamDate	datetime	
→@	examType	varchar(20)	

Script

```sql
CREATE proc Sp_InsGenerateExam @CourseID varchar(5),@ExamDate datetime ,@examType varchar(20)
as
begin
declare @examId int
set @examId =(select MAX(e.ExamID)+1 from Exam e)

declare @examDuration int

declare @nomOfQues int
if @examType ='Final'
begin
set @nomOfQues=10
set @examDuration=2
end
else
begin
set @nomOfQues=6
set @examDuration=1
end
--add generated exam
insert into Exam (ExamID,ExamNoOfQuest,ExamDatetime,ExamDuration,ExamType)
Values (@examId,@nomOfQues,@ExamDate,@examDuration,@examType)

declare @partID int
set @partID=(select isnull(MAX(P.PartID),0) from Participaton P)
set @partID+=1

insert into Participaton(QuestID,PartID,ExamID,QuestType)
select top (@nomOfQues/2) Q.QuestID,@partID,@examId ,'TorF'
from Question Q
where @CourseID =Q.CourseID and Q.QuestType='TorF'
order by NEWID()

insert into Participaton(QuestID,PartID,ExamID,QuestType)
select top (@nomOfQues/2) Q.QuestID, @partID,@examId ,'MCQ'
from Question Q
where @CourseID =Q.CourseID and Q.QuestType='MCQ'
order by NEWID()

insert into ExamQuest(ExamID , QuestID ,QuestType)
select ExamID,QuestID,QuestType
from Participaton
where ExamID=@examId

End
```

2.30. Procedure: Sp_instCheckPassword

Input/Output

	Name	Data type	Description
→@	instEmail	varchar(30)	
→@	password	varchar(20)	

Script

```sql
CREATE proc Sp_instCheckPassword @instEmail varchar(30), @password varchar(20) as
IF EXISTS(SELECT * FROM Instructor WHERE InstEmail = @instEmail AND InstPassword = @password)
SELECT 1 AS UserExists
ELSE
SELECT -1 AS UserExists
```

2.31. Procedure: Sp_instOnlyAddQuestion

Input/Output

	Name	Data type	Description
→@	quesID	int	
→@	quesType	varchar(20)	
→@	instID	int	
→@	questBody	varchar(200)	
→@	marks	int	
→@	courseID	varchar(5)	
→@	correctAns	varchar(100)	

Script

```
create proc Sp_instOnlyAddQuestion @quesID int ,@quesType varchar(20),@instID int,
@questBody varchar(200) ,@marks int , @courseID varchar(5),@correctAns varchar(100) as
begin
IF EXISTS(SELECT * FROM Instructor WHERE InstID = @InstID)
begin
insert into Question
values (@quesID,@quesType,@questBody,@marks,@courseID,@correctAns)
end
else
select 'Instructor only can Add question' as 'STATUS'
end
```

2.32. Procedure: Sp_instOnlyUpdateExam

Input/Output

	Name	Data type	Description
→@	examID	int	
→@	instID	int	
→@	ExamNoOfQuest	int	
→@	ExamType	varchar(20)	

Script

```
CREATE proc Sp_instOnlyUpdateExam @examID int ,@instID int , @ExamNoOfQuest int ,@ExamType varchar(20) as
begin
IF EXISTS(SELECT * FROM Instructor WHERE InstID = @InstID)
begin
update Exam
set ExamNoOfQuest=@ExamNoOfQuest , ExamType=@ExamType
where ExamID=@examID
end
else
select 'Instructor only can edit Exam' as 'STATUS'
end
```

2.33. Procedure: SP_SelectCoursesExams

Script

```
CREATE proc [dbo].[SP_SelectCoursesExams]
as
select distinct C.CourseName,C.CourseID
from Participaton P
inner join Question Q on P.QuestID=Q.QuestID
and P.QuestType=Q.QuestType
inner join Course C
on Q.CourseID=C.CourseID
```

2.34. Procedure: SP_SelectCoursesSpecificExam

Input/Output

	Name	Data type	Description
→@	CourseID	varchar(5)	

Script

```
CREATE proc [dbo].[SP_SelectCoursesSpecificExam] @CourseID varchar(5)
as
select distinct P.ExamID,E.ExamDatetime
from Participaton P
inner join Question Q on P.QuestID=Q.QuestID
and P.QuestType=Q.QuestType
inner join Course C
on Q.CourseID=C.CourseID
inner join Exam E on P.ExamID=E.ExamID
where C.CourseID=@CourseID and p.StudID is null
```

2.35. Procedure: SP_SelectDepartments

Script

```
create Proc [dbo].[SP_SelectDepartments]
as
Select  DeptID,DeptName
from Department
```

2.36. Procedure: SP_SelectInstructors

Script

```
CREATE Proc [dbo].[SP_SelectInstructors]
as
Select  InstID as InstructID,InstFName,InstLName
from Instructor
```

2.37. Procedure: Sp_ShowModelAnswers

Input/Output

	Name	Data type	Description
→@	ExamID	int	

Script

```
CREATE proc Sp_ShowModelAnswers @ExamID int as
begin
select distinct Q.QuestBody, Q.QuestType ,Q.CorrectAns
from Participaton P
inner join Question Q on Q.QuestID = P.QuestID and Q.QuestType =P.QuestType
inner join Choices C on C.QuestID = Q.QuestID and C.QuestType = Q.QuestType
inner join Course cor on cor.CourseID=Q.CourseID
where P.ExamID = @ExamID
end
```

2.38. Procedure: Sp_ShowStudentAnswers

Input/Output

	Name	Data type	Description
→@	ExamID	int	
→@	StudID	int	

Script

```
create proc Sp_ShowStudentAnswers @ExamID int, @StudID int as
begin
select distinct Q.QuestBody, P.StudAns
from Participaton P
inner join Question Q on Q.QuestID = P.QuestID and Q.QuestType =P.QuestType
where P.ExamID=@ExamID and P.StudID=@StudID
end
```

2.39. Procedure: Sp_StExamRetrival

Input/Output

	Name	Data type	Description
→@	ExamID	int	
→@	ExamDateTime	datetime	

Script

```
CREATE proc Sp_StExamRetrival @ExamID int , @ExamDateTime datetime
as
begin
select Q.QuestBody,Q.QuestType,Q.Marks,cor.CourseName ,E.ExamDatetime ,E.ExamNoOfQuest ,E.ExamType ,E.ExamDuration
from Participaton P
inner join Question Q on Q.QuestID = P.QuestID and Q.QuestType =P.QuestType
inner join Choices C on C.QuestID = Q.QuestID and C.QuestType = Q.QuestType
inner join Course cor on cor.CourseID=Q.CourseID
inner join Exam E on E.ExamID=P.ExamID
where E.ExamID = @ExamID and E.ExamDatetime=@ExamDateTime
end
```

2.40. Procedure: Sp_StudentCheckPassword

Input/Output

	Name	Data type	Description
→@	useremail	varchar(30)	
→@	password	varchar(20)	

Script

```
create PROCEDURE Sp_StudentCheckPassword
@useremail VARCHAR(30),
@password varchar(20)
AS
BEGIN
IF EXISTS(SELECT * FROM Students WHERE StudEmail = @useremail AND StudPassword = @password)
begin (
SELECT s.StudID
from Students s
where s.StudEmail=@useremail and s.StudPassword=@password)
end
ELSE
SELECT -1 AS UserExists
END
```

2.41. Procedure: Sp_StudentDidntExamined

Input/Output

	Name	Data type	Description
→@	courseID	varchar(5)	

Script

```
CREATE proc Sp_StudentDidntExamined @courseID varchar(5) as
begin
select  s.StudFName + ' '+ s.StudLName as 'Student Full name'
from Students s
where StudID in (
select sc.StudID
from StudCourse sc inner join Students s
on sc.StudID=s.StudID
where CourseID=@courseID except(
select StudID
from Participaton P))
end
```

2.42. Procedure: Sp_sumOfMarksPerCourse

Input/Output

	Name	Data type	Description
→@	CourseID	varchar(5)	
→@	ExamID	int	
→@	StudID	int	

Script

```
CREATE proc [dbo].[Sp_sumOfMarksPerCourse] @CourseID varchar(5) ,@ExamID int ,@StudID int
as
begin
declare @marksOfCourse int
set @marksOfCourse = (select sum (isnull(P.StudMarks,0))
from Participaton P inner join Question Q
on P.QuestID=Q.QuestID and P.QuestType =Q.QuestType
inner join Course C
on Q.CourseID =C.CourseID
where C.CourseID =@CourseID and P.ExamID=@ExamID and P.StudID=@StudID
)
update StudCourse
set CourseMarks = @marksOfCourse
where CourseID=@CourseID and StudID=@StudID

select sc.CourseMarks
from StudCourse sc
where sc.StudID=@StudID and sc.CourseID=@CourseID
end
```

2.43. Procedure: Sp_TakeAnswers

Input/Output

	Name	Data type	Description
→@	ExamID	int	
→@	CourseID	varchar(5)	
→@	StudID	int	
→@	StudAnswer	varchar(100)	
→@	quesID	int	
→@	quesType	varchar(20)	

Script

```
CREATE proc [dbo].[Sp_TakeAnswers] @ExamID int,@CourseID varchar(5) ,@StudID int, @StudAnswer varchar(100),@quesID int ,
@quesType varchar(20)
 as
begin
update Participaton
Set StudID=@StudID , StudAns=@StudAnswer
where ExamID =@ExamID and QuestID=@quesID and QuestType=@quesType
insert into StudCourse
(StudID,CourseID) values (@StudID,@CourseID)
end
```

2.44. Procedure: Sp_totalGrade

Input/Output

	Name	Data type	Description
→@	StuID	int	

Script

```
create proc Sp_totalGrade @StuID int as
begin

declare @totalGradePerc decimal
set @totalGradePerc=
(
select sum(ISNULL(SC.CourseMarks,0))
from StudCourse SC
where SC.StudID=@StuID
)/
(
select sum(C.CourseMarks)
from Course C inner join StudCourse SC
on C.CourseID = SC.CourseID and SC.StudID = @StuID
)
declare @totalGrade varchar(1)
set @totalGrade =
(
    CASE
        WHEN @totalGradePerc between 100 and 90 THEN 'A'
        WHEN @totalGradePerc between 89 and  80 THEN 'B'
        WHEN @totalGradePerc between 70 and 79 THEN 'C'
                      WHEN @totalGradePerc between 69 and 60 THEN 'D'
        WHEN @totalGradePerc between 0 and 59 THEN 'F'
    END )


update Students
set StudTotalGrade = @totalGrade
where StudID=@StuID

end
```

2.45. Procedure: Sp_updateCourseInstructor

Input/Output

	Name	Data type	Description
→@	courseID	varchar(5)	
→@	newInstID	int	

Script

```
create proc Sp_updateCourseInstructor @courseID varchar(5), @newInstID int as
begin
update Course
set InstID=@newInstID
where CourseID=@courseID
end
```

2.46. Procedure: Sp_updateInstructorData

Input/Output

	Name	Data type	Description
→@	InstID	int	
→@	newMail	varchar(30)	
→@	newPassword	varchar(20)	

Script

```
create proc Sp_updateInstructorData @InstID int ,@newMail varchar(30) , @newPassword varchar(20) as
begin
IF EXISTS(SELECT * FROM Instructor WHERE InstID = @InstID)
begin
update Instructor
set InstEmail =@newMail , InstPassword=@newPassword
where InstID=@InstID
end
else
select 'Instructor Not Found' as 'STATUS'
end
```

2.47. Procedure: Sp_updateStudentData

Input/Output

	Name	Data type	Description
→@	StudID	int	
→@	newMail	varchar(30)	
→@	newPassword	varchar(20)	

Script

```
CREATE proc Sp_updateStudentData @StudID int ,@newMail varchar(30) , @newPassword varchar(20) as
begin
IF EXISTS(SELECT * FROM Students WHERE StudID= @StudID)
begin
update Students
set StudEmail =@newMail , StudPassword=@newPassword
where StudID=@StudID
end
else
select 'Student Not Found' as 'STATUS'
end
```

2.48. Procedure: Sp_updateStudentMarks

Input/Output

	Name	Data type	Description
→@	StudID	int	
→@	courseID	varchar(5)	
→@	newMark	int	

Script

```
CREATE proc Sp_updateStudentMarks @StudID int , @courseID varchar(5) ,@newMark int as
begin
declare @courseMarks int
set @courseMarks =(select CourseMarks from Course where CourseID=@courseID)
if @newMark <= @courseMarks
begin
update StudCourse
set CourseMarks =@newMark
where StudID=@StudID and CourseID=@courseID
end
else
select 'Mark is not approved' as 'STATUS'
end
```

2.49. Procedure: Sp_updateStudentTotalGrade

Input/Output

	Name	Data type	Description
→@	StudID	int	
→@	newGrade	varchar(1)	

Script

```
create proc Sp_updateStudentTotalGrade @StudID int , @newGrade varchar(1) as
begin
update Students
set StudTotalGrade =@newGrade
where StudID=@StudID
end
```

2.50. Procedure: Sp_withdraw

Input/Output

	Name	Data type	Description
→@	studID	int	

Script

```
create proc Sp_withdraw @studID int as
  begin
  delete from Students
  where StudID=@studID
  end
```