

ExamSystem

Data Dictionary

2022-03-21

TRIAL
























TRIAL

Table of contents

ExamSystem	7
1. Tables	7
1.1. Table: Choice	7
1.2. Table: Course	8
1.3. Table: Crs_Top	9
1.4. Table: Department	10
1.5. Table: Exam	11
1.6. Table: Exm_Ques	12
1.7. Table: Ins_Crs	13
1.8. Table: Instructor	14
1.9. Table: Question	16
1.10. Table: Std_Crs	18
1.11. Table: Std_Ques_Exm	19
1.12. Table: Student	20
1.13. Table: User	22
2. Procedures	23
2.1. Procedure: DelCrsIns	23
2.2. Procedure: DeleteInstructor	24
2.3. Procedure: DeleteStdByld	25
2.4. Procedure: DeleteUserByUserName	26
2.5. Procedure: InsertInsForCrs	27
2.6. Procedure: InsertInstructor	28
2.7. Procedure: InsertNewStd	29
2.8. Procedure: InsertNewUser	30
2.9. Procedure: InsertQues	31
2.10. Procedure: SelectCrsIns	32
2.11. Procedure: SelectInstructor	33
2.12. Procedure: SelectStdByld	34
2.13. Procedure: SelectUserByld	35
2.14. Procedure: sp_AssignAnsToStd	36
2.15. Procedure: sp_CorrectStdExam	37
2.16. Procedure: sp_CourseNameWithStudentCount	39
2.17. Procedure: sp_deleteChoice	40
2.18. Procedure: sp_deleteCourse	41
2.19. Procedure: sp_deleteDepartment	42
2.20. Procedure: sp_deleteExam	43
2.21. Procedure: sp_DeleteQues	44
2.22. Procedure: sp_DeleteStdFromCrs	45
2.23. Procedure: sp_deleteTopic	46
2.24. Procedure: sp_ExamQuestionsAndAnswers	47
2.25. Procedure: sp_GenerateExam	48
2.26. Procedure: sp_InsertChoice	49
2.27. Procedure: sp_insertCourse	50
2.28. Procedure: sp_insertDepartment	51
2.29. Procedure: sp_insertExam	52

2.30.	Procedure: sp_InsertStdAnsForExm	53
2.31.	Procedure: sp_InsertStdForCrS	54
2.32.	Procedure: sp_insertTopic	55
2.33.	Procedure: sp_selectchoice	56
2.34.	Procedure: sp_selectCourse	57
2.35.	Procedure: sp_selectDepartment	58
2.36.	Procedure: sp_selectExam	59
2.37.	Procedure: sp_SelectQues	60
2.38.	Procedure: sp_SelectStdAnsForExm	61
2.39.	Procedure: sp_SelectStdOfCrS	62
2.40.	Procedure: sp_selectTopic	63
2.41.	Procedure: sp_StdExamAnswers	64
2.42.	Procedure: sp_StudentGradePerCourse	65
2.43.	Procedure: sp_StudentperDepartment	66
2.44.	Procedure: sp_TopicsPerCourse	67
2.45.	Procedure: sp_updateChoice	68
2.46.	Procedure: sp_updateCourse	69
2.47.	Procedure: sp_UpdateCrSForIns	70
2.48.	Procedure: sp_UpdateCrSOfStd	71
2.49.	Procedure: sp_updateDepartment	72
2.50.	Procedure: sp_updateExam	73
2.51.	Procedure: sp_updateTopic	74
2.52.	Procedure: UpdateInstructor	75
2.53.	Procedure: UpdateQues	76
2.54.	Procedure: UpdateStdInfo	77
2.55.	Procedure: UpdateUser	78

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

ExamSystem


1. Tables

1.1. Table: Choice

Columns

Name		Data type	Description / Attributes
	Cho_Id	int	Identity / Auto increment
	Ques_Id	int	Nullable References: Question
	Cho_Content	varchar(100)	Nullable
	Cho_Char	varchar(1)	Nullable

Links to

Table	Join	Title / Name / Description
 Question	ChoiceQues_Id = QuestionQues_Id	FK_Choice_Question



Linked from

Table	Join	Title / Name / Description
 Question	ChoiceCho_Id = QuestionModel_Ans	FK_Question_Choice










Unique keys

Columns	Name / Description
 Cho_Id	PK_Choice

Uses






Name
 Choice
 Question

Used By

Name
 Choice
 sp_CorrectStdExam
 sp_deleteChoice
 sp_ExamQuestionsAndAnswers
 sp_InsertChoice
 sp_selectchoice
 sp_StdExamAnswers
 sp_updateChoice
 Question

1.2. Table: Course

Columns

	Name	Data type	Description / Attributes
 	Crs_Id	int	Identity / Auto increment
	Crs_Name	varchar(50)	Nullable
	Crs_Desc	varchar(50)	Nullable
	Crs_Dur	int	Nullable








Linked from

	Table	Join	Title / Name / Description
→	Crs_Top	Course Crs_Id = Crs_TopCrs_Id	FK_Crs_Top_Course
→	Exam	Course Crs_Id = ExamCrs_Id	FK_Exam_Course
→	Ins_Crs	Course Crs_Id = Ins_CrsCrs_Id	FK_Ins_Crs_Course
→	Std_Crs	Course Crs_Id = Std_CrsCrs_Id	FK_Std_Crs_Course

Unique keys





	Columns	Name / Description
	Crs_Id	PK_Course

Used By


	Name
 Course	
	sp_CourseNameWithStudentCount
	sp_deleteCourse
	sp_insertCourse
	sp_selectCourse
	sp_StudentGradePerCourse
	sp_updateCourse
→	Crs_Top
→	Exam
→	Ins_Crs
→	Std_Crs

1.3. Table: Crs_Top


Columns

Name		Data type	Description / Attributes
	 Crs_Id	int	References: Course
	 Crs_Top	varchar(50)	



Links to

Table	Join	Title / Name / Description
 Course	Crs_Top Crs_Id = CourseCrs_Id	FK_Crs_Top_Course







Unique keys

Columns	Name / Description
 Crs_Id, Crs_Top	PK_Crs_Top

Uses






Name
 Crs_Top
 Course

Used By


Name
 Crs_Top
 sp_deleteTopic
 sp_insertTopic
 sp_selectTopic
 sp_TopicsPerCourse
 sp_updateTopic

1.4. Table: Department



Columns

Name		Data type	Description / Attributes
	Dept_Id	int	Identity / Auto increment
	Dept_Name	varchar(50)	Nullable
	Dept_Loc	varchar(50)	Nullable
	Dept_ManagerHireDate	date	Nullable
	Dept_ManagerId	int	Nullable References: Instructor

Links to

Table	Join	Title / Name / Description
 Instructor	Department Dept_ManagerId = InstructorIns_Id	FK_Department_Instructor



Linked from

Table	Join	Title / Name / Description
 Instructor	Department Dept_Id = InstructorDept_Id	FK_Instructor_Department
 Student	Department Dept_Id = StudentDept_Id	FK_Student_Department








Unique keys

Columns	Name / Description
 Dept_Id	PK_Department

Uses





Name
 Department
 Instructor

Used By



Name
 Department
 sp_deleteDepartment
 sp_insertDepartment
 sp_selectDepartment
 sp_updateDepartment
 Instructor
 Student

1.5. Table: Exam



Columns

	Name	Data type	Description / Attributes
	Exm_Id	int	Identity / Auto increment
	Crs_Id	int	Nullable References: Course
	Generator_Id	int	Nullable References: Instructor
	Exm_Grade	int	Nullable

Links to

	Table	Join	Title / Name / Description
	Course	Exam Crs_Id = CourseCrs_Id	FK_Exam_Course
	Instructor	Exam Generator_Id = InstructorIns_Id	FK_Exam_Instructor




Linked from

	Table	Join	Title / Name / Description
	Exm_Ques	Exam Exm_Id = Exm_QuesExm_Id	FK_Exm_Ques_Exam
	Std_Ques_Exm	Exam Exm_Id = Std_Ques_ExmExm_Id	FK_Std_Ques_Exm_Exam










Unique keys

	Columns	Name / Description
	Exm_Id	PK_Exam

Uses






	Name
	Exam
	Course
	Instructor

Used By



	Name
	Exam
	sp_CorrectStdExam
	sp_deleteExam
	sp_GenerateExam
	sp_insertExam
	sp_selectExam
	sp_updateExam
	Exm_Ques
	Std_Ques_Exm

1.6. Table: Exm_Ques


Columns

Name		Data type	Description / Attributes
	 Exm_Id	int	References: Exam
	 Ques_Id	int	References: Question
	Grade	int	Nullable Default: 1




Links to

Table	Join	Title / Name / Description
 Exam	Exm_Ques Exm_Id = ExamExm_Id	FK_Exm_Ques_Exam
 Question	Exm_Ques Ques_Id = QuestionQues_Id	FK_Exm_Ques_Question




Unique keys

Columns	Name / Description
 Exm_Id, Ques_Id	PK_Exm_Ques

Uses





Name
 Exm_Ques
 Exam
 Question

Used By



Name
 Exm_Ques
 sp_ExamQuestionsAndAnswers
 sp_GenerateExam

1.7. Table: Ins_Crs


Columns

Name		Data type	Description / Attributes
 	Ins_Id	int	References: Instructor
 	Crs_Id	int	References: Course




Links to

Table	Join	Title / Name / Description
 Course	Ins_Crs Crs_Id = CourseCrs_Id	FK_Ins_Crs_Course
 Instructor	Ins_Crs Ins_Id = InstructorIns_Id	FK_Ins_Crs_Instructor





Unique keys

Columns	Name / Description
 Ins_Id, Crs_Id	PK_Ins_Crs

Uses








Name
 Ins_Crs
 Course
 Instructor

Used By



Name
 Ins_Crs
 DelCrsIns
 InsertInsForCrs
 SelectCrsIns
 sp_CourseNameWithStudentCount
 sp_UpdateCrsForIns

1.8. Table: Instructor




Columns

Name		Data type	Description / Attributes
	Ins_Id	int	Identity / Auto increment
	U_Id	int	Nullable References: User
	Dept_Id	int	Nullable References: Department
	Ins_Fname	varchar(50)	Nullable
	Ins_Lname	varchar(50)	Nullable
	Ins_Degree	varchar(50)	Nullable
	Ins_Salary	int	Nullable

Links to

Table	Join	Title / Name / Description
 Department	Instructor Dept_Id = DepartmentDept_Id	FK_Instructor_Department
 User	Instructor U_Id = UserU_Id	FK_Instructor_User




Linked from

Table	Join	Title / Name / Description
 Department	Instructor Ins_Id = DepartmentDept_ManagerId	FK_Department_Instructor
 Exam	Instructor Ins_Id = ExamGenerator_Id	FK_Exam_Instructor
 Ins_Crs	Instructor Ins_Id = Ins_CrsIns_Id	FK_Ins_Crs_Instructor






Unique keys

Columns	Name / Description
 Ins_Id	PK_Instructor

Uses

Name
 Instructor
 Department
 User

Used By







Name
 Instructor
 DeletelInstructor
 InsertInstructor
 SelectInstructor
 UpdatelInstructor

Name	
← Department	
← Exam	
← Ins_Crs	


TRIAL

1.9. Table: Question




Columns

	Name	Data type	Description / Attributes
	Ques_Id	int	Identity / Auto increment
	Crs_Id	int	Nullable
	Ques_Content	varchar(300)	Nullable
	Ques_Grade	int	Nullable
	Ques_Type	varchar(3)	Nullable
	Model_Ans	int	Nullable References: Choice


Links to

Table	Join	Title / Name / Description
 Choice	Question Model_Ans = ChoiceCho_Id	FK_Question_Choice



Linked from

Table	Join	Title / Name / Description
 Choice	Question Ques_Id = ChoiceQues_Id	FK_Choice_Question
 Exm_Ques	Question Ques_Id = Exm_QuesQues_Id	FK_Exm_Ques_Question
 Std_Ques_Exm	Question Ques_Id = Std_Ques_ExmQues_Id	FK_Std_Ques_Exm_Question










Unique keys

Columns	Name / Description
 Ques_Id	PK_Question

Uses

Name
 Question
 Choice

Used By







Name
 Question
 InsertQues
 sp_CorrectStdExam
 sp_DeleteQues
 sp_ExamQuestionsAndAnswers
 sp_GenerateExam
 sp_SelectQues
 sp_StdExamAnswers
 UpdateQues

Name
→ Choice
→ Exm_Ques
→ Std_Ques_Exm



TRIAL

1.10. Table: Std_Crs


Columns

Name		Data type	Description / Attributes
	 Std_Id	int	References: Student
	 Crs_Id	int	References: Course
	Grade	int	Nullable
	Date	date	Nullable




Links to

Table		Join	Title / Name / Description
	Course	Std_Crs Crs_Id = CourseCrs_Id	FK_Std_Crs_Course
	Student	Std_Crs Std_Id = StudentStd_Id	FK_Std_Crs_Student









Unique keys

Columns	Name / Description
 Std_Id, Crs_Id	PK_Std_Crs

Uses





Name
 Std_Crs
 Course
 Student

Used By

Name
 Std_Crs
 sp_CorrectStdExam
 sp_CourseNameWithStudentCount
 sp_DeleteStdFromCrs
 sp_InsertStdForCrs
 sp_SelectStdOfCrs
 sp_StudentGradePerCourse
 sp_UpdateCrsOfStd

1.11. Table: Std_Ques_Exm

Columns

	Name	Data type	Description / Attributes
	Std_Id	int	References: Student
	Exm_Id	int	References: Exam
	Ques_Id	int	References: Question
	Std_Answer	varchar(1)	

Links to

	Table	Join	Title / Name / Description
➤	Exam	Std_Ques_Exm Exm_Id = ExamExm_Id	FK_Std_Ques_Exm_Exam
➤	Question	Std_Ques_Exm Ques_Id = QuestionQues_Id	FK_Std_Ques_Exm_Question
➤	Student	Std_Ques_Exm Std_Id = StudentStd_Id	FK_Std_Ques_Exm_Student







Unique keys

	Columns	Name / Description
	Std_Id, Exm_Id, Ques_Id	PK_Std_Ques_Exm

Uses








	Name
	Std_Ques_Exm
➤	Exam
➤	Question
➤	Student

Used By



	Name
	Std_Ques_Exm
	sp_AssignAnsToStd
	sp_CorrectStdExam
	sp_InsertStdAnsForExm
	sp_SelectStdAnsForExm
	sp_StdExamAnswers

1.12. Table: Student



Columns

	Name	Data type	Description / Attributes
	Std_Id	int	Identity / Auto increment
	U_Id	int	Nullable References: User
	Dept_Id	int	Nullable References: Department
	Std_Fname	varchar(50)	Nullable
	Std_Lname	varchar(50)	Nullable
	Std_BOD	date	Nullable
	Std_Address	varchar(100)	Nullable

Links to

Table	Join	Title / Name / Description
 Department	Student Dept_Id = DepartmentDept_Id	FK_Student_Department
 User	Student U_Id = UserU_Id	FK_Student_User




Linked from

Table	Join	Title / Name / Description
 Std_Crs	Student Std_Id = Std_CrsStd_Id	FK_Std_Crs_Student
 Std_Ques_Exm	Student Std_Id = Std_Ques_ExmStd_Id	FK_Std_Ques_Exm_Student








Unique keys

Columns	Name / Description
 Std_Id	PK_Student

Uses

Name
 Student
 Department
 User

Used By







Name
 Student
 DeleteStdById
 InsertNewStd
 SelectStdById
 sp_StudentGradePerCourse
 sp_StudentperDepartment
 UpdateStdInfo

Name
← Std_Crs
← Std_Ques_Exm

TRIAL

1.13. Table: User

Columns

	Name	Data type	Description / Attributes
	U_Id	int	Identity / Auto increment
	U_UserName	varchar(50)	
	U_Email	varchar(50)	Nullable
	U_Password	varchar(50)	
	U_Sex	varchar(1)	Nullable
	U_IsStd	bit	






Linked from

	Table	Join	Title / Name / Description
→	Instructor	UserU_Id = InstructorU_Id	FK_Instructor_User
→	Student	UserU_Id = StudentU_Id	FK_Student_User

Unique keys

	Columns	Name / Description
	U_Id	PK_User
	U_Id	IX_User

Used By

	Name
	User
	DeleteUserByUserName
	InsertNewUser
	SelectUserById
	UpdateUser
→	Instructor
→	Student

2. Procedures

2.1. Procedure: DelCrsIns

Input/Output

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

Uses

Name	
⚙ DelCrsIns	
📊 Ins_Crs	

Script

```
CREATE PROC DelCrsIns
AS
    BEGIN TRY
        DELETE FROM Ins_Crs
        WHERE Ins_Id = @ins_id
        AND Crs_Id = @crs_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
    @ins_id int,
    @crs_id int
```

2.2. Procedure: DeleteInstructor

Input/Output

Name		Data type	Description
→@	ins_id	int	

Uses

Name	
⚙ DeleteInstructor	
📄 Instructor	

Script

```
CREATE PROC DeleteInstructor @ins_id int
AS
    BEGIN TRY
        DELETE FROM Instructor
        WHERE Ins_Id = @ins_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```


2.3. Procedure: DeleteStdById

Input/Output

Name		Data type	Description
➔@	std_id	int	

Uses

Name	
⚙️	DeleteStdById
📄	Student

Script

```
CREATE PROC DeleteStdById
AS
    BEGIN TRY
        DELETE FROM Student
        WHERE Std_Id = @std_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
    @std_id int
```

2.4. Procedure: DeleteUserByUserName

Input/Output

	Name	Data type	Description
➔@	userName	varchar(50)	

Uses

Name
⚙ DeleteUserByUserName
🗃 User

Script

```
CREATE PROC DeleteUserByUserName @userName varchar(50)
AS
    BEGIN TRY
        DELETE FROM [User]
        WHERE U_UserName = @userName
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.5. Procedure: InsertInsForCrs

Input/Output

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

Uses

Name	
⚙️	InsertInsForCrs
📊	Ins_Crs

Script

```
CREATE PROC InsertInsForCrs
AS
    BEGIN TRY
        INSERT INTO Ins_Crs
        VALUES (@ins_id, @crs_id)
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.6. Procedure: InsertInstructor

Input/Output

Name		Data type	Description
→@	u_id	int	
→@	dept_id	int	
→@	fname	varchar(50)	
→@	lname	varchar(50)	
→@	deg	varchar(50)	
→@	sal	int	

Uses

Name	
⚙️	InsertInstructor
📊	Instructor

Script

```
CREATE PROC InsertInstructor
AS
    BEGIN TRY
        INSERT INTO Instructor
        VALUES (
            @u_id,
            @dept_id,
            @fname,
            @lname,
            @deg,
            @sal
        )
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.7. Procedure: InsertNewStd

Input/Output

Name		Data type	Description
→@	us_id	int	
→@	dept_id	int	
→@	fname	varchar(50)	
→@	lname	varchar(50)	
→@	bod	date	
→@	address	varchar(50)	

Uses

Name	
⚙️	InsertNewStd
📊	Student

Script

```
CREATE PROC InsertNewStd
AS
    BEGIN TRY
        INSERT INTO Student
        VALUES (
            @us_id,
            @dept_id,
            @fname,
            @lname,
            @bod,
            @address
        )
    END TRY
    BEGIN CATCH
        SELECT
        ERROR_STATE() AS ErrorState,
        ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.8. Procedure: InsertNewUser

Input/Output

Name		Data type	Description
➔@	us_name	varchar(50)	
➔@	us_mail	varchar(50)	
➔@	us_pass	varchar(50)	
➔@	us_sex	varchar(1)	
➔@	us_isStd	bit	

Uses

Name	
⚙️	InsertNewUser
📊	User

Script



```
CREATE PROC [dbo].[InsertNewUser]
AS
BEGIN TRY
    INSERT INTO [User]
    VALUES (@us_name, @us_mail, @us_pass, @us_sex, @us_isStd)
END TRY
BEGIN CATCH
    SELECT
        ERROR_STATE() AS ErrorState,
        ERROR_MESSAGE() AS ErrorMessage
END CATCH
```

2.9. Procedure: InsertQues

Input/Output

	Name	Data type	Description
→@	crs_id	int	
→@	cont	varchar(200)	
→@	grade	int	
→@	typ	varchar(3)	
→@	ans	int	

Uses

Name
 InsertQues
 Question

Script

```
CREATE PROC InsertQues
```

```
@crs_id int,  
@cont varchar(200),  
@grade int,  
@typ varchar(3),  
@ans int
```

```
AS
```

```
    BEGIN TRY  
        INSERT INTO Question  
        VALUES (  
            @crs_id,  
            @cont,  
            @grade,  
            @typ,  
            @ans  
        )  
    END TRY  
    BEGIN CATCH  
        SELECT  
            ERROR_STATE() AS ErrorState,  
            ERROR_MESSAGE() AS ErrorMessage  
    END CATCH
```

2.10. Procedure: SelectCrsIns

Input/Output

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

Uses

Name	
⚙️	SelectCrsIns
📊	Ins_Crs

Script

```
CREATE PROC SelectCrsIns

AS

    BEGIN TRY
        SELECT *
        FROM Ins_Crs
        WHERE Ins_Id = @ins_id
        AND Crs_Id = @crs_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
        END CATCH

/*****
```


2.11. Procedure: SelectInstructor

Input/Output

Name		Data type	Description
→@	ins_id	int	

Uses

Name	
⚙️	SelectInstructor
📊	Instructor

Script

```
CREATE PROC SelectInstructor @ins_id int
AS
    BEGIN TRY
        SELECT *
        FROM Instructor
        WHERE Ins_Id = @ins_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.12. Procedure: SelectStdById

Input/Output

Name		Data type	Description
→@	std_id	int	

Uses

Name	
⚙️	SelectStdById
📊	Student

Script

```
CREATE PROC SelectStdById
AS
    BEGIN TRY
        SELECT *
        FROM Student
        WHERE Std_Id = @std_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
    @std_id int
```

2.13. Procedure: SelectUserById

Input/Output

Name		Data type	Description
➔@	us_id	int	

Uses

Name	
⚙️	SelectUserById
📊	User

Script

```
CREATE PROC [dbo].[SelectUserById] @us_id int
AS
    BEGIN TRY
        SELECT *
        FROM [User]
        WHERE U_Id = @us_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.14. Procedure: sp_AssignAnsToStd

Input/Output

Name		Data type	Description
→@	std_id	int	
→@	exm_id	int	
→@	ans_dict	table type	

Uses

Name	
⚙️	sp_AssignAnsToStd
📊	Std_Ques_Exm

Script

```
CREATE PROC sp_AssignAnsToStd
AS
    BEGIN TRY
        DECLARE @LOCAL_TABLEVARIABLE TABLE
        (
            std_id int NULL,
            exm_id int NULL,
            ques_id int,
            ans varchar(1)
        )

        INSERT INTO @LOCAL_TABLEVARIABLE (ques_id, ans)
        SELECT * FROM @ans_dict

        UPDATE @LOCAL_TABLEVARIABLE
        SET
            std_id = @std_id,
            exm_id = @exm_id

        INSERT INTO Std_Ques_Exm
        SELECT * FROM @LOCAL_TABLEVARIABLE
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.15. Procedure: sp_CorrectStdExam

Input/Output

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

Uses

Name	
⚙ sp_CorrectStdExam	
📄 Choice	
📄 Exam	
📄 Question	
📄 Std_Crs	
📄 Std_Ques_Exm	

Script

```
CREATE PROC [dbo].[sp_CorrectStdExam]

AS

BEGIN TRY

    DECLARE @LOCAL_TABLEVARIABLE TABLE
    (
        mod_ans varchar(1),
        std_ans varchar(1)
    )

    INSERT INTO @LOCAL_TABLEVARIABLE
    SELECT Cho_Char, Std_Answer
    FROM Choice c
    JOIN Question q ON (q.Model_Ans = c.Cho_Id)
    JOIN Std_Ques_Exm sqe ON (sqe.Ques_Id = q.Ques_Id)
    WHERE Exm_Id = @exm_id
    AND Std_Id = @std_id
    declare c cursor
    FOR SELECT * FROM @LOCAL_TABLEVARIABLE

    DECLARE @mod varchar(1), @ans varchar(1)
    DECLARE @scr int, @grade int, @crs_id int
    SET @scr = 0
    SET @grade = 0
    OPEN c
    FETCH c INTO @mod, @ans
    while @@FETCH_STATUS = 0
    BEGIN
        IF @mod = @ans
        BEGIN
            SET @scr = @scr + 1
        END
        FETCH c INTO @mod, @ans
    END
    SELECT @grade = Exm_Grade,
           @crs_id = Crs_Id
    FROM Exam
    WHERE Exm_Id = @exm_id

    DECLARE @ques_grade int
    SET @ques_grade = @grade / 10
    SET @scr = @scr * @ques_grade
    UPDATE Std_Crs
    SET
        Grade = @scr,
        Date = GETDATE()
    WHERE Std_Id = @std_id
    AND Crs_Id = @crs_id

END TRY
BEGIN CATCH
    SELECT
        ERROR_STATE() AS ErrorState,
        ERROR_MESSAGE() AS ErrorMessage
END CATCH
```

```
@std_id int,
@exm_id int
```

2.16. Procedure: sp_CourseNameWithStudentCount

Input/Output

Name		Data type	Description
→@	ins_id	int	

Uses

Name	
⚙️	sp_CourseNameWithStudentCount
📄	Course
📄	Ins_Crs
📄	Std_Crs

Script

```
CREATE PROC sp_CourseNameWithStudentCount @ins_id INT
as
SELECT Crs_Name,
COUNT(Std_Id) AS [Students count]
FROM dbo.Course
INNER JOIN dbo.Ins_Crs
ON dbo.Course.Crs_Id = dbo.Ins_Crs.Crs_Id
INNER JOIN dbo.Std_Crs
ON Std_Crs.Crs_Id = Course.Crs_Id
WHERE dbo.Ins_Crs.Ins_Id = @ins_id GROUP BY Crs_Name;
```

2.17. Procedure: sp_deleteChoice

Input/Output

Name		Data type	Description
→@	cho_id	int	

Uses

Name	
⚙️	sp_deleteChoice
📊	Choice

Script

```
CREATE PROC sp_deleteChoice @cho_id INT
AS
BEGIN TRY
    DELETE FROM dbo.Choice
    WHERE Cho_Id = @cho_id;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```


2.18. Procedure: sp_deleteCourse

Input/Output

Name		Data type	Description
➔@	crs_id	int	

Uses

Name	
⚙️	sp_deleteCourse
📄	Course

Script

```
CREATE PROC sp_deleteCourse @crs_id INT
AS
BEGIN TRY
    DELETE FROM dbo.Course
    WHERE Crs_Id = @crs_id;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```

2.19. Procedure: sp_deleteDepartment

Input/Output

Name		Data type	Description
➔@	deptId	int	

Uses

Name	
⚙️	sp_deleteDepartment
📊	Department

Script

```
CREATE PROC sp_deleteDepartment @deptId INT
AS
BEGIN TRY
    DELETE FROM dbo.Department WHERE Dept_Id=@deptId;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH
```

2.20. Procedure: sp_deleteExam

Input/Output

Name		Data type	Description
→@	exam_id	int	

Uses

Name	
⚙ sp_deleteExam	
📄 Exam	

Script

```
CREATE PROC sp_deleteExam @exam_id INT
AS
BEGIN TRY
    delete FROM dbo.Exam WHERE Exm_Id =@exam_id
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH
```

2.21. Procedure: sp_DeleteQues

Input/Output

Name		Data type	Description
→@	ques_id	int	

Uses

Name	
⚙ sp_DeleteQues	
📄 Question	

Script

```
CREATE PROC sp_DeleteQues
AS
    BEGIN TRY
        DELETE FROM Question
        WHERE Ques_Id = @ques_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH

/*****
```

2.22. Procedure: sp_DeleteStdFromCrs

Input/Output

Name		Data type	Description
➔@	std_id	int	
➔@	crs_id	int	

Uses

Name	
⚙ sp_DeleteStdFromCrs	
📊 Std_Crs	

Script

```
CREATE PROC sp_DeleteStdFromCrs
AS
    BEGIN TRY
        DELETE FROM Std_Crs
        WHERE Std_Id = @std_id
        AND Crs_Id = @crs_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.23. Procedure: sp_deleteTopic

Input/Output

Name		Data type	Description
➔@	crs_id	int	
➔@	crs_top	varchar(50)	

Uses

Name	
⚙️	sp_deleteTopic
📊	Crs_Top

Script

```
CREATE PROC sp_deleteTopic
    @crs_id INT,
    @crs_top VARCHAR(50)
AS
BEGIN TRY
    DELETE FROM dbo.Crs_Top
    WHERE Crs_Id = @crs_id
    AND Crs_Top = @crs_top;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```

2.24. Procedure: sp_ExamQuestionsAndAnswers

Input/Output

Name		Data type	Description
→@	exm_id	int	

Uses

Name	
⚙ sp_ExamQuestionsAndAnswers	
📄 Choice	
📄 Exm_Ques	
📄 Question	

Script

```
CREATE PROC sp_ExamQuestionsAndAnswers @exm_id INT
as
SELECT Ques_Content, Cho_Content
FROM Exm_Ques eq
INNER JOIN Question q
    ON (q.Ques_Id = eq.Ques_Id)
INNER JOIN Choice c
    ON (c.Ques_Id = q.Ques_Id)
WHERE eq.Exm_Id = @exm_id
```

2.25. Procedure: sp_GenerateExam

Input/Output

Name		Data type	Description
→@	mcq	int	
→@	tfq	int	
→@	crs_id	int	
→@	ins_id	int	
→@	grade	int	

Uses

Name
⚙️ sp_GenerateExam
📄 Exam
📄 Exm_Ques
📄 Question

Script

```
CREATE PROC [dbo].[sp_GenerateExam]
AS
BEGIN TRY
    INSERT INTO Exam
    VALUES (@crs_id, @ins_id, @grade);

    DECLARE @exm_id int;
    SET @exm_id = (SELECT MAX(Exm_Id) FROM Exam);

    DECLARE @LOCAL_TABLEVARIABLE TABLE
    (
        ex_id int NULL,
        ques_id int NULL,
        grade int NULL
    )

    INSERT INTO @LOCAL_TABLEVARIABLE (ques_id)
    SELECT TOP(@mcq) Ques_Id
    FROM Question
    WHERE Ques_Type = 'MCQ' AND Crs_Id = @crs_id
    ORDER BY NEWID()

    INSERT INTO @LOCAL_TABLEVARIABLE (ques_id)
    SELECT TOP(@tfq) Ques_Id
    FROM Question
    WHERE Ques_Type = 'TFQ' AND Crs_Id = @crs_id
    ORDER BY NEWID()

    UPDATE @LOCAL_TABLEVARIABLE
    SET ex_id = @exm_id,
        grade = 1

    INSERT INTO Exm_Ques
    SELECT * FROM @LOCAL_TABLEVARIABLE
END TRY
BEGIN CATCH
    SELECT
    ERROR_STATE() AS ErrorState,
    ERROR_MESSAGE() AS ErrorMessage
END CATCH

/*****/
```


2.26. Procedure: sp_InsertChoice

Input/Output

Name		Data type	Description
→@	ques_id	int	
→@	cho_content	varchar(200)	
→@	cho_char	varchar(1)	

Uses

Name	
⚙️	sp_InsertChoice
📊	Choice

Script

```
--choice
CREATE PROC sp_InsertChoice
    @ques_id INT,
    @cho_content VARCHAR(200),
    @cho_char VARCHAR(1)
AS
BEGIN TRY
    INSERT INTO dbo.Choice
    (
        Ques_Id,
        Cho_Content,
        Cho_Char
    )
    VALUES
    (
        @ques_id,      -- Ques_Id - int
        @cho_content,  -- Cho_Content - varchar(200)
        @cho_char      -- Cho_Char - varchar(1)
    );
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```

2.27. Procedure: sp_insertCourse

Input/Output

	Name	Data type	Description
→@	crs_name	varchar(50)	
→@	crs_desc	varchar(50)	
→@	crs_dur	int	

Uses

Name
⚙️ sp_insertCourse
📄 Course

Script

```
CREATE PROC sp_insertCourse
    @crs_name VARCHAR(50),
    @crs_desc VARCHAR(50),
    @crs_dur INT
AS
BEGIN TRY
    INSERT INTO dbo.Course
    (
        Crs_Name,
        Crs_Desc,
        Crs_Dur
    )
    VALUES
    (
        @crs_name, -- Crs_Name - varchar(50)
        @crs_desc, -- Crs_Desc - varchar(50)
        @crs_dur  -- Crs_Dur - int
    );
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```

2.28. Procedure: sp_insertDepartment

Input/Output

	Name	Data type	Description
→@	dept_name	varchar(50)	
→@	dept_loc	varchar(50)	
→@	dept_managerHireDate	date	
→@	dept_managerId	int	

Uses

Name
⚙️ sp_insertDepartment
📊 Department

Script

```
CREATE PROC sp_insertDepartment
    @dept_name VARCHAR(50),
    @dept_loc VARCHAR(50),
    @dept_managerHireDate DATE,
    @dept_managerId INT
AS
BEGIN TRY
    INSERT INTO dbo.Department
    (
        Dept_Name,
        Dept_Loc,
        Dept_ManagerHireDate,
        Dept_ManagerId
    )
    VALUES
    (
        @dept_name,           -- Dept_Name - varchar(50)
        @dept_loc,            -- Dept_Loc - varchar(50)
        @dept_managerHireDate, -- Dept_ManagerHireDate - date
        @dept_managerId       -- Dept_ManagerId - int
    );
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```

2.29. Procedure: sp_insertExam

Input/Output

Name		Data type	Description
→@	crs_id	int	
→@	generator_id	int	
→@	exam_grade	int	

Uses

Name	
⚙️	sp_insertExam
📊	Exam

Script



```
--exam
CREATE PROC sp_insertExam @crs_id INT,@generator_id INT ,@exam_grade INT
AS
BEGIN TRY
    INSERT INTO dbo.Exam
    (
        Crs_Id,
        Generator_Id,
        Exm_Grade
    )
    VALUES
    (
        @crs_id, -- Crs_Id - int
        @generator_id, -- Generator_Id - int
        @exam_grade -- Exm_Grade - int
    )
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH
```

2.30. Procedure: sp_InsertStdAnsForExm

Input/Output

	Name	Data type	Description
→@	std_id	int	
→@	Exm_id	int	
→@	ques_id	int	
→@	ans	varchar(1)	

Uses

Name	
	sp_InsertStdAnsForExm
	Std_Ques_Exm

Script

/ ***** /

```
CREATE PROC sp_InsertStdAnsForExm
```

AS

BEGIN TRY

```
INSERT INTO Std_Ques_Exm
VALUES (
```

```
@std_id ,
@Exm_id ,
@ques_id ,
@ans
```

)

END TRY

```
BEGIN CATCH
```

SELECT

```

ERROR_STATE() AS ErrorState,
ERROR_MESSAGE() AS ErrorMessage
END CATCH

```

```
@std_id int,  
@Exm_id int,  
@ques_id int,  
@ans varchar(1)
```

2.31. Procedure: sp_InsertStdForCrs

Input/Output

Name		Data type	Description
→@	std_id	int	
→@	crs_id	int	
→@	grade	int	
→@	date	date	

Uses

Name	
⚙️	sp_InsertStdForCrs
📊	Std_Crs

Script

```
CREATE PROC sp_InsertStdForCrs
AS
    BEGIN TRY
        INSERT INTO Std_Crs
        VALUES (
            @std_id,
            @crs_id,
            @grade,
            @date
        )
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.32. Procedure: sp_insertTopic

Input/Output

Name		Data type	Description
➔@	crs_id	int	
➔@	crs_top	varchar(50)	

Uses

Name	
⚙️	sp_insertTopic
📊	Crs_Top

Script

```
CREATE PROC sp_insertTopic
    @crs_id INT,
    @crs_top VARCHAR(50)
AS
BEGIN TRY
    INSERT INTO dbo.Crs_Top
    (
        Crs_Id,
        Crs_Top
    )
    VALUES
    (
        @crs_id, -- Crs_Id - int
        @crs_top -- Crs_Top - varchar(50)
    );
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```

2.33. Procedure: sp_selectchoice

Input/Output

Name		Data type	Description
→@	id	int	

Uses

Name	
⚙️	sp_selectchoice
📄	Choice

Script

```
CREATE PROC sp_selectchoice @id INT
AS
BEGIN TRY
    SELECT *
    FROM dbo.Choice
    WHERE Cho_Id = @id;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;

---course
```


2.34. Procedure: sp_selectCourse

Input/Output

Name		Data type	Description
→@	crs_id	int	

Uses

Name	
⚙ sp_selectCourse	
📄 Course	

Script

```
CREATE PROC sp_selectCourse @crs_id INT
AS
BEGIN TRY
    SELECT *
    FROM dbo.Course
    WHERE Crs_Id = @crs_id;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;

--crs_top
```

2.35. Procedure: sp_selectDepartment

Input/Output

Name		Data type	Description
→@	dept_id	int	

Uses

Name	
⚙ sp_selectDepartment	
📊 Department	

Script

```
CREATE PROC sp_selectDepartment @dept_id INT
AS
BEGIN TRY
    SELECT * FROM dbo.Department WHERE Dept_Id =@dept_id;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH
```

2.36. Procedure: sp_selectExam

Input/Output

Name		Data type	Description
→@	exam_id	int	

Uses

Name	
⚙️	sp_selectExam
📊	exam

Script

```
CREATE PROC sp_selectExam @exam_id INT
AS
BEGIN TRY
    SELECT * FROM exam WHERE Exm_Id =@exam_id;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH
```

2.37. Procedure: sp_SelectQues

Input/Output

Name		Data type	Description
→@	ques_id	int	

Uses

Name	
⚙️	sp_SelectQues
📄	Question

Script

```
CREATE PROC sp_SelectQues
AS
    BEGIN TRY
        SELECT *
        FROM Question
        WHERE Ques_Id = @ques_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.38. Procedure: sp_SelectStdAnsForExm

Input/Output

Name		Data type	Description
→@	std_id	int	
→@	Exm_id	int	
→@	ques_id	int	

Uses

Name	
⚙ sp_SelectStdAnsForExm	
📊 Std_Ques_Exm	

Script

```
CREATE PROC sp_SelectStdAnsForExm
AS
    BEGIN TRY
        SELECT *
        FROM Std_Ques_Exm
        WHERE Std_Id = @std_id
        AND Exm_Id = @Exm_id
        AND Ques_Id = @ques_id
    END TRY
    BEGIN CATCH
        SELECT
        ERROR_STATE() AS ErrorState,
        ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.39. Procedure: sp_SelectStdOfCrs

Input/Output

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

Uses

Name	
⚙ sp_SelectStdOfCrs	
📊 Std_Crs	

Script

```
CREATE PROC sp_SelectStdOfCrs
AS
    BEGIN TRY
        SELECT *
        FROM Std_Crs
        WHERE Std_Id = @std_id
        AND Crs_Id = @crs_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.40. Procedure: sp_selectTopic

Input/Output

Name		Data type	Description
➔@	crs_id	int	
➔@	crs_topic	varchar(50)	

Uses

Name	
⚙️	sp_selectTopic
📊	Crs_Top

Script

```
CREATE PROC sp_selectTopic
    @crs_id INT,
    @crs_topic VARCHAR(50)
AS
BEGIN TRY
    SELECT *
    FROM dbo.Crs_Top
    WHERE Crs_Id = @crs_id
        AND Crs_Top = @crs_topic;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
        ERROR_MESSAGE() AS ErrorMessage;
END CATCH;

--dept
```

2.41. Procedure: sp_StdExamAnswers

Input/Output

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

Uses

Name	
⚙ sp_StdExamAnswers	
📄 Choice	
📄 Question	
📄 Std_Ques_Exm	

Script

```
CREATE PROC sp_StdExamAnswers @exm_id INT, @std_id INT
as
SELECT Ques_Content, Std_Answer
FROM Std_Ques_Exm sqe
INNER JOIN Question q
    ON (q.Ques_Id = sqe.Ques_Id)
INNER JOIN Choice c
    ON (sqe.Std_Answer = c.Cho_Id)
WHERE sqe.Exm_Id = @exm_id
    AND sqe.Std_Id = @std_id
```


2.42. Procedure: sp_StudentGradePerCourse

Input/Output

Name		Data type	Description
➔@	std_id	int	

Uses

Name	
⚙️	sp_StudentGradePerCourse
📄	Course
📄	Std_Crs
📄	Student

Script

```
CREATE PROC sp_StudentGradePerCourse @std_id INT
AS
SELECT CONCAT(Std_Fname, ' ', Std_Lname) AS Student, c.Crs_Name AS Course, grade AS Grade
FROM Std_Crs sc
JOIN Student s
ON (s.Std_Id = sc.Std_Id)
JOIN Course c
ON (c.Crs_Id = sc.Crs_Id)
WHERE sc.Std_Id = @std_id
AND sc.Grade IS NOT NULL
```

2.43. Procedure: sp_StudentperDepartment

Input/Output

Name		Data type	Description
→@	dept_id	int	

Uses

Name	
⚙	sp_StudentperDepartment
📊	Student

Script

```
CREATE PROC sp_StudentperDepartment @dept_id INT
as
SELECT * FROM dbo.Student s WHERE s.Dept_Id = @dept_id;
```

2.44. Procedure: sp_TopicsPerCourse

Input/Output

Name		Data type	Description
➔@	crs_id	int	

Uses

Name	
⚙️	sp_TopicsPerCourse
📊	Crs_Top

Script



```
CREATE PROC sp_TopicsPerCourse @crs_id INT
as
SELECT dbo.Crs_Top.Crs_Top AS Topics FROM dbo.Crs_Top WHERE crs_id =@crs_id
```

2.45. Procedure: sp_updateChoice

Input/Output

	Name	Data type	Description
→@	cho_id	int	
→@	ques_id	int	
→@	cho_content	varchar(200)	
→@	cho_char	varchar(1)	

Uses

Name
 sp_updateChoice
 Choice

Script

```
CREATE PROC sp_updateChoice
    @cho_id INT,
    @ques_id INT,
    @cho_content VARCHAR(200),
    @cho_char VARCHAR(1)
AS
BEGIN TRY
    UPDATE dbo.Choice
    SET Ques_Id = @ques_id,
        Cho_Content = @cho_content,
        Cho_Char = @cho_char
    WHERE Cho_Id = @cho_id;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```

2.46. Procedure: sp_updateCourse

Input/Output

Name		Data type	Description
➔@	crs_id	int	
➔@	crs_name	varchar(50)	
➔@	crs_desc	varchar(50)	
➔@	crs_dur	int	

Uses

Name	
⚙️	sp_updateCourse
📊	Course

Script

```
CREATE PROC sp_updateCourse
    @crs_id INT,
    @crs_name VARCHAR(50),
    @crs_desc VARCHAR(50),
    @crs_dur INT
AS
BEGIN TRY
    UPDATE dbo.Course
    SET Crs_Name = @crs_name,
        Crs_Desc = @crs_desc,
        Crs_Dur = @crs_dur
    WHERE Crs_Id = @crs_id;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```

2.47. Procedure: sp_UpdateCrsForIns

Input/Output

Name		Data type	Description
→@	ins_id	int	
→@	old_crs_id	int	
→@	new_crs_id	int	

Uses

Name	
⚙️	sp_UpdateCrsForIns
📊	Ins_Crs

Script

```
CREATE PROC sp_UpdateCrsForIns
AS
    BEGIN TRY
        UPDATE Ins_Crs
        SET
            Crs_Id = @new_crs_id
        WHERE Crs_Id = @ins_id
        AND Crs_Id = @old_crs_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
    @ins_id int,
    @old_crs_id int,
    @new_crs_id int
```

2.48. Procedure: sp_UpdateCrsOfStd

Input/Output

	Name	Data type	Description
→@	std_id	int	
→@	old_crs_id	int	
→@	new_crs_id	int	
→@	grade	int	
→@	date	date	

Uses

Name
⚙️ sp_UpdateCrsOfStd
📊 Std_Crs

Script

```
CREATE PROC sp_UpdateCrsOfStd
```

```
@std_id int,  
@old_crs_id int,  
@new_crs_id int,  
@grade int = NULL,  
@date date = NULL
```

```
AS
```

```
    BEGIN TRY  
        UPDATE Std_Crs  
        SET  
            Crs_Id = @new_crs_id,  
            Grade = @grade,  
            [Date] = @date  
        WHERE Std_Id = @std_id  
        AND Crs_Id = @old_crs_id  
    END TRY  
    BEGIN CATCH  
        SELECT  
            ERROR_STATE() AS ErrorState,  
            ERROR_MESSAGE() AS ErrorMessage  
    END CATCH
```

2.49. Procedure: sp_updateDepartment

Input/Output

Name		Data type	Description
➔@	dept_id	int	
➔@	dept_name	varchar(50)	
➔@	dept_loc	varchar(50)	
➔@	dept_managerHireDate	date	
➔@	dept_managerId	int	

Uses

Name
⚙️ sp_updateDepartment
📊 Department

Script

```
CREATE PROC sp_updateDepartment
    @dept_id INT,
    @dept_name VARCHAR(50),
    @dept_loc VARCHAR(50),
    @dept_managerHireDate DATE,
    @dept_managerId INT
AS
BEGIN TRY
    UPDATE dbo.Department
    SET Dept_Name = @dept_name,
        Dept_Loc = @dept_loc,
        Dept_ManagerHireDate = @dept_managerHireDate,
        Dept_ManagerId = @dept_managerId
    WHERE Dept_Id = @dept_id;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```


2.50. Procedure: sp_updateExam

Input/Output

Name		Data type	Description
→@	exam_id	int	
→@	crs_id	int	
→@	generator_id	int	
→@	exam_grade	int	

Uses

Name	
⚙️ sp_updateExam	
📊 Exam	

Script

```
CREATE PROC sp_updateExam @exam_id INT ,@crs_id INT ,@generator_id INT ,@exam_grade INT
AS
BEGIN TRY
    UPDATE dbo.Exam SET Crs_Id =@crs_id,Generator_Id =@generator_id,Exm_Grade=@exam_grade
    WHERE Exm_Id =@exam_id;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH
```

2.51. Procedure: sp_updateTopic

Input/Output

Name		Data type	Description
→@	crs_id	int	
→@	crs_top	varchar(50)	

Uses

Name	
⚙️	sp_updateTopic
📊	Crs_Top

Script



```
CREATE PROC sp_updateTopic
    @crs_id INT,
    @crs_top VARCHAR(50)
AS
BEGIN TRY
    UPDATE dbo.Crs_Top
    SET Crs_Top = @crs_top
    WHERE Crs_Id = @crs_id
        AND Crs_Top = @crs_top;
END TRY
BEGIN CATCH
    SELECT ERROR_STATE() AS ErrorState,
           ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```

2.52. Procedure: UpdateInstructor

Input/Output

	Name	Data type	Description
→@	ins_id	int	
→@	u_id	int	
→@	dept_id	int	
→@	fname	varchar(50)	
→@	lname	varchar(50)	
→@	deg	varchar(50)	
→@	sal	int	

Uses

Name
 UpdateInstructor
 Instructor

Script

CREATE PROC UpdateInstructor

AS

```
BEGIN TRY
    UPDATE Instructor
    SET
        U_Id =@u_id,
        Dept_Id = @dept_id,
        Ins_Fname = @fname,
        Ins_Lname = @lname,
        Ins_Degree = @deg,
        Ins_Salary = @sal
    WHERE Ins_Id = @ins_id
END TRY
BEGIN CATCH
    SELECT
        ERROR_STATE() AS ErrorState,
        ERROR_MESSAGE() AS ErrorMessage
END CATCH

/*****/
```

2.53. Procedure: UpdateQues

Input/Output

Name		Data type	Description
↗@	ques_id	int	
↗@	crs_id	int	
↗@	cont	varchar(200)	
↗@	grade	int	
↗@	typ	varchar(3)	
↗@	ans	int	

Uses

Name	
⚙ UpdateQues	
📄 Question	

Script

```
CREATE PROC UpdateQues
AS
    BEGIN TRY
        UPDATE Question
        SET
            Crs_Id = @crs_id,
            Ques_Content = @cont,
            Ques_Grade = @grade,
            Ques_Type = @typ,
            Model_Ans = @ans
        WHERE Ques_Id = @ques_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.54. Procedure: UpdateStdInfo

Input/Output

Name		Data type	Description
→@	std_id	int	
→@	us_id	int	
→@	dept_id	int	
→@	fname	varchar(50)	
→@	lname	varchar(50)	
→@	bod	date	
→@	address	varchar(50)	

Uses

Name	
⚙ UpdateStdInfo	
📄 Student	

Script

```
CREATE PROC UpdateStdInfo
AS
    BEGIN TRY
        UPDATE Student
        SET
            U_Id = @us_id,
            Dept_Id = @dept_id,
            Std_Fname = @fname,
            Std_Lname = @lname,
            Std_BOD = @bod,
            Std_Address = @address
        WHERE Std_Id = @std_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

2.55. Procedure: UpdateUser

Input/Output

	Name	Data type	Description
→@	us_id	int	
→@	us_name	varchar(50)	
→@	us_mail	varchar(50)	
→@	us_pass	varchar(50)	
→@	us_sex	varchar(1)	
→@	us_isStd	bit	

Uses

Name
⚙ UpdateUser
📊 User

Script

```
CREATE PROC [dbo].[UpdateUser]
    @us_id int,
    @us_name varchar(50),
    @us_mail varchar(50),
    @us_pass varchar(50),
    @us_sex varchar(1),
    @us_isStd bit
AS
    BEGIN TRY
        UPDATE [User]
        SET
            U_UserName = @us_name,
            U_Email = @us_mail,
            U_Password = @us_pass,
            U_Sex = @us_sex,
            U_IsStd = @us_isStd
        WHERE U_Id = @us_id
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_STATE() AS ErrorState,
            ERROR_MESSAGE() AS ErrorMessage
    END CATCH
```

TRIAL