Online-Examination-system

Online-Examination-System

Server	DESKTOP-EKCK12U
Author	Team (7)
Created	Tuesday, February 28, 2023 3:24:24 PM
File Path	C:\Users\RadwaElgammal\Desktop\(local)__documentation-2023-02-28T15-24-24.pdf

Online Examination System which enables instructors to generate exams according to their courses and decide how many questions will be in the exam and take student answers and calculate grades to each student

Table of Contents

DESKTOP-EKCK12U

Databases (1)

- Online-Examination-System

Server Properties

Property	Value
Product	Microsoft SQL Server
Version	14.0.2047.8
Language	English (United States)
Platform	NT x64
Edition	Developer Edition (64-bit)
Engine Edition	3 (Enterprise)
Processors	8
OS Version	6.3 (22621)
Physical Memory	8057
Is Clustered	False
Root Directory	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL
Collation	SQL_Latin1_General_CP1_CI_AS

Server Settings

Property	Value
Default data file path	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\
Default backup file path	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\Backup
Default log file path	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\
Recovery Interval (minutes)	0
Default index fill factor	0
Default backup media retention	0
Compress Backup	False

Advanced Server Settings

Property	Value
Locks	0
Nested triggers enabled	True

Allow triggers to fire others	True
Default language	English
Network packet size	4096
Default fulltext language LCID	1033
Two-digit year cutoff	2049
Remote login timeout	10
Cursor threshold	-1
Max text replication size	65536
Parallelism cost threshold	5
Max degree of parallelism	0
Min server memory	16
Max server memory	2147483647
Scan for startup procs	False
Transform noise words	False
CLR enabled	False
Blocked process threshold	0
Filestream access level	False
Optimize for ad hoc workloads	False
CLR strict security	True

User databases

Databases (1)

- Online-Examination-System

Online-Examination-System Database

Database Properties

Property	Value
SQL Server Version	SQL Server 2017
Compatibility Level	SQL Server 2017
Last backup time	-
Last log backup time	-
Creation date	Feb 20 2023
Users	6
Database Encryption Enabled	False
Database Encryption Algorithm	None
Database size	16.00 MB
Unallocated space	3.55 MB

Database Options

Property	Value
Compatibility Level	140
Database collation	SQL_Latin1_General_CP1_CI_AS
Restrict access	MULTI_USER
Is read-only	False
Auto close	False
Auto shrink	False
Database status	ONLINE
In standby	False
Cleanly shutdown	False
Supplemental logging enabled	False
Snapshot isolation state	OFF
Read committed snapshot on	False
Recovery model	FULL
Page verify option	CHECKSUM
Auto create statistics	True
Auto update statistics	True
Auto update statistics asynchronously	False
ANSI NULL default	False
ANSI NULL enabled	False
ANSI padding enabled	False

ANSI warnings enabled	False
Arithmetic abort enabled	False
Concatenating NULL yields NULL	False
Numeric roundabort enabled	False
Quoted Identifier On	False
Recursive triggers enabled	False
Close cursors on commit	False
Local cursors by default	False
Fulltext enabled	True
Trustworthy	False
Database chaining	False
Forced parameterization	False
Master key encrypted by server	False
Published	False
Subscribed	False
Merge published	False
Is distribution database	False
Sync with backup	False
Service broker GUID	ff7e174e-1ea2-4ca3-af2d-bb8e9603af30
Service broker enabled	False
Log reuse wait	NOTHING
Date correlation	False
CDC enabled	False
Encrypted	False
Honor broker priority	False
Default language	English
Default fulltext language LCID	1033
Nested triggers enabled	True
Transform noise words	False
Two-digit year cutoff	2049
Containment	NONE
Target recovery time	60
Database owner	DESKTOP-EKCK12U\RadwaElgammal

Files

Name	Type	Size	Maxsize	Autogrowth	File Name
Online-Examination-System	Data	8.00 MB	unlimited	64.00 MB	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\Online-Examination-System.mdf

Online-Examination-System_log	Log	8.00 MB	2048.00 GB	64.00 MB	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\Online-Examination-System_log.ldf

Tables

Objects

Name
dbo.Courses
dbo.Department
dbo.Exam_Questions
dbo.Exams
dbo.Instructor
dbo.Instructor_Course
dbo.Question_Choice
dbo.Questions
dbo.Student
dbo.Student_Course
dbo.Student_Exam_Question
dbo.Topic
dbo.Work_In

[dbo].[Courses]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	8
Created	5:07:50 PM Monday, February 20, 2023
Last Modified	12:53:39 AM Monday, February 27, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK	Crs_id	int	4	NOT NULL
	Name	varchar(50)	50	NULL allowed
	Duration	int	4	NULL allowed
	Description	varchar(200)	200	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK	PK_Courses	Crs_id	True

SQL Script

```
CREATE TABLE [dbo].[Courses]
(
[Crs_id] [int] NOT NULL,
[Name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Duration] [int] NULL,
[Description] [varchar] (200) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Courses] ADD CONSTRAINT [PK_Courses] PRIMARY KEY CLUSTERED
([Crs_id]) ON [PRIMARY]
GO
```

Used By

[dbo].[Exams]
[dbo].[Instructor_Course]

[dbo].[Questions]
[dbo].[Student_Course]
[dbo].[Topic]
[dbo].[addCourse]
[dbo].[courseTopics]
[dbo].[CreateExam]
[dbo].[deleteCourse]
[dbo].[getInstructorCourses]
[dbo].[instructorCourses]
[dbo].[studentGrades]
[dbo].[updateCourse]

[dbo].[Department]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	5
Created	5:07:50 PM Monday, February 20, 2023
Last Modified	10:56:18 AM Sunday, February 26, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK C	Did	int	4	NOT NULL
	Dname	varchar(50)	50	NULL allowed
	Description	varchar(200)	200	NULL allowed
	Location	nvarchar(200)	400	NULL allowed
FK	Mng_id	int	4	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Department	Did	True

Foreign Keys

Name	Update	Delete	Columns
FK_Department_Instructor	Cascade	SetNull	Mng_id->[dbo].[Instructor].[Ins_id]

SQL Script

```
CREATE TABLE [dbo].[Department]
(
[Did] [int] NOT NULL,
[Dname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Description] [varchar] (200) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Location] [nvarchar] (200) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Mng_id] [int] NULL
) ON [PRIMARY]
GO
```

```
ALTER TABLE [dbo].[Department] ADD CONSTRAINT [PK_Department] PRIMARY KEY CLUSTERED
([Did]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Department] ADD CONSTRAINT [FK_Department_Instructor] FOREIGN KEY
([Mng_id]) REFERENCES [dbo].[Instructor] ([Ins_id]) ON DELETE SET NULL ON UPDATE
CASCADE
GO
```

Uses

[dbo].[Instructor]

Used By

[dbo].[Student]
[dbo].[Work_In]
[dbo].[addDepartment]
[dbo].[deleteDepartment]
[dbo].[departmentStudents]
[dbo].[updateDepartment]

[dbo].[Exam_Questions]

Properties

Property	Value
Row Count (~)	80
Created	10:56:11 AM Sunday, February 26, 2023
Last Modified	12:53:39 AM Monday, February 27, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK C FK	Exm_id	int	4	NOT NULL
PK C FK	Qst_id	int	4	NOT NULL

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Exam_Questions	Exm_id, Qst_id	True

Foreign Keys

Name	Columns
FK_Exam_Questions_Exams	Exm_id->[dbo].[Exams].[Exm_id]
FK_Exam_Questions_Questions	Qst_id->[dbo].[Questions].[Qst_id]

SQL Script

```
CREATE TABLE [dbo].[Exam_Questions]
(
[Exm_id] [int] NOT NULL,
[Qst_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Exam_Questions] ADD CONSTRAINT [PK_Exam_Questions] PRIMARY KEY CLUSTERED ([Exm_id], [Qst_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Exam_Questions] ADD CONSTRAINT [FK_Exam_Questions_Exams] FOREIGN KEY ([Exm_id]) REFERENCES [dbo].[Exams] ([Exm_id])
```

```
GO
ALTER TABLE [dbo].[Exam_Questions] ADD CONSTRAINT [FK_Exam_Questions_Questions] FOREIGN KEY ([Qst_id]) REFERENCES [dbo].[Questions] ([Qst_id])
GO
```

Uses

[dbo].[Exams]
[dbo].[Questions]

Used By

[dbo].[CreateExam]
[dbo].[deleteExamByCrsId]
[dbo].[DeleteExamByID]
[dbo].[getExam]
[dbo].[PrintExam]
[dbo].[StudentExamAnswers]

[dbo].[Exams]

Properties

Property	Value
Row Count (~)	20
Created	10:56:11 AM Sunday, February 26, 2023
Last Modified	12:53:39 AM Monday, February 27, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
PK C	Exm_id	int	4	NOT NULL	1 - 1
	Duration	int	4	NULL allowed	
FK	Crs_id	int	4	NULL allowed	
	Date	date	3	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Exams	Exm_id	True

Foreign Keys

Name	Columns
FK_Exams_Courses	Crs_id->[dbo].[Courses].[Crs_id]

SQL Script

```
CREATE TABLE [dbo].[Exams]
(
[Exm_id] [int] NOT NULL IDENTITY(1, 1),
[Duration] [int] NULL,
[Crs_id] [int] NULL,
[Date] [date] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Exams] ADD CONSTRAINT [PK_Exams] PRIMARY KEY CLUSTERED ([Exm_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Exams] ADD CONSTRAINT [FK_Exams_Courses] FOREIGN KEY ([Crs_id])
```

```
REFERENCES [dbo].[Courses] ([Crs_id])
GO
```

Uses

[dbo].[Courses]

Used By

[dbo].[Exam_Questions]
[dbo].[Student_Exam_Question]
[dbo].[CreateExam]
[dbo].[deleteExamByCrsId]
[dbo].[DeleteExamByID]
[dbo].[Exam_Correction]
[dbo].[Get_All_Exams]
[dbo].[PrintExam]
[dbo].[StudentExamAnswers]
[dbo].[updateExam]

[dbo].[Instructor]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	7
Created	5:07:50 PM Monday, February 20, 2023
Last Modified	10:56:18 AM Sunday, February 26, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK C	Ins_id	int	4	NOT NULL
	Fname	varchar(50)	50	NULL allowed
	Lname	varchar(50)	50	NULL allowed
	Academics_Degree	varchar(100)	100	NULL allowed
	Salary	int	4	NULL allowed
	Address	varchar(200)	200	NULL allowed
	Phone	int	4	NULL allowed
FK	Super_id	int	4	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Instructor	Ins_id	True

Foreign Keys

Name	Columns
FK_Instructor_Instructor1	Super_id->[dbo].[Instructor].[Ins_id]

SQL Script

```
CREATE TABLE [dbo].[Instructor]
(
[Ins_id] [int] NOT NULL,
[Fname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Lname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
```

```
[Academics_Degree] [varchar] (100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Salary] [int] NULL,
[Address] [varchar] (200) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Phone] [int] NULL,
[Super_id] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Instructor] ADD CONSTRAINT [PK_Instructor] PRIMARY KEY CLUSTERED
([Ins_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Instructor] ADD CONSTRAINT [FK_Instructor_Instructor1] FOREIGN KEY
([Super_id]) REFERENCES [dbo].[Instructor] ([Ins_id])
GO
```

Used By

[dbo].[Department]
[dbo].[Instructor_Course]
[dbo].[Work_In]
[dbo].[AddInstructor]
[dbo].[deleteInstructorByID]
[dbo].[getInstructor]
[dbo].[getInstructorCourses]
[dbo].[instructorCourses]
[dbo].[updateInstructoByID]

[dbo].[Instructor_Course]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	10
Created	5:07:50 PM Monday, February 20, 2023
Last Modified	10:56:18 AM Sunday, February 26, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK FK	Ins_id	int	4	NOT NULL
PK FK	Crs_id	int	4	NOT NULL
	Evaluation	varchar(150)	150	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK	PK_Instructor_Course	Ins_id, Crs_id	True

Foreign Keys

Name	Update	Delete	Columns
FK_Instructor_Course_Courses			Crs_id->[dbo].[Courses].[Crs_id]
FK_Instructor_Course_Instructor	Cascade	Cascade	Ins_id->[dbo].[Instructor].[Ins_id]

SQL Script

```
CREATE TABLE [dbo].[Instructor_Course]
(
[Ins_id] [int] NOT NULL,
[Crs_id] [int] NOT NULL,
[Evaluation] [varchar] (150) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Instructor_Course] ADD CONSTRAINT [PK_Instructor_Course] PRIMARY KEY
```

```
CLUSTERED ([Ins_id], [Crs_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Instructor_Course] ADD CONSTRAINT [FK_Instructor_Course_Courses]
FOREIGN KEY ([Crs_id]) REFERENCES [dbo].[Courses] ([Crs_id])
GO
ALTER TABLE [dbo].[Instructor_Course] ADD CONSTRAINT [FK_Instructor_Course_Instructor]
FOREIGN KEY ([Ins_id]) REFERENCES [dbo].[Instructor] ([Ins_id]) ON DELETE CASCADE ON
UPDATE CASCADE
GO
```

Uses

[dbo].[Courses]
[dbo].[Instructor]

Used By

[dbo].[addInstructorCourse]
[dbo].[deleteInstructorCourses]
[dbo].[getInstructorCourses]
[dbo].[instructorCourses]
[dbo].[updateInstructorCourse]

[dbo].[Question_Choice]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	186
Created	5:07:50 PM Monday, February 20, 2023
Last Modified	12:53:39 AM Monday, February 27, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK C FK	Qst_id	int	4	NOT NULL
PK C	Choice	varchar(150)	150	NOT NULL
	choiceID	varchar(2)	2	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Question_Choice_1	Qst_id, Choice	True

Foreign Keys

Name	Columns
FK_Question_Choice_Questions	Qst_id->[dbo].[Questions].[Qst_id]

SQL Script

```
CREATE TABLE [dbo].[Question_Choice]
(
[Qst_id] [int] NOT NULL,
[Choice] [varchar] (150) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[choiceID] [varchar] (2) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Question_Choice] ADD CONSTRAINT [PK_Question_Choice_1] PRIMARY KEY
CLUSTERED ([Qst_id], [Choice]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Question_Choice] ADD CONSTRAINT [FK_Question_Choice_Questions]
```

```
FOREIGN KEY ([Qst_id]) REFERENCES [dbo].[Questions] ([Qst_id])
GO
```

Uses

[dbo].[Questions]

Used By

[dbo].[deleteQuestion]
[dbo].[getQuestion]
[dbo].[insertQuestion]
[dbo].[PrintExam]
[dbo].[updateQuestionByID]

[dbo].[Questions]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	62
Created	12:53:39 AM Monday, February 27, 2023
Last Modified	12:53:39 AM Monday, February 27, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK C	Qst_id	int	4	NOT NULL
FK	Crs_id	int	4	NULL allowed
	Question	varchar(200)	200	NULL allowed
	Type	varchar(50)	50	NULL allowed
	Model_Answer	char(1)	1	NULL allowed
	Qst_Grade	int	4	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Questions	Qst_id	True

Foreign Keys

Name	Columns
FK_Questions_Courses	Crs_id->[dbo].[Courses].[Crs_id]

SQL Script

```
CREATE TABLE [dbo].[Questions]
(
[Qst_id] [int] NOT NULL,
[Crs_id] [int] NULL,
[Question] [varchar] (200) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Type] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Model_Answer] [char] (1) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Qst_Grade] [int] NULL
```

```
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Questions] ADD CONSTRAINT [PK_Questions] PRIMARY KEY CLUSTERED
([Qst_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Questions] ADD CONSTRAINT [FK_Questions_Courses] FOREIGN KEY
([Crs_id]) REFERENCES [dbo].[Courses] ([Crs_id])
GO
```

Uses

[dbo].[Courses]

Used By

[dbo].[Exam_Questions]
[dbo].[Question_Choice]
[dbo].[Student_Exam_Question]
[dbo].[CreateExam]
[dbo].[deleteQuestion]
[dbo].[Exam_Correction]
[dbo].[Exam_Student_Answers]
[dbo].[getQuestion]
[dbo].[insertQuestion]
[dbo].[PrintExam]
[dbo].[updateQuestionByID]

[dbo].[Student]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	4
Created	10:44:12 PM Monday, February 20, 2023
Last Modified	12:53:39 AM Monday, February 27, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK C	Stu_id	int	4	NOT NULL
	Fname	varchar(50)	50	NULL allowed
	Lname	varchar(50)	50	NULL allowed
	Phone	int	4	NULL allowed
	Email	varchar(50)	50	NULL allowed
	Address	varchar(50)	50	NULL allowed
FK	Leader_id	int	4	NULL allowed
FK	Did	int	4	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Student	Stu_id	True

Foreign Keys

Name	Columns
FK_Student_Department	Did->[dbo].[Department].[Did]
FK_Student_Leader	Leader_id->[dbo].[Student].[Stu_id]

SQL Script

```
CREATE TABLE [dbo].[Student]
(
[Stu_id] [int] NOT NULL,
[Fname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
```

```
[Lname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Phone] [int] NULL,
[Email] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Address] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Leader_id] [int] NULL,
[Did] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student] ADD CONSTRAINT [PK_Student] PRIMARY KEY CLUSTERED
([Stu_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student] ADD CONSTRAINT [FK_Student_Department] FOREIGN KEY ([Did])
REFERENCES [dbo].[Department] ([Did])
GO
ALTER TABLE [dbo].[Student] ADD CONSTRAINT [FK_Student_Leader] FOREIGN KEY
([Leader_id]) REFERENCES [dbo].[Student] ([Stu_id])
GO
```

Uses

[dbo].[Department]

Used By

[dbo].[Student_Course]
[dbo].[Student_Exam_Question]
[dbo].[Add_Student]
[dbo].[Delete_All_Students]
[dbo].[Delete_Student_By_Id]
[dbo].[departmentStudents]
[dbo].[Get_All_Students]
[dbo].[Get_Student_By_Email]
[dbo].[Get_Student_By_Id]
[dbo].[StudentExamAnswers]
[dbo].[studentGrades]
[dbo].[Update_Student_By_Id]

[dbo].[Student_Course]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	13
Created	5:07:50 PM Monday, February 20, 2023
Last Modified	8:09:33 PM Monday, February 27, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK C FK	Stu_id	int	4	NOT NULL
PK C FK	Crs_id	int	4	NOT NULL
	grade	int	4	NULL allowed
	percentageGrade	varchar(10)	10	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Student_Course	Stu_id, Crs_id	True

Foreign Keys

Name	Columns
FK_Student_Course_Courses	Crs_id->[dbo].[Courses].[Crs_id]
FK_Student_Course_Student	Stu_id->[dbo].[Student].[Stu_id]

SQL Script

```
CREATE TABLE [dbo].[Student_Course]
(
[Stu_id] [int] NOT NULL,
[Crs_id] [int] NOT NULL,
[grade] [int] NULL,
[percentageGrade] [varchar] (10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON [PRIMARY]
```

```
GO
ALTER TABLE [dbo].[Student_Course] ADD CONSTRAINT [PK_Student_Course] PRIMARY KEY
CLUSTERED ([Stu_id], [Crs_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student_Course] ADD CONSTRAINT [FK_Student_Course_Courses] FOREIGN
KEY ([Crs_id]) REFERENCES [dbo].[Courses] ([Crs_id])
GO
ALTER TABLE [dbo].[Student_Course] ADD CONSTRAINT [FK_Student_Course_Student] FOREIGN
KEY ([Stu_id]) REFERENCES [dbo].[Student] ([Stu_id])
GO
```

Uses

[dbo].[Courses]
[dbo].[Student]

Used By

[dbo].[Add_StudentGarde]
[dbo].[delete_garde]
[dbo].[delete_student_course]
[dbo].[Exam_Correction]
[dbo].[Get_Student_Course]
[dbo].[Get_Student_Course_by_CourseId]
[dbo].[Get_Student_Course_by_StudentId]
[dbo].[instructorCourses]
[dbo].[studentGrades]
[dbo].[update_student_grade]

[dbo].[Student_Exam_Question]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	100
Created	12:53:39 AM Monday, February 27, 2023
Last Modified	12:53:39 AM Monday, February 27, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK FK	Stu_id	int	4	NOT NULL
PK FK	Exm_id	int	4	NOT NULL
PK FK	Qst_id	int	4	NOT NULL
	Student_Answer	char(1)	1	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK	PK_Student_Exam_Question	Stu_id, Exm_id, Qst_id	True

Foreign Keys

Name	Columns
FK_Student_Exam_Question_Exams	Exm_id->[dbo].[Exams].[Exm_id]
FK_Student_Exam_Question_Questions	Qst_id->[dbo].[Questions].[Qst_id]
FK_Student_Exam_Question_Student	Stu_id->[dbo].[Student].[Stu_id]

SQL Script

```
CREATE TABLE [dbo].[Student_Exam_Question]
(
[Stu_id] [int] NOT NULL,
[Exm_id] [int] NOT NULL,
```

```
[Qst_id] [int] NOT NULL,
[Student_Answer] [char] (1) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student_Exam_Question] ADD CONSTRAINT [PK_Student_Exam_Question]
PRIMARY KEY CLUSTERED ([Stu_id], [Exm_id], [Qst_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student_Exam_Question] ADD CONSTRAINT [FK_Student_Exam_Question_-
Exams] FOREIGN KEY ([Exm_id]) REFERENCES [dbo].[Exams] ([Exm_id])
GO
ALTER TABLE [dbo].[Student_Exam_Question] ADD CONSTRAINT [FK_Student_Exam_Question_-
Questions] FOREIGN KEY ([Qst_id]) REFERENCES [dbo].[Questions] ([Qst_id])
GO
ALTER TABLE [dbo].[Student_Exam_Question] ADD CONSTRAINT [FK_Student_Exam_Question_-
Student] FOREIGN KEY ([Stu_id]) REFERENCES [dbo].[Student] ([Stu_id])
GO
```

Uses

[dbo].[Exams]
[dbo].[Questions]
[dbo].[Student]

Used By

[dbo].[deleteExamByCrsId]
[dbo].[DeleteExamByID]
[dbo].[Exam_Correction]
[dbo].[Exam_Student_Answers]
[dbo].[StudentExamAnswers]

[dbo].[Topic]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	16
Created	5:07:50 PM Monday, February 20, 2023
Last Modified	5:07:52 PM Monday, February 20, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK C	Top_id	int	4	NOT NULL
	Top_Name	varchar(50)	50	NULL allowed
FK	Crs_id	int	4	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Topic	Top_id	True

Foreign Keys

Name	Columns
FK_Topic_Courses1	Crs_id->[dbo].[Courses].[Crs_id]

SQL Script

```
CREATE TABLE [dbo].[Topic]
(
[Top_id] [int] NOT NULL,
[Top_Name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Crs_id] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Topic] ADD CONSTRAINT [PK_Topic] PRIMARY KEY CLUSTERED ([Top_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Topic] ADD CONSTRAINT [FK_Topic_Courses1] FOREIGN KEY ([Crs_id]) REFERENCES [dbo].[Courses] ([Crs_id])
```

```
GO
```

Uses

[dbo].[Courses]

Used By

[dbo].[addTopic]
[dbo].[courseTopics]
[dbo].[deleteTopicByCrsId]
[dbo].[deleteTopicById]
[dbo].[GetTopicData]
[dbo].[updateTopic]

[dbo].[Work_In]

Properties

Property	Value
Row Count (~)	9
Created	5:07:50 PM Monday, February 20, 2023
Last Modified	10:56:18 AM Sunday, February 26, 2023

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK C FK	Ins_id	int	4	NOT NULL
PK C FK	Did	int	4	NOT NULL
	Working_Hours	int	4	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Work_In	Ins_id, Did	True

Foreign Keys

Name	Update	Delete	Columns
FK_Work_In_Department			Did->[dbo].[Department].[Did]
FK_Work_In_Instructor	Cascade	Cascade	Ins_id->[dbo].[Instructor].[Ins_id]

SQL Script

```
CREATE TABLE [dbo].[Work_In]
(
[Ins_id] [int] NOT NULL,
[Did] [int] NOT NULL,
[Working_Hours] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Work_In] ADD CONSTRAINT [PK_Work_In] PRIMARY KEY CLUSTERED
([Ins_id], [Did]) ON [PRIMARY]
GO
```

```
ALTER TABLE [dbo].[Work_In] ADD CONSTRAINT [FK_Work_In_Department] FOREIGN KEY ([Did])
REFERENCES [dbo].[Department] ([Did])
GO
ALTER TABLE [dbo].[Work_In] ADD CONSTRAINT [FK_Work_In_Instructor] FOREIGN KEY
([Ins_id]) REFERENCES [dbo].[Instructor] ([Ins_id]) ON DELETE CASCADE ON UPDATE CASCADE
GO
```

Uses

[dbo].[Department]
[dbo].[Instructor]

Used By

[dbo].[AddWorkIn]
[dbo].[deleteWorkInByDepId]
[dbo].[deleteWorkInByDepIdAndInsId]
[dbo].[deleteWorkInByInsId]
[dbo].[getWorkIn]
[dbo].[updateWorkIn]

Stored Procedures

Objects

Name
dbo.Add_Student
dbo.Add_StudentGarde
dbo.addCourse
dbo.addDepartment
dbo.AddInstructor
dbo.addInstructorCourse
dbo.addTopic
dbo.AddWorkIn
dbo.courseTopics
dbo.CreateExam
dbo.Delete_All_Students
dbo.delete_garde
dbo.Delete_Student_By_Id
dbo.delete_student_course
dbo.deleteCourse
dbo.deleteDepartment
dbo.deleteExamByCrsId
dbo.DeleteExamByID
dbo.deleteInstructorByID
dbo.deleteInstructorCourses
dbo.deleteQuestion
dbo.deleteTopicByCrsId
dbo.deleteTopicById
dbo.deleteWorkInByDepId
dbo.deleteWorkInByDepIdAndInsId
dbo.deleteWorkInByInsId
dbo.departmentStudents
dbo.Exam_Correction
dbo.Exam_Student_Answers
dbo.Get_All_Exams
dbo.Get_All_Students
dbo.Get_Student_By_Email
dbo.Get_Student_By_Id
dbo.Get_Student_Course

dbo.Get_Student_Course_by_CourseId
dbo.Get_Student_Course_by_StudentId
dbo.getExam
dbo.getInstructor
dbo.getInstructorCourses
dbo.getQuestion
dbo.GetTopicData
dbo.getWorkIn
dbo.insertQuestion
dbo.instructorCourses
dbo.PrintExam
dbo.selectCourse
dbo.selectDepartment
dbo.StudentExamAnswers
dbo.studentGrades
dbo.Update_Student_By_Id
dbo.update_student_grade
dbo.updateCourse
dbo.updateDepartment
dbo.updateExam
dbo.updateInstructoByID
dbo.updateInstructorCourse
dbo.updateQuestionByID
dbo.updateTopic
dbo.updateWorkIn

[dbo].[Add_Student]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@fname	varchar(20)	20
@lName	varchar(20)	20
@phone	int	4
@email	varchar(20)	20
@address	varchar(20)	20
@leaderId	int	4
@departmentId	int	4

Permissions

Type	Action	Owning Principal
Deny	EXECUTE	Instructor_Mohammed
Deny	ALTER	Instructor_Mohammed
Deny	CONTROL	Instructor_Mohammed
Deny	TAKE OWNERSHIP	Instructor_Mohammed
Deny	VIEW DEFINITION	Instructor_Mohammed

SQL Script

```
CREATE PROC [dbo].[Add_Student] @id INT, @fname VARCHAR(20), @lName VARCHAR(20),
    @phone INT, @email VARCHAR(20),@address VARCHAR(20), @leaderId INT, @departmentId
INT

WITH ENCRYPTION
AS
    BEGIN TRY
            INSERT INTO  Student
            VALUES (@id,@fname,@lName,@phone,@email,@address,@leaderId,@departmentId)
```

```
    END TRY
    BEGIN CATCH
            SELECT 'Error'
    END CATCH
GO
DENY ALTER ON  [dbo].[Add_Student] TO [Instructor_Mohammed]
GO
DENY CONTROL ON  [dbo].[Add_Student] TO [Instructor_Mohammed]
GO
DENY EXECUTE ON  [dbo].[Add_Student] TO [Instructor_Mohammed]
GO
DENY TAKE OWNERSHIP ON  [dbo].[Add_Student] TO [Instructor_Mohammed]
GO
DENY VIEW DEFINITION ON  [dbo].[Add_Student] TO [Instructor_Mohammed]
GO
```

Uses

[dbo].[Student]

[dbo].[Add_StudentGarde]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@StuId	int	4
@CrsId	int	4
@grade	int	4

Permissions

Type	Action	Owning Principal
Deny	EXECUTE	Student_A
Deny	ALTER	Student_A
Deny	CONTROL	Student_A
Deny	TAKE OWNERSHIP	Student_A
Deny	VIEW DEFINITION	Student_A

SQL Script

```
CREATE PROC [dbo].[Add_StudentGarde] @StuId INT, @CrsId INT, @grade INT

WITH ENCRYPTION
AS
    BEGIN TRY
            INSERT INTO  Student_Course
            VALUES (@StuId, @CrsId, @grade)
    END TRY
    BEGIN CATCH
            SELECT 'Error'
    END CATCH
GO
DENY ALTER ON  [dbo].[Add_StudentGarde] TO [Student_A]
GO
DENY CONTROL ON  [dbo].[Add_StudentGarde] TO [Student_A]
```

```
GO
DENY EXECUTE ON  [dbo].[Add_StudentGarde] TO [Student_A]
GO
DENY TAKE OWNERSHIP ON  [dbo].[Add_StudentGarde] TO [Student_A]
GO
DENY VIEW DEFINITION ON  [dbo].[Add_StudentGarde] TO [Student_A]
GO
```

Uses

[dbo].[Student_Course]

[dbo].[addCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@cr_id	int	4
@name	varchar(20)	20
@duration	int	4
@description	varchar(70)	70

SQL Script

```
CREATE PROC [dbo].[addCourse] @cr_id INT, @name VARCHAR(20), @duration INT,
@description VARCHAR(70)=NULL
AS
BEGIN TRY
INSERT INTO courses VALUES(@cr_id,@name,@duration,@description)
END TRY
BEGIN CATCH
SELECT ERROR_MESSAGE()
END CATCH
GO
```

Uses

[dbo].[Courses]

[dbo].[addDepartment]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@did	int	4
@name	varchar(20)	20
@location	nvarchar(200)	400
@description	varchar(50)	50
@mgr_id	int	4

SQL Script

```
CREATE proc [dbo].[addDepartment] @did INT , @name VARCHAR(20),@location NVARCHAR(200),
@description VARCHAR(50)=NULL,@mgr_id INT=NULL
AS
BEGIN try
INSERT INTO department VALUES (@did,@name,@description,@location,@mgr_id)
END TRY
BEGIN CATCH
SELECT ERROR_MESSAGE()
END catch
GO
```

Uses

[dbo].[Department]

[dbo].[AddInstructor]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@fname	nvarchar(70)	140
@lname	nvarchar(70)	140
@degree	nvarchar(70)	140
@sal	int	4
@address	nvarchar(70)	140
@phone	int	4
@supervisor	int	4

SQL Script

```
CREATE proc [dbo].[AddInstructor]  @id int, @fname nvarchar(70),@lname nvarchar(70)
,@degree NVARCHAR(70), @sal int , @address nvarchar(70)=null,@phone
int=null,@supervisor int
as
begin try
insert into Instructor
values(@id,@fname,@lname,@degree,@sal,@address,@phone,@supervisor)
end try
begin catch
select 'Error'
end CATCH
GO
```

Uses

[dbo].[Instructor]

[dbo].[addInstructorCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@ins_id	int	4
@crs_id	int	4
@eval	nvarchar(70)	140

SQL Script

```
create proc [dbo].[addInstructorCourse] @ins_id int ,@crs_id INT ,@eval NVARCHAR(70)
as
begin try
insert into Instructor_Course values(@ins_id, @crs_id,@eval)
end try
begin catch
select 'Error'
end catch
GO
```

Uses

[dbo].[Instructor_Course]

[dbo].[addTopic]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@name	varchar(50)	50
@crs	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[addTopic] @id INT , @name VARCHAR(50), @crs INT
AS
    IF NOT EXISTS (SELECT t.Top_id FROM Topic t WHERE @id = t.Top_id)
        INSERT INTO Topic (Top_id,Top_Name,Crs_id)
        VALUES     (@id,@name,@crs)
    ELSE
        SELECT'Duplicted ID'
GO
```

Uses

[dbo].[Topic]

[dbo].[AddWorkIn]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@insId	int	4
@depId	int	4
@WorkingHours	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[AddWorkIn] @insId INT , @depId INT , @WorkingHours INT
WITH ENCRYPTION
AS
BEGIN TRY
    IF NOT EXISTS (SELECT * FROM Work_In wi WHERE wi.Ins_id=@insId AND wi.Did=@depId)
        INSERT INTO Work_In (Ins_id,Did,Working_Hours)
        VALUES    (@insId,@depId,@WorkingHours)
    ELSE
        SELECT'Duplicted ID'
END TRY
BEGIN CATCH
    SELECT 'Error'
END CATCH
GO
```

Uses

[dbo].[Work_In]

[dbo].[courseTopics]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@course_id	int	4

SQL Script

```
CREATE PROC [dbo].[courseTopics] @course_id INT
AS
IF EXISTS(SELECT crs_id FROM Courses  WHERE Crs_id=@course_id)
BEGIN
SELECT topic.top_name
FROM Topic , courses
WHERE topic.Crs_id=Courses.Crs_id AND Courses.Crs_id=@course_id
END
ELSE
SELECT 'There is no course with this id' AS 'result'
GO
```

Uses

[dbo].[Courses]
[dbo].[Topic]

[dbo].[CreateExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@crs_name	varchar(50)	50
@N_mcq	int	4
@N_tf	int	4
@duration	int	4
@date	date	3

Permissions

Type	Action	Owning Principal
Grant	EXECUTE	Student_A
Grant	ALTER	Student_A
Grant	CONTROL	Student_A
Grant	TAKE OWNERSHIP	Student_A
Grant	VIEW DEFINITION	Student_A

SQL Script

```
CREATE PROC [dbo].[CreateExam] @crs_name VARCHAR(50) , @N_mcq INT=0 ,@N_tf INT=0 ,
@duration INT = 60 , @date DATE =NULL
WITH ENCRYPTION
AS
BEGIN TRY
--Check if the Course Exists and The Number OF Questions =10
    IF NOT EXISTS (SELECT * FROM Courses C WHERE Name =@crs_name)
    BEGIN
        SELECT 'This Course Does Not Exists'
    END
    ELSE IF (@N_mcq +@N_tf !=10 OR @N_mcq< 0 OR @N_tf<0 )
    BEGIN
```

```
        SELECT 'Enter Valid Number Of Question'
    END
    -- Else (Our Code)
    ELSE
    BEGIN
        ---1- Get Course ID
        DECLARE @crs_id INT
        SELECT @crs_id= Crs_id FROM Courses WHERE Name=@crs_name
        -- insert in exam ( exm id - duration- crs id -date )
        if @date is null set @date =getDate()
        INSERT INTO Exams
        VALUES (@duration,@crs_id,@date)


    -- get the latest add identity value
        DECLARE @new_exam_id int = @@IDENTITY




        INSERT INTO Exam_Questions
        SELECT TOP (@N_mcq) @new_exam_id,Qst_id FROM Questions WHERE Crs_id = @crs_id
and type ='mcq' ORDER BY NEWID()

        Insert Into Exam_Questions
        SELECT TOP (@N_tf) @new_exam_id, Qst_id FROM Questions WHERE Crs_id = @crs_id
and type = 't/f' ORDER BY NEWID()

        SELECT 'Exam Created'
    END
END TRY
BEGIN CATCH
SELECT 'Error'
END CATCH
GO
GRANT ALTER ON  [dbo].[CreateExam] TO [Student_A]
GO
GRANT CONTROL ON  [dbo].[CreateExam] TO [Student_A]
GO
GRANT EXECUTE ON  [dbo].[CreateExam] TO [Student_A]
GO
GRANT TAKE OWNERSHIP ON  [dbo].[CreateExam] TO [Student_A]
GO
GRANT VIEW DEFINITION ON  [dbo].[CreateExam] TO [Student_A]
GO
```

Uses

[dbo].[Courses]
[dbo].[Exam_Questions]
[dbo].[Exams]
[dbo].[Questions]

[dbo].[Delete_All_Students]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

SQL Script

```
CREATE PROC [dbo].[Delete_All_Students]
WITH ENCRYPTION
AS
        BEGIN TRY
                DELETE FROM Student
        END TRY

        BEGIN CATCH
                SELECT 'error'
        END CATCH
GO
```

Uses

[dbo].[Student]

[dbo].[delete_garde]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@StuId	int	4
@crsId	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[delete_garde] @StuId INT , @crsId INT
WITH ENCRYPTION
AS
    BEGIN TRY
            DELETE grade FROM Student_Course sc
            WHERE sc.Stu_id = @StuId AND sc.Crs_id = @crsId
    END TRY

    BEGIN CATCH
            SELECT 'Error'
    END CATCH
GO
```

Uses

[dbo].[Student_Course]

[dbo].[Delete_Student_By_Id]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4

SQL Script

```
CREATE PROC [dbo].[Delete_Student_By_Id] @id INT
WITH ENCRYPTION
AS

    BEGIN TRY
            if exists(SELECT s.Stu_id from Student s  where s.Stu_id = @id)

            UPDATE Student  SET Leader_id = NULL WHERE Leader_id = @id

            delete from Student where Stu_id = @id
    END TRY

    BEGIN CATCH
            SELECT 'Error'
    END CATCH
GO
```

Uses

[dbo].[Student]

[dbo].[delete_student_course]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4

SQL Script

```
CREATE PROC [dbo].[delete_student_course] @id INT
WITH ENCRYPTION
AS
    BEGIN TRY
            DELETE FROM Student_Course
            WHERE Stu_id = @id
    END TRY

    BEGIN CATCH
            SELECT 'Error'
    END CATCH
GO
```

Uses

[dbo].[Student_Course]

[dbo].[deleteCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@cr_id	int	4

SQL Script

```
CREATE PROC [dbo].[deleteCourse] @cr_id INT
AS
BEGIN TRY
DELETE courses WHERE Crs_id=@cr_id
END TRY
BEGIN CATCH
SELECT ERROR_MESSAGE()
END CATCH
GO
```

Uses

[dbo].[Courses]

[dbo].[deleteDepartment]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@did	int	4

SQL Script

```
CREATE PROC [dbo].[deleteDepartment] @did INT
AS
BEGIN TRY
DELETE FROM department WHERE did=@did
END TRY
BEGIN CATCH
SELECT ERROR_MESSAGE()
END CATCH
GO
```

Uses

[dbo].[Department]

[dbo].[deleteExamByCrsId]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@crsId	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[deleteExamByCrsId] @crsId INT
WITH ENCRYPTION
AS

BEGIN
    IF EXISTS(SELECT e.Crs_id FROM Exams e WHERE e.Crs_id=@crsId)
    BEGIN
        BEGIN TRY
            BEGIN TRANSACTION;

                DELETE FROM Student_Exam_Question WHERE Exm_id IN (SELECT Exm_id FROM
Exams WHERE Crs_id = @crsId)

                DELETE FROM Exam_Questions WHERE Exm_id IN (SELECT Exm_id FROM Exams
WHERE Crs_id = @crsId)

                DELETE FROM Exams WHERE Crs_id=@crsId

            COMMIT TRANSACTION;
        END TRY
        BEGIN CATCH
            IF @@TRANCOUNT > 0 ROLLBACK TRANSACTION;
                SELECT'Error happend while deleting'
        END CATCH
    END
    ELSE
    BEGIN
        SELECT 'Exam does not exist'
    END
END
```

```
GO
```

Uses

[dbo].[Exam_Questions]

[dbo].[Exams]

[dbo].[Student_Exam_Question]

[dbo].[DeleteExamByID]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@ExamID	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[DeleteExamByID]  @ExamID INT
AS
BEGIN

    IF EXISTS (SELECT Exm_id FROM Exams WHERE Exm_id = @ExamID)
    BEGIN
        BEGIN TRY
            BEGIN TRANSACTION;

            DELETE FROM Student_Exam_Question WHERE Exm_id = @ExamID;

            DELETE FROM Exam_Questions WHERE Exm_id = @ExamID;

            DELETE FROM Exams WHERE Exm_id = @ExamID;

            COMMIT TRANSACTION;
        END TRY
        BEGIN CATCH
            IF @@TRANCOUNT > 0 ROLLBACK TRANSACTION;
            SELECT'Error happend while deleting';
        END CATCH
    END
    ELSE
    BEGIN
        SELECT 'Exam does not exist'
    END
END
GO
```

Uses

[dbo].[Exam_Questions]

[dbo].[Exams]

[dbo].[Student_Exam_Question]

[dbo].[deleteInstructorByID]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4

SQL Script

```
CREATE proc [dbo].[deleteInstructorByID]  @id int
AS
BEGIN TRY
if exists(select Ins_id from Instructor i where i.Ins_id = @id)
BEGIN

UPDATE Instructor  SET Super_id=NULL WHERE Super_id=@id

delete from Instructor where Ins_id = @id

end
else select 'Not Found'
END TRY
BEGIN CATCH
 select 'Error'
END CATCH
GO
```

Uses

[dbo].[Instructor]

[dbo].[deleteInstructorCourses]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@ins_id	int	4
@crs_id	int	4

SQL Script

```
CREATE proc [dbo].[deleteInstructorCourses] @ins_id int = -1 ,@crs_id int =-1
as
BEGIN TRY
if(@ins_id<>-1 and @crs_id =-1)
BEGIN
        IF EXISTS (SELECT Ins_id FROM Instructor_Course ic WHERE Ins_id=@ins_id)
        BEGIN
        delete from Instructor_course where Ins_id=@ins_id
        END
        ELSE SELECT 'Not Found'
END
else if (@ins_id =-1 and @crs_id <>-1)
BEGIN
        IF EXISTS (SELECT Crs_id FROM Instructor_Course ic WHERE Crs_id=@crs_id)
        begin
        delete from Instructor_course where Crs_id=@crs_id
        END
        ELSE SELECT 'Not Found'
END
else if(@ins_id<>-1 and @crs_id<>-1)
BEGIN
        IF EXISTS (SELECT Crs_id FROM Instructor_Course ic WHERE Crs_id=@crs_id) AND
EXISTS (SELECT Ins_id FROM Instructor_Course ic WHERE Ins_id=@ins_id)
        begin
        delete from Instructor_course where Ins_id=@ins_id and Crs_id=@crs_id
        END
        ELSE SELECT 'Not Found'
END
END TRY
```

```
begin catch
select 'Error'
end CATCH
GO
```

Uses

[dbo].[Instructor_Course]

[dbo].[deleteQuestion]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4

SQL Script

```
CREATE proc [dbo].[deleteQuestion] @id INT
as
BEGIN TRY
if exists(select qst_id from  Questions where qst_id = @id)
    BEGIN
        UPDATE Questions  SET Crs_id=NULL WHERE qst_id=@id

        delete from Question_Choice where qst_id = @id
        delete from Questions where qst_id = @id
    end
else
    select 'Not Found'
END TRY
BEGIN CATCH
    select 'Error'
END CATCH
GO
```

Uses

[dbo].[Question_Choice]
[dbo].[Questions]

[dbo].[deleteTopicByCrsId]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@crsId	int	4

SQL Script

```
CREATE PROC [dbo].[deleteTopicByCrsId] @crsId INT
WITH ENCRYPTION
AS
BEGIN TRY
    IF EXISTS(SELECT crs_id FROM Topic t WHERE @crsId=T.Crs_id)
        BEGIN
            DELETE FROM Topic WHERE @crsId=Crs_id
        END
    ELSE
        SELECT 'Topic does not exist'
END TRY
BEGIN CATCH
    SELECT 'error'
END CATCH
GO
```

Uses

[dbo].[Topic]

[dbo].[deleteTopicById]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@topId	int	4

SQL Script

```
CREATE PROC [dbo].[deleteTopicById] @topId INT
WITH ENCRYPTION
AS
BEGIN TRY
    IF EXISTS(SELECT t.Top_id FROM Topic t WHERE @topId=T.Top_id)
        BEGIN
            DELETE FROM Topic WHERE @topId=Top_id
        END
    ELSE
        SELECT 'Topic does not exist'
END TRY
BEGIN CATCH
    SELECT 'error'
END CATCH
GO
```

Uses

[dbo].[Topic]

[dbo].[deleteWorkInByDepId]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@depId	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[deleteWorkInByDepId] @depId INT
WITH ENCRYPTION
AS
BEGIN TRY
    IF EXISTS(SELECT wi.Ins_id FROM Work_In wi WHERE @depId=wi.Did)
        BEGIN
            DELETE FROM Work_In WHERE @depId=Did
        END
    ELSE
        SELECT 'Wrong Id'
END TRY
BEGIN CATCH
    SELECT'Error'
END CATCH
GO
```

Uses

[dbo].[Work_In]

[dbo].[deleteWorkInByDepIdAndInsId]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@depId	int	4
@insId	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[deleteWorkInByDepIdAndInsId] @depId INT , @insId INT
WITH ENCRYPTION
AS
BEGIN TRY
    IF EXISTS(SELECT wi.Ins_id FROM Work_In wi WHERE @depId=wi.Did AND @insId=Ins_id)
        BEGIN
            DELETE FROM Work_In WHERE @depId=Did AND @insId=Ins_id
        END
    ELSE
        SELECT 'Wrong Id'
END TRY
BEGIN CATCH
    SELECT'Error'
END CATCH
GO
```

Uses

[dbo].[Work_In]

[dbo].[deleteWorkInByInsId]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@insId	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[deleteWorkInByInsId] @insId INT
WITH ENCRYPTION
AS
BEGIN TRY
    IF EXISTS(SELECT wi.Ins_id FROM Work_In wi WHERE @insId=wi.Ins_id)
        BEGIN
            DELETE FROM Work_In WHERE @insId=Ins_id
        END
    ELSE
        SELECT 'Wrong Id'
END TRY
BEGIN CATCH
    SELECT'Error'
END CATCH
GO
```

Uses

[dbo].[Work_In]

[dbo].[departmentStudents]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@dept_id	int	4

SQL Script

```
CREATE PROC [dbo].[departmentStudents] @dept_id INT
AS
IF EXISTS(SELECT did FROM department WHERE did = @dept_id)
BEGIN
IF EXISTS(SELECT s.did,D.did FROM student s, department D WHERE s.did=D.did AND
D.did=@dept_id)
BEGIN
SELECT s.* FROM student s, department d
WHERE s.did=D.did AND D.did=@dept_id
END--end of secont if
ELSE
SELECT 'no students in this department'
END --end of first if
ELSE
SELECT 'department does not exist' AS 'result'
GO
```

Uses

[dbo].[Department]
[dbo].[Student]

[dbo].[Exam_Correction]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@ExamId	int	4
@StudentId	int	4

Permissions

Type	Action	Owning Principal
Deny	EXECUTE	Student_A
Deny	ALTER	Student_A
Deny	CONTROL	Student_A
Deny	TAKE OWNERSHIP	Student_A
Deny	VIEW DEFINITION	Student_A

SQL Script

```
CREATE PROCEDURE [dbo].[Exam_Correction]
    @ExamId INT,
    @StudentId INT

    WITH ENCRYPTION
AS
BEGIN
    BEGIN TRY
        IF NOT EXISTS(SELECT* FROM Student_Exam_Question seq WHERE seq.Stu_id=@Student-
Id AND seq.Exm_id=@ExamId)
            SELECT 'Wrong student or exam'
        ELSE
            BEGIN
                    DECLARE @TotalScore INT = 0;
                    DECLARE @QuestionId INT;
                    DECLARE @CorrectAnswer CHAR(1);
```

```
                    DECLARE @StudentAnswer CHAR(1);
                    DECLARE @gradeAnswer INT ;
                    DECLARE @totalGrade INT = 0;
                    DECLARE @CrsId INT;

                    SELECT @CrsId=e.Crs_id
                    FROM Exams e
                    WHERE e.Exm_id=@ExamId


                    DECLARE question_cursor CURSOR FOR
                    SELECT q.Qst_id, q.Model_Answer,q.Qst_Grade
                    FROM Questions q
                    INNER JOIN Student_Exam_Question seq ON q.Qst_id = seq.Qst_id
                    WHERE seq.Exm_id = @ExamId AND seq.Stu_id = @StudentId;


                    OPEN question_cursor;
                    FETCH NEXT FROM question_cursor INTO @QuestionId, @Correct-
Answer,@gradeAnswer;

                    WHILE @@FETCH_STATUS = 0
                        BEGIN
                            SELECT @StudentAnswer = seq.Student_Answer
                            FROM Questions q , Student_Exam_Question seq
                            WHERE seq.Exm_id = @ExamId AND seq.Stu_id = @StudentId AND
seq.Qst_id = @QuestionId;

                            IF @StudentAnswer = @CorrectAnswer
                            SET @TotalScore = @TotalScore + @gradeAnswer;

                            SET @totalGrade = @totalGrade + @gradeAnswer


                            FETCH NEXT FROM question_cursor INTO @QuestionId, @Correct-
Answer,@gradeAnswer;
                        END

                    UPDATE  Student_Course
                    SET percentageGrade = CONCAT((@TotalScore/@totalGrade)*100,'%')   ,
grade =@TotalScore
                    WHERE Stu_id=@StudentId AND Crs_id = @CrsId

                    CLOSE question_cursor;
                    DEALLOCATE question_cursor;

    END



        END TRY

        BEGIN CATCH
            SELECT ERROR_MESSAGE();
```



```
        END CATCH
END
GO
DENY ALTER ON  [dbo].[Exam_Correction] TO [Student_A]
GO
DENY CONTROL ON  [dbo].[Exam_Correction] TO [Student_A]
GO
DENY EXECUTE ON  [dbo].[Exam_Correction] TO [Student_A]
GO
DENY TAKE OWNERSHIP ON  [dbo].[Exam_Correction] TO [Student_A]
GO
DENY VIEW DEFINITION ON  [dbo].[Exam_Correction] TO [Student_A]
GO
```

Uses

[dbo].[Exams]
[dbo].[Questions]
[dbo].[Student_Course]
[dbo].[Student_Exam_Question]

[dbo].[Exam_Student_Answers]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@exam_id	int	4
@st_id	int	4

SQL Script

```
CREATE PROC [dbo].[Exam_Student_Answers] @exam_id INT, @st_id INT
AS
SELECT seq.Exm_id  ,q.Question, q.Type, q.Model_Answer,seq.Student_Answer
FROM Student_Exam_Question seq , Questions q
WHERE seq.Qst_id = q.Qst_id AND seq.Exm_id = @exam_id and seq.Stu_id = @st_id
GO
```

Uses

[dbo].[Questions]
[dbo].[Student_Exam_Question]

[dbo].[Get_All_Exams]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@ExmId	int	4
@crsId	int	4
@date	date	3

SQL Script

```
CREATE PROC [dbo].[Get_All_Exams] @ExmId INT=-1, @crsId INT =-1,@date DATE ='1600-01-01'

WITH ENCRYPTION
AS
BEGIN TRY

    IF (@ExmId!=-1)
        BEGIN
            IF EXISTS(    SELECT * FROM Exams e WHERE e.Exm_id=@ExmId)
            BEGIN
                SELECT *
                FROM Exams e
                WHERE e.Exm_id=@ExmId
            END
            else
            SELECT 'Exam does not exist'
     END


    IF(@crsId!=-1)
    BEGIN
        IF EXISTS(    SELECT * FROM Exams e WHERE e.Crs_id=@crsId)
        BEGIN
            SELECT *
            FROM Exams e
            WHERE e.Crs_id=@crsId
```

```
        END
        else
        SELECT 'Exam does not exist'
    END

    IF(@date!='1600-01-01')
    BEGIN
        IF EXISTS(    SELECT * FROM Exams e WHERE e.Date=@date)
            BEGIN
                SELECT *
                FROM Exams e
                WHERE e.Date=@date
            END
        else
        SELECT 'Exam does not exist'
    END

END TRY
BEGIN CATCH
    SELECT 'error'
END CATCH
GO
```

Uses

[dbo].[Exams]

[dbo].[Get_All_Students]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

SQL Script

```
CREATE PROCEDURE [dbo].[Get_All_Students] WITH ENCRYPTION
AS
BEGIN
    SELECT *
    FROM dbo.Student

END
GO
```

Uses

[dbo].[Student]

[dbo].[Get_Student_By_Email]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@email	varchar(20)	20

SQL Script

```
CREATE PROCEDURE [dbo].[Get_Student_By_Email]  @email VARCHAR(20)
WITH ENCRYPTION
AS
BEGIN
    SELECT *
    FROM Student
    WHERE Email = @email
END
GO
```

Uses

[dbo].[Student]

[dbo].[Get_Student_By_Id]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[Get_Student_By_Id]  @id INT
WITH ENCRYPTION
AS
BEGIN
    SELECT *
    FROM Student
    WHERE Stu_id = @id
END
GO
```

Uses

[dbo].[Student]

[dbo].[Get_Student_Course]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

SQL Script

```
CREATE PROCEDURE [dbo].[Get_Student_Course] WITH ENCRYPTION
AS
BEGIN
    SELECT *
    FROM Student_Course

END
GO
```

Uses

[dbo].[Student_Course]

[dbo].[Get_Student_Course_by_CourseId]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4

SQL Script

```
CREATE PROC [dbo].[Get_Student_Course_by_CourseId] @id INT
WITH ENCRYPTION
AS
    BEGIN TRY
            SELECT * FROM Student_Course sc
            WHERE sc.Crs_id = @id
    END TRY

    BEGIN CATCH
            SELECT 'Error'
    END CATCH
GO
```

Uses

[dbo].[Student_Course]

[dbo].[Get_Student_Course_by_StudentId]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4

SQL Script

```
CREATE PROC [dbo].[Get_Student_Course_by_StudentId] @id INT
WITH ENCRYPTION
AS
    BEGIN TRY
            SELECT * FROM Student_Course sc
            WHERE sc.Stu_id = @id
    END TRY

    BEGIN CATCH
            SELECT 'Error'
    END CATCH
GO
```

Uses

[dbo].[Student_Course]

[dbo].[getExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4

SQL Script

```
CREATE PROC [dbo].[getExam] @id int    =-1
AS
if  @id != -1
BEGIN
    IF exists(select * from Exam_Questions eq where eq.Exm_id = @id)
        BEGIN
            DECLARE @getexam TABLE (qstID int,question varchar(200),answer
varchar(200),a varchar(200),b VARCHAR(200),c VARCHAR(200),d VARCHAR(200));
            DECLARE c1 CURSOR
            FOR SELECT eq.Qst_id FROM Exam_Questions eq WHERE eq.Exm_id = @id
            for read ONLY
            declare @qstID int
            open c1
            fetch c1 into @qstID
            while @@fetch_Status=0
            BEGIN
                INSERT INTO @getexam
                EXECUTE getquestion @qstID
                fetch c1 into @qstID
            end
            SELECT * FROM @getexam
            close c1
            deallocate c1
        END
    ELSE
    BEGIN
        SELECT 'not found'
    END
END
ELSE
    BEGIN
```

```
        SELECT 'enter an id'
    END
GO
```

Uses

[dbo].[Exam_Questions]

[dbo].[getQuestion]

[dbo].[getInstructor]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@fname	nvarchar(70)	140

SQL Script

```
CREATE proc [dbo].[getInstructor] @id int=-1 ,@fname nvarchar(70)=' '
as
if @id = -1 and @fname !=' '
begin
        if exists(select * from Instructor i where i.Fname = @fname)
        begin
        select i.Ins_id ID , i.Fname +' '+i.Lname [Full name], i.Academics_Degree
,i.Salary , i.Address, i.Phone
        FROM Instructor i
        WHERE i.Fname=@fname
        end
        else select 'Not Found'
end
else if  @id != -1 and @fname =' '
begin
            if exists(select * from Instructor i where i.Ins_id = @id)
        begin
        select i.Ins_id ID , i.Fname +' '+i.Lname [Full name], i.Academics_Degree
,i.Salary , i.Address, i.Phone
        FROM Instructor i
        WHERE i.Ins_id=@id
        end
        else select 'Not Found'
END

else /*For the First If */
begin
    select i.Ins_id ID , i.Fname +' '+i.Lname [Full name], i.Academics_Degree ,i.Salary
, i.Address, i.Phone
    FROM Instructor i
```

```
end
GO
```

Uses

[dbo].[Instructor]

[dbo].[getInstructorCourses]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@ins_id	int	4
@crs_id	int	4

SQL Script

```
CREATE proc [dbo].[getInstructorCourses] @ins_id int=-1 ,@crs_id int =-1
as
BEGIN TRY
if(@ins_id=-1 and @crs_id<>-1)
    BEGIN
        if exists(select ic.Crs_id from Instructor_Course ic  WHERE ic.Crs_id =
@crs_id)
        BEGIN
        SELECT i.Fname+' '+i.Lname AS [Full Name] , c.Name ,ic.Evaluation
        FROM Courses c ,Instructor i,Instructor_Course ic
        WHERE c.Crs_id=ic.Crs_id AND ic.Ins_id=i.Ins_id AND ic.Crs_id=@crs_id
        END
        ELSE SELECT 'Not Found'
    END
ELSE if(@ins_id<>-1 and @crs_id=-1)
    BEGIN
        if exists(select ic.Ins_id from Instructor_Course ic  WHERE ic.Ins_id  =
@ins_id)
        BEGIN
        SELECT i.Fname+' '+i.Lname AS [Full Name] , c.Name ,ic.Evaluation
        FROM Courses c ,Instructor i,Instructor_Course ic
        WHERE c.Crs_id=ic.Crs_id AND ic.Ins_id=i.Ins_id AND ic.Ins_id  = @ins_id
        END
        ELSE SELECT 'Not Found'
    END
ELSE IF (@ins_id<>-1 and @crs_id<>-1)
    BEGIN
        if exists (select ic.Crs_id from Instructor_Course ic  WHERE ic.Crs_id =
@crs_id) AND EXISTS (select ic.Ins_id from Instructor_Course ic  WHERE ic.Ins_id  =
@ins_id)
```

```
        BEGIN
        SELECT i.Fname+' '+i.Lname AS [Full Name] , c.Name ,ic.Evaluation
        FROM Courses c ,Instructor i,Instructor_Course ic
        WHERE c.Crs_id=ic.Crs_id AND ic.Ins_id=i.Ins_id AND ic.Ins_id  = @ins_id AND
ic.Crs_id=@crs_id
        END
        ELSE SELECT 'Not Found'
    END

ELSE
    BEGIN
        SELECT i.Fname+' '+i.Lname AS [Full Name] , c.Name ,ic.Evaluation
        FROM Courses c ,Instructor i,Instructor_Course ic
        WHERE c.Crs_id=ic.Crs_id AND ic.Ins_id=i.Ins_id
    END
END TRY
BEGIN CATCH
SELECT 'Error'
END CATCH
GO
```

Uses

[dbo].[Courses]
[dbo].[Instructor]
[dbo].[Instructor_Course]

[dbo].[getQuestion]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4

SQL Script

```
CREATE PROC [dbo].[getQuestion] @id int=-1
as
if  @id != -1
    BEGIN
    IF exists(select * from Questions where Qst_id = @id)
        DECLARE @type VARCHAR(20) =(SELECT type FROM questions WHERE qst_id = @id)
        IF @type = 'mcq'
        BEGIN
            SELECT Questions.Qst_id, Questions.Question, Questions.Model_Answer, Pivot-
Data.a, PivotData.b, PivotData.c, PivotData.d
            FROM
            (
            SELECT *
            FROM
            (
                SELECT qst_id,choiceId , choice
                FROM question_choice
            ) d
            pivot
            (
                MAX(choice)
                FOR choiceId in (a, b, c, d)
            ) piv
            WHERE piv.qst_id =@id
            ) as PivotData
            INNER JOIN Questions ON PivotData.qst_id = Questions.Qst_id
        END
        ELSE IF @type='t/f'
        BEGIN
            SELECT q.Qst_id, q.Question, q.Model_Answer,'true' AS a ,'false' AS b ,''
AS c , '' AS d
```

```
            FROM Questions q
            WHERE q.Qst_id=@id
        END
    ELSE
        SELECT 'Not Found'
    END
ELSE  /*For the First If */
    BEGIN
        SELECT 'enter an id'
    END
GO
```

Uses

[dbo].[Question_Choice]
[dbo].[Questions]

Used By

[dbo].[getExam]

[dbo].[GetTopicData]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@name	varchar(50)	50
@id	int	4
@crsId	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[GetTopicData] @name VARCHAR(50)='',@id INT=-1, @crsId INT =-1
WITH ENCRYPTION
AS
BEGIN TRY

    IF (@id!=-1)
    BEGIN
        IF EXISTS(    SELECT * FROM Topic t WHERE Top_id=@id)
        BEGIN
            SELECT *
            FROM Topic t
            WHERE t.Top_id=@id
        END
        else
        SELECT 'topic does not exist'
    END

    IF(@name!='')
    BEGIN
        IF EXISTS(    SELECT * FROM Topic t WHERE t.Top_Name=@name)
        BEGIN
            SELECT *
            FROM Topic t
            WHERE t.Top_Name=@name
        END
        else
        SELECT 'topic does not exist'
```

```
    END

    IF(@crsId!=-1)
    BEGIN
        IF EXISTS(    SELECT * FROM Topic t WHERE T.Crs_id=@crsId)
        BEGIN
            SELECT *
            FROM Topic t
            WHERE T.Crs_id=@crsId
        END
        else
        SELECT 'topic does not exist'
    END

END TRY
BEGIN CATCH
    SELECT 'error'
END catch
GO
```

Uses

[dbo].[Topic]

[dbo].[getWorkIn]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@insId	int	4
@depId	int	4
@WorkingHours	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[getWorkIn] @insId INT=-1 , @depId INT=-1 , @WorkingHours INT=-1
 WITH ENCRYPTION
 AS
 BEGIN TRY

    IF(@insId!=-1) AND (@depId!=-1)
    BEGIN
            IF EXISTS(    SELECT * FROM Work_In wi WHERE wi.Ins_id=@insId AND
wi.Did=@depId)
            BEGIN
                SELECT *
                FROM Work_In wi
                WHERE wi.Ins_id=@insId AND Did=@depId
            END
            else
                SELECT ' Does not exist'
        END

     ELSE IF(@insId!=-1)
        BEGIN
            IF EXISTS(    SELECT * FROM Work_In wi WHERE wi.Ins_id=@insId )
            BEGIN
                SELECT *
                FROM Work_In wi
                WHERE wi.Ins_id=@insId
            END
            else
```

```
                SELECT ' Does not exist'
        END

     ELSE IF(@depId!=-1)
        BEGIN
            IF EXISTS(    SELECT * FROM Work_In wi WHERE wi.Did=@depId)
            BEGIN
                SELECT *
                FROM Work_In wi
                WHERE wi.Did=@depId
            END
            else
                SELECT ' Does not exist'
        END

    ELSE  IF(@WorkingHours!=-1)
        BEGIN
            IF EXISTS(    SELECT * FROM Work_In wi WHERE wi.Working_Hours=@Working-
Hours)
            BEGIN
                SELECT *
                FROM Work_In wi
                WHERE  wi.Working_Hours=@WorkingHours
            END
            else
                SELECT ' Does not exist'
        END

 END TRY
 BEGIN CATCH
    SELECT 'Error'
 END CATCH
GO
```

Uses

[dbo].[Work_In]

[dbo].[insertQuestion]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@crsId	int	4
@qstContent	varchar(200)	200
@type	varchar(10)	10
@ModelAns	varchar(10)	10
@Qstgrd	int	4
@choiceA	varchar(200)	200
@choiceB	varchar(200)	200
@choiceC	varchar(200)	200
@choiceD	varchar(200)	200

SQL Script

```
CREATE proc [dbo].[insertQuestion]
(@crsId INT
,@qstContent VARCHAR(200)
,@type VARCHAR(10)
,@ModelAns VARCHAR(10)
,@Qstgrd INT
,@choiceA VARCHAR(200) = NULL
,@choiceB VARCHAR(200) = NULL
,@choiceC VARCHAR(200) = NULL
,@choiceD VARCHAR(200) = NULL
)
as
begin TRY
    DECLARE @qstId INT
    SET @qstId = (SELECT TOP 1 Qst_id FROM Questions ORDER BY Qst_id DESC)
    SET @qstId += 1

    INSERT INTO Questions
    VALUES (@qstId, @crsId, @qstContent, @type, @ModelAns, @Qstgrd)
```

```
    IF @type = 'mcq'
        BEGIN
            INSERT INTO Question_Choice
            VALUES(@qstId,@choiceA,'a'),(@qstId,@choiceB,'b'),(@qstId,@choice-
C,'c'),(@qstId,@choiceD,'d')
        END
    ELSE
        BEGIN
            INSERT INTO Question_Choice
            VALUES(@qstId,'false','f'),(@qstId,'true','t')
        END
end try
begin catch
    select 'error'
end CATCH
GO
```

Uses

[dbo].[Question_Choice]
[dbo].[Questions]

[dbo].[instructorCourses]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@ins_id	int	4

SQL Script

```
CREATE PROC [dbo].[instructorCourses] @ins_id INT
AS
IF EXISTS(SELECT ins_id FROM Instructor  WHERE Ins_id=@ins_id)
BEGIN
SELECT c.name AS 'course name' , COUNT(sc.stu_id) AS 'students number' FROM Student_-
Course sc, Courses c,Instructor_Course ic
WHERE c.Crs_id=sc.Crs_id AND c.Crs_id=ic.Crs_id AND ic.Ins_id=@ins_id
GROUP BY c.name
END
ELSE
SELECT 'instructor does not exist'
GO
```

Uses

[dbo].[Courses]
[dbo].[Instructor]
[dbo].[Instructor_Course]
[dbo].[Student_Course]

[dbo].[PrintExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@exam_id	int	4

SQL Script

```
create proc [dbo].[PrintExam] @exam_id int
as
select eq.Exm_id ,q.Question, q.type, qc.Choice, q.Model_Answer from Exams  e
    inner join Exam_Questions  eq on e.Exm_id = eq.Exm_id
    inner join Questions  q on eq.Qst_id = q.Qst_id
    inner join Question_Choice qc on q.Qst_id = qc.Qst_id
    where e.Exm_id = @exam_id
GO
```

Uses

[dbo].[Exam_Questions]
[dbo].[Exams]
[dbo].[Question_Choice]
[dbo].[Questions]

[dbo].[selectCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@columns	varchar(50)	50

SQL Script

```
CREATE PROC [dbo].[selectCourse] @id INT=-1, @columns varchar(50)
AS BEGIN
IF @id!=-1
EXECUTE ('SELECT '+@columns+' from courses where crs_id='+@id)
ELSE
EXECUTE ('SELECT '+@columns+' from courses')
END
GO
```

[dbo].[selectDepartment]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@columns	varchar(50)	50

SQL Script

```
create PROC [dbo].[selectDepartment] @id INT=-1, @columns varchar(50)
AS BEGIN
IF @id!=-1
EXECUTE ('SELECT '+@columns+' from department where did='+@id)
ELSE
EXECUTE ('SELECT '+@columns+' from department')
END
GO
```

[dbo].[StudentExamAnswers]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@exam_id	int	4
@std_id	int	4
@q_1	varchar(50)	50
@q_2	varchar(50)	50
@q_3	varchar(50)	50
@q_4	varchar(50)	50
@q_5	varchar(50)	50
@q_6	varchar(50)	50
@q_7	varchar(50)	50
@q_8	varchar(50)	50
@q_9	varchar(50)	50
@q_10	varchar(50)	50

SQL Script

```
CREATE PROC [dbo].[StudentExamAnswers]
@exam_id int, @std_id INT, @q_1 varchar(50),
@q_2 varchar(50),@q_3 varchar(50),@q_4 varchar(50),@q_5 varchar(50),@q_6 varchar(50),
@q_7 varchar(50),@q_8 varchar(50),@q_9 varchar(50),@q_10 varchar(50)

WITH ENCRYPTION
AS
BEGIN TRY
    IF NOT EXISTS (SELECT * FROM Student  WHERE Stu_id=@std_id)
    BEGIN
    SELECT 'Student Does Not Exists'
    END
    ELSE IF NOT EXISTS(SELECT * FROM Exams e WHERE e.Exm_id=@exam_id)
    BEGIN
    SELECT 'Exam does Not Exists'
```

```
    END
    ELSE -- My Code
    BEGIN

            DECLARE @Answers table(answer VARCHAR(50))
            INSERT INTO @Answers VALUES(@q_1), (@q_2), (@q_3),(@q_4), (@q_5),
(@q_6),(@q_7), (@q_8), (@q_9),(@q_10)

            INSERT INTO Student_Exam_Question
            SELECT s.Stu_id,eq.Exm_id,eq.Qst_id, NULL  --Must Insert Value for Every
Column
            FROM Student s, Exam_Questions eq
            WHERE s.Stu_id=@std_id AND eq.Exm_id=@exam_id
            --Loop To Add the Anwers
            DECLARE c1 CURSOR
            FOR SELECT answer FROM @Answers
            FOR READ ONLY
            DECLARE @addAnswer VARCHAR(50)
            OPEN c1
            FETCH c1 INTO @addAnswer
            ------
            DECLARE c2 CURSOR --Pointer on the Student_Exam_Question
            FOR SELECT seq.Qst_id FROM Student_Exam_Question seq
            WHERE seq.Exm_id=@exam_id AND seq.Stu_id=@std_id
            FOR UPDATE
            DECLARE @oldValue int
            OPEN c2
            FETCH c2 INTO @oldValue


                    WHILE @@FETCH_STATUS=0
                    BEGIN
                    UPDATE Student_Exam_Question SET Student_Answer =@addAnswer WHERE
Exm_id=@exam_id and Stu_id=@std_id and Qst_id=@oldValue
                    FETCH c1 INTO @addAnswer
                    FETCH c2 INTO @oldValue
                    END

            CLOSE c1
            CLOSE c2
            DEALLOCATE c1
            DEALLOCATE c2

    END

END TRY
BEGIN CATCH
SELECT 'Error'
END CATCH
GO
```


Uses

[dbo].[Exam_Questions]

[dbo].[Exams]

[dbo].[Student]

[dbo].[Student_Exam_Question]

[dbo].[studentGrades]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@std_id	int	4

SQL Script

```
CREATE PROC [dbo].[studentGrades] @std_id INT
AS
IF EXISTS(SELECT stu_id from Student WHERE stu_id=@std_id)
BEGIN
SELECT  s.Fname +' '+ s.Lname AS 'Full Name' , c.name  ,sc.grade ,sc.percentageGrade
FROM courses c,Student_Course sc ,dbo.Student s
WHERE c.Crs_id=sc.Crs_id AND sc.Stu_id=@std_id AND s.Stu_id= sc.Stu_id
END
ELSE
SELECT 'Student does not exist' AS 'result'
GO
```

Uses

[dbo].[Courses]
[dbo].[Student]
[dbo].[Student_Course]

[dbo].[Update_Student_By_Id]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@fname	varchar(20)	20
@lName	varchar(20)	20
@phone	int	4
@email	varchar(20)	20
@address	varchar(20)	20
@leaderId	int	4
@departmentId	int	4

SQL Script

```
CREATE PROC [dbo].[Update_Student_By_Id] @id INT=-1, @fname VARCHAR(20)='', @lName
VARCHAR(20)='',
    @phone INT=-1, @email VARCHAR(20)='',@address VARCHAR(20)='', @leaderId INT=-1,
@departmentId INT =-1
WITH ENCRYPTION
AS
    BEGIN TRY
        IF @fname!=''
            BEGIN
                UPDATE Student SET Fname = @fname WHERE Stu_id = @id
            END

        IF @lname!=''
            BEGIN
                UPDATE Student SET Lname = @lName WHERE Stu_id = @id
            END

        IF @phone!=-1
            BEGIN
                UPDATE Student SET Phone = @phone WHERE Stu_id = @id
            END
```

```
        IF @email!=''
            BEGIN
                UPDATE Student SET Email = @email WHERE Stu_id = @id
            END

        IF @address!=''
            BEGIN
                UPDATE Student SET Address = @address WHERE Stu_id = @id
            END

        IF @leaderId!=-1
            BEGIN
                UPDATE Student SET Leader_id = @leaderId WHERE Stu_id = @id
            END

        IF @departmentId!=-1
            BEGIN
                UPDATE Student SET Did = @departmentId WHERE Stu_id = @id
            END

    END TRY

    BEGIN CATCH
            SELECT 'Error'
    END CATCH
GO
```

Uses

[dbo].[Student]

[dbo].[update_student_grade]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@stuId	int	4
@crsId	int	4
@Grade	int	4

SQL Script

```
CREATE PROC [dbo].[update_student_grade] @stuId INT , @crsId INT , @Grade INT
WITH ENCRYPTION

AS
    BEGIN TRY
            UPDATE Student_Course
            SET grade = @Grade
            WHERE Stu_id = @stuId AND Crs_id = @crsId
    END TRY

    BEGIN CATCH
            SELECT 'Error'
    END CATCH
GO
```

Uses

[dbo].[Student_Course]

[dbo].[updateCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@crs_id	int	4
@name	nvarchar(5)	10
@duration	int	4
@description	varchar(50)	50

SQL Script

```
CREATE PROC [dbo].[updateCourse] @crs_id INT, @name NVARCHAR(5)='', @duration INT=-
1,@description VARCHAR(50)=''
AS
BEGIN TRY
IF @name!=''
UPDATE courses SET Name=@name WHERE crs_id=@crs_id
IF @duration!=-1
UPDATE courses SET Duration=@duration WHERE Crs_id=@crs_id
IF @description!=''
UPDATE courses SET Description=@description WHERE Crs_id=@crs_id
END TRY
BEGIN CATCH
SELECT ERROR_MESSAGE()
END CATCH
GO
```

Uses

[dbo].[Courses]

[dbo].[updateDepartment]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@did	int	4
@name	varchar(10)	10
@description	varchar(100)	100
@location	nvarchar(20)	40
@mgr_id	int	4

SQL Script

```
CREATE PROC [dbo].[updateDepartment] @did INT , @name VARCHAR(10)='',@description
VARCHAR(100)='', @location NVARCHAR(20)='',@mgr_id INT=-1
AS
BEGIN TRY
IF @name!=''
UPDATE department SET dname=@name WHERE did=@did
IF @description!=''
UPDATE department SET Description=@description WHERE did=@did
IF @location!=''
UPDATE Department SET Location=@location
IF @mgr_id!=-1
UPDATE department SET Mng_id=@mgr_id WHERE Did=@did
END TRY
BEGIN CATCH
SELECT ERROR_MESSAGE()
END CATCH
GO
```

Uses

[dbo].[Department]

[dbo].[updateExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@ExmId	int	4
@date	date	3
@duration	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[updateExam] @ExmId INT=-1,@date DATE ='1600-01-01',@duration
INT =-1
WITH ENCRYPTION

AS
BEGIN
IF EXISTS (SELECT e.Exm_id FROM Exams e WHERE e.Exm_id=@ExmId)
BEGIN TRY

BEGIN

    IF(@date!='1600-01-01')
        BEGIN
            UPDATE Exams SET Date=@date WHERE @ExmId = Exm_id
        END

    IF(@duration!=-1)
    BEGIN
        IF EXISTS(    SELECT * FROM Exams e WHERE e.Duration=e.Duration)
            BEGIN
                UPDATE Exams SET Duration=@duration WHERE @ExmId = Exm_id
            END

    END

END
END TRY
```

```
BEGIN CATCH
    SELECT 'error'
END CATCH

ELSE
    select'Exam does not exist'
END
GO
```

Uses

[dbo].[Exams]

[dbo].[updateInstructoByID]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@fname	nvarchar(70)	140
@lname	nvarchar(70)	140
@degree	nvarchar(70)	140
@sal	int	4
@address	nvarchar(70)	140
@phone	int	4
@supervisor	int	4

SQL Script

```
CREATE proc [dbo].[updateInstructoByID] @id INT, @fname nvarchar(70)='',@lname
nvarchar(70) ='',@degree NVARCHAR(70)='', @sal INT=-1 , @address nvarchar(70)='',@phone
int=-1,@supervisor INT=-1
as
if exists(select Ins_id from Instructor where Ins_id = @id)
begin
begin try

if @fname != ''
update Instructor set Fname=@fname where Ins_id = @id

if @lname != ''
update Instructor set Lname=@lname where Ins_id = @id

if @degree != ''
update Instructor set Academics_Degree=@degree where Ins_id = @id
if @sal != -1
update Instructor set Salary=@sal where Ins_id = @id
if @phone != 0
update Instructor set Phone=@phone where Ins_id = @id

if @address != ''
```

```
update Instructor set Address=@address where Ins_id = @id
if @supervisor != -1
update Instructor set Super_id=@supervisor where Ins_id = @id

end try
begin catch
select 'Error'
end catch
end
else select 'Instructor Does Not Exists'
GO
```

Uses

[dbo].[Instructor]

[dbo].[updateInstructorCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@ins_id	int	4
@crs_id	int	4
@eval	nvarchar(70)	140
@ins_new	int	4
@crs_new	int	4

SQL Script

```
CREATE proc [dbo].[updateInstructorCourse]  @ins_id INT  ,@crs_id INT, @eval
NVARCHAR(70)='', @ins_new INT =-1, @crs_new INT =-1
as
begin try
if(@ins_new!=-1)
    BEGIN
        IF EXISTS(SELECT * FROM Instructor_Course ic WHERE Ins_id=@ins_id  AND
Crs_id=@crs_id)
        BEGIN
        update Instructor_Course set Ins_id=@ins_new where  Ins_id=@ins_id  AND
Crs_id=@crs_id
        END
        ELSE SELECT 'Not Found'

    END
if(@crs_new!=-1 )
    BEGIN
    IF EXISTS(SELECT * FROM Instructor_Course ic WHERE Ins_id=@ins_id  AND
Crs_id=@crs_id)
        BEGIN
        update Instructor_Course set Crs_id=@crs_new where  Ins_id=@ins_id  AND
Crs_id=@crs_id
        END
        ELSE SELECT 'Not Found'

    END
if( @eval !='')
```

```
    BEGIN
    IF EXISTS(SELECT * FROM Instructor_Course ic WHERE Ins_id=@ins_id  AND
Crs_id=@crs_id)
        BEGIN
        update Instructor_Course set Evaluation=@eval  where Ins_id=@ins_id  AND
Crs_id=@crs_id
        END
        ELSE SELECT 'Not Found'


    END


end try
begin catch
select 'Error'
end CATCH
GO
```

Uses

[dbo].[Instructor_Course]

[dbo].[updateQuestionByID]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@qstContent	varchar(200)	200
@ModelAns	varchar(10)	10
@Qstgrd	int	4
@choiceA	varchar(200)	200
@choiceB	varchar(200)	200
@choiceC	varchar(200)	200
@choiceD	varchar(200)	200

SQL Script

```
CREATE proc [dbo].[updateQuestionByID]
(@id INT
,@qstContent VARCHAR(200) = ''
,@ModelAns VARCHAR(10)=''
,@Qstgrd INT= -1
,@choiceA VARCHAR(200) = ''
,@choiceB VARCHAR(200) = ''
,@choiceC VARCHAR(200) = ''
,@choiceD VARCHAR(200) = ''
)
as
if exists(select Qst_id from Questions where Qst_id = @id)
begin
begin

if @qstContent != ''
update Questions set Question=@qstContent where Qst_id = @id

if @ModelAns != ''
update Questions set Model_Answer=@ModelAns where Qst_id = @id
```

```
if @Qstgrd != -1
update Questions set Qst_Grade=@Qstgrd where Qst_id = @id

DECLARE @type VARCHAR(20) =(SELECT type FROM questions WHERE qst_id = @id)

If @type = 'mcq'
BEGIN
    if @choiceA != ''
    begin
    update Question_Choice set Choice=@choiceA
    where choice = (SELECT choice FROM question_choice WHERE qst_id = @id
    ORDER BY qst_id OFFSET 0 ROWS FETCH NEXT 1 ROWS ONLY)
    END

    if @choiceB != ''
    BEGIN
    update Question_Choice set Choice=@choiceB
    where choice = (SELECT choice FROM question_choice WHERE qst_id = @id
    ORDER BY qst_id OFFSET 1 ROWS FETCH NEXT 1 ROWS ONLY)
    END

    if @choiceC != ''
    BEGIN
    update Question_Choice set Choice=@choiceC
    where choice = (SELECT choice FROM question_choice WHERE qst_id = @id
    ORDER BY qst_id OFFSET 2 ROWS FETCH NEXT 1 ROWS ONLY)
    END

    if @choiceD != ''
    BEGIN
    update Question_Choice set Choice=@choiceD
    where choice = (SELECT choice FROM question_choice WHERE qst_id = @id
    ORDER BY qst_id OFFSET 3 ROWS FETCH NEXT 1 ROWS ONLY)
    END
END
ELSE
    BEGIN
        SELECT 'true & false'
    END

end
--begin catch
    --select 'Error'
--end catch
end
else select 'no question found'
GO
```

Uses

[dbo].[Question_Choice]

[dbo].[Questions]

[dbo].[updateTopic]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@topId	int	4
@topName	varchar(50)	50
@crsId	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[updateTopic]
(@topId INT=-1 , @topName VARCHAR(50)='', @crsId INT=-1)

WITH ENCRYPTION

AS
IF EXISTS (SELECT Top_id FROM Topic t WHERE @topId=t.Top_id)
BEGIN TRY

BEGIN
    IF (@topName!='')
        BEGIN
            UPDATE Topic SET Top_Name=@topName WHERE Top_id = @topId
        END

     IF (@crsId!=-1)
        BEGIN
            UPDATE Topic SET Crs_id=@crsId WHERE Top_id = @topId
        END
    IF (@topName!='') AND (@crsId!=-1)
    UPDATE Topic SET Top_Name=@topName,Crs_id=@crsId WHERE Top_id = @topId
END
END TRY

BEGIN CATCH
    SELECT 'error'
END CATCH
```

```
ELSE
    select'Topic does not exist'
GO
```

Uses

[dbo].[Topic]

[dbo].[updateWorkIn]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True
Encrypted	True

Parameters

Name	Data Type	Max Length (Bytes)
@insId	int	4
@depId	int	4
@WorkingHours	int	4
@newInsId	int	4
@newDepId	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[updateWorkIn] (@insId INT=-1 , @depId INT=-1 , @WorkingHours
INT=-1,@newInsId INT=-1 , @newDepId INT=-1)
WITH ENCRYPTION
AS
BEGIN TRY
    IF(@newInsId!=-1)
        BEGIN
            UPDATE Work_In SET Ins_id=@newInsId
            WHERE @insId=Ins_id AND @depId=Did
        END

     IF(@newDepId!=-1)
        BEGIN
            UPDATE Work_In SET Did=@newDepId
            WHERE @insId=Ins_id AND @depId=Did
        END

     IF(@WorkingHours!=-1)
        BEGIN
            UPDATE Work_In SET Working_Hours=@WorkingHours
            WHERE( @insId=Ins_id AND @depId=Did )
        END

END TRY
```

```
BEGIN CATCH
    SELECT 'Error'
END CATCH
GO
```

Uses

[dbo].[Work_In]

Users

Objects

Name
dbo
guest
Instructor_Mohammed
Student_A

dbo

Properties

Property	Value
Type	WindowsUser
Login Name	DESKTOP-EKCK12U\RadwaElgammal
Default Schema	dbo

Database Level Permissions

Type	Action
CONNECT	Grant

SQL Script

```
GO
```

guest

Properties

Property	Value
Type	SqlUser
Default Schema	guest

SQL Script

```
GO
```

Instructor_Mohammed

Properties

Property	Value
Type	SqlUser
Login Name	Instructor_Mohammed
Default Schema	dbo

Database Level Permissions

Type	Action
CONNECT	Grant

SQL Script

```
CREATE USER [Instructor_Mohammed] FOR LOGIN [Instructor_Mohammed]
GO
```

Student_A

Properties

Property	Value
Type	SqlUser
Login Name	Student_A
Default Schema	dbo

Database Level Permissions

Type	Action
CONNECT	Grant

SQL Script

```
CREATE USER [Student_A] FOR LOGIN [Student_A]
GO
```

Database Roles

Objects

Name
db_accessadmin
db_backupoperator
db_datareader
db_datawriter
db_ddladmin
db_denydatareader
db_denydatawriter
db_owner
db_securityadmin
public

db_accessadmin

Properties

Property	Value
Owner	dbo

db_backupoperator

Properties

Property	Value
Owner	dbo

db_datareader

Properties

Property	Value
Owner	dbo

db_datawriter

Properties

Property	Value
Owner	dbo

db_ddladmin

Properties

Property	Value
Owner	dbo

db_denydatareader

Properties

Property	Value
Owner	dbo

db_denydatawriter

Properties

Property	Value
Owner	dbo

db_owner

Properties

Property	Value
Owner	dbo

db_securityadmin

Properties

Property	Value
Owner	dbo

public

Properties

Property	Value
Owner	dbo