(local)\SQLEXPRESS Documentation

ITI_ExamSystem

Server	LAPTOP-D5LRBLH1\SQLEXPRESS
Author	Salem
Created	Thursday, December 12, 2024 8:46:02 PM
File Path	D:\ITI_Professional_45\9- Advanced DB\Project\Final Version\DB_DOC-2024-12-12T20-46-02.docx

Team 6

- **Fatma Ali Ali Elshihna**
- **Mostafa Emad Mostafa Helal**
- **Safiya Kamal Ahmed Masoud**
- **Salem Gamal Salem Mohamed**

Table of Contents

LAPTOP-D5LRBLH1\SQLEXPRESS

Databases (1)

- ITI_ExamSystem

Server Properties

Property	Value
Product	Microsoft SQL Server
Version	16.0.1000.6
Language	English (United States)
Platform	NT x64
Edition	Express Edition (64-bit)
Engine Edition	4 (Express)
Processors	16
OS Version	6.3 (26100)
Physical Memory	16003
Is Clustered	False
Root Directory	C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL
Collation	SQL_Latin1_General_CP1_CI_AS

Server Settings

Property	Value
Default data file path	C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL\DATA\
Default backup file path	C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL\Backup
Default log file path	C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL\DATA\
Recovery Interval (minutes)	0
Default index fill factor	0
Default backup media retention	0

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)

- ITI_ExamSystem

ITI_ExamSystem Database

Database Properties

Property	Value
SQL Server Version	Max
Compatibility Level	Max
Last backup time	12/12/2024
Last log backup time	-
Creation date	Dec 5 2024
Users	4
Database Encryption Enabled	False
Database Encryption Algorithm	None
Database size	80.00 MB
Unallocated space	62.85 MB

Database Options

Property	Value
Compatibility Level	160
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	SIMPLE
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	40c72baa-54a9-499d-b141-bcb9aedb95a9
Service broker enabled	False
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	LAPTOP-D5LRBLH1\Salem

Files

Name	Type	Size	Maxsize	Autogrowth	File Name
ITI_ExamSystem	Data	72.00 MB	unlimited	64.00 MB	C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL\DATA\ITI_ExamSystem.mdf
ITI_ExamSystem_log	Log	8.00 MB	2048.00 GB	64.00 MB	C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL\DATA\ITI_ExamSystem_log.ldf

Tables

Objects

Name
dbo.Branch *Branch Table defines each branch by id and name of it*
dbo.Branches_tracks *Branchs_track table shows the track in each branch*
dbo.Course *Course define each course by its name*
dbo.Crs_track *Crs_track shows Courses in the track*
dbo.Exam *Exam table defines exam data like id, name, date and duration*
dbo.Exam_questions *Exam_questions shows all questions in each exam*
dbo.Ins_course *Ins_course table shows instructors who teach this course*
dbo.Instructor *Instructor table deffine each instructor by his id, name, address and other data*
dbo.Instructors_in_track *Instructors_in_track table shoes all instructors who work in track*
dbo.Question *Question table define each question by id and other question data like question itself, model answer and grade*
dbo.Question_choices
dbo.Student *Student table deffine each student by his id, name, address and other data*
dbo.Student_crs *Student_crs shows students who assigned to this course*
dbo.Student_exam *Student_exam table shows each student and the grade of each exam he did*
dbo.Student_exam_questions *Student_exam_questions table shows student answers in specific exam he did*
dbo.Topic
dbo.Track *Track table defines each track by id, name and the track manager*

[dbo].[Branch]

MS_Description

Branch Table defines each branch by id and name of it

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	4
Created	9:49:29 PM Wednesday, December 4, 2024
Last Modified	12:44:27 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
PK C	id	int	4	NOT NULL	1 - 1
	name	varchar(50)	50	NOT NULL	

Indexes

Key	Name	Key Columns	Unique
PK C	PK__Branch__3213E83F4F9C8B34	id	True

SQL Script

```
CREATE TABLE [dbo].[Branch]
(
[id] [int] NOT NULL IDENTITY(1, 1),
[name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Branch] ADD CONSTRAINT [PK__Branch__3213E83F4F9C8B34] PRIMARY KEY
CLUSTERED ([id]) ON [PRIMARY]
GO
EXEC sp_addextendedproperty N'MS_Description', N'Branch Table defines each branch by id
and name of it', 'SCHEMA', N'dbo', 'TABLE', N'Branch', NULL, NULL
GO
```

Used By

[dbo].[Branches_tracks]
[dbo].[Sp_deleteBranch]
[dbo].[Sp_InsertBranch]
[dbo].[Sp_SelectBranch]
[dbo].[Sp_updateBranch]

[dbo].[Branches_tracks]

MS_Description

Branchs_track table shows the track in each branch

Properties

Property	Value
Row Count (~)	10
Created	10:08:35 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

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

Indexes

Key	Name	Key Columns	Unique
PK C	PK__Branches__1710FB5CA4CF2F6B	branch_id, track_id	True

Foreign Keys

Name	Columns
FK__Branches___branc__48CFD27E	branch_id->[dbo].[Branch].[id]
FK__Branches___track__49C3F6B7	track_id->[dbo].[Track].[id]

SQL Script

```
CREATE TABLE [dbo].[Branches_tracks]
(
[branch_id] [int] NOT NULL,
[track_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Branches_tracks] ADD CONSTRAINT [PK__Branches__1710FB5CA4CF2F6B]
```

```
PRIMARY KEY CLUSTERED ([branch_id], [track_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Branches_tracks] ADD CONSTRAINT [FK__Branches___branc__48CFD27E]
FOREIGN KEY ([branch_id]) REFERENCES [dbo].[Branch] ([id])
GO
ALTER TABLE [dbo].[Branches_tracks] ADD CONSTRAINT [FK__Branches___track__49C3F6B7]
FOREIGN KEY ([track_id]) REFERENCES [dbo].[Track] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Branchs_track  table shows the track in
each branch', 'SCHEMA', N'dbo', 'TABLE', N'Branches_tracks', NULL, NULL
GO
```

Uses

[dbo].[Branch]
[dbo].[Track]

Used By

[dbo].[Sp_deleteBranch]
[dbo].[Sp_deleteBranchesTracks]
[dbo].[Sp_deleteTrack]
[dbo].[Sp_insertBranchesTracks]
[dbo].[Sp_SelectBranchesTracks]
[dbo].[Sp_updateBranch]
[dbo].[Sp_updateBranchesTracks]
[dbo].[Sp_updateTrack]

[dbo].[Course]

MS_Description

Course define each course by its name

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	5
Created	9:58:06 PM Wednesday, December 4, 2024
Last Modified	9:19:15 AM Thursday, December 12, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
PK C	id	int	4	NOT NULL	1 - 1
	name	varchar(50)	50	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
PK C	PK__Course__3213E83F198E67E7	id	True

SQL Script

```
CREATE TABLE [dbo].[Course]
(
[id] [int] NOT NULL IDENTITY(1, 1),
[name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Course] ADD CONSTRAINT [PK__Course__3213E83F198E67E7] PRIMARY KEY
CLUSTERED ([id]) ON [PRIMARY]
GO
EXEC sp_addextendedproperty N'MS_Description', N'Course define each course by its name',
'SCHEMA', N'dbo', 'TABLE', N'Course', NULL, NULL
GO
```

Used By

[dbo].[Crs_track]
[dbo].[Exam]
[dbo].[Ins_course]
[dbo].[Question]
[dbo].[Student_crs]
[dbo].[Topic]
[dbo].[Insert_Stu_Crs]
[dbo].[Sp_deleteCourse]
[dbo].[Sp_insertCourse]
[dbo].[Sp_InsertCrsTrack]
[dbo].[SP_insertQuestions]
[dbo].[SP_r2GetStdGradeByStdId]
[dbo].[SP_r3GetCoursesInsTeach]
[dbo].[SP_r4GetTopicsInCourse]
[dbo].[SP_r5QuestionAndChoicesinExam]
[dbo].[Sp_selectCourse]
[dbo].[Sp_UpdateCourse]
[dbo].[Update_Ins_Course]
[dbo].[Update_Stu_Crs]

[dbo].[Crs_track]

MS_Description

Crs_track shows Courses in the track

Properties

Property	Value
Row Count (~)	12
Created	10:11:47 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK FK	track_id	int	4	NOT NULL
PK FK	crs_id	int	4	NOT NULL

Indexes

Key	Name	Key Columns	Unique
PK	PK__Crs_trac__6A263D19FD90F571	track_id, crs_id	True

Foreign Keys

Name	Columns
FK__Crs_track__crs_i__5165187F	crs_id->[dbo].[Course].[id]
FK__Crs_track__track__5070F446	track_id->[dbo].[Track].[id]

SQL Script

```
CREATE TABLE [dbo].[Crs_track]
(
[track_id] [int] NOT NULL,
[crs_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Crs_track] ADD CONSTRAINT [PK__Crs_trac__6A263D19FD90F571] PRIMARY KEY
```

```
CLUSTERED ([track_id], [crs_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Crs_track] ADD CONSTRAINT [FK__Crs_track__crs_i__5165187F] FOREIGN KEY
([crs_id]) REFERENCES [dbo].[Course] ([id])
GO
ALTER TABLE [dbo].[Crs_track] ADD CONSTRAINT [FK__Crs_track__track__5070F446] FOREIGN KEY
([track_id]) REFERENCES [dbo].[Track] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Crs_track shows Courses in the track',
'SCHEMA', N'dbo', 'TABLE', N'Crs_track', NULL, NULL
GO
```

Uses

[dbo].[Course]
[dbo].[Track]

Used By

[dbo].[Sp_deleteCourse]
[dbo].[Sp_DeleteCrsTrack]
[dbo].[Sp_deleteTrack]
[dbo].[Sp_InsertCrsTrack]
[dbo].[Sp_SelectCrsTrack]
[dbo].[Sp_UpdateCourse]
[dbo].[Sp_UpdateCrsTrack]
[dbo].[Sp_updateTrack]

[dbo].[Exam]

MS_Description

Exam table defines exam data like id, name, date and duration

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	7
Created	10:00:22 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
PK C	id	int	4	NOT NULL	1 - 1
	name	varchar(50)	50	NULL allowed	
	ex_date	date	3	NULL allowed	
	duration	int	4	NULL allowed	
FK	crs_id	int	4	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
PK C	PK__Exam__3213E83F8D5FA1A9	id	True

Foreign Keys

Name	Columns
FK__Exam__crs_id__778AC167	crs_id->[dbo].[Course].[id]

SQL Script

```
CREATE TABLE [dbo].[Exam]
(
[id] [int] NOT NULL IDENTITY(1, 1),
[name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[ex_date] [date] NULL,
[duration] [int] NULL,
[crs_id] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Exam] ADD CONSTRAINT [PK__Exam__3213E83F8D5FA1A9] PRIMARY KEY
CLUSTERED ([id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Exam] ADD CONSTRAINT [FK__Exam__crs_id__778AC167] FOREIGN KEY
([crs_id]) REFERENCES [dbo].[Course] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Exam table defines exam data like id,
name, date and duration', 'SCHEMA', N'dbo', 'TABLE', N'Exam', NULL, NULL
GO
```

Uses

[dbo].[Course]

Used By

[dbo].[Exam_questions]
[dbo].[Student_exam]
[dbo].[Student_exam_questions]
[dbo].[SP_correctExam]
[dbo].[SP_deleteExam]
[dbo].[SP_generateExam]
[dbo].[SP_insertStudentExamQuestionAns]
[dbo].[SP_r2GetStdGradeByStdId]
[dbo].[SP_r5QuestionAndChoicesinExam]
[dbo].[SP_selectExam]
[dbo].[SP_updateExam]

[dbo].[Exam_questions]

MS_Description

Exam_questions shows all questions in each exam

Properties

Property	Value
Row Count (~)	69
Created	10:21:49 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK FK	ex_id	int	4	NOT NULL
PK FK	q_id	int	4	NOT NULL

Indexes

Key	Name	Key Columns	Unique
PK	PK__Exam_que__E5067FB8DB0A382F	ex_id, q_id	True

Foreign Keys

Name	Columns
FK__Exam_ques__ex_id__60A75C0F	ex_id->[dbo].[Exam].[id]
FK__Exam_quest__q_id__619B8048	q_id->[dbo].[Question].[id]

SQL Script

```
CREATE TABLE [dbo].[Exam_questions]
(
[ex_id] [int] NOT NULL,
[q_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Exam_questions] ADD CONSTRAINT [PK__Exam_que__E5067FB8DB0A382F]
PRIMARY KEY CLUSTERED ([ex_id], [q_id]) ON [PRIMARY]
```

```
GO
ALTER TABLE [dbo].[Exam_questions] ADD CONSTRAINT [FK__Exam_ques__ex_id__60A75C0F]
FOREIGN KEY ([ex_id]) REFERENCES [dbo].[Exam] ([id])
GO
ALTER TABLE [dbo].[Exam_questions] ADD CONSTRAINT [FK__Exam_quest__q_id__619B8048]
FOREIGN KEY ([q_id]) REFERENCES [dbo].[Question] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Exam_questions shows all questions in
each exam', 'SCHEMA', N'dbo', 'TABLE', N'Exam_questions', NULL, NULL
GO
```

Uses

[dbo].[Exam]
[dbo].[Question]

Used By

[dbo].[SP_correctExam]
[dbo].[SP_deleteExam]
[dbo].[SP_deleteExamQuestions]
[dbo].[SP_deleteQuestion]
[dbo].[SP_generateExam]
[dbo].[SP_insertExamQuestions]
[dbo].[SP_insertStudentExamQuestionAns]
[dbo].[SP_r5QuestionAndChoicesinExam]
[dbo].[SP_selectExamQuestion]
[dbo].[SP_updateExamQuestions]

[dbo].[Ins_course]

MS_Description

Ins_course table shows instructors who teach this course

Properties

Property	Value
Row Count (~)	11
Created	10:15:11 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

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

Indexes

Key	Name	Key Columns	Unique
PK C	PK__Ins_cour__F56421A7D00BEA77	crs_id, ins_id	True

Foreign Keys

Name	Columns
FK__Ins_cours__crs_i__5441852A	crs_id->[dbo].[Course].[id]
FK__Ins_cours__ins_i__5535A963	ins_id->[dbo].[Instructor].[id]

SQL Script

```
CREATE TABLE [dbo].[Ins_course]
(
[crs_id] [int] NOT NULL,
[ins_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Ins_course] ADD CONSTRAINT [PK__Ins_cour__F56421A7D00BEA77] PRIMARY
KEY CLUSTERED ([crs_id], [ins_id]) ON [PRIMARY]
```

```
GO
ALTER TABLE [dbo].[Ins_course] ADD CONSTRAINT [FK__Ins_cours__crs_i__5441852A] FOREIGN
KEY ([crs_id]) REFERENCES [dbo].[Course] ([id])
GO
ALTER TABLE [dbo].[Ins_course] ADD CONSTRAINT [FK__Ins_cours__ins_i__5535A963] FOREIGN
KEY ([ins_id]) REFERENCES [dbo].[Instructor] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Ins_course table shows instructors who
teach this course', 'SCHEMA', N'dbo', 'TABLE', N'Ins_course', NULL, NULL
GO
```

Uses

[dbo].[Course]
[dbo].[Instructor]

Used By

[dbo].[Delete_Ins_Course]
[dbo].[DeleteInstructor]
[dbo].[Insert_Ins_Course]
[dbo].[Select_Ins_Course]
[dbo].[SP_r3GetCoursesInsTeach]
[dbo].[Update_Ins_Course]

[dbo].[Instructor]

MS_Description

Instructor table deffine each instructor by his id, name, address and other data

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	9
Created	9:56:26 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
PK C	id	int	4	NOT NULL	1 - 1
	fname	varchar(50)	50	NOT NULL	
	lname	varchar(50)	50	NULL allowed	
	city	varchar(50)	50	NULL allowed	
	street	varchar(100)	100	NULL allowed	
	bdate	date	3	NULL allowed	
	phone	varchar(25)	25	NULL allowed	
	salary	int	4	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
PK C	PK__Instruct__3213E83FBFF0BB47	id	True

SQL Script

```
CREATE TABLE [dbo].[Instructor]
(
[id] [int] NOT NULL IDENTITY(1, 1),
[fname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[lname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[city] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[street] [varchar] (100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[bdate] [date] NULL,
[phone] [varchar] (25) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[salary] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Instructor] ADD CONSTRAINT [PK__Instruct__3213E83FBFF0BB47] PRIMARY
KEY CLUSTERED ([id]) ON [PRIMARY]
GO
EXEC sp_addextendedproperty N'MS_Description', N'Instructor table deffine each instructor
by his id, name, address and other data', 'SCHEMA', N'dbo', 'TABLE', N'Instructor', NULL,
NULL
GO
```

Used By

[dbo].[Ins_course]

[dbo].[Instructors_in_track]

[dbo].[Track]

[dbo].[DeleteInstructor]

[dbo].[SelectInstructor]

[dbo].[SP_insertInstructor]

[dbo].[SP_r3GetCoursesInsTeach]

[dbo].[UpdateInstructor]

[dbo].[Instructors_in_track]

MS_Description

Instructors_in_track table shoes all instructors who work in track

Properties

Property	Value
Row Count (~)	10
Created	10:10:20 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK FK	track_id	int	4	NOT NULL
PK FK	ins_id	int	4	NOT NULL

Indexes

Key	Name	Key Columns	Unique
PK	PK__Instruct__3D27BAFCB901F1FC	track_id, ins_id	True

Foreign Keys

Name	Columns
FK__Instructo__ins_i__4D94879B	ins_id->[dbo].[Instructor].[id]
FK__Instructo__track__4CA06362	track_id->[dbo].[Track].[id]

SQL Script

```
CREATE TABLE [dbo].[Instructors_in_track]
(
[track_id] [int] NOT NULL,
[ins_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Instructors_in_track] ADD CONSTRAINT [PK__Instruct__3D27BAFCB901F1FC]
```

```
PRIMARY KEY CLUSTERED ([track_id], [ins_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Instructors_in_track] ADD CONSTRAINT [FK__Instructo__ins_i__4D94879B]
FOREIGN KEY ([ins_id]) REFERENCES [dbo].[Instructor] ([id])
GO
ALTER TABLE [dbo].[Instructors_in_track] ADD CONSTRAINT [FK__Instructo__track__4CA06362]
FOREIGN KEY ([track_id]) REFERENCES [dbo].[Track] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Instructors_in_track table shoes all
instructors who work in track', 'SCHEMA', N'dbo', 'TABLE', N'Instructors_in_track', NULL,
NULL
GO
```

Uses

[dbo].[Instructor]
[dbo].[Track]

Used By

[dbo].[DeleteInstructor]
[dbo].[SelectInstructorinTrack]
[dbo].[SP_insertInstructorInTrack]

[dbo].[Question]

MS_Description

Question table define each question by id and other question data like question itself, model answer and grade

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	171
Created	10:02:24 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
PK	id	int	4	NOT NULL	1 - 1
	q_head	varchar(400)	400	NULL allowed	
	model_ans	varchar(100)	100	NULL allowed	
	grade	int	4	NULL allowed	
	type	varchar(10)	10	NULL allowed	
FK	crs_id	int	4	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
PK	PK__Question__3213E83F30981880	id	True

Foreign Keys

Name	Columns
FK__Question__crs_id__787EE5A0	crs_id->[dbo].[Course].[id]

SQL Script

```
CREATE TABLE [dbo].[Question]
(
[id] [int] NOT NULL IDENTITY(1, 1),
[q_head] [varchar] (400) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[model_ans] [varchar] (100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[grade] [int] NULL,
[type] [varchar] (10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[crs_id] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Question] ADD CONSTRAINT [PK__Question__3213E83F30981880] PRIMARY KEY
CLUSTERED ([id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Question] ADD CONSTRAINT [FK__Question__crs_id__787EE5A0] FOREIGN KEY
([crs_id]) REFERENCES [dbo].[Course] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Question table define each question by
id and other question data like question itself, model answer and grade', 'SCHEMA',
N'dbo', 'TABLE', N'Question', NULL, NULL
GO
```

Uses

[dbo].[Course]

Used By

[dbo].[Exam_questions]
[dbo].[Question_choices]
[dbo].[Student_exam_questions]
[dbo].[SP_correctExam]
[dbo].[Sp_deleteCourse]
[dbo].[SP_deleteQuestion]
[dbo].[SP_deleteQuestionChoices]
[dbo].[SP_generateExam]
[dbo].[SP_insertQuestions]
[dbo].[SP_r5QuestionAndChoicesinExam]
[dbo].[SP_r6QuestionAndAnswersinExam]
[dbo].[SP_selectExamQuestion]
[dbo].[SP_selectQuestionChoices]
[dbo].[SP_selectQuestions]
[dbo].[SP_updateQuestionChoices]
[dbo].[SP_updateQuestions]

[dbo].[Question_choices]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	494
Created	10:27:04 PM Wednesday, December 4, 2024
Last Modified	10:27:04 PM Wednesday, December 4, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK FK	q_id	int	4	NOT NULL
PK	choice	varchar(100)	100	NOT NULL

Indexes

Key	Name	Key Columns	Unique
PK	PK__Question__2A2ABA40110052A2	q_id, choice	True

Foreign Keys

Name	Columns
FK__Question_c__q_id__6A30C649	q_id->[dbo].[Question].[id]

SQL Script

```
CREATE TABLE [dbo].[Question_choices]
(
[q_id] [int] NOT NULL,
[choice] [varchar] (100) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Question_choices] ADD CONSTRAINT [PK__Question__2A2ABA40110052A2]
PRIMARY KEY CLUSTERED ([q_id], [choice]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Question_choices] ADD CONSTRAINT [FK__Question_c__q_id__6A30C649]
FOREIGN KEY ([q_id]) REFERENCES [dbo].[Question] ([id])
GO
```

Uses

[dbo].[Question]

Used By

[dbo].[SP_deleteQuestion]
[dbo].[SP_deleteQuestionChoices]
[dbo].[SP_insertQuestions]
[dbo].[SP_r5QuestionAndChoicesinExam]
[dbo].[SP_selectQuestionChoices]
[dbo].[SP_selectQuestions]
[dbo].[SP_updateQuestionChoices]
[dbo].[SP_updateQuestions]

[dbo].[Student]

MS_Description

Student table deffine each student by his id, name, address and other data

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	5
Created	9:57:02 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
PK C	id	int	4	NOT NULL	1 - 1
	fname	varchar(50)	50	NOT NULL	
	lname	varchar(50)	50	NULL allowed	
	city	varchar(50)	50	NULL allowed	
	street	varchar(100)	100	NULL allowed	
	bdate	date	3	NULL allowed	
	phone	varchar(25)	25	NULL allowed	
FK	track_id	int	4	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
PK C	PK__Student__3213E83F548A5ACA	id	True

Foreign Keys

Name	Columns
FK__Student__track_i__797309D9	track_id->[dbo].[Track].[id]

SQL Script

```
CREATE TABLE [dbo].[Student]
```

```
(
[id] [int] NOT NULL IDENTITY(1, 1),
[fname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[lname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[city] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[street] [varchar] (100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[bdate] [date] NULL,
[phone] [varchar] (25) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[track_id] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student] ADD CONSTRAINT [PK__Student__3213E83F548A5ACA] PRIMARY KEY
CLUSTERED ([id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student] ADD CONSTRAINT [FK__Student__track_i__797309D9] FOREIGN KEY
([track_id]) REFERENCES [dbo].[Track] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Student table deffine each student by
his id, name, address and other data', 'SCHEMA', N'dbo', 'TABLE', N'Student', NULL, NULL
GO
```

Uses

[dbo].[Track]

Used By

[dbo].[Student_crs]
[dbo].[Student_exam]
[dbo].[Student_exam_questions]
[dbo].[Delete_Student]
[dbo].[Insert_Stu_Crs]
[dbo].[Insert_Student]
[dbo].[Select_Student]
[dbo].[SP_correctExam]
[dbo].[SP_insertStudentExamQuestionAns]
[dbo].[SP_r1GetStudentInTrack]
[dbo].[SP_r2GetStdGradeByStdId]
[dbo].[SP_r3GetCoursesInsTeach]
[dbo].[SP_r6QuestionAndAnswersinExam]
[dbo].[Update_Student]

[dbo].[Student_crs]

MS_Description

Student_crs shows students who assigned to this course

Properties

Property	Value
Row Count (~)	25
Created	10:16:51 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK FK	crs_id	int	4	NOT NULL
PK FK	stu_id	int	4	NOT NULL

Indexes

Key	Name	Key Columns	Unique
PK	PK__Student___E2FC99C73D12A01C	crs_id, stu_id	True

Foreign Keys

Name	Columns
FK__Student_c__crs_i__5812160E	crs_id->[dbo].[Course].[id]
FK__Student_c__stu_i__59063A47	stu_id->[dbo].[Student].[id]

SQL Script

```
CREATE TABLE [dbo].[Student_crs]
(
[crs_id] [int] NOT NULL,
[stu_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student_crs] ADD CONSTRAINT [PK__Student___E2FC99C73D12A01C] PRIMARY
KEY CLUSTERED ([crs_id], [stu_id]) ON [PRIMARY]
```

```
GO
ALTER TABLE [dbo].[Student_crs] ADD CONSTRAINT [FK__Student_c__crs_i__5812160E] FOREIGN
KEY ([crs_id]) REFERENCES [dbo].[Course] ([id])
GO
ALTER TABLE [dbo].[Student_crs] ADD CONSTRAINT [FK__Student_c__stu_i__59063A47] FOREIGN
KEY ([stu_id]) REFERENCES [dbo].[Student] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Student_crs shows students who assigned
to this course', 'SCHEMA', N'dbo', 'TABLE', N'Student_crs', NULL, NULL
GO
```

Uses

[dbo].[Course]
[dbo].[Student]

Used By

[dbo].[Delete_Stu_Crs]
[dbo].[Delete_Student]
[dbo].[Insert_Stu_Crs]
[dbo].[Select_Stu_Crs]
[dbo].[Sp_deleteCourse]
[dbo].[SP_r3GetCoursesInsTeach]
[dbo].[Update_Stu_Crs]

[dbo].[Student_exam]

MS_Description

Student_exam table shows each student and the grade of each exam he did

Properties

Property	Value
Row Count (~)	1
Created	10:18:59 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK FK	stu_id	int	4	NOT NULL
PK FK	ex_id	int	4	NOT NULL
	grade	int	4	NOT NULL

Indexes

Key	Name	Key Columns	Unique
PK	PK__Student___F8802E3BCB0CFBBE	ex_id, stu_id	True

Foreign Keys

Name	Columns
FK__Student_e__ex_id__5CD6CB2B	ex_id->[dbo].[Exam].[id]
FK__Student_e__stu_i__5BE2A6F2	stu_id->[dbo].[Student].[id]

SQL Script

```
CREATE TABLE [dbo].[Student_exam]
(
[stu_id] [int] NOT NULL,
[ex_id] [int] NOT NULL,
[grade] [int] NOT NULL
) ON [PRIMARY]
GO
```

```
ALTER TABLE [dbo].[Student_exam] ADD CONSTRAINT [PK__Student___F8802E3BCB0CFBBE] PRIMARY
KEY CLUSTERED ([ex_id], [stu_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student_exam] ADD CONSTRAINT [FK__Student_e__ex_id__5CD6CB2B] FOREIGN
KEY ([ex_id]) REFERENCES [dbo].[Exam] ([id])
GO
ALTER TABLE [dbo].[Student_exam] ADD CONSTRAINT [FK__Student_e__stu_i__5BE2A6F2] FOREIGN
KEY ([stu_id]) REFERENCES [dbo].[Student] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Student_exam table shows each student
and the grade of each exam he did', 'SCHEMA', N'dbo', 'TABLE', N'Student_exam', NULL,
NULL
GO
```

Uses

[dbo].[Exam]
[dbo].[Student]

Used By

[dbo].[Delete_Student]
[dbo].[SP_deleteStudentExam]
[dbo].[SP_insertStudentExam]
[dbo].[SP_r2GetStdGradeByStdId]
[dbo].[SP_selectStudentExam]
[dbo].[SP_updateStudentExam]

[dbo].[Student_exam_questions]

MS_Description

Student_exam_questions table shows student answers in specific exam he did

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	10
Created	10:25:30 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK FK	stu_id	int	4	NOT NULL
PK FK	ex_id	int	4	NOT NULL
PK FK	q_id	int	4	NOT NULL
	stu_ans	varchar(100)	100	NULL allowed

Indexes

Key	Name	Key Columns	Unique
PK	PK__Student___7B6CCCDA2C504F08	stu_id, ex_id, q_id	True

Foreign Keys

Name	Columns
FK__Student_e__ex_id__66603565	ex_id->[dbo].[Exam].[id]
FK__Student_ex__q_id__6754599E	q_id->[dbo].[Question].[id]
FK__Student_e__stu_i__656C112C	stu_id->[dbo].[Student].[id]

SQL Script

```
CREATE TABLE [dbo].[Student_exam_questions]
```

```
(
[stu_id] [int] NOT NULL,
[ex_id] [int] NOT NULL,
[q_id] [int] NOT NULL,
[stu_ans] [varchar] (100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student_exam_questions] ADD CONSTRAINT
[PK__Student___7B6CCCDA2C504F08] PRIMARY KEY CLUSTERED ([stu_id], [ex_id], [q_id]) ON
[PRIMARY]
GO
ALTER TABLE [dbo].[Student_exam_questions] ADD CONSTRAINT
[FK__Student_e__ex_id__66603565] FOREIGN KEY ([ex_id]) REFERENCES [dbo].[Exam] ([id])
GO
ALTER TABLE [dbo].[Student_exam_questions] ADD CONSTRAINT
[FK__Student_ex__q_id__6754599E] FOREIGN KEY ([q_id]) REFERENCES [dbo].[Question] ([id])
GO
ALTER TABLE [dbo].[Student_exam_questions] ADD CONSTRAINT
[FK__Student_e__stu_i__656C112C] FOREIGN KEY ([stu_id]) REFERENCES [dbo].[Student] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Student_exam_questions table shows
student answers in specific exam he did', 'SCHEMA', N'dbo', 'TABLE',
N'Student_exam_questions', NULL, NULL
GO
```

Uses

[dbo].[Exam]
[dbo].[Question]
[dbo].[Student]

Used By

[dbo].[Delete_Student]
[dbo].[SP_correctExam]
[dbo].[SP_deleteStudentExamQuestions]
[dbo].[SP_insertStudentExam]
[dbo].[SP_insertStudentExamQuestionAns]
[dbo].[SP_r6QuestionAndAnswersinExam]
[dbo].[SP_selectStudentExamQuestions]
[dbo].[SP_updateStudentExamQuestions]

[dbo].[Topic]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	25
Created	9:18:56 AM Thursday, December 12, 2024
Last Modified	4:13:02 PM Thursday, December 12, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK	id	int	4	NOT NULL
	name	nvarchar(max)	max	NULL allowed
FK	crs_id	int	4	NOT NULL

Indexes

Key	Name	Key Columns	Unique
PK	PK_Topic	id	True

Foreign Keys

Name	Columns
FK__Topic__crs_id__625A9A57	crs_id->[dbo].[Course].[id]

SQL Script

```
CREATE TABLE [dbo].[Topic]
(
[id] [int] NOT NULL,
[name] [nvarchar] (max) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[crs_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Topic] ADD CONSTRAINT [PK_Topic] PRIMARY KEY CLUSTERED ([id]) ON
[PRIMARY]
GO
ALTER TABLE [dbo].[Topic] ADD CONSTRAINT [FK__Topic__crs_id__625A9A57] FOREIGN KEY
([crs_id]) REFERENCES [dbo].[Course] ([id])
```

```
GO
```

Uses

[dbo].[Course]

Used By

[dbo].[Sp_deleteCourse]
[dbo].[Sp_DeleteTopic]
[dbo].[Sp_InsertTopic]
[dbo].[SP_r4GetTopicsInCourse]
[dbo].[Sp_SelectTopic]
[dbo].[Sp_UpdateCourse]
[dbo].[Sp_UpdateTopic]

[dbo].[Track]

MS_Description

Track table defines each track by id, name and the track manager

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	6
Created	9:51:31 PM Wednesday, December 4, 2024
Last Modified	1:02:11 AM Sunday, December 8, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
PK C	id	int	4	NOT NULL	1 - 1
	name	varchar(50)	50	NOT NULL	
FK	track_mgr	int	4	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
PK C	PK__Track__3213E83FD5B44C38	id	True

Foreign Keys

Name	Columns
FK__Track__track_mgr__75A278F5	track_mgr->[dbo].[Instructor].[id]

SQL Script

```
CREATE TABLE [dbo].[Track]
(
[id] [int] NOT NULL IDENTITY(1, 1),
[name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[track_mgr] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Track] ADD CONSTRAINT [PK__Track__3213E83FD5B44C38] PRIMARY KEY
```

```
CLUSTERED ([id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Track] ADD CONSTRAINT [FK__Track__track_mgr__75A278F5] FOREIGN KEY
([track_mgr]) REFERENCES [dbo].[Instructor] ([id])
GO
EXEC sp_addextendedproperty N'MS_Description', N'Track table defines each track by id,
name and the track manager', 'SCHEMA', N'dbo', 'TABLE', N'Track', NULL, NULL
GO
```

Uses

[dbo].[Instructor]

Used By

[dbo].[Branches_tracks]
[dbo].[Crs_track]
[dbo].[Instructors_in_track]
[dbo].[Student]
[dbo].[DeleteInstructor]
[dbo].[Insert_Student]
[dbo].[Sp_deleteTrack]
[dbo].[Sp_InsertCrsTrack]
[dbo].[Sp_insertTrack]
[dbo].[SP_r1GetStudentInTrack]
[dbo].[Sp_selectTrack]
[dbo].[Sp_updateTrack]

Stored Procedures

Objects

Name
dbo.Delete_Ins_Course
dbo.Delete_Instructor_in_Track
dbo.Delete_Stu_Crs
dbo.Delete_Student
dbo.DeleteInstructor
dbo.Insert_Ins_Course
dbo.Insert_Stu_Crs
dbo.Insert_Student
dbo.Select_Ins_Course
dbo.Select_Stu_Crs
dbo.Select_Student
dbo.SelectInstructor
dbo.SelectInstructorinTrack
dbo.SP_correctExam
dbo.Sp_deleteBranch
dbo.Sp_deleteBranchesTracks
dbo.Sp_deleteCourse
dbo.Sp_DeleteCrsTrack
dbo.SP_deleteExam
dbo.SP_deleteExamQuestions
dbo.SP_deleteQuestion
dbo.SP_deleteQuestionChoices
dbo.SP_deleteStudentExam
dbo.SP_deleteStudentExamQuestions
dbo.Sp_DeleteTopic
dbo.Sp_deleteTrack
dbo.SP_generateExam
dbo.Sp_InsertBranch
dbo.Sp_insertBranchesTracks
dbo.Sp_insertCourse
dbo.Sp_InsertCrsTrack
dbo.SP_insertExamQuestions
dbo.SP_insertInstructor
dbo.SP_insertInstructorInTrack

dbo.SP_insertQuestions
dbo.SP_insertStudentExam
dbo.SP_insertStudentExamQuestionAns
dbo.Sp_InsertTopic
dbo.Sp_insertTrack
dbo.SP_r1GetStudentInTrack
dbo.SP_r2GetStdGradeByStdId
dbo.SP_r3GetCoursesInsTeach
dbo.SP_r4GetTopicsInCourse
dbo.SP_r5QuestionAndChoicesinExam
dbo.SP_r6QuestionAndAnswersinExam
dbo.Sp_SelectBranch
dbo.Sp_SelectBranchesTracks
dbo.Sp_selectCourse
dbo.Sp_SelectCrsTrack
dbo.SP_selectExam
dbo.SP_selectExamQuestion
dbo.SP_selectQuestionChoices
dbo.SP_selectQuestions
dbo.SP_selectStudentExam
dbo.SP_selectStudentExamQuestions
dbo.Sp_SelectTopic
dbo.Sp_selectTrack
dbo.Sp_updateBranch
dbo.Sp_updateBranchesTracks
dbo.Sp_UpdateCourse
dbo.Sp_UpdateCrsTrack
dbo.SP_updateExam
dbo.SP_updateExamQuestions
dbo.SP_updateQuestionChoices
dbo.SP_updateQuestions
dbo.SP_updateStudentExam
dbo.SP_updateStudentExamQuestions
dbo.Sp_UpdateTopic
dbo.Sp_updateTrack
dbo.Update_Ins_Course
dbo.Update_Instructor_in_Track
dbo.Update_Stu_Crs
dbo.Update_Student
dbo.UpdateInstructor

[dbo].[Delete_Ins_Course]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROCEDURE [dbo].[Delete_Ins_Course]
    @instructor_id INT,
    @crs_id INT
AS
BEGIN
    IF EXISTS (SELECT 1 FROM dbo.Ins_Course WHERE ins_id = @instructor_id AND crs_id =
@crs_id)
    BEGIN
        DELETE FROM dbo.Ins_Course
        WHERE ins_id = @instructor_id AND crs_id = @crs_id;
    END
    ELSE
    BEGIN
        PRINT 'Instructor in Course record does not exist.';
    END
END;
GO
```

Uses

[dbo].[Ins_course]

[dbo].[Delete_Instructor_in_Track]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@instructor_id	int	4
@track_id	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[Delete_Instructor_in_Track]
    @instructor_id INT,
    @track_id INT
AS
BEGIN
    IF EXISTS (SELECT 1 FROM dbo.Instructor_in_Track WHERE ins_id = @instructor_id AND
track_id = @track_id)
    BEGIN
        DELETE FROM dbo.Instructor_in_Track
        WHERE ins_id = @instructor_id AND track_id = @track_id;
    END
    ELSE
    BEGIN
        PRINT 'Instructor in Track record does not exist.';
    END
END;
GO
```

[dbo].[Delete_Stu_Crs]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```sql
CREATE PROCEDURE [dbo].[Delete_Stu_Crs]
    @student_id INT = NULL,
    @crs_id INT = NULL
AS
BEGIN
    IF @student_id IS NOT NULL AND @crs_id IS NOT NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Student_crs WHERE stu_id = @student_id AND crs_id =
@crs_id)
        BEGIN
            DELETE FROM dbo.Student_crs
            WHERE stu_id = @student_id AND crs_id = @crs_id;

            PRINT 'Student-course record deleted successfully.';
        END
        ELSE
        BEGIN
            PRINT 'Record not found for the specified student and course.';
        END
    END

    ELSE IF @student_id IS NOT NULL AND @crs_id IS NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Student_crs WHERE stu_id = @student_id)
        BEGIN
            DELETE FROM dbo.Student_crs
            WHERE stu_id = @student_id;

            PRINT 'All courses for the student deleted successfully.';
        END
        ELSE
        BEGIN
            PRINT 'No courses found for the specified student.';
        END
    END
END;
GO
```

Uses

[dbo].[Student_crs]

[dbo].[Delete_Student]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROCEDURE [dbo].[Delete_Student]
    @id INT
AS
BEGIN
    IF NOT EXISTS (SELECT 1 FROM dbo.Student WHERE id = @id)
    BEGIN
        PRINT 'Student not found in the Student table.';
        RETURN;
    END

    IF EXISTS (SELECT 1 FROM dbo.Student_crs WHERE stu_id = @id)
    BEGIN
        DELETE FROM dbo.Student_crs WHERE stu_id = @id;
        PRINT 'Student record deleted from Student_crs table.';
    END
    ELSE
    BEGIN
        PRINT 'Student not found in Student_crs table.';
    END

    IF EXISTS (SELECT 1 FROM dbo.Student_exam WHERE stu_id = @id)
    BEGIN
        DELETE FROM dbo.Student_exam WHERE stu_id = @id;
        PRINT 'Student record deleted from Student_exam table.';
    END
    ELSE
    BEGIN
        PRINT 'Student not found in Student_exam table.';
    END
    IF EXISTS (SELECT 1 FROM dbo.Student_exam_questions WHERE stu_id = @id)
    BEGIN
        DELETE FROM dbo.Student_exam_questions WHERE stu_id = @id;
        PRINT 'Student record deleted from Student_exam_questions table.';
    END
    ELSE
    BEGIN
        PRINT 'Student not found in Student_exam_questions table.';
    END

    DELETE FROM dbo.Student WHERE id = @id;
    PRINT 'Student record deleted from Student table.';
END;
GO
```

Uses

[dbo].[Student]
[dbo].[Student_crs]
[dbo].[Student_exam]
[dbo].[Student_exam_questions]

[dbo].[DeleteInstructor]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROCEDURE [dbo].[DeleteInstructor]
    @id INT
AS
BEGIN
    IF EXISTS (SELECT 1 FROM dbo.Instructor WHERE id = @id)
    BEGIN
        delete from Instructors_in_track where ins_id = @id
        delete from Ins_course where ins_id = @id
        update Track set track_mgr = NULL
        DELETE FROM dbo.Instructor
        WHERE id = @id;
    END
    ELSE
    BEGIN
        PRINT 'Instructor with this ID does not exist.';
    END
END;
GO
```

Uses

[dbo].[Ins_course]
[dbo].[Instructor]
[dbo].[Instructors_in_track]
[dbo].[Track]

[dbo].[Insert_Ins_Course]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROCEDURE [dbo].[Insert_Ins_Course]
    @instructor_id INT,
    @crs_id INT
AS
BEGIN
    INSERT INTO dbo.Ins_Course (crs_id,ins_id)
    VALUES (@crs_id,@instructor_id )
END;
GO
```

Uses

[dbo].[Ins_course]

[dbo].[Insert_Stu_Crs]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROCEDURE [dbo].[Insert_Stu_Crs]
    @student_id INT,
    @crs_id INT
AS
BEGIN
    IF NOT EXISTS (SELECT 1 FROM dbo.Student WHERE id = @student_id)
    BEGIN
        PRINT 'Student does not exist in the Student table.';
        RETURN;
    END

    IF NOT EXISTS (SELECT 1 FROM dbo.Course WHERE id = @crs_id)
    BEGIN
        PRINT 'Course does not exist in the Course table.';
        RETURN;
    END

    INSERT INTO dbo.Student_crs (crs_id, stu_id)
    VALUES (@crs_id, @student_id);

    PRINT 'Student-course record inserted successfully.';
END;
GO
```

Uses

[dbo].[Course]

[dbo].[Student]

[dbo].[Student_crs]

[dbo].[Insert_Student]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@fname	varchar(50)	50
@lname	varchar(50)	50
@city	varchar(50)	50
@street	varchar(50)	50
@bdate	date	3
@phone	varchar(15)	15
@track_id	int	4

SQL Script

```sql
CREATE PROCEDURE [dbo].[Insert_Student]
    @fname VARCHAR(50),
    @lname VARCHAR(50),
    @city VARCHAR(50),
    @street VARCHAR(50),
    @bdate DATE,
    @phone VARCHAR(15),
    @track_id INT
AS
BEGIN
    IF EXISTS (SELECT 1 FROM dbo.Track WHERE id = @track_id)
    BEGIN
        INSERT INTO dbo.Student (fname, lname, city, street, bdate, phone, track_id)
        VALUES (@fname, @lname, @city, @street, @bdate, @phone, @track_id);
        PRINT 'Student inserted successfully.';
    END
    ELSE
    BEGIN
        PRINT 'The provided track_id does not exist in the Track table.';
    END
END;
GO
```

Uses

[dbo].[Student]

[dbo].[Track]

[dbo].[Select_Ins_Course]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROCEDURE [dbo].[Select_Ins_Course]
    @instructor_id INT = NULL,
    @crs_id INT = NULL
AS
BEGIN
    IF @instructor_id IS NOT NULL AND @crs_id IS NOT NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Ins_Course WHERE ins_id = @instructor_id AND crs_id
= @crs_id)
        BEGIN
            SELECT * FROM dbo.Ins_Course WHERE ins_id = @instructor_id AND crs_id =
@crs_id;
        END
        ELSE
        BEGIN
            PRINT 'Instructor in Course record does not exist for the given
combination.';
        END
    END
    ELSE IF @instructor_id IS NOT NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Ins_Course WHERE ins_id = @instructor_id)
        BEGIN
            SELECT * FROM dbo.Ins_Course WHERE ins_id = @instructor_id;
        END
        ELSE
        BEGIN
            PRINT 'Instructor record does not exist for the given instructor ID.';
        END
    END
    ELSE IF @crs_id IS NOT NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Ins_Course WHERE crs_id = @crs_id)
        BEGIN
            SELECT * FROM dbo.Ins_Course WHERE crs_id = @crs_id;
        END
        ELSE
        BEGIN
            PRINT 'Course record does not exist for the given course ID.';
        END
    END
    ELSE
    BEGIN
        PRINT 'Please provide either an instructor ID or a course ID to query.';
    END
END;
GO
```

Uses

[dbo].[Ins_course]

[dbo].[Select_Stu_Crs]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROCEDURE [dbo].[Select_Stu_Crs]
    @student_id INT = NULL,
    @crs_id INT = NULL
AS
BEGIN
    IF @student_id IS NOT NULL AND @crs_id IS NOT NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Student_crs WHERE stu_id = @student_id AND crs_id =
@crs_id)
        BEGIN
            SELECT *
            FROM dbo.Student_crs
            WHERE stu_id = @student_id AND crs_id = @crs_id;
        END
        ELSE
        BEGIN
            PRINT 'Record not found for the specified student and course.';
        END
    END
    ELSE IF @student_id IS NOT NULL AND @crs_id IS NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Student_crs WHERE stu_id = @student_id)
        BEGIN
            SELECT *
            FROM dbo.Student_crs
            WHERE stu_id = @student_id;
        END
        ELSE
        BEGIN
```

```
            PRINT 'No courses found for the specified student.';
        END
    END

    ELSE IF @crs_id IS NOT NULL AND @student_id IS NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Student_crs WHERE crs_id = @crs_id)
        BEGIN
            SELECT *
            FROM dbo.Student_crs
            WHERE crs_id = @crs_id;
        END
        ELSE
        BEGIN
            PRINT 'No students found for the specified course.';
        END
    END

    ELSE
    BEGIN
        PRINT 'Please provide at least one parameter: student_id or crs_id.';
    END
END;
GO
```

Uses

[dbo].[Student_crs]

[dbo].[Select_Student]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROCEDURE [dbo].[Select_Student]
    @id INT
AS
BEGIN
    IF NOT EXISTS (SELECT 1 FROM dbo.Student WHERE id = @id)
    BEGIN
        PRINT 'Student not found.';
        RETURN;
    END

    Select * FROM dbo.Student WHERE id = @id;
END;
GO
```

Uses

[dbo].[Student]

[dbo].[SelectInstructor]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROCEDURE [dbo].[SelectInstructor]
    @id INT
AS
BEGIN
    IF EXISTS (SELECT 1 FROM dbo.Instructor WHERE id = @id)
    BEGIN
        Select * from Instructor where id =@id
    END
    ELSE
    BEGIN
        PRINT 'Instructor with this ID does not exist.';
    END
END;
GO
```

Uses

[dbo].[Instructor]

[dbo].[SelectInstructorinTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@instructor_id	int	4
@track_id	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[SelectInstructorinTrack]
    @instructor_id INT = NULL,
    @track_id INT = NULL
AS
BEGIN
    IF @instructor_id IS NOT NULL AND @track_id IS NOT NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Instructors_in_track WHERE ins_id = @instructor_id
AND track_id = @track_id)
        BEGIN
            SELECT * FROM dbo.Instructors_in_track WHERE ins_id = @instructor_id AND
track_id = @track_id;
        END
        ELSE
        BEGIN
            PRINT 'Instructor in Track record does not exist for the given combination.';
        END
    END
    ELSE IF @instructor_id IS NOT NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Instructors_in_track WHERE ins_id = @instructor_id)
        BEGIN
            SELECT * FROM dbo.Instructors_in_track WHERE ins_id = @instructor_id;
        END
        ELSE
        BEGIN
            PRINT 'Instructor record does not exist for the given instructor ID.';
        END
    END
    ELSE IF @track_id IS NOT NULL
    BEGIN
        IF EXISTS (SELECT 1 FROM dbo.Instructors_in_track WHERE track_id = @track_id)
        BEGIN
            SELECT * FROM dbo.Instructors_in_track WHERE track_id = @track_id;
        END
        ELSE
        BEGIN
            PRINT 'Track record does not exist for the given track ID.';
        END
    END
    ELSE
    BEGIN
        PRINT 'Please provide either an instructor ID or a track ID to query.';
    END
END;
GO
```

Uses

[dbo].[Instructors_in_track]

[dbo].[SP_correctExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)	Direction
@stu_id	int	4	
@ex_id	int	4	
@examPercentage	float	8	Out

SQL Script

```
CREATE proc [dbo].[SP_correctExam]
    @stu_id int,
    @ex_id int,
    @examPercentage FLOAT OUTPUT
as
begin
    begin transaction;
    begin try
        if NOT EXISTS (select 1 from Student where id = @stu_id)
        begin
            select 'The specified student does not exist.'
        end

        if NOT EXISTS (select 1 from .Exam where id = @ex_id)
        begin
            select 'The specified exam does not exist.'
        end
        DECLARE @examGrade INT;
        select @examGrade = SUM(q.grade)
        FROM Student_exam_questions seq
        inner join Question q
            ON q.id = seq.q_id
        WHERE seq.stu_id = @stu_id
          and seq.ex_id = @ex_id
          and seq.stu_ans = q.model_ans;

        if @examGrade IS NULL
            set @examGrade = 0;
        DECLARE @maxGrade INT;
        SELECT @maxGrade = SUM(q.grade)
        FROM Question q inner join Exam_questions eq
        on q.id = eq.ex_id
        WHERE eq.ex_id = @ex_id;

        IF @maxGrade = 0
            SET @examPercentage = 0;
        ELSE
            SET @examPercentage = (@examGrade * 100.0) / @maxGrade;
        commit transaction;
    end try
    begin catch
        rollback transaction;
        throw;
    end catch
end;
GO
```

Uses

[dbo].[Exam]
[dbo].[Exam_questions]
[dbo].[Question]
[dbo].[Student]
[dbo].[Student_exam_questions]

[dbo].[Sp_deleteBranch]

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].[Sp_deleteBranch]
    @Id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Branch WHERE id = @Id)
        BEGIN
        -- delete first in table based on branch
            DELETE FROM Branches_tracks WHERE branch_id = @Id;
            --then delete from branch
         DELETE FROM Branch WHERE id = @Id;
        END
        ELSE
        BEGIN
            PRINT 'Branch does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while deleting the branch.';
    END CATCH
END;
GO
```

Uses

[dbo].[Branch]
[dbo].[Branches_tracks]

[dbo].[Sp_deleteBranchesTracks]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@BranchId	int	4
@TrackId	int	4

SQL Script

```
CREATE PROC [dbo].[Sp_deleteBranchesTracks]
    @BranchId INT,
    @TrackId INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Branches_tracks WHERE branch_id = @BranchId AND track_id
= @TrackId)
        BEGIN
            DELETE FROM Branches_tracks
            WHERE branch_id = @BranchId AND track_id = @TrackId;
        END
        ELSE
        BEGIN
            PRINT 'The specified Branch-Track relationship does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while deleting the Branch-Track relationship.';
    END CATCH
END;
GO
```

Uses

[dbo].[Branches_tracks]

[dbo].[Sp_deleteCourse]

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].[Sp_deleteCourse]
    @Id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Course WHERE id = @Id)
        BEGIN
        --added
         update Question SET crs_id = NULL
         DELETE FROM Student_crs where crs_id = @id
         --
         DELETE FROM Crs_track WHERE crs_id = @Id;
         DELETE FROM Course WHERE id = @Id;
         DELETE FROM Topic where crs_id = @Id
        END
        ELSE
        BEGIN
            PRINT 'Course does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while deleting the course.';
    END CATCH
END;
GO
```

Uses

[dbo].[Course]

[dbo].[Crs_track]
[dbo].[Question]
[dbo].[Student_crs]
[dbo].[Topic]

[dbo].[Sp_DeleteCrsTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```sql
CREATE PROC [dbo].[Sp_DeleteCrsTrack]
    @TrackId INT,
    @CrsId INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Crs_track WHERE track_id = @TrackId AND crs_id = @CrsId)
        BEGIN
            DELETE FROM Crs_track
            WHERE track_id = @TrackId AND crs_id = @CrsId;
        END
        ELSE
        BEGIN
            PRINT 'The record does not exist in Crs_track.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while deleting the record from Crs_track.';
    END CATCH
END;
GO
```

Uses

[dbo].[Crs_track]

[dbo].[SP_deleteExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```sql
create proc [dbo].[SP_deleteExam]
@examID int
as
begin
    IF EXISTS (SELECT 1 FROM Exam WHERE id = @examId)
    begin
        delete from Exam_questions where ex_id = @examID
        delete from Exam where id = @examID
    end
    else
    begin
        select 'This exam does not exist'
    end
end
GO
```

Uses

[dbo].[Exam]
[dbo].[Exam_questions]

[dbo].[SP_deleteExamQuestions]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@examID	int	4
@QuestionId	int	4

SQL Script

```
create proc [dbo].[SP_deleteExamQuestions]
@examID int ,
@QuestionId int
as
begin
    IF EXISTS (SELECT 1 FROM Exam_questions WHERE ex_id = @examID and q_id = @questionid)
    begin
        delete from Exam_questions where ex_id = @examID and q_id = @QuestionId
    end
    else
    begin
        select 'This question or this exam does not exist'
    end
end
GO
```

Uses

[dbo].[Exam_questions]

[dbo].[SP_deleteQuestion]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```sql
create proc [dbo].[SP_deleteQuestion]
@questionID int
as
begin
    IF Exists (select 1 from Question where id = @questionId)
    begin
        delete from Exam_questions where q_id = @questionID
        delete from Question_choices where Question_choices.q_id = @questionId
        delete from Question where id = @questionID
    end
    else
    begin
        select 'This Question does not in the system'
    end
end
GO
```

Uses

[dbo].[Exam_questions]
[dbo].[Question]
[dbo].[Question_choices]

[dbo].[SP_deleteQuestionChoices]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE proc [dbo].[SP_deleteQuestionChoices]
@questionID int
as
begin

    delete from Question_choices where Question_choices.q_id = @questionId
        delete from Question where id = @questionID
end
GO
```

Uses

[dbo].[Question]
[dbo].[Question_choices]

[dbo].[SP_deleteStudentExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@stu_id	int	4
@ex_id	int	4

SQL Script

```
create proc [dbo].[SP_deleteStudentExam]
@stu_id int,
@ex_id int
as
begin
    begin transaction
    begin try
        if NOT EXISTS (select 1 from Student_exam s where s.stu_id  = @stu_id)
        begin
            select 'The specified student does not exist.'
        end

        if NOT EXISTS (select 1 from Student_exam s where s.ex_id = @ex_id)
        begin
            select 'The specified exam does not exist.'
        end
        delete from Student_exam
        where stu_id = @stu_id and ex_id =@ex_id

    commit transaction;
    end try
    begin catch
        rollback transaction;
        throw;
    end catch
end

GO
```

Uses

[dbo].[Student_exam]

[dbo].[SP_deleteStudentExamQuestions]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@stu_id	int	4
@ex_id	int	4

SQL Script

```
create proc [dbo].[SP_deleteStudentExamQuestions]
@stu_id int , @ex_id int
as
begin
    begin transaction
    begin try
        IF NOT EXISTS (
            select 1
            from Student_exam_questions
            where ex_id = @ex_id and stu_id = @stu_id
        )
        begin
            select 'No records found in this table according to your data'
        end

        delete from Student_exam_questions
        where stu_id = @stu_id and ex_id = @ex_id

        commit transaction
        end try
        begin catch
            ROLLBACK TRANSACTION;
            THROW;
        end catch
end
GO
```

Uses

[dbo].[Student_exam_questions]

[dbo].[Sp_DeleteTopic]

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].[Sp_DeleteTopic]
    @Id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Topic WHERE id = @Id)
        BEGIN
            DELETE FROM Topic
            WHERE id = @Id;

            PRINT 'Topic successfully deleted.';
        END
        ELSE
        BEGIN
            PRINT 'Topic does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while deleting the topic.';
    END CATCH
END;
GO
```

Uses

[dbo].[Topic]

[dbo].[Sp_deleteTrack]

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].[Sp_deleteTrack]
    @Id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Track WHERE id = @Id)
        BEGIN
            DELETE FROM Crs_track WHERE track_id = @Id;
            DELETE FROM Branches_tracks WHERE track_id = @Id;
            DELETE FROM Track WHERE id = @Id;
        END
        ELSE
        BEGIN
            PRINT 'Track does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while deleting the Track.';
    END CATCH
END;
GO
```

Uses

[dbo].[Branches_tracks]
[dbo].[Crs_track]
[dbo].[Track]

[dbo].[SP_generateExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)	Direction
@exName	varchar(50)	50	
@numOfTF	int	4	
@numOFMCQ	int	4	
@exDate	date	3	
@exDuration	int	4	
@courseId	int	4	
@exId	int	4	Out

SQL Script

```
CREATE proc [dbo].[SP_generateExam]
    @exName varchar(50) , @numOfTF int , @numOFMCQ int ,
    @exDate date , @exDuration int , @courseId int , @exId int output
AS
begin
    BEGIN TRY
    insert into Exam (ex_date , duration , name , crs_id)
    values (@exDate , @exDuration , @exName , @courseId)
    set @exId = SCOPE_IDENTITY()

    insert into Exam_questions ( ex_id , q_id)
    select @exId , id
    from
    (
        select top (@numOfTF) id from Question where crs_id = @courseId
        and type = 'TF' order by NEWID()
        UNION ALL
        select top (@numOFMCQ) id from Question where crs_id = @courseId
        and type = 'MCQ' order by NEWID()
    ) as RandomQuestions
    END TRY
    BEGIN CATCH
```

```
        PRINT 'Error: ' + ERROR_MESSAGE();
        ROLLBACK TRANSACTION;
    END CATCH
END;
GO
```

Uses

[dbo].[Exam]
[dbo].[Exam_questions]
[dbo].[Question]

[dbo].[Sp_InsertBranch]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@Name	varchar(50)	50

SQL Script

```
CREATE PROC [dbo].[Sp_InsertBranch]
       @Name VARCHAR(50) = NULL
AS
BEGIN
    BEGIN TRY
        INSERT INTO Branch (name)
        VALUES (ISNULL(@Name, NULL));
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while inserting the branch.';
    END CATCH
END;
GO
```

Uses

[dbo].[Branch]

[dbo].[Sp_insertBranchesTracks]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@BranchId	int	4
@TrackId	int	4

SQL Script

```
CREATE PROC [dbo].[Sp_insertBranchesTracks]
       @BranchId INT=NULL ,
       @TrackId INT =NULL
AS
BEGIN
    BEGIN TRY
        -- If both branch_id and track_id are NULL
        IF @BranchId IS NULL OR @TrackId IS NULL
        BEGIN
            PRINT 'Both BranchId and TrackId are required.'; -- because of constraint of
not null on these colums
            RETURN;
        END
        IF NOT EXISTS (SELECT 1 FROM Branches_tracks WHERE branch_id = @BranchId AND
track_id = @TrackId)
        BEGIN
            INSERT INTO Branches_tracks (branch_id, track_id)
            VALUES (@BranchId, @TrackId);
        END
        ELSE
        BEGIN
            PRINT 'The record already exists in Branches_tracks.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while inserting the Branch-Track relationship.';
    END CATCH
END;
GO
```

Uses

[dbo].[Branches_tracks]

[dbo].[Sp_insertCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@Name	varchar(50)	50

SQL Script

```
CREATE PROC [dbo].[Sp_insertCourse]
    @Name VARCHAR(50) = NULL
AS
BEGIN
    BEGIN TRY
        INSERT INTO Course (name)
        VALUES (@Name);
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while inserting the course.';
    END CATCH
END;
GO
```

Uses

[dbo].[Course]

[dbo].[Sp_InsertCrsTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROC [dbo].[Sp_InsertCrsTrack]
    @TrackId INT = NULL,
    @CrsId INT = NULL
AS
BEGIN
    BEGIN TRY
        -- Check for NULL values
        IF @TrackId IS NULL OR @CrsId IS NULL
        BEGIN
            PRINT 'Both track_id and crs_id must be provided.';
        END
        ELSE
        BEGIN
            -- Check if the provided crs_id exists in the Course table
            IF NOT EXISTS (SELECT 1 FROM Course WHERE id = @CrsId)
            BEGIN
                PRINT 'The provided crs_id does not exist in the Course table.';
            END
            -- Check if the provided track_id exists in the Track table
            ELSE IF NOT EXISTS (SELECT 1 FROM Track WHERE id = @TrackId)
            BEGIN
                PRINT 'The provided track_id does not exist in the Track table.';
            END
            -- Check if the record already exists in Crs_track
            ELSE IF EXISTS (SELECT 1 FROM Crs_track WHERE track_id = @TrackId AND crs_id
= @CrsId)
            BEGIN
                PRINT 'The record already exists in Crs_track.';
            END
            ELSE
            BEGIN
                -- Insert the record into Crs_track
                INSERT INTO Crs_track (track_id, crs_id)
                VALUES (@TrackId, @CrsId);
                PRINT 'Record inserted successfully into Crs_track.';
            END
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while inserting the record into Crs_track.';
    END CATCH
END;
GO
```

Uses

[dbo].[Course]
[dbo].[Crs_track]
[dbo].[Track]

[dbo].[SP_insertExamQuestions]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@examId	int	4
@questionId	int	4

SQL Script

```
create proc [dbo].[SP_insertExamQuestions]
@examId int ,
@questionId int
as
begin
    insert into Exam_questions(ex_id , q_id)
    values (@examId , @questionId)
end
GO
```

Uses

[dbo].[Exam_questions]

[dbo].[SP_insertInstructor]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@fname	varchar(50)	50
@lname	varchar(50)	50
@city	varchar(50)	50
@street	varchar(100)	100
@bdate	date	3
@phone	varchar(25)	25
@salary	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[SP_insertInstructor]
    @fname VARCHAR(50),
    @lname VARCHAR(50),
    @city VARCHAR(50),
    @street VARCHAR(100),
    @bdate DATE,
    @phone VARCHAR(25),
    @salary INT
AS
BEGIN
    INSERT INTO Instructor (fname, lname, city, street, bdate, phone, salary)
    VALUES (@fname, @lname, @city, @street, @bdate, @phone, @salary)
END
GO
```

Uses

[dbo].[Instructor]

[dbo].[SP_insertInstructorInTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@instructor_id	int	4
@track_id	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[SP_insertInstructorInTrack]
    @instructor_id INT,
    @track_id INT
AS
BEGIN
    IF NOT EXISTS (SELECT 1 FROM dbo.Instructors_in_track WHERE track_id = @track_id AND
ins_id = @instructor_id)
    BEGIN
        INSERT INTO dbo.Instructors_in_track (track_id, ins_id)
        VALUES (@track_id, @instructor_id);
    END
    ELSE
    BEGIN
        PRINT 'This instructor is already assigned to this track.';
    END
END;
GO
```

Uses

[dbo].[Instructors_in_track]

[dbo].[SP_insertQuestions]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)	Direction
@questionHead	varchar(400)	400	
@modelAnswer	varchar(200)	200	
@grade	int	4	
@type	varchar(10)	10	
@courseId	int	4	
@choiceOne	varchar(200)	200	
@choiceTwo	varchar(200)	200	
@choiceThree	varchar(200)	200	
@choiceFour	varchar(200)	200	
@questionId	int	4	Out

SQL Script

```
create proc [dbo].[SP_insertQuestions]
@questionHead varchar(400),
@modelAnswer varchar(200),
@grade int = NULL,
@type varchar(10),
@courseId int,
@choiceOne varchar(200) = NULL,
@choiceTwo varchar(200) = NULL,
@choiceThree varchar(200) = NULL,
@choiceFour varchar(200) = NULL,
@questionId int output
as
begin
    IF Exists (select 1 from Course where id = @courseId)
    begin
        insert into Question (q_head , model_ans , grade , type ,crs_id)
        values (@questionHead , @modelAnswer ,ISNULL(@grade , 10) , @type , @courseId)
        set @questionId = SCOPE_IDENTITY()
        if @type = 'TF'
        begin
            insert into Question_choices (q_id , choice)
            values (@questionId , 'True') ,(@questionId , 'False')
        end
        else
            begin
            IF @choiceOne IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceOne);
            IF @choiceTwo IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceTwo);
            IF @choiceThree IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceThree);
            IF @choiceFour IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceFour);
        end
    end
    else
    begin
        select 'this course does not exit'
    end
end
GO
```

Uses

[dbo].[Course]
[dbo].[Question]
[dbo].[Question_choices]

[dbo].[SP_insertStudentExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@stu_id	int	4
@ex_id	int	4
@grade	int	4

SQL Script

```
create proc [dbo].[SP_insertStudentExam]
@stu_id int,
@ex_id int,
@grade int
as
begin
    begin transaction
    begin try
        if NOT EXISTS (select 1 from Student_exam_questions s where s.stu_id  = @stu_id)
        begin
            select 'The specified student does not exist.'
        end

        if NOT EXISTS (select 1 from Student_exam_questions s where s.ex_id = @ex_id)
        begin
            select 'The specified exam does not exist.'
        end
        insert into Student_exam (stu_id , ex_id , grade)
        values (@stu_id ,@ex_id ,@grade)

    commit transaction;
    end try
    begin catch
        rollback transaction;
        throw;
    end catch

end
```

```
GO
```

Uses

[dbo].[Student_exam]
[dbo].[Student_exam_questions]

[dbo].[SP_insertStudentExamQuestionAns]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@stu_id	int	4
@ex_id	int	4
@answers	AnswerTableType	max

SQL Script

```
CREATE proc [dbo].[SP_insertStudentExamQuestionAns]
@stu_id int , @ex_id int , @answers dbo.AnswerTableType READONLY
as
begin
    begin transaction
    begin try
    IF NOT EXISTS (SELECT 1 FROM Student s WHERE s.id = @stu_id)
        BEGIN
            THROW 50002, 'The specified student does not exist in the Student table.', 1;
        END
    IF NOT EXISTS (SELECT 1 FROM dbo.Exam WHERE id = @ex_id)
        BEGIN
            THROW 50003, 'The specified exam does not exist in the Exam table.', 1;
        END
    IF EXISTS (
            SELECT 1
            FROM @answers a
            LEFT JOIN Exam_questions eq
                ON eq.ex_id = @ex_id AND eq.q_id = a.q_id
            WHERE eq.q_id IS NULL
        )
        BEGIN
            THROW 50001, 'One or more questions do not belong to the specified exam.', 1;
        END
        INSERT INTO Student_exam_questions (stu_id, ex_id, q_id, stu_ans)
        SELECT @stu_id, @ex_id, a.q_id, a.stu_ans
        FROM @answers a;
        COMMIT TRANSACTION;
    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION;
        THROW;
    END CATCH
END;
GO
```

Uses

[dbo].[Exam]
[dbo].[Exam_questions]
[dbo].[Student]
[dbo].[Student_exam_questions]
[dbo].[AnswerTableType]

[dbo].[Sp_InsertTopic]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@Name	nvarchar(50)	100
@CrsId	int	4

SQL Script

```
CREATE PROC [dbo].[Sp_InsertTopic]
    @Name NVARCHAR(50),
    @CrsId INT = NULL
AS
BEGIN
    BEGIN TRY
        INSERT INTO Topic (name, crs_id)
        VALUES (@Name, @CrsId);

        PRINT 'Topic successfully inserted.';
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while inserting the topic.';
    END CATCH
END;
GO
```

Uses

[dbo].[Topic]

[dbo].[Sp_insertTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROC [dbo].[Sp_insertTrack]
       @Name VARCHAR(50) = NULL,  -- default val
       @TrackMgr INT = NULL
AS
BEGIN
    BEGIN TRY
        INSERT INTO Track (name, track_mgr)
        VALUES (ISNULL(@Name, NULL), ISNULL(@TrackMgr, NULL));
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while inserting the Track.';
    END CATCH
END;
GO
```

Uses

[dbo].[Track]

[dbo].[SP_r1GetStudentInTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE proc [dbo].[SP_r1GetStudentInTrack] @trk_id int
as
    select s.fname+' '+s.lname as fullName , s.city , s.bdate , s.phone
    from Student s , Track t
    where t.id = @trk_id and t.id = s.track_id

GO
```

Uses

[dbo].[Student]
[dbo].[Track]

[dbo].[SP_r2GetStdGradeByStdId]

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].[SP_r2GetStdGradeByStdId] @std_id int
as
    select s.fname+' '+s.lname as fullName ,c.name , se.grade
    from Course c , Student s , Student_exam se , Exam e
    where s.id= @std_id and  e.id = se.ex_id and s.id = se.stu_id and c.id = e.crs_id
GO
```

Uses

[dbo].[Course]
[dbo].[Exam]
[dbo].[Student]
[dbo].[Student_exam]

[dbo].[SP_r3GetCoursesInsTeach]

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].[SP_r3GetCoursesInsTeach] @ins_id int
as
    select  i.fname+' '+i.lname as fullName,c.name , count(stu_id) as numOfStudents
    from Instructor i , Course c , Ins_course ic , Student s, Student_crs sc
    where ic.ins_id =@ins_id and c.id = ic.crs_id and i.id = ic.ins_id and s.id =
sc.stu_id and c.id= sc.crs_id
    group by c.name , i.fname+' '+i.lname
GO
```

Uses

[dbo].[Course]
[dbo].[Ins_course]
[dbo].[Instructor]
[dbo].[Student]
[dbo].[Student_crs]

[dbo].[SP_r4GetTopicsInCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE proc [dbo].[SP_r4GetTopicsInCourse] @crs_id int
as
    select t.name  , c.name as crsName
    from Topic t, Course c
    where c.id=@crs_id  and c.id = t.crs_id
GO
```

Uses

[dbo].[Course]
[dbo].[Topic]

[dbo].[SP_r5QuestionAndChoicesinExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE proc [dbo].[SP_r5QuestionAndChoicesinExam]
@ExamID INT
AS
BEGIN
    SELECT
        c.name,
        eq.ex_id AS ExamID,
        q.id AS QuestionID,
        q.q_head AS QuestionText,
        STRING_AGG(qc.choice, ', ') AS Choices
    FROM
        Exam_Questions eq
    INNER JOIN
        Question q ON eq.q_id = q.id
    LEFT JOIN
        Question_Choices qc ON q.id = qc.q_id
    INNER JOIN
        Exam e ON e.id = eq.ex_id
    INNER JOIN
        Course c ON  c.id = e.crs_id
    WHERE
        eq.ex_id = @ExamID
    GROUP BY
        eq.ex_id, q.id, q.q_head , c.name
    ORDER BY
        q.id
END

GO
```

Uses

[dbo].[Course]
[dbo].[Exam]
[dbo].[Exam_questions]
[dbo].[Question]
[dbo].[Question_choices]

[dbo].[SP_r6QuestionAndAnswersinExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@ex_id	int	4
@stu_id	int	4

SQL Script

```sql
CREATE PROC [dbo].[SP_r6QuestionAndAnswersinExam]
@ex_id INT,
@stu_id INT
AS
BEGIN
    SELECT
        s.fname +' '+s.lname as fullName,
        q.id AS QuestionID,
        q.q_head AS QuestionText,
        q.model_ans AS ModelAnswer,
        seq.stu_ans AS StudentAnswer
    FROM
        Student s
    INNER JOIN
        Student_exam_questions seq ON s.id = seq.stu_id
    INNER JOIN
        Question q ON seq.q_id = q.id


    WHERE
        seq.ex_id = @ex_id
        AND seq.stu_id = @stu_id;
END
GO
```

Uses

[dbo].[Question]
[dbo].[Student]
[dbo].[Student_exam_questions]

[dbo].[Sp_SelectBranch]

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].[Sp_SelectBranch]
    @Id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Branch WHERE id = @Id)
        BEGIN
            SELECT *
            FROM Branch
            WHERE id = @Id;
        END
        ELSE
        BEGIN
            PRINT 'Branch does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while selecting the branch.';
    END CATCH
END;
GO
```

Uses

[dbo].[Branch]

[dbo].[Sp_SelectBranchesTracks]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@BranchId	int	4
@TrackId	int	4

SQL Script

```
CREATE PROC [dbo].[Sp_SelectBranchesTracks]
    @BranchId INT = NULL,
    @TrackId INT = NULL
as
BEGIN
    BEGIN TRY
        -- If both BranchId and TrackId are provided
        IF @BranchId IS NOT NULL AND @TrackId IS NOT NULL
        BEGIN
            SELECT branch_id, track_id
            FROM Branches_tracks
            WHERE branch_id = @BranchId AND track_id = @TrackId;
        END
        -- If only BranchId is provided
        ELSE IF @BranchId IS NOT NULL
        BEGIN
            SELECT branch_id, track_id
            FROM Branches_tracks
            WHERE branch_id = @BranchId;
        END
        -- If only TrackId is provided
        ELSE IF @TrackId IS NOT NULL
        BEGIN
            SELECT branch_id, track_id
            FROM Branches_tracks
            WHERE track_id = @TrackId;
        END
        -- If no parameters are provided
        ELSE
        BEGIN
            PRINT 'Please provide either BranchId or TrackId to perform the selection.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while selecting from Branches_tracks.';
    END CATCH
END;
GO
```

Uses

[dbo].[Branches_tracks]

[dbo].[Sp_selectCourse]

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].[Sp_selectCourse]
    @Id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Course WHERE id = @Id)
        BEGIN
            SELECT *
            FROM Course
            WHERE id = @Id;
        END
        ELSE
        BEGIN
            PRINT 'Course does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while selecting the course.';
    END CATCH
END;
GO
```

Uses

[dbo].[Course]

[dbo].[Sp_SelectCrsTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROC [dbo].[Sp_SelectCrsTrack]
    @TrackId INT = NULL,
    @CrsId INT = NULL
AS
BEGIN
    BEGIN TRY
        IF @TrackId IS NOT NULL AND @CrsId IS NOT NULL
        BEGIN
            SELECT track_id, crs_id
            FROM Crs_track
            WHERE track_id = @TrackId AND crs_id = @CrsId;
        END
        -- If only TrackId is provided
        ELSE IF @TrackId IS NOT NULL
        BEGIN
            SELECT track_id, crs_id
            FROM Crs_track
            WHERE track_id = @TrackId;
        END
        -- If only CrsId is provided
        ELSE IF @CrsId IS NOT NULL
        BEGIN
            SELECT track_id, crs_id
            FROM Crs_track
            WHERE crs_id = @CrsId;
        END
        -- If no parameters are provided,
        ELSE
         BEGIN
```

```
            PRINT 'Please provide either TrackId or CrsId to perform the selection.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while selecting the records from Crs_track.';
    END CATCH
END;
GO
```

Uses

[dbo].[Crs_track]

[dbo].[SP_selectExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROC [dbo].[SP_selectExam]
    @ex_id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Exam WHERE id = @ex_id)
        BEGIN
            SELECT * FROM Exam WHERE id = @ex_id;
        END
        ELSE
        BEGIN
            PRINT 'This Exam Does not Exist';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while selecting the exam.';
    END CATCH
END
GO
```

Uses

[dbo].[Exam]

[dbo].[SP_selectExamQuestion]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROC [dbo].[SP_selectExamQuestion]
    @ex_id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Exam_questions WHERE ex_id = @ex_id)
        BEGIN
            SELECT
                eq.q_id, q.q_head, q.model_ans
            FROM
                Exam_questions eq INNER JOIN Question q
            ON eq.q_id = q.id
            WHERE eq.ex_id = @ex_id;
        END
        ELSE
        BEGIN
            PRINT 'No questions found for the given exam ID.';
        END
    END TRY
    BEGIN CATCH
        -- Handle any errors that occur
        PRINT 'An error occurred while selecting the exam questions.';
    END CATCH
END;
GO
```

Uses

[dbo].[Exam_questions]

[dbo].[Question]

[dbo].[SP_selectQuestionChoices]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create PROC [dbo].[SP_selectQuestionChoices]
    @q_id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Question WHERE id = @q_id)
        BEGIN
             SELECT *
            FROM Question_choices
            WHERE q_id = @q_id;
        END
        ELSE
        BEGIN
            PRINT 'This question does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while retrieving the question.';
    END CATCH
END
GO
```

Uses

[dbo].[Question]
[dbo].[Question_choices]

[dbo].[SP_selectQuestions]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROC [dbo].[SP_selectQuestions]
    @q_id INT
AS
BEGIN
    BEGIN TRY
        -- Check if the question exists
        IF EXISTS (SELECT 1 FROM Question WHERE id = @q_id)
        BEGIN
            SELECT *
            FROM Question
            WHERE id = @q_id;

             SELECT *
            FROM Question_choices
            WHERE q_id = @q_id;
        END
        ELSE
        BEGIN
            PRINT 'This question does not exist.';
        END
    END TRY
    BEGIN CATCH
        -- Handle any errors that occur
        PRINT 'An error occurred while retrieving the question.';
    END CATCH
END
GO
```

Uses

[dbo].[Question]

[dbo].[Question_choices]

[dbo].[SP_selectStudentExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@stu_id	int	4
@ex_id	int	4

SQL Script

```
CREATE PROC [dbo].[SP_selectStudentExam]
    @stu_id INT = NULL,
    @ex_id INT = NULL
AS
BEGIN
    BEGIN TRY
        IF @stu_id IS NOT NULL AND @ex_id IS NOT NULL
        BEGIN
            IF EXISTS (SELECT 1 FROM Student_exam WHERE stu_id = @stu_id AND ex_id =
@ex_id)
            BEGIN
                SELECT *
                FROM Student_exam
                WHERE stu_id = @stu_id AND ex_id = @ex_id;
            END
            ELSE
            BEGIN
                PRINT 'This data does not exist.';
            END
        END
        ELSE IF @stu_id IS NOT NULL
        BEGIN
            IF EXISTS (SELECT 1 FROM Student_exam WHERE stu_id = @stu_id)
            BEGIN
                SELECT *
                FROM Student_exam
                WHERE stu_id = @stu_id;
            END
            ELSE
            BEGIN
                PRINT 'No records found for the specified student.';
            END
        END
        ELSE IF @ex_id IS NOT NULL
        BEGIN
            IF EXISTS (SELECT 1 FROM Student_exam WHERE ex_id = @ex_id)
            BEGIN
                SELECT *
                FROM Student_exam
                WHERE ex_id = @ex_id;
            END
            ELSE
            BEGIN
                PRINT 'No records found for the specified exam.';
            END
        END
        ELSE
        BEGIN
            PRINT 'Please provide at least one parameter: @stu_id or @ex_id.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while retrieving the student exam record.';
    END CATCH
END
GO
```

Uses

[dbo].[Student_exam]

[dbo].[SP_selectStudentExamQuestions]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@stu_id	int	4
@ex_id	int	4

SQL Script

```
CREATE PROC [dbo].[SP_selectStudentExamQuestions]
    @stu_id INT,
    @ex_id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Student_exam_questions WHERE stu_id = @stu_id AND ex_id
= @ex_id)
        BEGIN
            SELECT *
            FROM Student_exam_questions
            WHERE stu_id = @stu_id AND ex_id = @ex_id;
        END
        ELSE
        BEGIN
            PRINT 'This data does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while retrieving student exam questions.';
    END CATCH
END;
GO
```

Uses

[dbo].[Student_exam_questions]

[dbo].[Sp_SelectTopic]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROC [dbo].[Sp_SelectTopic]
    @Id INT = NULL,
    @CrsId INT = NULL
AS
BEGIN
    BEGIN TRY
        -- If both Id and CrsId are provided
        IF @Id IS NOT NULL AND @CrsId IS NOT NULL
        BEGIN
            SELECT id, name, crs_id
            FROM Topic
            WHERE id = @Id AND crs_id = @CrsId;
        END
        -- If only Id is provided
        ELSE IF @Id IS NOT NULL
        BEGIN
            SELECT id, name, crs_id
            FROM Topic
            WHERE id = @Id;
        END
        -- If only CrsId is provided
        ELSE IF @CrsId IS NOT NULL
        BEGIN
            SELECT id, name, crs_id
            FROM Topic
            WHERE crs_id = @CrsId;
        END
        -- If neither Id nor CrsId is provided
        ELSE
        BEGIN
            PRINT 'Please provide either Id or CrsId to perform the selection.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while selecting topics.';
    END CATCH
END;
GO
```

Uses

[dbo].[Topic]

[dbo].[Sp_selectTrack]

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].[Sp_selectTrack]
    @Id INT
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Track WHERE id = @Id)
        BEGIN
            SELECT id, name, track_mgr
            FROM Track
            WHERE id = @Id;
        END
        ELSE
        BEGIN
            PRINT 'Track does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while selecting the Track.';
    END CATCH
END;
GO
```

Uses

[dbo].[Track]

[dbo].[Sp_updateBranch]

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

SQL Script

```
create PROC [dbo].[Sp_updateBranch]
    @Id INT,
    @Name VARCHAR(50) = NULL
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Branch WHERE id = @Id)
        BEGIN
            UPDATE Branch
            SET name = ISNULL(@Name, name)
            WHERE id = @Id;

            -- Update related records in Branches_Tracks اصلا مفيش بس
            UPDATE Branches_tracks
            SET branch_id = @Id
            WHERE branch_id = @Id; -- No change in ID, but ensures related records are
consistent  بس بيتاكد
        END
        ELSE
        BEGIN
            PRINT 'Branch does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while updating the branch.';
    END CATCH
END;
GO
```

Uses

[dbo].[Branch]
[dbo].[Branches_tracks]

[dbo].[Sp_updateBranchesTracks]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@OldBranchId	int	4
@OldTrackId	int	4
@NewBranchId	int	4
@NewTrackId	int	4

SQL Script

```
CREATE PROC [dbo].[Sp_updateBranchesTracks]
    @OldBranchId INT,
    @OldTrackId INT,
    @NewBranchId INT = NULL,
    @NewTrackId INT = NULL
AS
BEGIN
    BEGIN TRY
        -- Ensure the old pair exists
        IF EXISTS (SELECT 1 FROM Branches_tracks WHERE branch_id = @OldBranchId AND
track_id = @OldTrackId)
        BEGIN
            UPDATE Branches_tracks
            SET branch_id = ISNULL(@NewBranchId, branch_id),
                track_id = ISNULL(@NewTrackId, track_id)
            WHERE branch_id = @OldBranchId AND track_id = @OldTrackId;
        END
        ELSE
        BEGIN
            PRINT 'The specified Branch-Track relationship does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while updating the Branch-Track relationship.';
    END CATCH
END;
GO
```

Uses

[dbo].[Branches_tracks]

[dbo].[Sp_UpdateCourse]

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

SQL Script

```
create PROC [dbo].[Sp_UpdateCourse]
    @Id INT,
    @Name VARCHAR(50) = NULL
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Course WHERE id = @Id)
        BEGIN
            UPDATE Course
            SET name = ISNULL(@Name, name)
            WHERE id = @Id;

            -- Update related records in Topic
            UPDATE Topic
            SET crs_id = @Id
            WHERE crs_id = @Id;

            -- Update related records in Crs_Track
            UPDATE Crs_track
            SET crs_id = @Id
            WHERE crs_id = @Id;
        END
        ELSE
        BEGIN
            PRINT 'Course does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while updating the course.';
    END CATCH
END;
GO
```

Uses

[dbo].[Course]
[dbo].[Crs_track]
[dbo].[Topic]

[dbo].[Sp_UpdateCrsTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```sql
CREATE PROC [dbo].[Sp_UpdateCrsTrack]
    @TrackId INT = NULL,
    @CrsId INT = NULL
AS
BEGIN
    BEGIN TRY
       --return an error message
        IF @TrackId IS NULL OR @CrsId IS NULL
        BEGIN
            PRINT 'Both track_id and crs_id must be provided.';
        END
        ELSE
        BEGIN
            IF EXISTS (SELECT 1 FROM Crs_track WHERE track_id = @TrackId AND crs_id =
@CrsId)
            BEGIN
                UPDATE Crs_track
                SET track_id = @TrackId, crs_id = @CrsId
                WHERE track_id = @TrackId AND crs_id = @CrsId;
            END
            ELSE
            BEGIN
                PRINT 'The record does not exist in Crs_track.';
            END
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while updating the record in Crs_track.';
    END CATCH
```

```
END;
GO
```

Uses

[dbo].[Crs_track]

[dbo].[SP_updateExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@examId	int	4
@examName	nvarchar(50)	100
@examDate	date	3
@examDuration	int	4
@crsId	int	4

SQL Script

```
CREATE Procedure [dbo].[SP_updateExam]
    @examId INT, @examName NVARCHAR(50) = NULL ,
    @examDate DATE = NULL, @examDuration INT = NULL, @crsId INT = NULL
as
begin
        IF EXISTS (SELECT 1 FROM Exam WHERE id = @examId)
        begin
            update Exam set
            name = ISNULL(@examName, Exam.name),
            ex_date = ISNULL(@examDate, ex_date),
            duration = ISNULL(@examDuration, duration),
            crs_id = ISNULL(@crsId, crs_id)
            where Exam.id = @examId
        end
        else
        begin
            select 'Exam does not exist'
        end
end
GO
```

Uses

[dbo].[Exam]

[dbo].[SP_updateExamQuestions]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@examid	int	4
@questionId	int	4
@newQuestionId	int	4

SQL Script

```
create proc [dbo].[SP_updateExamQuestions]
@examid int , @questionId int , @newQuestionId int
as
begin
    IF EXISTS (SELECT 1 FROM Exam_questions WHERE ex_id = @examId and q_id =
@questionid )
    begin
        update Exam_questions
        set q_id = @newQuestionId
         WHERE ex_id = @examId and q_id = @questionid
    end
    else
    begin
        select 'Enter real data'
    end
end
GO
```

Uses

[dbo].[Exam_questions]

[dbo].[SP_updateQuestionChoices]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@questionId	int	4
@choiceOne	varchar(200)	200
@choiceTwo	varchar(200)	200
@choiceThree	varchar(200)	200
@choiceFour	varchar(200)	200

SQL Script

```
create proc [dbo].[SP_updateQuestionChoices]
    @questionId INT,
    @choiceOne varchar(200) = NULL,
    @choiceTwo varchar(200) = NULL,
    @choiceThree varchar(200) = NULL,
    @choiceFour varchar(200) = NULL
as
BEGIN
    begin TRANSACTION;
    begin TRY
        -- Check if the question exists
        IF EXISTS (SELECT 1 FROM Question WHERE id = @questionId)
        begin
            -- Delete existing choices for the question
            DELETE FROM Question_choices WHERE q_id = @questionId;

            -- Insert new choices if provided
            IF @choiceOne IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceOne);
            IF @choiceTwo IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceTwo);
            IF @choiceThree IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceThree);
```

```
            IF @choiceFour IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceFour);
        end
        ELSE
        BEGIN
            SELECT 'Question does not exist'
            ROLLBACK TRANSACTION;
            RETURN;
        end

        COMMIT TRANSACTION;
    end TRY
    begin CATCH
        ROLLBACK TRANSACTION;
        PRINT 'An error occurred: ' + ERROR_MESSAGE();
    end CATCH
end;
GO
```

Uses

[dbo].[Question]
[dbo].[Question_choices]

[dbo].[SP_updateQuestions]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@questionId	int	4
@modelAnswer	varchar(200)	200
@grade	int	4
@type	varchar(10)	10
@courseId	int	4
@choiceOne	varchar(200)	200
@choiceTwo	varchar(200)	200
@choiceThree	varchar(200)	200
@choiceFour	varchar(200)	200

SQL Script

```
CREATE proc [dbo].[SP_updateQuestions]
@questionId int,
@modelAnswer varchar(200) = NULL,
@grade int = NULL,
@type varchar(10)= NULL,
@courseId int = NULL,
@choiceOne varchar(200) = NULL,
@choiceTwo varchar(200) = NULL,
@choiceThree varchar(200) = NULL,
@choiceFour varchar(200) = NULL
as
begin
    BEGIN TRANSACTION;
    begin try
    IF Exists (select 1 from Question where id = @questionId)
    begin
        update Question
        set model_ans = ISNULL(@modelAnswer , model_ans) ,
            grade = ISNULL(@grade , grade),
            type = ISNULL(@type , type) ,
            crs_id = ISNULL(@courseId , crs_id)
        where id = @questionId
        --update choices
        if @type = 'TF'
        begin
            delete from Question_choices where q_id = @questionId
            insert into Question_choices (q_id , choice)
            values (@questionId , 'True') ,(@questionId , 'False')
        end
        else
        begin
            DELETE FROM Question_choices WHERE q_id = @questionId
            IF @choiceOne IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceOne)
            IF @choiceTwo IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceTwo)
            IF @choiceThree IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceThree)
            IF @choiceFour IS NOT NULL
                INSERT INTO Question_choices (q_id, choice) VALUES (@questionId,
@choiceFour)
        end
    end
    else
    begin
        SELECT 'Question does not exist' AS ErrorMessage;
        ROLLBACK TRANSACTION;
    end
    COMMIT TRANSACTION;
end try
BEGIN CATCH
        -- Handle any errors and rollback the transaction
        ROLLBACK TRANSACTION;
        PRINT 'An error occurred: ' + ERROR_MESSAGE();
    END CATCH
end
GO
```

Uses

[dbo].[Question]
[dbo].[Question_choices]

[dbo].[SP_updateStudentExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@stu_id	int	4
@ex_id	int	4
@newGrade	int	4

SQL Script

```
CREATE proc [dbo].[SP_updateStudentExam]
@stu_id int,
@ex_id int,
@newGrade int
as
begin
    begin transaction
    begin try
        if NOT EXISTS (select 1 from Student_exam s where s.stu_id  = @stu_id)
        begin
            select 'The specified student does not exist.'
        end

        if NOT EXISTS (select 1 from Student_exam s where s.ex_id = @ex_id)
        begin
            select 'The specified exam does not exist.'
        end
        update Student_exam
        set grade = @newGrade
        where stu_id = @stu_id and ex_id = @ex_id

    commit transaction;
    end try
    begin catch
        rollback transaction;
        throw;
    end catch
end
```

```
GO
```

Uses

[dbo].[Student_exam]

[dbo].[SP_updateStudentExamQuestions]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@stu_id	int	4
@ex_id	int	4
@q_id	int	4
@new_ans	varchar(200)	200

SQL Script

```sql
create proc [dbo].[SP_updateStudentExamQuestions]
@stu_id int , @ex_id int , @q_id int , @new_ans varchar(200)
as
begin
    begin transaction
    begin try
        IF NOT EXISTS (
                select 1
                from Student_exam_questions
                where ex_id = @ex_id and stu_id = @stu_id
            )
        begin
            select 'No records found in this table according to your data'
        end
        update Student_exam_questions
        set stu_ans = @new_ans
        where stu_id = @stu_id and ex_id = @ex_id and q_id=@q_id

        commit transaction
    end try
    begin catch
        ROLLBACK TRANSACTION;
        THROW;
    end catch
end
GO
```

Uses

[dbo].[Student_exam_questions]

[dbo].[Sp_UpdateTopic]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@Id	int	4
@Name	nvarchar(50)	100
@CrsId	int	4

SQL Script

```
CREATE PROC [dbo].[Sp_UpdateTopic]
    @Id INT,
    @Name NVARCHAR(50) = NULL,
    @CrsId INT = NULL
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Topic WHERE id = @Id)
        BEGIN
            UPDATE Topic
            SET
                name = COALESCE(@Name, name),
                crs_id = @CrsId
            WHERE id = @Id;

            PRINT 'Topic successfully updated.';
        END
        ELSE
        BEGIN
            PRINT 'Topic does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while updating the topic.';
    END CATCH
END;
GO
```

Uses

[dbo].[Topic]

[dbo].[Sp_updateTrack]

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
@TrackMgr	int	4

SQL Script

```sql
create PROC [dbo].[Sp_updateTrack]
    @Id INT,
    @Name VARCHAR(50) = NULL,
    @TrackMgr INT = NULL
AS
BEGIN
    BEGIN TRY
        IF EXISTS (SELECT 1 FROM Track WHERE id = @Id)
        BEGIN
            UPDATE Track
            SET name = ISNULL(@Name, name),
                track_mgr = ISNULL(@TrackMgr, track_mgr)
            WHERE id = @Id;

            -- Update related records in Branches_Tracks >>   مفيش قيم بس مجرد بناكد بس
انو مزال ال id ذي ماهووو
            UPDATE Branches_tracks
            SET track_id = @Id
            WHERE track_id = @Id;

            -- Update related records in Crs_Track
            UPDATE Crs_track
            SET track_id = @Id
            WHERE track_id = @Id;
        END
        ELSE
        BEGIN
            PRINT 'Track does not exist.';
        END
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred while updating the track.';
    END CATCH
END;
GO
```

Uses

[dbo].[Branches_tracks]
[dbo].[Crs_track]
[dbo].[Track]

[dbo].[Update_Ins_Course]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE PROCEDURE [dbo].[Update_Ins_Course]
    @instructor_id INT = NULL,
    @crs_id INT = NULL,
    @newCrsId int
AS
BEGIN
    IF EXISTS (SELECT 1 FROM dbo.Ins_Course WHERE ins_id = @instructor_id AND crs_id =
@crs_id) and
     EXISTS (SELECT 1 FROM dbo.Course WHERE id = @newCrsId)
    BEGIN
        UPDATE dbo.Ins_Course
        SET crs_id = ISNULL(@newCrsId , crs_id)
        WHERE ins_id = @instructor_id and crs_id = @crs_id;
    END
    ELSE
    BEGIN
        PRINT 'Instructor in Course record does not exist.';
    END
END;
GO
```

Uses

[dbo].[Course]
[dbo].[Ins_course]

[dbo].[Update_Instructor_in_Track]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@instructor_id	int	4
@track_id	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[Update_Instructor_in_Track]
    @instructor_id INT,
    @track_id INT
AS
BEGIN
    IF EXISTS (SELECT 1 FROM dbo.Instructor_in_Track WHERE ins_id = @instructor_id AND
track_id = @track_id)
    BEGIN
        UPDATE dbo.Instructor_in_Track
        SET track_id = @track_id
        WHERE ins_id = @instructor_id;
    END
    ELSE
    BEGIN
        PRINT 'Instructor in Track record does not exist.';
    END
END;
GO
```

[dbo].[Update_Stu_Crs]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@student_id	int	4
@old_crs_id	int	4
@new_crs_id	int	4

SQL Script

```
CREATE PROCEDURE [dbo].[Update_Stu_Crs]
    @student_id INT,
    @old_crs_id INT,
    @new_crs_id INT
AS
BEGIN
    IF EXISTS (SELECT 1 FROM dbo.Student_crs WHERE stu_id = @student_id AND crs_id =
@old_crs_id)
    AND EXISTS (SELECT 1 FROM dbo.Course WHERE id = @new_crs_id)
    BEGIN
        UPDATE dbo.Student_crs
        SET crs_id = @new_crs_id
        WHERE stu_id = @student_id AND crs_id = @old_crs_id;
    END
END;
GO
```

Uses

[dbo].[Course]
[dbo].[Student_crs]

[dbo].[Update_Student]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@fname	varchar(50)	50
@lname	varchar(50)	50
@city	varchar(50)	50
@street	varchar(50)	50
@bdate	date	3
@phone	varchar(15)	15
@track_id	int	4

SQL Script

```sql
CREATE PROCEDURE [dbo].[Update_Student]
    @id INT,
    @fname VARCHAR(50) = NULL,
    @lname VARCHAR(50) = NULL,
    @city VARCHAR(50) = NULL,
    @street VARCHAR(50) = NULL,
    @bdate DATE = NULL,
    @phone VARCHAR(15) = NULL,
    @track_id INT = NULL
AS
BEGIN
    IF NOT EXISTS (SELECT 1 FROM dbo.Student WHERE id = @id)
    BEGIN
        PRINT 'Student not found.';
        RETURN;
    END

    UPDATE dbo.Student
    SET
        fname = COALESCE(@fname, fname),
        lname = COALESCE(@lname, lname),
        city = COALESCE(@city, city),
        street = COALESCE(@street, street),
        bdate = COALESCE(@bdate, bdate),
        phone = COALESCE(@phone, phone),
        track_id = COALESCE(@track_id, track_id)
    WHERE id = @id;

    PRINT 'Student information updated successfully.';
END;
GO
```

Uses

[dbo].[Student]

[dbo].[UpdateInstructor]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@id	int	4
@fname	varchar(50)	50
@lname	varchar(50)	50
@bdate	date	3
@phone	varchar(15)	15
@city	varchar(50)	50
@street	varchar(100)	100

SQL Script

```
CREATE PROCEDURE [dbo].[UpdateInstructor]
    @id INT,
    @fname VARCHAR(50) = NULL,
    @lname VARCHAR(50) = NULL,
    @bdate DATE = NULL,
    @phone VARCHAR(15) = NULL,
    @city varchar(50) = NULL,
    @street varchar(100)=NULL
AS
BEGIN
    IF EXISTS (SELECT 1 FROM dbo.Instructor WHERE id = @id)
    BEGIN
        UPDATE dbo.Instructor
        SET
            fname = ISNULL(@fname, fname),
            lname = ISNULL(@lname, lname),
            bdate = ISNULL(@bdate, bdate),
            phone = ISNULL(@phone, phone),
            city = ISNULL(@city , city),
            street = ISNULL(@street,street)
        WHERE id = @id;
    END
    ELSE
    BEGIN
        PRINT 'Instructor with this ID does not exist.';
    END
END;
GO
```

Uses

[dbo].[Instructor]

User-Defined Table Types

Objects

Name
dbo.AnswerTableType

[dbo].[AnswerTableType]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Heap	True

Columns

Name	Data Type	Max Length (Bytes)	Nullability
q_id	int	4	NULL allowed
stu_ans	varchar(200)	200	NULL allowed

SQL Script

```
CREATE TYPE [dbo].[AnswerTableType] AS TABLE
(
[q_id] [int] NULL,
[stu_ans] [varchar] (200) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
)
GO
```

Used By

[dbo].[SP_insertStudentExamQuestionAns]

Users

Objects

Name
dbo
guest

dbo

Properties

Property	Value
Type	WindowsUser
Login Name	LAPTOP-D5LRBLH1\Salem
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
```

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