. Documentation

ITIProject

Server	.
Author	SoftLaptop
Created	Thursday, December 12, 2024 8:46:29 PM
File Path	D:\ITI\DataBase\Project\Examination System Documentation-2024-12-12T20-46-29.pdf

Table of Contents

.

Databases (1)

- ITIProject

Server Properties

Property	Value
Product	Microsoft SQL Server
Version	16.0.1135.2
Language	English (United States)
Platform	NT x64
Edition	Developer Edition (64-bit)
Engine Edition	3 (Enterprise)
Processors	8
OS Version	6.3 (19045)
Physical Memory	16299
Is Clustered	False
Root Directory	C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL
Collation	SQL_Latin1_General_CP1_CI_AS

Server Settings

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

Advanced Server Settings

Property	Value
Locks	0
Nested triggers enabled	True
Allow triggers to fire others	True
Default language	English

Network packet size	4096
Default fulltext language LCID	1033
Two-digit year cutoff	2049
Remote login timeout	10
Cursor threshold	-1
Max text replication size	65536
Parallelism cost threshold	5
Max degree of parallelism	8
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)

- ITIProject

ITIProject 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 9 2024
Users	4
Database Encryption Enabled	False
Database Encryption Algorithm	None
Database size	16.00 MB
Unallocated space	0.23 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	FULL
Page verify option	CHECKSUM
Auto create statistics	True
Auto update statistics	True
Auto update statistics asynchronously	False
ANSI NULL default	False
ANSI NULL enabled	False
ANSI padding enabled	False

ANSI warnings enabled	False
Arithmetic abort enabled	False
Concatenating NULL yields NULL	False
Numeric roundabort enabled	False
Quoted Identifier On	False
Recursive triggers enabled	False
Close cursors on commit	False
Local cursors by default	False
Fulltext enabled	True
Trustworthy	False
Database chaining	False
Forced parameterization	False
Master key encrypted by server	False
Published	False
Subscribed	False
Merge published	False
Is distribution database	False
Sync with backup	False
Service broker GUID	7d42fd1c-ea60-477f-a2e7-f2705434df48
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	DESKTOP-2Q5RJ97\SoftLaptop

Files

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

Tables

Objects

Name
dbo.Br_Track
dbo.Branch
dbo.Course
dbo.Ex_question
dbo.Exam
dbo.Instructor
dbo.Ques_choice
dbo.Question
dbo.St_answer
dbo.St_course
dbo.St_exam
dbo.Student
dbo.Topic
dbo.Track

[dbo].[Br_Track]

Properties

Property	Value
Row Count (~)	8
Created	9:00:45 PM Wednesday, December 4, 2024
Last Modified	11:31:06 PM Monday, December 9, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
	Br_id	int	4	NOT NULL
	Tr_id	int	4	NOT NULL

Indexes

Key	Name	Key Columns	Unique
	PK_Br_Track	Br_id, Tr_id	True

Foreign Keys

Name	Columns
FK_Br_Track_Branch	Br_id->[dbo].[Branch].[Br_id]
FK_Br_Track_Track	Tr_id->[dbo].[Track].[Tr_id]

SQL Script

```
CREATE TABLE [dbo].[Br_Track]
(
[Br_id] [int] NOT NULL,
[Tr_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Br_Track] ADD CONSTRAINT [PK_Br_Track] PRIMARY KEY CLUSTERED ([Br_id],
[Tr_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Br_Track] ADD CONSTRAINT [FK_Br_Track_Branch] FOREIGN KEY ([Br_id]) REFERENCES
[dbo].[Branch] ([Br_id])
GO
ALTER TABLE [dbo].[Br_Track] ADD CONSTRAINT [FK_Br_Track_Track] FOREIGN KEY ([Tr_id]) REFERENCES
[dbo].[Track] ([Tr_id])
```

```
GO
```

Uses

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

Used By

[dbo].[add_br_track]
[dbo].[delete_br_track]
[dbo].[delete_branch]
[dbo].[get_all_br_tracks]
[dbo].[update_br_track]

[dbo].[Branch]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	10
Created	1:15:38 AM Saturday, December 7, 2024
Last Modified	1:15:39 AM Saturday, December 7, 2024

Columns

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

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Branch	Br_id	True

SQL Script

```
CREATE TABLE [dbo].[Branch]
(
[Br_id] [int] NOT NULL IDENTITY(1, 1),
[Br_name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[Br_phone] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Branch] ADD CONSTRAINT [PK_Branch] PRIMARY KEY CLUSTERED ([Br_id]) ON
[PRIMARY]
GO
```

Used By

[dbo].[Br_Track]
[dbo].[add_br_track]
[dbo].[add_branch]
[dbo].[delete_branch]

[dbo].[get_all_branches]
[dbo].[update_br_track]
[dbo].[update_branch]

[dbo].[Course]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	10
Created	11:28:22 PM Monday, December 9, 2024
Last Modified	11:32:41 PM Monday, December 9, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
FK C	Cr_id	int	4	NOT NULL	1 - 1
	Cr_name	varchar(50)	50	NOT NULL	
	Cr_duration	int	4	NULL allowed	
FK	Ins_id	int	4	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
FK C	PK_Course	Cr_id	True

Foreign Keys

Name	Columns
FK_Course_Instructor	Ins_id->[dbo].[Instructor].[Ins_id]

SQL Script

```
CREATE TABLE [dbo].[Course]
(
[Cr_id] [int] NOT NULL IDENTITY(1, 1),
[Cr_name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[Cr_duration] [int] NULL,
[Ins_id] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Course] ADD CONSTRAINT [PK_Course] PRIMARY KEY CLUSTERED ([Cr_id]) ON
[PRIMARY]
GO
```

```
ALTER TABLE [dbo].[Course] ADD CONSTRAINT [FK_Course_Instructor] FOREIGN KEY ([Ins_id])
REFERENCES [dbo].[Instructor] ([Ins_id])
GO
```

Uses

[dbo].[Instructor]

Used By

[dbo].[Exam]
[dbo].[Question]
[dbo].[St_course]
[dbo].[Topic]
[dbo].[add_course]
[dbo].[add_exam]
[dbo].[delete_course]
[dbo].[Exam_correction]
[dbo].[GenerateExam]
[dbo].[get_all_courses]
[dbo].[get_instructor_courses]
[dbo].[get_student_grade]
[dbo].[InsertIntoTopic]
[dbo].[InsertStCourse]
[dbo].[update_course]
[dbo].[update_exam]
[dbo].[UpdateQuestion]
[dbo].[UpdateTopic]

[dbo].[Ex_question]

Properties

Property	Value
Row Count (~)	479
Created	12:25:41 AM Thursday, December 5, 2024
Last Modified	11:30:11 PM Monday, December 9, 2024

Columns

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

Indexes

Key	Name	Key Columns	Unique
PK C	PK_Ex_question	Ques_id, Ex_id	True

Foreign Keys

Name	Columns
FK_Ex_question_Exam	Ex_id->[dbo].[Exam].[Ex_id]
FK_Ex_question_Question	Ques_id->[dbo].[Question].[Ques_id]

SQL Script

```
CREATE TABLE [dbo].[Ex_question]
(
[Ex_id] [int] NOT NULL,
[Ques_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Ex_question] ADD CONSTRAINT [PK_Ex_question] PRIMARY KEY CLUSTERED ([Ques_id],
[Ex_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Ex_question] ADD CONSTRAINT [FK_Ex_question_Exam] FOREIGN KEY ([Ex_id])
REFERENCES [dbo].[Exam] ([Ex_id])
GO
ALTER TABLE [dbo].[Ex_question] ADD CONSTRAINT [FK_Ex_question_Question] FOREIGN KEY ([Ques_id])
```

```
REFERENCES [dbo].[Question] ([Ques_id])
GO
```

Uses

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

Used By

[dbo].[add_ex_question]
[dbo].[delete_ex_question]
[dbo].[Exam_correction]
[dbo].[GenerateExam]
[dbo].[get_all_ex_questions]
[dbo].[get_exam_questions]
[dbo].[get_exam_questions_with_answers]
[dbo].[insert_student_answers]
[dbo].[InsertexammAnswers]
[dbo].[update_ex_question]

[dbo].[Exam]

Properties

Property	Value
Row Count (~)	36
Created	11:29:27 PM Monday, December 9, 2024
Last Modified	8:48:28 AM Thursday, December 12, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
	Ex_id	int	4	NOT NULL	1 - 1
	date	date	3	NULL allowed	
	Start_ex	time	5	NULL allowed	
	End_ex	time	5	NULL allowed	
	Cr_id	int	4	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
	PK_Exam	Ex_id	True

Foreign Keys

Name	Columns
FK_Exam_Course	Cr_id->[dbo].[Course].[Cr_id]

SQL Script

```
CREATE TABLE [dbo].[Exam]
(
[Ex_id] [int] NOT NULL IDENTITY(1, 1),
[date] [date] NULL,
[Start_ex] [time] NULL,
[End_ex] [time] NULL,
[Cr_id] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Exam] ADD CONSTRAINT [PK_Exam] PRIMARY KEY CLUSTERED ([Ex_id]) ON [PRIMARY]
```

```
GO
ALTER TABLE [dbo].[Exam] ADD CONSTRAINT [FK_Exam_Course] FOREIGN KEY ([Cr_id]) REFERENCES
[dbo].[Course] ([Cr_id])
GO
```

Uses

[dbo].[Course]

Used By

[dbo].[Ex_question]
[dbo].[St_answer]
[dbo].[St_exam]
[dbo].[add_ex_question]
[dbo].[add_exam]
[dbo].[delete_exam]
[dbo].[Exam_correction]
[dbo].[GenerateExam]
[dbo].[get_all_exams]
[dbo].[get_student_grade]
[dbo].[insert_student_answers]
[dbo].[InsertexammAnswers]
[dbo].[InsertStAnswer]
[dbo].[update_ex_question]
[dbo].[update_exam]

[dbo].[Instructor]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	9
Created	11:29:40 PM Monday, December 9, 2024
Last Modified	11:31:06 PM Monday, December 9, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
FK C	Ins_id	int	4	NOT NULL	1 - 1
	Ins_name	varchar(50)	50	NOT NULL	
	Ins_age	int	4	NULL allowed	
	Ins_address	varchar(50)	50	NULL allowed	
	Ins_salary	int	4	NULL allowed	
	Ins_Degree	varchar(50)	50	NULL allowed	
FK	Tr_id	int	4	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
FK C	PK_Instructor	Ins_id	True

Foreign Keys

Name	Columns
FK_Instructor_Track1	Tr_id->[dbo].[Track].[Tr_id]

SQL Script

```
CREATE TABLE [dbo].[Instructor]
(
[Ins_id] [int] NOT NULL IDENTITY(1, 1),
[Ins_name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[Ins_age] [int] NULL,
[Ins_address] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Ins_salary] [int] NULL,
[Ins_Degree] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Tr_id] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Instructor] ADD CONSTRAINT [PK_Instructor] PRIMARY KEY CLUSTERED ([Ins_id]) ON
[PRIMARY]
GO
ALTER TABLE [dbo].[Instructor] ADD CONSTRAINT [FK_Instructor_Track1] FOREIGN KEY ([Tr_id])
REFERENCES [dbo].[Track] ([Tr_id])
GO
```

Uses

[dbo].[Track]

Used By

[dbo].[Course]
[dbo].[Track]
[dbo].[add_course]
[dbo].[AddInstructor]
[dbo].[DeleteInstructor]
[dbo].[GetInstructorByID]
[dbo].[InsertIntoTrack]
[dbo].[update_course]
[dbo].[UpdateInstructor]
[dbo].[UpdateTrack]

[dbo].[Ques_choice]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	300
Created	9:00:45 PM Wednesday, December 4, 2024
Last Modified	11:36:07 PM Monday, December 9, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
PK FK	Ques_id	int	4	NOT NULL
PK	choise	varchar(50)	50	NOT NULL

Indexes

Key	Name	Key Columns	Unique
PK	PK_Ques_choice	Ques_id, choise	True

Foreign Keys

Name	Update	Delete	Columns
FK_Ques_choice_Question	Cascade	Cascade	Ques_id->[dbo].[Question].[Ques_id]

SQL Script

```
CREATE TABLE [dbo].[Ques_choice]
(
[Ques_id] [int] NOT NULL,
[choise] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Ques_choice] ADD CONSTRAINT [PK_Ques_choice] PRIMARY KEY CLUSTERED ([Ques_id],
[choise]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Ques_choice] ADD CONSTRAINT [FK_Ques_choice_Question] FOREIGN KEY ([Ques_id])
REFERENCES [dbo].[Question] ([Ques_id]) ON DELETE CASCADE ON UPDATE CASCADE
GO
```

Uses

[dbo].[Question]

Used By

[dbo].[AddQuesChoice]
[dbo].[DeleteQuesChoice]
[dbo].[get_exam_questions]
[dbo].[GetQuesChoice]
[dbo].[UpdateQuesChoice]

[dbo].[Question]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	101
Created	11:30:11 PM Monday, December 9, 2024
Last Modified	8:48:28 AM Thursday, December 12, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
	Ques_id	int	4	NOT NULL	1 - 1
	Ques_content	varchar(500)	500	NOT NULL	
	type	varchar(50)	50	NOT NULL	
	Correct_ans	varchar(500)	500	NOT NULL	
	Ques_point	int	4	NULL allowed	
	Cr_id	int	4	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
	PK_Question	Ques_id	True

Foreign Keys

Name	Columns
FK_Question_Course	Cr_id->[dbo].[Course].[Cr_id]

SQL Script

```
CREATE TABLE [dbo].[Question]
(
[Ques_id] [int] NOT NULL IDENTITY(1, 1),
[Ques_content] [varchar] (500) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[type] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[Correct_ans] [varchar] (500) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[Ques_point] [int] NULL,
[Cr_id] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Question] ADD CONSTRAINT [PK_Question] PRIMARY KEY CLUSTERED ([Ques_id]) ON
[PRIMARY]
GO
ALTER TABLE [dbo].[Question] ADD CONSTRAINT [FK_Question_Course] FOREIGN KEY ([Cr_id]) REFERENCES
[dbo].[Course] ([Cr_id])
GO
```

Uses

[dbo].[Course]

Used By

[dbo].[Ex_question]
[dbo].[Ques_choice]
[dbo].[St_answer]
[dbo].[add_ex_question]
[dbo].[AddQuesChoice]
[dbo].[DeleteQuesChoice]
[dbo].[DeleteQuestion]
[dbo].[Exam_correction]
[dbo].[GenerateExam]
[dbo].[get_exam_questions]
[dbo].[get_exam_questions_with_answers]
[dbo].[GetQuestion]
[dbo].[InsertStAnswer]
[dbo].[update_ex_question]
[dbo].[UpdateQuesChoice]
[dbo].[UpdateQuestion]

[dbo].[St_answer]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	60
Created	8:48:28 AM Thursday, December 12, 2024
Last Modified	8:48:28 AM Thursday, December 12, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Default
PK FK	St_id	int	4	NOT NULL	
PK FK	Ex_id	int	4	NOT NULL	
PK FK	Ques_id	int	4	NOT NULL	
	answer	varchar(200)	200	NOT NULL	
	points	int	4	NULL allowed	((0))

Indexes

Key	Name	Key Columns	Unique
PK	PK_St_answer	St_id, Ex_id, Ques_id	True

Triggers

Name	ANSI Nulls On	Quoted Identifier On	On
t1	True	True	After Insert

Foreign Keys

Name	Update	Delete	Columns
FK_St_answer_Exam1			Ex_id->[dbo].[Exam].[Ex_id]
FK_St_answer_Question1			Ques_id->[dbo].[Question].[Ques_id]
FK_St_answer_Student	Cascade	Cascade	St_id->[dbo].[Student].[St_id]

SQL Script

```
CREATE TABLE [dbo].[St_answer]
(
[St_id] [int] NOT NULL,
[Ex_id] [int] NOT NULL,
[Ques_id] [int] NOT NULL,
[answer] [varchar] (200) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[points] [int] NULL CONSTRAINT [DF_St_answer_points] DEFAULT ((0))
) ON [PRIMARY]
GO
create trigger [dbo].[t1]
on [dbo].[St_answer]
after insert
as
  UPDATE st_answer
SET points = CASE
                WHEN (select answer from inserted) = (select Correct_ans from Question where
Ques_id = (select Ques_id from inserted))
                THEN (select Ques_point from Question where Ques_id = (select Ques_id from
inserted))
                ELSE 0
             END
FROM st_answer
JOIN question ON st_answer.Ques_id = question.Ques_id
where St_answer.St_id = (select st_id from inserted) and St_answer.Ex_id = (select ex_id from
inserted)
and St_answer.Ques_id = (select Ques_id from inserted);
GO
ALTER TABLE [dbo].[St_answer] ADD CONSTRAINT [PK_St_answer] PRIMARY KEY CLUSTERED ([St_id],
[Ex_id], [Ques_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[St_answer] ADD CONSTRAINT [FK_St_answer_Exam1] FOREIGN KEY ([Ex_id])
REFERENCES [dbo].[Exam] ([Ex_id])
GO
ALTER TABLE [dbo].[St_answer] ADD CONSTRAINT [FK_St_answer_Question1] FOREIGN KEY ([Ques_id])
REFERENCES [dbo].[Question] ([Ques_id])
GO
ALTER TABLE [dbo].[St_answer] ADD CONSTRAINT [FK_St_answer_Student] FOREIGN KEY ([St_id])
REFERENCES [dbo].[Student] ([St_id]) ON DELETE CASCADE ON UPDATE CASCADE
GO
```

Uses

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

Used By

[dbo].[DeleteStAnswer]
[dbo].[Exam_correction]
[dbo].[get_exam_questions_with_answers]

[dbo].[GetStudentAnswer]
[dbo].[insert_student_answers]
[dbo].[InsertexammAnswers]
[dbo].[InsertStAnswer]
[dbo].[UpdateStAnswer]

[dbo].[St_course]

Properties

Property	Value
Row Count (~)	30
Created	9:00:45 PM Wednesday, December 4, 2024
Last Modified	11:32:41 PM Monday, December 9, 2024

Columns

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

Indexes

Key	Name	Key Columns	Unique
PK	PK_St_course	St_id, Cr_id	True

Foreign Keys

Name	Update	Delete	Columns
FK_St_course_Course	Cascade	Cascade	Cr_id->[dbo].[Course].[Cr_id]
FK_St_course_Student1			St_id->[dbo].[Student].[St_id]

SQL Script

```
CREATE TABLE [dbo].[St_course]
(
[St_id] [int] NOT NULL,
[Cr_id] [int] NOT NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[St_course] ADD CONSTRAINT [PK_St_course] PRIMARY KEY CLUSTERED ([St_id],
[Cr_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[St_course] ADD CONSTRAINT [FK_St_course_Course] FOREIGN KEY ([Cr_id])
REFERENCES [dbo].[Course] ([Cr_id]) ON DELETE CASCADE ON UPDATE CASCADE
GO
ALTER TABLE [dbo].[St_course] ADD CONSTRAINT [FK_St_course_Student1] FOREIGN KEY ([St_id])
REFERENCES [dbo].[Student] ([St_id])
```

```
GO
```

Uses

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

Used By

[dbo].[DeleteStCourse]
[dbo].[get_instructor_courses]
[dbo].[GetStudentCourse]
[dbo].[insert_student_answers]
[dbo].[InsertexammAnswers]
[dbo].[InsertStCourse]

[dbo].[St_exam]

Properties

Property	Value
Row Count (~)	6
Created	9:21:55 PM Wednesday, December 4, 2024
Last Modified	11:33:42 PM Monday, December 9, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability
	St_id	int	4	NOT NULL
	Ex_id	int	4	NOT NULL
	Total_degree	int	4	NOT NULL

Indexes

Key	Name	Key Columns	Unique
	PK_St_exam_1	St_id, Ex_id	True

Triggers

Name	ANSI Nulls On	Quoted Identifier On	On
t2	True	True	After Insert

Foreign Keys

Name	Update	Delete	Columns
FK_St_exam_Exam			Ex_id->[dbo].[Exam].[Ex_id]
FK_St_exam_Student	Cascade	Cascade	St_id->[dbo].[Student].[St_id]

SQL Script

```
CREATE TABLE [dbo].[St_exam]
(
[St_id] [int] NOT NULL,
[Ex_id] [int] NOT NULL,
[Total_degree] [int] NOT NULL
) ON [PRIMARY]
GO
create trigger [dbo].[t2]
    on [dbo].[St_exam]
    after insert
    as
    update St_exam
    set Total_degree = (select sum(points) from St_answer
    where st_id = (select st_id from inserted) and Ex_id = (select Ex_id from inserted))
    where st_id = (select st_id from inserted) and Ex_id = (select Ex_id from inserted)
GO
ALTER TABLE [dbo].[St_exam] ADD CONSTRAINT [PK_St_exam_1] PRIMARY KEY CLUSTERED ([St_id],
[Ex_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[St_exam] ADD CONSTRAINT [FK_St_exam_Exam] FOREIGN KEY ([Ex_id]) REFERENCES
[dbo].[Exam] ([Ex_id])
GO
ALTER TABLE [dbo].[St_exam] ADD CONSTRAINT [FK_St_exam_Student] FOREIGN KEY ([St_id]) REFERENCES
[dbo].[Student] ([St_id]) ON DELETE CASCADE ON UPDATE CASCADE
GO
```

Uses

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

Used By

[dbo].[Delete_StExam]
[dbo].[get_student_grade]
[dbo].[InsertexammAnswers]

[dbo].[Student]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	15
Created	11:30:30 PM Monday, December 9, 2024
Last Modified	8:48:28 AM Thursday, December 12, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
FK C	St_id	int	4	NOT NULL	1 - 1
	St_fname	varchar(50)	50	NOT NULL	
	St_lname	varchar(50)	50	NULL allowed	
	age	int	4	NULL allowed	
	address	varchar(50)	50	NULL allowed	
FK	Tr_id	int	4	NULL allowed	
	Join_date	date	3	NULL allowed	
	duration(M)	smallint	2	NULL allowed	
	gender	varchar(1)	1	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
FK C	PK_Student	St_id	True

Check Constraints

Name	On Column	Constraint
c1	gender	([gender]='M' OR [gender]='F')

Foreign Keys

Name	Columns
FK_Student_Track	Tr_id->[dbo].[Track].[Tr_id]

SQL Script

```
CREATE TABLE [dbo].[Student]
(
[St_id] [int] NOT NULL IDENTITY(1, 1),
[St_fname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[St_lname] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[age] [int] NULL,
[address] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[Tr_id] [int] NULL,
[Join_date] [date] NULL,
[duration(M)] [smallint] NULL,
[gender] [varchar] (1) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Student] ADD CONSTRAINT [c1] CHECK (([gender]='M' OR [gender]='F'))
GO
ALTER TABLE [dbo].[Student] ADD CONSTRAINT [PK_Student] PRIMARY KEY CLUSTERED ([St_id]) ON
[PRIMARY]
GO
ALTER TABLE [dbo].[Student] ADD CONSTRAINT [FK_Student_Track] FOREIGN KEY ([Tr_id]) REFERENCES
[dbo].[Track] ([Tr_id])
GO
```

Uses

[dbo].[Track]

Used By

[dbo].[St_answer]
[dbo].[St_course]
[dbo].[St_exam]
[dbo].[DeleteStudentDetails]
[dbo].[Exam_correction]
[dbo].[get_student_grade]
[dbo].[get_student_info]
[dbo].[insert_student_answers]
[dbo].[InsertexammAnswers]
[dbo].[InsertNewStudent]
[dbo].[InsertStAnswer]
[dbo].[InsertStCourse]
[dbo].[UpdateStudent]

[dbo].[Topic]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	20
Created	5:46:07 PM Thursday, December 5, 2024
Last Modified	11:28:23 PM Monday, December 9, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
PK	Top_id	int	4	NOT NULL	1 - 1
	Top_name	varchar(50)	50	NOT NULL	
FK	Cr_id	int	4	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
PK	PK_Topic	Top_id	True

Foreign Keys

Name	Columns
FK_Topic_Course	Cr_id->[dbo].[Course].[Cr_id]

SQL Script

```
CREATE TABLE [dbo].[Topic]
(
[Top_id] [int] NOT NULL IDENTITY(1, 1),
[Top_name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[Cr_id] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Topic] ADD CONSTRAINT [PK_Topic] PRIMARY KEY CLUSTERED ([Top_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Topic] ADD CONSTRAINT [FK_Topic_Course] FOREIGN KEY ([Cr_id]) REFERENCES
[dbo].[Course] ([Cr_id])
GO
```

Uses

[dbo].[Course]

Used By

[dbo].[DeleteTopic]
[dbo].[get_course_topics]
[dbo].[InsertIntoTopic]
[dbo].[UpdateTopic]

[dbo].[Track]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Row Count (~)	10
Created	11:31:05 PM Monday, December 9, 2024
Last Modified	11:31:06 PM Monday, December 9, 2024

Columns

Key	Name	Data Type	Max Length (Bytes)	Nullability	Identity
🔑	Tr_id	int	4	NOT NULL	1 - 1
	Tr_name	varchar(50)	50	NOT NULL	
	Tr_decs	varchar(50)	50	NULL allowed	
🔑	manager_id	int	4	NULL allowed	
	hire_date	date	3	NULL allowed	

Indexes

Key	Name	Key Columns	Unique
🔑	PK_Track	Tr_id	True

Foreign Keys

Name	Columns
FK_Track_Instructor	manager_id->[dbo].[Instructor].[Ins_id]

SQL Script

```
CREATE TABLE [dbo].[Track]
(
[Tr_id] [int] NOT NULL IDENTITY(1, 1),
[Tr_name] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
[Tr_decs] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
[manager_id] [int] NULL,
[hire_date] [date] NULL
) ON [PRIMARY]
GO
```

```
ALTER TABLE [dbo].[Track] ADD CONSTRAINT [PK_Track] PRIMARY KEY CLUSTERED ([Tr_id]) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Track] ADD CONSTRAINT [FK_Track_Instructor] FOREIGN KEY ([manager_id])
REFERENCES [dbo].[Instructor] ([Ins_id])
GO
```

Uses

[dbo].[Instructor]

Used By

[dbo].[Br_Track]
[dbo].[Instructor]
[dbo].[Student]
[dbo].[add_br_track]
[dbo].[AddInstructor]
[dbo].[DeleteTrack]
[dbo].[InsertIntoTrack]
[dbo].[InsertNewStudent]
[dbo].[update_br_track]
[dbo].[UpdateInstructor]
[dbo].[UpdateStudent]
[dbo].[UpdateTrack]

Stored Procedures

Objects

Name
dbo.add_br_track
dbo.add_branch
dbo.add_course
dbo.add_ex_question
dbo.add_exam
dbo.AddInstructor
dbo.AddQuesChoice
dbo.delete_br_track
dbo.delete_branch
dbo.delete_course
dbo.delete_ex_question
dbo.delete_exam
dbo.Delete_StExam
dbo.DeleteInstructor
dbo.DeleteQuesChoice
dbo.DeleteQuestion
dbo.DeleteStAnswer
dbo.DeleteStCourse
dbo.DeleteStudentDetails
dbo.DeleteTopic
dbo.DeleteTrack
dbo.Exam_correction
dbo.GenerateExam
dbo.get_all_br_tracks
dbo.get_all_branches
dbo.get_all_courses
dbo.get_all_ex_questions
dbo.get_all_exams
dbo.get_course_topics
dbo.get_exam_questions
dbo.get_exam_questions_with_answers
dbo.get_instructor_courses
dbo.Get_St_Exam
dbo.get_student_grade

dbo.get_student_info
dbo.GetInstructorByID
dbo.GetQuesChoice
dbo.GetQuestion
dbo.GetStudentAnswer
dbo.GetStudentCourse
dbo.GetStudentDetails
dbo.GetTopicDetails
dbo.GetTrackDetails
dbo.insert_student_answers
dbo.InsertexammAnswers
dbo.InsertIntoTopic
dbo.InsertIntoTrack
dbo.InsertNewStudent
dbo.InsertStAnswer
dbo.InsertStCourse
dbo.update_br_track
dbo.update_branch
dbo.update_course
dbo.update_ex_question
dbo.update_exam
dbo.UpdateInstructor
dbo.UpdateQuesChoice
dbo.UpdateQuestion
dbo.UpdateStAnswer
dbo.UpdateStudent
dbo.UpdateTopic
dbo.UpdateTrack

[dbo].[add_br_track]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@br_id	int	4
@tr_id	int	4

SQL Script

```
create procedure [dbo].[add_br_track]
  @br_id int,
  @tr_id int
as
begin
  begin try
    begin transaction
    if not exists(select 1 from Br_Track where Br_id=@br_id and Tr_id=@tr_id)
     begin
    if not exists(select 1 from Branch where Br_id = @br_id)
      select 'Error: Branch ID does not exist in the Branch table.'
    else if not exists(select 1 from Track where Tr_id = @tr_id)
      select 'Error: Track ID does not exist in the Track table.'
    else
    begin
      insert into Br_Track (Br_id, Tr_id)
      values(@br_id, @tr_id)
      select 'Data inserted successfully into Br_Track.'
    end
    end
    else
        select'error dublicte key'

    commit transaction
  end try

  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

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

[dbo].[add_branch]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create procedure [dbo].[add_branch]
  @br_name varchar(50),
  @br_phone varchar(50)
as
begin
  begin try
    begin transaction

    insert into Branch (Br_name, Br_phone)
    values (@br_name, @br_phone)

    commit transaction
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

[dbo].[Branch]

[dbo].[add_course]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@cr_name	varchar(50)	50
@cr_duration	int	4
@ins_id	int	4

SQL Script

```
create procedure [dbo].[add_course]
  @cr_name varchar(50),
  @cr_duration int,
  @ins_id int
as
begin
  begin try
    if not exists(select 1 from instructor where ins_id = @ins_id)
      select 'Error: instructor ID does not exist in the instructor table.'
    else
    begin
      insert into course (cr_name, cr_duration, ins_id)
      values (@cr_name, @cr_duration, @ins_id)
      select 'Data inserted successfully into course.'
    end
  end try
  begin catch
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

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

[dbo].[add_ex_question]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create proc [dbo].[add_ex_question]
  @ex_id int,
  @ques_id int
as
begin
  begin try
    begin transaction

    if not exists(select 1 from Ex_question where Ex_id = @ex_id and Ques_id = @ques_id)
    begin
      if not exists(select 1 from Exam where Ex_id = @ex_id)
        select 'Error: Exam ID does not exist in the Exam table.'
      else if not exists(select 1 from Question where Ques_id = @ques_id)
        select 'Error: Question ID does not exist in the Question table.'
      else
      begin
        insert into Ex_question (Ex_id, Ques_id)
        values (@ex_id, @ques_id)
        select 'Data inserted successfully into Ex_question.'
      end
    end
    else
      select 'Duplicate key'

    commit transaction
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

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

[dbo].[add_exam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@date	date	3
@start_ex	time	5
@end_ex	time	5
@cr_id	int	4

SQL Script

```
create proc [dbo].[add_exam]
  @date date,
  @start_ex time,
  @end_ex time,
  @cr_id int
as
begin
  begin try
    begin transaction

    if not exists(select 1 from Course where Cr_id = @cr_id)
    begin
      select 'Error: Course ID does not exist in the Course table.'
    end
    else
    begin
      insert into Exam (date, Start_ex, End_ex, Cr_id)
      values (@date, @start_ex, @end_ex, @cr_id)
      select 'Data inserted successfully into exam.'
    end

    commit transaction
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
```

```
end
GO
```

Uses

[dbo].[Course]
[dbo].[Exam]

[dbo].[AddInstructor]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@Ins_Name	varchar(50)	50
@Ins_Age	int	4
@Ins_Address	varchar(50)	50
@Ins_Salary	int	4
@Ins_Degree	varchar(50)	50
@Tr_id	int	4

SQL Script

```sql
create Proc [dbo].[AddInstructor]
    @Ins_Name varchar(50),
    @Ins_Age int = NULL,
    @Ins_Address varchar(50) = NULL,
    @Ins_Salary int = 5000,
    @Ins_Degree varchar(50) = NULL,
    @Tr_id int = NULL
as
begin
    begin Try
        IF not exists (select 1 from Track where Tr_id = @Tr_id)
        begin

            select 'Error: The Track ID does not exist in the Track table.' AS Message;
            return;
        end

        -- Insert the new Instructor record
        insert into Instructor(Ins_name, Ins_age,
                               Ins_address, Ins_salary, Ins_Degree,Tr_id)
        values ( @Ins_Name, @Ins_Age, @Ins_Address,
                @Ins_Salary, @Ins_Degree, @Tr_id);

        -- Success message
        SELECT 'Data inserted successfully into Instructor table.' AS Message;

    end Try
    begin Catch
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage
    end Catch
end
GO
```

Uses

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

[dbo].[AddQuesChoice]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create Proc [dbo].[AddQuesChoice]
    @Ques_id int,
    @Choice varchar(50)
as
begin
    begin try
        if not exists (select 1 from Question where Ques_id = @Ques_id)
        begin
            select 'Error: Question ID does not exist in the Question table.' AS Message;
            return;
        end

        -- Check if the combination of Ques_id and choice already exists
        if exists (select 1 from Ques_choice where Ques_id = @Ques_id and choise = @Choice)
        begin
            select 'Error: The choice already exists for this Question ID.' AS Message;
            return;
        end
        -- Insert the new choice record
        insert into Ques_choice
        values (@Ques_id, @Choice);

        select 'Data inserted successfully into Ques_choice table.' AS Message;
    end try
    begin catch
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage;
    end catch
end;
GO
```

Uses

[dbo].[Ques_choice]
[dbo].[Question]

[dbo].[delete_br_track]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@br_id	int	4
@tr_id	int	4

SQL Script

```
create procedure [dbo].[delete_br_track]
  @br_id int,
  @tr_id int
as
begin
  begin try
    begin transaction

    if exists (select 1 from br_track where br_id = @br_id and tr_id = @tr_id)
    begin
      delete from br_track where br_id = @br_id and tr_id = @tr_id
      print 'Br_Track deleted successfully.'
      commit transaction
    end
    else
    begin
      print 'Error: Br_Track record does not exist.'
      rollback transaction
    end
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end

GO
```

Uses

[dbo].[Br_Track]

[dbo].[delete_branch]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create procedure [dbo].[delete_branch]
  @br_id int
as
begin
  begin try
    begin transaction

    if exists (select 1 from Branch where Br_id = @br_id)
    begin
      if exists (select 1 from Br_Track where Br_id = @br_id)
      begin
        select 'Error: The branch ID has related records in Br_Track. Cannot delete.'
      end
      else
      begin
        delete from Branch where Br_id = @br_id
        print 'Branch deleted successfully.'
      end
    end
    else
    begin
      print 'Error: Branch ID does not exist.'
    end

    commit transaction
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
```

```
GO
```

Uses

[dbo].[Br_Track]
[dbo].[Branch]

[dbo].[delete_course]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create proc [dbo].[delete_course]
  @cr_id int
as
begin
  begin try
    begin transaction

    if exists(select 1 from course where cr_id = @cr_id)
    begin
      delete from course where cr_id = @cr_id
      print 'Course deleted successfully.'
      commit transaction
    end
    else
    begin
      print 'Error: Course ID does not exist.'
      rollback transaction
    end
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

[dbo].[Course]

[dbo].[delete_ex_question]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create proc [dbo].[delete_ex_question]
  @ex_id int,
  @ques_id int
  as begin
  begin try
  begin transaction
        if exists (select 1 from Ex_question where Ex_id=@ex_id and Ques_id=@ques_id)
            begin
                delete from Ex_question where Ex_id=@ex_id and Ques_id=@ques_id
                print' ex_questiondeleted successfully.'
            end
        else
            print'Error: ex_question record does not exist.'
    commit transaction
    end try
    begin catch
    rollback transaction
 print 'Error: ' + error_message()
  end catch

    end
GO
```

Uses

[dbo].[Ex_question]

[dbo].[delete_exam]

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].[delete_exam]
  @ex_id int
as
begin
  begin try
    begin transaction

    if exists(select 1 from exam where ex_id = @ex_id)
    begin
      delete from exam where ex_id = @ex_id
      print 'Exam deleted successfully.'
    end
    else
      print 'Error: The exam ID does not exist in the Exam table.'

    commit transaction
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

[dbo].[Exam]

[dbo].[Delete_StExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create Proc [dbo].[Delete_StExam]
@Sid int
as
    begin
        begin try
            if not exists (select 1 from St_exam where  St_id = @Sid )
                print 'Error : Because This Row Is Not Exsite In St_Exam To Delete';

                delete from St_exam
                where St_id = @Sid
        end try
        begin catch
            print 'An Error occured ' + Error_Message()
        end catch
    end;
GO
```

Uses

[dbo].[St_exam]

[dbo].[DeleteInstructor]

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].[DeleteInstructor] @Ins_Id int
as
begin
    begin Try
        if not exists (select 1 from Instructor where Ins_id = @Ins_Id)
        begin
            select 'Error: Instructor ID does not exist in the Instructor table.'  AS Message;
            return;
        end
        -- Delete the Instructor record
        DELETE FROM Instructor WHERE Ins_id = @Ins_Id;

        --success message
        select 'Instructor record deleted successfully.' AS Message;

    end try
    begin catch
        -- Handle any errors that occur during the procedure
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage;
    end Catch
end;
GO
```

Uses

[dbo].[Instructor]

[dbo].[DeleteQuesChoice]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create Proc [dbo].[DeleteQuesChoice]
    @Ques_id int,
    @Choice varchar(50)
as
begin
    begin try
        -- Check if the Ques_id exists
        if not exists (select 1 from Question where Ques_id = @Ques_id)
        begin
            select 'Error: Question ID does not exist in the Question table.' AS Message;
            RETURN;
        end

        -- Check if the combination of Ques_id and choice exists
        if not exists (select 1 from Ques_choice where Ques_id = @Ques_id and choise = @Choice)
        begin
            select 'Error: The choice does not exist for this Question ID.' AS Message;
            return;
        end

        --Delete The Choice record
        delete from Ques_choice
        where Ques_id = @Ques_id and choise = @Choice;

        select 'Data deleted successfully from Ques_choice table.' AS Message;

    end try
    begin catch
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage;
    end catch;
end;
GO
```

Uses

[dbo].[Ques_choice]
[dbo].[Question]

[dbo].[DeleteQuestion]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create Proc [dbo].[DeleteQuestion]
    @Ques_id int
as
begin
    begin try
        if not exists (select 1 from Question where Ques_id = @Ques_id)
        begin
            select 'Error: Question ID does not exist in the Question table.' AS Message;
            return;
        end

        -- Delete the question record
        delete from Question
        where Ques_id = @Ques_id

        select 'Data deleted successfully from Question table.' AS Message;
    end try
    begin catch
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage
    end catch
end;
GO
```

Uses

[dbo].[Question]

[dbo].[DeleteStAnswer]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@St_id	int	4
@Ex_id	int	4
@Ques_id	int	4

SQL Script

```
create Proc [dbo].[DeleteStAnswer]
    @St_id INT,
    @Ex_id INT,
    @Ques_id INT
as
begin
    begin try
        -- Check if the combination of St_id, Ex_id, and Ques_id already exists
        if not exists (select 1 from St_answer where St_id = @St_id
                                            and Ex_id = @Ex_id
                                            and Ques_id = @Ques_id)
        begin
            select 'Error: The answer does not exist for this Student, Exam, and Question.' AS
Message;
            return;
        end

        -- Delete the answer for the student
        delete from St_answer
        where St_id = @St_id and Ex_id = @Ex_id and Ques_id = @Ques_id;

        select 'Data deleted successfully from St_answer table.' AS Message;
    end try
    begin catch
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage
    end catch
end;
GO
```

Uses

[dbo].[St_answer]

[dbo].[DeleteStCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@St_id	int	4
@Cr_id	int	4

SQL Script

```sql
CREATE PROC [dbo].[DeleteStCourse]
    @St_id INT,
    @Cr_id INT
AS
BEGIN

    -- Check if record exists
    IF NOT EXISTS (SELECT 1 FROM St_course WHERE St_id = @St_id AND Cr_id = @Cr_id)
    BEGIN
        select 'Record does not exist'
        RETURN;
    END

    -- Delete the record
    DELETE FROM St_course
    WHERE St_id = @St_id AND Cr_id = @Cr_id;

    SELECT 'Course enrollment deleted successfully' AS Message;
END
GO
```

Uses

[dbo].[St_course]

[dbo].[DeleteStudentDetails]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create Proc [dbo].[DeleteStudentDetails]
@Sid int
as
    begin
        begin try
            if not exists (select 1 from Student where St_id = @Sid)
                print 'Error Because St_id Is Not Exists To Delete';
            delete from Student
            where St_id = @Sid
        end try
        begin catch
            Print 'Error Occured ' + Error_Message();
        end catch
    end;
GO
```

Uses

[dbo].[Student]

[dbo].[DeleteTopic]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create Proc [dbo].[DeleteTopic]
(@TopId int)
as
    begin
        begin try
                if not exists (select 1 from Topic where Top_id= @TopId)
                begin
                    print 'Error Because Topic_id Is Not Exists To Delete';
                    return;
                end
            delete from Topic
            where Top_id = @TopId
        end try
        begin catch
            print 'Error Occuerd ' + Error_Message();
        end catch
    end;
GO
```

Uses

[dbo].[Topic]

[dbo].[DeleteTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
Create Proc [dbo].[DeleteTrack]
(@TrID int)
as
    begin
        begin try
                if not exists (select 1 from Track where Tr_id= @TrID)
                begin
                    print 'Error Because TrackID Is Not Exists To Delete';
                    return;
                end
            delete from Track
            where Tr_id = @TrID
        end try
        begin catch
            print 'Error Occuerd ' + Error_Message();
        end catch
    end;
GO
```

Uses

[dbo].[Track]

[dbo].[Exam_correction]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE procedure [dbo].[Exam_correction]
    @exam_id int,
    @student_id int
as
begin
    begin try
        if exists(select 1 from Exam where Ex_id=@exam_id)
            begin
                 if exists (select 1 from St_answer where St_id=@student_id)
                    begin
                        select st.St_fname+' '+st.St_lname as student_name,
                               e.Ex_id,
                               c.Cr_name as course_name,
                               sum(case when sa.answer = q.correct_ans then isnull(sa.points, 0)
end) as student_score,
                              cast(sum(case when sa.answer = q.correct_ans then isnull(sa.points,
0) end) * 100.0 / sum(isnull(q.Ques_point, 0)) as decimal(5, 2)) as percent_grade
                        from st_answer sa
                        inner join question q
                            on sa.ques_id = q.ques_id
                        inner join ex_question eq
                            on eq.ques_id = q.ques_id
                        inner join exam e
                            on eq.ex_id = e.ex_id
                        inner join course c
                            on e.Cr_id = c.Cr_id
                        inner join student st
                            on sa.st_id = st.st_id
                        where e.ex_id = @exam_id
                          and st.st_id = @student_id
                        group by st.St_fname,st.St_lname, e.Ex_id, c.Cr_name;
                    end
                    else
                        begin
                            select 'Error: answers do not exist yet'
                        end
            end
            else
                begin
                    select 'Error: Exam does not exist'
                end
    end try
    begin catch
        print 'An error occurred during execution. Please check your input values.';
    end catch
END
GO
```

Uses

[dbo].[Course]
[dbo].[Ex_question]
[dbo].[Exam]
[dbo].[Question]
[dbo].[St_answer]
[dbo].[Student]

[dbo].[GenerateExam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@CrId	int	4
@ExamDate	date	3
@StartTime	time	5
@EndTime	time	5
@TOrFQuestions	int	4
@McqQuestions	int	4

SQL Script

```
CREATE Proc [dbo].[GenerateExam]
(@CrId int , @ExamDate date , @StartTime Time  , @EndTime Time , @TOrFQuestions int , @Mcq-
Questions int )
as
    begin
        begin try

                -- Check if Cr_id exists in the Course table
                 IF NOT EXISTS (SELECT 1 FROM Course WHERE Cr_id = @CrId)
                 BEGIN
                     print 'Invalid Cr_id: No matching Course found.'
                     return;
                 END;

                -- Ensure that we request at least 2 questions
                if (@TOrFQuestions + @McqQuestions) < 2
                begin
                    print 'You must request at least 2 questions.'
                    return;
                end

                -- Start a transaction
                BEGIN TRANSACTION;
```

```
              -- Step 1: Insert the exam record into the Exam table
              declare @Ex_id int; -- Variable to store the newly created Exam ID

              insert into Exam (date, Start_ex, End_ex, Cr_id)
              values (@ExamDate, @StartTime, @EndTime, @CrId);

                 -- Get the newly created Exam ID
                 Set @Ex_id = SCOPE_IDENTITY();



                 -- Step 3: insert  True/False questions from the Question table and insert them
into Ex_Question
                 Insert into Ex_Question (Ex_id, Ques_id)
                 select top (@TOrFQuestions) @Ex_id, Ques_id
                 from Question
                 where Cr_id = @CrId AND type = 'TorF' and not exists( select 1 from Ex_Question
eq where eq.Ques_id = Question.Ques_id AND eq.Ex_id = @Ex_id)
                 order by NEWID();  -- Randomly select True/False questions

                 -- Step 4: insert MCQ questions from the Question table and insert them into Ex_-
Question
                 insert into Ex_Question (Ex_id, Ques_id)
                 select top (@McqQuestions) @Ex_id, Ques_id
                 from Question
                 where Cr_id = @CrId AND type = 'MCQ' and not exists (select 1 from Ex_Question eq
where eq.Ques_id = Question.Ques_id AND eq.Ex_id = @Ex_id)
                 order by NEWID();  -- Randomly select MCQ questions

                 -- Step 5: Fetch the questions linked to the exam and display them
                 select q.Ques_id, q.Ques_content, q.type, q.Correct_ans, q.Ques_point
                 from Question q
                 inner join Ex_Question eq on q.Ques_id = eq.Ques_id
                 where eq.Ex_id = @Ex_id;

                 -- Commit the transaction
                 COMMIT TRANSACTION;

                 -- Optional: Return the Exam ID
                 Select @Ex_id AS GeneratedExamID;
        end try

        BEGIN CATCH
            -- Rollback the transaction in case of error
            ROLLBACK TRANSACTION;

            -- Return error information
            THROW;
        END CATCH;
    end;
GO
```

Uses

[dbo].[Course]
[dbo].[Ex_question]
[dbo].[Exam]
[dbo].[Question]

[dbo].[get_all_br_tracks]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

SQL Script

```
create proc [dbo].[get_all_br_tracks]
as
begin
  begin try
    begin transaction

    select * from br_track

    commit transaction
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

[dbo].[Br_Track]

[dbo].[get_all_branches]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

SQL Script

```
create proc [dbo].[get_all_branches]
as
begin
  begin try
    select * from Branch
  end try
  begin catch
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

[dbo].[Branch]

[dbo].[get_all_courses]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

SQL Script

```
create proc [dbo].[get_all_courses]
as
begin
  begin try
    select * from course
  end try
  begin catch
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

[dbo].[Course]

[dbo].[get_all_ex_questions]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

SQL Script

```
create proc [dbo].[get_all_ex_questions]
as begin
begin try
    begin transaction
  select * from Ex_question
    commit transaction
end try
begin catch
rollback transaction
 print 'Error: ' + error_message()
  end catch
 end
GO
```

Uses

[dbo].[Ex_question]

[dbo].[get_all_exams]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

SQL Script

```
create proc [dbo].[get_all_exams]
as
begin
  begin try
    begin transaction

    select * from exam

    commit transaction
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

[dbo].[Exam]

[dbo].[get_course_topics]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create procedure [dbo].[get_course_topics]
    @cr_id int
as
begin
    select
        top_id,
        top_name
    from topic
    where cr_id = @cr_id;
end;
GO
```

Uses

[dbo].[Topic]

[dbo].[get_exam_questions]

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 procedure [dbo].[get_exam_questions]
    @ex_id int
as
begin
    select
        q.ques_id,
        q.ques_content,
        qc.choise
    from ex_question eq
    inner join question q on eq.ques_id = q.ques_id
    left join ques_choice qc on q.ques_id = qc.ques_id
    where eq.ex_id = @ex_id;
end;

GO
```

Uses

[dbo].[Ex_question]
[dbo].[Ques_choice]
[dbo].[Question]

[dbo].[get_exam_questions_with_answers]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
CREATE procedure [dbo].[get_exam_questions_with_answers]
    @ex_id int,
    @st_id int
as
begin
    select
        q.ques_id,
        q.ques_content,
        sa.answer as student_answer
        , q.Correct_ans
    from ex_question eq
    inner join question q on eq.ques_id = q.ques_id
    left join st_answer sa on q.ques_id = sa.ques_id and sa.st_id = @st_id and sa.ex_id = @ex_id
    where eq.ex_id = @ex_id;
end;
GO
```

Uses

[dbo].[Ex_question]
[dbo].[Question]
[dbo].[St_answer]

[dbo].[get_instructor_courses]

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 procedure [dbo].[get_instructor_courses]
    @ins_id int
as
begin
    set nocount on;
    select
        c.cr_name,
        count(sc.st_id) as student_count
    from course c
    left join st_course sc on c.cr_id = sc.cr_id
    where c.ins_id = @ins_id
    group by c.cr_name;
end;
GO
```

Uses

[dbo].[Course]
[dbo].[St_course]

[dbo].[Get_St_Exam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@x	nvarchar(20)	40

SQL Script

```
Create Proc [dbo].[Get_St_Exam] @x nvarchar(20)
as
    If @x  = '*'
        begin
            exec ('Select * From St_Exam')
        end
    else
        begin
         exec ('select ' + @x +' From St_Exam ')
        end
GO
```

[dbo].[get_student_grade]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create procedure [dbo].[get_student_grade]
    @student_id int
as
begin
    select
        s.st_id,
        s.St_fname + ' ' + s.St_lname,
        c.Cr_name,
        Total_degree
  from Student s
        join St_exam se on s.St_id = se.St_id
        join Exam e on e.Ex_id = se.Ex_id
        join Course c on c.Cr_id = e.Cr_id

  where s.st_id = @student_id;
end;
GO
```

Uses

[dbo].[Course]
[dbo].[Exam]
[dbo].[St_exam]
[dbo].[Student]

[dbo].[get_student_info]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
  create procedure [dbo].[get_student_info]
 @TR_No INT
as
begin
        select *from  Student
        where
            Tr_id = @TR_No;
end;
GO
```

Uses

[dbo].[Student]

[dbo].[GetInstructorByID]

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].[GetInstructorByID]
@Ins_Id int
as
begin
    select *
    from Instructor
    where Ins_id = @Ins_id;
end;
GO
```

Uses

[dbo].[Instructor]

[dbo].[GetQuesChoice]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create Proc [dbo].[GetQuesChoice] @Ques_id int
as
begin
    select *
    from Ques_choice
    where Ques_id = @Ques_id;
end
GO
```

Uses

[dbo].[Ques_choice]

[dbo].[GetQuestion]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

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

SQL Script

```
create Proc [dbo].[GetQuestion] @Ques_id int
as
begin
    select *
    from Question
    where Ques_id = @Ques_id;
end
GO
```

Uses

[dbo].[Question]

[dbo].[GetStudentAnswer]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@St_id	int	4
@Ex_id	int	4
@Ques_id	int	4

SQL Script

```
create Proc [dbo].[GetStudentAnswer]
    @St_id int,
    @Ex_id int,
    @Ques_id int
as
begin
    select *
    from St_answer
    where St_id = @St_id and Ex_id = @Ex_id and Ques_id = @Ques_id;
end
GO
```

Uses

[dbo].[St_answer]

[dbo].[GetStudentCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@St_id	int	4
@Cr_id	int	4

SQL Script

```
create Proc [dbo].[GetStudentCourse]
    @St_id int,
    @Cr_id int
as
begin
    select *
    from St_course
    where St_id = @St_id and Cr_id = @Cr_id
end
GO
```

Uses

[dbo].[St_course]

[dbo].[GetStudentDetails]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@x	nvarchar(20)	40

SQL Script

```
Create Proc [dbo].[GetStudentDetails] @x nvarchar(20)
as
    If @x  = '*'
        begin
            exec ('Select * From Student')
        end
    else
        begin
         exec ('select ' + @x +' From Student')
        end
GO
```

[dbo].[GetTopicDetails]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@x	nvarchar(20)	40

SQL Script

```
Create Proc [dbo].[GetTopicDetails] @x nvarchar(20)
as
    If @x  = '*'
        begin
            exec ('Select * From Topic')
        end
    else
        begin
         exec ('select ' + @x +' From Topic')
        end
GO
```

[dbo].[GetTrackDetails]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@x	nvarchar(20)	40

SQL Script

```
Create Proc [dbo].[GetTrackDetails] @x nvarchar(20)
as
    If @x  = '*'
        begin
            exec ('Select * From Track')
        end
    else
        begin
         exec ('select ' + @x +' From Track')
        end
GO
```

[dbo].[insert_student_answers]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@student_id	int	4
@exam_id	int	4
@answer1	varchar(50)	50
@answer2	varchar(50)	50
@answer3	varchar(50)	50
@answer4	varchar(50)	50
@answer5	varchar(50)	50
@answer6	varchar(50)	50
@answer7	varchar(50)	50
@answer8	varchar(50)	50
@answer9	varchar(50)	50
@answer10	varchar(50)	50

SQL Script

```
CREATE procedure [dbo].[insert_student_answers]
    @student_id int,
    @exam_id int,
    @answer1 varchar(50),
    @answer2 varchar(50),
    @answer3 varchar(50),
    @answer4 varchar(50),
    @answer5 varchar(50),
    @answer6 varchar(50),
    @answer7 varchar(50),
    @answer8 varchar(50),
    @answer9 varchar(50),
    @answer10 varchar(50)
as
begin
    begin try
        -- التحقق من وجود الطالب
        if not exists (select 1 from student where st_id = @student_id)
        begin
            print 'Error: Student with ID ' + cast(@student_id as varchar) + ' does not exist.';
            return;
        end

        -- التحقق من وجود الامتحان
        if not exists (select 1 from exam where ex_id = @exam_id)
        begin
            print 'Error: Exam with ID ' + cast(@exam_id as varchar) + ' does not exist.';
            return;
        end

        -- التحقق من أن الطالب مشترك في الكورس الخاص بالامتحان
        if exists (
            select 1
            from st_course s
            join exam e on s.cr_id = e.cr_id
            where e.ex_id = @exam_id and s.st_id = @student_id
        )
        begin
            -- إدخال الإجابات
            ;with question_with_row as (
                select eq.ques_id, row_number() over (order by eq.ques_id) as row_num
                from ex_question eq
                where eq.ex_id = @exam_id
            )
            insert into st_answer (st_id, ex_id, answer, ques_id)
            select
                @student_id,
                @exam_id,
                case
                    when row_num = 1 then @answer1
                    when row_num = 2 then @answer2
                    when row_num = 3 then @answer3
                    when row_num = 4 then @answer4
                    when row_num = 5 then @answer5
                    when row_num = 6 then @answer6
                    when row_num = 7 then @answer7
                    when row_num = 8 then @answer8
                    when row_num = 9 then @answer9
                    when row_num = 10 then @answer10
                end as answer,
                eq.ques_id
            from question_with_row eq
            where row_num <= 10;

            print 'Answers have been successfully inserted for student ID  in exam ID ';
        end
        else
```

```
        begin
            print 'Error: The student with ID  is not enrolled in the course for exam ID ';
        end
    end try
    begin catch
        print 'An error occurred: ' + ERROR_MESSAGE();
    end catch
end
GO
```

Uses

[dbo].[Ex_question]
[dbo].[Exam]
[dbo].[St_answer]
[dbo].[St_course]
[dbo].[Student]

[dbo].[InsertexammAnswers]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@St_id	int	4
@Ex_id	int	4
@ans1	varchar(100)	100
@ans2	varchar(100)	100
@ans3	varchar(100)	100
@ans4	varchar(100)	100
@ans5	varchar(100)	100
@ans6	varchar(100)	100
@ans7	varchar(100)	100
@ans8	varchar(100)	100
@ans9	varchar(100)	100
@ans10	varchar(100)	100

SQL Script

```
CREATE PROC [dbo].[InsertexammAnswers]
    @St_id INT,
    @Ex_id INT,
    @ans1 varchar(100) ,
    @ans2 varchar(100) ,
    @ans3 varchar(100) ,
    @ans4 varchar(100) ,
    @ans5 varchar(100) ,
    @ans6 varchar(100) ,
    @ans7 varchar(100) ,
    @ans8 varchar(100) ,
    @ans9 varchar(100) ,
    @ans10 varchar(100)
AS
BEGIN
```

```
    BEGIN TRANSACTION;

    BEGIN TRY
        -- 1. Validate the student exists in the Track
        IF NOT EXISTS (
            SELECT 1
            FROM Student S
            WHERE St_id = @St_id
        )
        BEGIN
            SELECT 'Invalid student'
            ROLLBACK TRANSACTION;
            RETURN;
        END
        -- 2. Validate the course of the exam exists in the courses of the student
        IF ((SELECT Cr_id FROM dbo.Exam WHERE Ex_id = @ex_id) IN (SELECT Cr_id FROM dbo.St_course
WHERE St_id = @st_id))
        BEGIN
            select  'The course of this exam does not belong to the student'
            ROLLBACK TRANSACTION;
            RETURN;
        END

        -- 3. Check if the student has already taken the exam
        if exists (
            select 1
            from St_answer
            where Ex_id = @Ex_id and St_id = @St_id
        )
        BEGIN
            select 'Student has already taken this exam'
            ROLLBACK TRANSACTION;
            RETURN;
        END

        -- 4. Temporary table for processing student answers
        CREATE TABLE #examQuestions (virtual_id int IDENTITY,
            Question_id INT
        );

          WITH OrderedQuestions AS (
            SELECT
                Ques_id,
                ROW_NUMBER() OVER (ORDER BY ques_id) AS ques_order
             FROM dbo.Ex_question WHERE Ex_id = @Ex_id
        )


        INSERT INTO st_answer (st_id, ex_id, ques_id, answer)
        SELECT
            @st_id AS st_id,
            @ex_id AS ex_id,
            Ques_id AS ques_id,
            CASE ques_order
                WHEN 1 THEN @ans1
                WHEN 2 THEN @ans2
                WHEN 3 THEN @ans3
                WHEN 4 THEN @ans4
                WHEN 5 THEN @ans5
                WHEN 6 THEN @ans6
                WHEN 7 THEN @ans7
                WHEN 8 THEN @ans8
                WHEN 9 THEN @ans9
                WHEN 10 THEN @ans10
            END AS answer
        FROM OrderedQuestions;



       INSERT INTO dbo.St_exam
       (
           St_id,
           Ex_id
       )
       VALUES
       (   @st_id,
           @ex_id
       )

        -- 10. Commit the transaction
        COMMIT TRANSACTION;

        -- 11. Return success message
    --     PRINT 'Answers submitted successfully. Total Marks: ' + CAST(@TotalMarks AS NVARCHAR);

        -- 12. Clean up temporary table
     --    DROP TABLE #ProcessedAnswers;
    END TRY
    BEGIN CATCH
        -- Handle errors and roll back
        IF @@TRANCOUNT > 0
            ROLLBACK TRANSACTION;

        -- Print error details
        PRINT 'An error occurred: ' + ERROR_MESSAGE();
    END CATCH
END
GO
```

Uses

[dbo].[Ex_question]
[dbo].[Exam]
[dbo].[St_answer]

[dbo].[St_course]

[dbo].[St_exam]

[dbo].[Student]

[dbo].[InsertIntoTopic]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@TopName	varchar(20)	20
@CrID	int	4

SQL Script

```
Create Proc [dbo].[InsertIntoTopic]
 (@TopName varchar(20) , @CrID int)
as
    begin
        begin try
            if not exists (Select 1 from Course Where Cr_id = @CrID)
                begin
                    print 'Error Because This CrId Is Not Existe In Table Course';
                    return;
                end
                insert into Topic
                values ( @TopName , @CrID)
        end try
        begin catch
            print 'Error Ouccured' + Error_Message();
        end catch
    end;
GO
```

Uses

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

[dbo].[InsertIntoTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@TrName	varchar(20)	20
@TrDecs	varchar(20)	20
@ManagerID	int	4
@HireDate	date	3

SQL Script

```
create Proc [dbo].[InsertIntoTrack]
( @TrName varchar (20) , @TrDecs varchar(20) , @ManagerID int , @HireDate date)
as
    begin
        begin try
            if not exists(select 1 from Instructor where Ins_id = @ManagerID)
                begin
                    print 'Error Because This Id Is Aleardy Not Exsits'
                    return;
                end
                insert into Track
                values (@TrName, @TrDecs , @ManagerID , @HireDate)
        end try
        begin catch
            print 'Error Ouccerd ' + Error_Message();
        end catch
    end;
GO
```

Uses

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

[dbo].[InsertNewStudent]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@St_FirstName	varchar(20)	20
@St_LastName	varchar(20)	20
@age	int	4
@address	varchar(20)	20
@TrId	int	4
@joinDate	date	3
@duration	int	4
@gender	varchar	1

SQL Script

```
create Proc [dbo].[InsertNewStudent]
(
 @St_FirstName varchar (20) , @St_LastName varchar (20),
@age int , @address varchar(20)= NULL , @TrId int , @joinDate date = NULL ,
@duration int , @gender varchar(1)
)
as
    begin
        begin try

            if not exists (select 1 from Track where Tr_id = @TrId)
                print 'Error Beccause Tr You Enter Is Not Exists In Table Track';
            if @gender != 'M' and @gender != 'F'
                print 'Error Because You Must Enter "M" Or "F"';

            insert into Student
            values (@St_FirstName,@St_LastName,@age,@address,@TrId,@joinDate,@duration,@gender)
        end try
        begin catch
            print 'Error Occuerd' + Error_Message();
        end catch
    end;

GO
```

Uses

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

[dbo].[InsertStAnswer]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@St_id	int	4
@Ex_id	int	4
@Ques_id	int	4
@Answer	varchar(200)	200

SQL Script

```
create Proc [dbo].[InsertStAnswer]
    @St_id INT,
    @Ex_id INT,
    @Ques_id INT,
    @Answer VARCHAR(200) = NULL
as
begin
    begin try
        -- Check if the St_id exists in Student table
        if not exists (select 1 from Student where St_id = @St_id)
        begin
            select 'Error: Student ID does not exist in the Student table.' AS Message;
            return;
        end

        -- Check if the Ex_id exists in Exam table
        if not exists (select 1 from Exam where Ex_id = @Ex_id)
        begin
            select 'Error: Exam ID does not exist in the Exam table.' AS Message;
            return;
        end

        -- Check if the Ques_id exists in Question table
        if not exists (select 1 from Question where Ques_id = @Ques_id)
        begin
            select 'Error: Question ID does not exist in the Question table.' AS Message;
            return;
        end
        -- Check if the combination of St_id, Ex_id, and Ques_id already exists
        if exists (select 1 from St_answer where St_id = @St_id
                                            and Ex_id = @Ex_id
                                            and Ques_id = @Ques_id)
        begin
            select 'Error: The answer already exists for this Student, Exam, and Question.' AS
Message;
            return;
        end

        insert into St_answer (St_id,Ex_id,Ques_id,answer)
        values (@St_id, @Ex_id, @Ques_id, @Answer);

        select 'Data inserted successfully into St_answer table.' AS Message;
    end try
    begin catch
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage
    end catch
end;
GO
```

Uses

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

[dbo].[InsertStCourse]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@St_id	int	4
@Cr_id	int	4

SQL Script

```
create Proc [dbo].[InsertStCourse]
    @St_id int,
    @Cr_id int
as
begin
    begin try
        -- Check if the St_id exists in Student table
        if not exists (select 1 from Student where St_id = @St_id)
        begin
            select 'Error: Student ID does not exist in the Student table.' AS Message;
            return;
        end

        -- Check if the Cr_id exists in Course table
        if not exists (select 1 from course where Cr_id = @Cr_id)
        begin
            select 'Error: Course ID does not exist in the Course table.' AS Message;
            return;
        end

        -- Check if the combination of St_id and Cr_id already exists
        if exists (select 1 from St_course where Cr_id = @Cr_id and St_id = @St_id)
        begin
            select 'Error: The Course already exists for this Student ID.' AS Message;
            return;
        end

        -- Insert the new record
        insert into St_course
        values (@St_id, @Cr_id);

        select 'Data inserted successfully into St_course table.' AS Message;
    end try
    begin catch
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage
    end catch
end;
GO
```

Uses

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

[dbo].[update_br_track]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@old_br_id	int	4
@old_tr_id	int	4
@new_br_id	int	4
@new_tr_id	int	4

SQL Script

```
create procedure [dbo].[update_br_track]
    @old_br_id int,
    @old_tr_id int,
    @new_br_id int,
    @new_tr_id int
as
begin
    begin try
        -- ??? ????????
        begin transaction

        if exists (select 1 from br_track where br_id = @old_br_id and tr_id = @old_tr_id)
        begin
            if exists (select 1 from branch where br_id = @new_br_id) and exists (select 1 from
track where tr_id = @new_tr_id)
            begin
                delete from br_track where br_id = @old_br_id and tr_id = @old_tr_id;

                insert into br_track (br_id, tr_id)
                values (@new_br_id, @new_tr_id);

                print 'br_track record updated successfully.';
            end
            else
            begin
                print 'error: the new br_id or tr_id does not exist in the respective tables.';
                -- ??? ?? ???? ??? ????? ?????? ??? ??????? ?? ????????
                rollback transaction
            end
        end
        else
        begin
            print 'error: br_track record not found.';
            -- ??? ?? ??? ?????? ??? ?????? ??? ??????? ?? ????????
            rollback transaction
        end

        -- ??? ??? ??????? ?????? ??? ????? ?????????
        commit transaction
    end try
    begin catch
        -- ?? ???? ???? ???? ??? ??????? ?? ????????
        rollback transaction
        print 'error: ' + error_message()
    end catch
end
GO
```

Uses

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

[dbo].[update_branch]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@br_id	int	4
@br_name	varchar(50)	50
@br_phone	varchar(50)	50

SQL Script

```
create proc [dbo].[update_branch]
  @br_id int,
  @br_name varchar(50),
  @br_phone varchar(50)
as
begin
  begin try
    begin transaction

    if exists(select 1 from Branch where Br_id = @br_id)
    begin
      update Branch
      set Br_name = @br_name, Br_phone = @br_phone
      where Br_id = @br_id
      print 'Branch updated successfully.'
    end
    else
    begin
      print 'Error: Branch ID does not exist.'
    end

    commit transaction
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
```

```
GO
```

Uses

[dbo].[Branch]

[dbo].[update_course]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@cr_id	int	4
@cr_name	varchar(50)	50
@cr_duration	int	4
@ins_id	int	4

SQL Script

```
create proc [dbo].[update_course]
  @cr_id int,
  @cr_name varchar(50),
  @cr_duration int,
  @ins_id int
as
begin
  begin try
    begin transaction

    if exists(select 1 from course where cr_id = @cr_id)
    begin
      if exists(select 1 from instructor where ins_id = @ins_id)
      begin
        update course
        set cr_name = @cr_name, cr_duration = @cr_duration, ins_id = @ins_id
        where cr_id = @cr_id
        print 'Course updated successfully.'
      end
      else
        print 'Error: The instructor does not exist.'
    end
    else
      print 'Error: Course ID does not exist.'

    commit transaction
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

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

[dbo].[update_ex_question]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@oldex_id	int	4
@oldques_id	int	4
@newex_id	int	4
@newques_id	int	4

SQL Script

```
create proc [dbo].[update_ex_question]
  @oldex_id int,
  @oldques_id int,
   @newex_id int,
  @newques_id int
  as begin
  begin try
    begin transaction
  if exists(select 1 from Ex_question  where Ex_id=@oldex_id and Ques_id=@oldques_id)
    begin
        if exists(select 1 from Exam where Ex_id=@newex_id)
            begin
                if exists(select 1 from Question where Ques_id=@newques_id)
                    begin

                        delete from Ex_question
                        where Ex_id=@oldex_id and Ques_id=@oldques_id
                        insert into Ex_question (Ex_id,Ques_id)
                        values(@newex_id,@newques_id)
                        print'keys updated successfully.'
                    end
                else
                    print'Error: The new ques_id does not exist in the question table.'
            end
        else
            print'Error: The new ex_id does not exist in the exam table.'
    end
  else
    print'Error: The record does not exist in ex_question.'
    commit transaction
    end try
    begin catch
    rollback transaction
 print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

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

[dbo].[update_exam]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@ex_id	int	4
@date	date	3
@start_ex	time	5
@end_ex	time	5
@newcr_id	int	4

SQL Script

```
create proc [dbo].[update_exam]
  @ex_id int,
  @date date,
  @start_ex time,
  @end_ex time,
  @newcr_id int
as
begin
  begin try
    begin transaction

    if exists(select 1 from exam where ex_id = @ex_id)
    begin
      if exists(select 1 from course where cr_id = @newcr_id)
      begin
        update exam
        set date = @date,
            start_ex = @start_ex,
            end_ex = @end_ex,
            cr_id = @newcr_id
        where ex_id = @ex_id
        print 'Exam updated successfully.'
      end
      else
        print 'Error: The new Course ID does not exist in the Course table.'
    end
    else
      print 'Error: The exam ID does not exist in the Exam table.'

    commit transaction
  end try
  begin catch
    rollback transaction
    print 'Error: ' + error_message()
  end catch
end
GO
```

Uses

[dbo].[Course]
[dbo].[Exam]

[dbo].[UpdateInstructor]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@Ins_Id	int	4
@Ins_Name	varchar(50)	50
@Ins_Age	int	4
@Ins_Address	varchar(50)	50
@Ins_Salary	int	4
@Ins_Degree	varchar(50)	50
@Tr_id	int	4

SQL Script

```
create Proc [dbo].[UpdateInstructor]
    @Ins_Id int,
    @Ins_Name varchar(50),
    @Ins_Age int = NULL,
    @Ins_Address varchar(50) = NULL,
    @Ins_Salary int = 5000,
    @Ins_Degree varchar(50) = NULL,
    @Tr_id int = NULL
as
begin
    begin Try
        -- Check if the Instructor exists
        if exists (select 1 from Instructor where Ins_id = @Ins_Id)
        begin
            -- Check if the Track ID exists
            if exists (select 1 from Track where Tr_id = @Tr_id)
            begin
                -- Update the Instructor record
                update Instructor
                set Ins_name = @Ins_Name,
                    Ins_age = @Ins_Age,
                    Ins_address = @Ins_Address,
                    Ins_salary = @Ins_Salary,
                    Ins_Degree = @Ins_Degree,
                    Tr_id = @Tr_id
                where Ins_id = @Ins_Id
                --success message after updating
                SELECT 'Data updated successfully into Instructor table.' AS Message
            end
            else
            begin
                -- If Track ID does not exist
                select 'Error: The new Track ID does not exist in the Track table.' AS Message
            end
        end
        ELSE
        begin
            -- If Instructor ID does not exist
            select 'Error: The Instructor ID does not exist in the Instructor table.' AS Message;
        end
    end Try
    begin Catch
        -- Handle any errors that occur during the procedure
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage;
    end Catch
end;
GO
```

Uses

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

[dbo].[UpdateQuesChoice]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@Ques_id	int	4
@Choice	varchar(50)	50
@New_Choice	varchar(50)	50

SQL Script

```
create Proc [dbo].[UpdateQuesChoice]
    @Ques_id int,
    @Choice varchar(50),
    @New_Choice varchar(50)
as
begin
    begin try
        -- Check if the Ques_id exists
        if not exists (select 1 from Question where Ques_id = @Ques_id)
        begin
            select 'Error: Question ID does not exist in the Question table.' AS Message;
            return;
        end
        -- Check if the combination of Ques_id and Choice exists
        if not exists (select 1 from Ques_choice where Ques_id = @Ques_id and choise = @Choice)
        begin
            select 'Error: The choice does not exist for this Question ID.' AS Message;
            return;
        end
        -- Check if the new choice already exists
        if exists (select 1 from Ques_choice where Ques_id = @Ques_id and choise = @New_Choice)
        begin
            select 'Error: The new choice already exists for this Question ID.' AS Message;
            return;
        end

        -- Update the choice
        update Ques_choice
        set choise = @New_Choice
        where Ques_id = @Ques_id and choise = @Choice;

        select 'Data updated successfully in Ques_choice table.' AS Message;
    end try
    begin catch
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage;
    end catch
end;
GO
```

Uses

[dbo].[Ques_choice]
[dbo].[Question]

[dbo].[UpdateQuestion]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@Ques_id	int	4
@Ques_content	varchar(500)	500
@Type	varchar(50)	50
@Correct_ans	varchar(500)	500
@Ques_point	int	4
@Cr_id	int	4

SQL Script

```
create Proc [dbo].[UpdateQuestion]
    @Ques_id INT,
    @Ques_content VARCHAR(500),
    @Type VARCHAR(50),
    @Correct_ans VARCHAR(500),
    @Ques_point INT = NULL,
    @Cr_id INT = NULL
as
begin
    begin try
        -- Check if the Ques_id already exists
        if not exists (select 1 from Question where Ques_id = @Ques_id)
        begin
            select 'Error: Question ID does not exist in the Question table.' AS Message;
            return;
        end

        --check if it exists in Course table
        if @Cr_id is not Null and not exists (select 1 from Course where Cr_id = @Cr_id)
        begin
            select 'Error: Course ID does not exist in the Course table.' AS Message;
            return;
        end
        -- Update the question details
        update Question
        set Ques_content = @Ques_content,
            Type = @Type,
            Correct_ans = @Correct_ans,
            Ques_point = @Ques_point,
            Cr_id = @Cr_id
        WHERE Ques_id = @Ques_id

        select 'Data updated successfully in Question table.' AS Message;
    end try
    begin catch
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage
    end catch
end;
GO
```

Uses

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

[dbo].[UpdateStAnswer]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@St_id	int	4
@Ex_id	int	4
@Ques_id	int	4
@Answer	varchar(50)	50
@Points	int	4

SQL Script

```
create Proc [dbo].[UpdateStAnswer]
    @St_id INT,
    @Ex_id INT,
    @Ques_id INT,
    @Answer VARCHAR(50),
    @Points INT
as
begin
    begin try
        -- Check if the combination of St_id, Ex_id, and Ques_id already exists
        if not exists (select 1 from St_answer where St_id = @St_id
                                    and Ex_id = @Ex_id
                                    and Ques_id = @Ques_id)
        begin
            select 'Error: The answer does not exist for this Student, Exam, and Question.' AS
Message;
            return;
        end

        -- Update the answer for the student
        UPDATE St_answer
        set answer = @Answer,
            points = @Points
        where St_id = @St_id AND Ex_id = @Ex_id AND Ques_id = @Ques_id;

        select 'Data updated successfully in St_answer table.' AS Message;
    end try
    begin catch
        select 'An error occurred: ' + ERROR_MESSAGE() AS ErrorMessage
    end catch
end;
GO
```

Uses

[dbo].[St_answer]

[dbo].[UpdateStudent]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@Sid	int	4
@St_FirstName	varchar(20)	20
@St_LastName	varchar(20)	20
@age	int	4
@address	varchar(20)	20
@TrId	int	4
@joinDate	date	3
@duration	int	4
@gender	varchar	1

SQL Script

```
create proc [dbo].[UpdateStudent]
(
@Sid int , @St_FirstName varchar (20) , @St_LastName varchar (20),
@age int , @address varchar(20)= NULL , @TrId int , @joinDate date = NULL ,
@duration int , @gender varchar(1)
)

as
    begin
        begin try
        if not exists (select 1 from student where St_id = @Sid)
            begin
                Print 'Error Because StId You Enter Is Not exsits in Table Student '
                return;
            end
            if not exists (select 1 from Track where Tr_id = @TrId)
                print 'Error Because TrId You Enter Is Not exsits in Table Track';
            if @gender != 'M' and @gender != 'F'
                print 'Error Because You Must Enter "M" Or "F"';
            update Student
            set St_fname = @St_FirstName , St_lname = @St_LastName ,
            age = @age , address = @address , Tr_id = @TrId , Join_date = @joinDate,
            [duration(M)] = @duration , gender = @gender
            where St_id = @Sid
        end try
        begin catch
            print 'An Error Occured' + Error_Message();
        end catch
    end;
GO
```

Uses

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

[dbo].[UpdateTopic]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@TopID	int	4
@TopName	varchar(20)	20
@CrID	int	4

SQL Script

```
create Proc [dbo].[UpdateTopic]
(@TopID int,@TopName varchar(20) , @CrID int)
as
    begin
        begin try
                -- Check if the record with the given ID exists
        if not exists(SELECT 1 FROM Topic WHERE Top_id= @TopID)
            BEGIN
                PRINT 'Error: Record with the given ID does not exist';
            RETURN;
            END
            if not Exists(Select 1 from Course where cr_id = @CrID)
                begin
                    print 'Error Becasue This CrId Is not Exsits In Table Cousre';
                    return;
                end
                update Topic
                set Top_name = @TopName , Cr_id = @CrID
                where Top_id =@TopID
        end try
        begin catch
            print 'Error Occurred: ' + ERROR_MESSAGE();
        end catch
    end;
GO
```

Uses

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

[dbo].[UpdateTrack]

Properties

Property	Value
ANSI Nulls On	True
Quoted Identifier On	True

Parameters

Name	Data Type	Max Length (Bytes)
@TrID	int	4
@TrName	varchar(20)	20
@TrDecs	varchar(20)	20
@ManagerID	int	4
@HireDate	date	3

SQL Script

```
create Proc [dbo].[UpdateTrack]
(@TrID int , @TrName varchar (20) , @TrDecs varchar(20) , @ManagerID int , @HireDate date)
as
    begin
        begin try
                -- Check if the record with the given ID exists
        if not exists(SELECT 1 FROM Track WHERE Tr_id= @TrID)
            BEGIN
                PRINT 'Error: Record with the given ID does not exist';
            RETURN;
            END
            if not Exists(Select 1 from Instructor where Ins_id = @ManagerID)
                begin
                    print 'Error Becasue This InsID Is not Exsits In Table Instrcutors';
                    return;
                end
                update Track
                set Tr_name = @TrName, Tr_decs= @TrDecs , manager_id = @ManagerID ,hire_date =
@HireDate
                where Tr_id =@TrID
        end try
        begin catch
            print 'Error Occurred: ' + ERROR_MESSAGE();
        end catch
    end;
```

```
GO
```

Uses

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

User-Defined Table Types

Objects

Name
dbo.ExamAnswersType

[dbo].[ExamAnswersType]

Properties

Property	Value
Collation	SQL_Latin1_General_CP1_CI_AS
Heap	True

Columns

Name	Data Type	Max Length (Bytes)	Nullability
QuesID	int	4	NULL allowed
Answer	varchar(50)	50	NULL allowed

SQL Script

```
CREATE TYPE [dbo].[ExamAnswersType] AS TABLE
(
[QuesID] [int] NULL,
[Answer] [varchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
)
GO
```

Users

Objects

Name
dbo
guest

dbo

Properties

Property	Value
Type	WindowsUser
Login Name	DESKTOP-2Q5RJ97\SoftLaptop
Default Schema	dbo

Database Level Permissions

Type	Action
CONNECT	Grant

SQL Script

```
GO
```

Project > . > User databases > ITIProject > Security > Users > guest

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