# IST769 Homework 3 Submission

## Basic Information

Your Name: Srihari Busam   
Your SUID: sbusam  
Your Email: sbusam@syr.edu  
Date Due: 10/21/2021  
Homework #: 3

## QUESTIONS:

1. In the demo database, create two tables:
   1. The first table **players** should have columns player id (int pk), player name (varchar), shots attempted (int) shots made (int)
   2. The second table **shots** should have columns shot id (int pk), player id (int fk to players), clock time (datetime) shot made (bit)
   3. Add two players to the players table. Mary and Sue initialize the players with 0 shots attempted and made.
2. Write transaction safe code as a stored procedure which when given a player id, clock time, and whether the shot was made (bit value) will add the record to the **shots** table and update the player record in the **players** table. For example, If Mary takes a shot and makes it, then misses the next one, there would be two records in the **shots** table and her row in the **players** table should have 2 attempt and 1 shot made. Execute the stored procedure to demonstrate the transaction is ACID compliant.
3. Alter the **players** table to be a system-versioned temporal table.
4. Execute your stored procedure from part 2 to create at least 15 shot records over a 5-minute period. Make sure there are records in the first ½ of the 5-minute period and at few in the last minute of the 5-minute period.
5. Write SQL queries to show:
   1. The player statistics at the end of the 5-minute period (current statistics).
   2. The player statistics exactly 2 minutes and 30 seconds into the period.
   3. The player statistics in the last minute of the period.

## ANSWERS:

1. Screen shot to show the results of the creation of tables with appropriate primary and foreign keys

A screenshot of a computer

Description automatically generated with medium confidence

|  |
| --- |
| Code for First question |
| USE demo;  GO  DROP TABLE IF EXISTS dbo.shots;  GO  IF OBJECT\_ID(N'dbo.players\_history', N'U') IS NOT NULL  ALTER TABLE dbo.players SET ( SYSTEM\_VERSIONING = OFF)  GO  DROP TABLE IF EXISTS dbo.players\_history;  GO  DROP TABLE IF EXISTS dbo.players;  GO  CREATE TABLE dbo.players (  player\_id int NOT NULL PRIMARY KEY,  player\_name varchar(255) NOT NULL,  shots\_attempted int NOT NULL,  shots\_made int NOT NULL  );  GO  CREATE TABLE dbo.shots (  shot\_id int NOT NULL IDENTITY PRIMARY KEY,  player\_id int FOREIGN KEY REFERENCES dbo.players(player\_id),  clock\_time DATETIME2 NOT NULL,  shot\_made bit NOT NULL  );  GO  INSERT INTO dbo.players (player\_id, player\_name, shots\_attempted, shots\_made)  VALUES (1, 'Mary', 0 , 0);  GO  INSERT INTO dbo.players (player\_id, player\_name, shots\_attempted, shots\_made)  VALUES (2, 'Sue', 0 , 0);  GO  SELECT \* FROM dbo.players  SELECT \* FROM dbo.shots  GO |

1. Screenshot for stored procedure code and execution for Mary with first time shot made and second time shot did not made. Show the tables after running the stored procedure

Graphical user interface

Description automatically generated

|  |
| --- |
| Transactioinal stored procedure code |
| DROP PROCEDURE IF EXISTS dbo.p\_add\_player\_shot;  GO  CREATE PROCEDURE dbo.p\_add\_player\_shot  (  @player\_id int,  @clock\_time DATETIME2,  @shot\_made bit  )  AS  BEGIN  BEGIN TRY  BEGIN TRANSACTION    -- Insert into shots table  INSERT INTO dbo.shots (player\_id, clock\_time, shot\_made)  VALUES (@player\_id, @clock\_time, @shot\_made);  -- update the player table  UPDATE dbo.players  SET shots\_attempted = shots\_attempted + 1, -- increment the shots attempted by 1  shots\_made = shots\_made + CAST(@shot\_made AS BIT) -- Add 1 to shots\_made if the @shot\_made is set.  WHERE player\_id = @player\_id;  IF @@ROWCOUNT <> 1 THROW 50100 , 'players table is not updated', 0;    COMMIT TRANSACTION;  PRINT 'Commited transaction';  END TRY  BEGIN CATCH  ROLLBACK TRANSACTION;  SELECT error\_number() as error, error\_message() as message;  PRINT 'Rolled back transaction';  THROW;  END CATCH  END;  GO  -- first call with shot made  EXEC dbo.p\_add\_player\_shot 1, '2021-10-01', 1;  GO  --second call with shot did not made  EXEC dbo.p\_add\_player\_shot 1, '2021-10-02', 0;  GO  SELECT \* FROM dbo.players;  SELECT \* FROM dbo.shots;  GO |

1. Alter the **players** table to be a system-versioned temporal table

Graphical user interface, text, application

Description automatically generated

|  |
| --- |
| Code |
| -- Add required columns for versioning  ALTER TABLE dbo.players  ADD  valid\_from datetime2 (2) GENERATED ALWAYS AS ROW START HIDDEN  CONSTRAINT df\_valid\_from DEFAULT DATEADD(second, -1, SYSUTCDATETIME()),  valid\_to datetime2 (2) GENERATED ALWAYS AS ROW END HIDDEN  CONSTRAINT df\_valid\_to DEFAULT '9999.12.31 23:59:59.99',  PERIOD FOR SYSTEM\_TIME (valid\_from, valid\_to);  GO  -- Add history table  ALTER TABLE dbo.players  SET( SYSTEM\_VERSIONING = ON (HISTORY\_TABLE = dbo.players\_history));  GO  SELECT \* FROM dbo.players  SELECT \* FROM dbo.players\_history  GO |

1. Execute your stored procedure from part 2 to create at least 15 shot records over a 5-minute period. Make sure there are records in the first ½ of the 5-minute period and at few in the last minute of the 5-minute period.

Graphical user interface, application

Description automatically generated

|  |
| --- |
| CODE |
| /\*  Loop for 5 minutes to create synthetic data.  On each iteration player is switched and the shot\_made is flipped and data will be  inserted in the table using the stored procedure  \*/  DECLARE @EndTime as datetime2 = DATEADD(minute, 5, GETDATE())  DECLARE @clock\_time datetime2  DECLARE @shot\_made\_player1 BIT = 0  DECLARE @shot\_made\_player2 BIT = 0  DECLARE @shot\_made BIT = 0  DECLARE @player\_id INT = 1  WHILE GETDATE() < @EndTime  BEGIN  -- Using the current time as the clock\_time  SET @clock\_time = GETDATE()    -- switch player and shot made every time  IF @player\_id = 1  BEGIN  SET @player\_id = 2;  SET @shot\_made\_player2 = ~@shot\_made\_player2;  SET @shot\_made = @shot\_made\_player2;  END  ELSE  BEGIN  SET @player\_id = 1;  SET @shot\_made\_player1 = ~@shot\_made\_player1;  SET @shot\_made = @shot\_made\_player1;  END    -- Switch whether shaot made or not  SET @shot\_made = ~@shot\_made  -- Add data  EXEC dbo.p\_add\_player\_shot @player\_id, @clock\_time, @shot\_made;  -- wair for 15 sec  WAITFOR DELAY '00:00:15'  END  GO  SELECT \* FROM dbo.players  SELECT \* FROM dbo.players\_history  GO |

1. Write SQL queries to show:
   1. The player statistics at the end of the 5-minute period (current statistics).

**For current stats we can just do a select on the players table**

Graphical user interface, text, application

Description automatically generated

* 1. The player statistics exactly 2 minutes and 30 seconds into the period.

Graphical user interface, text, application

Description automatically generated

|  |
| --- |
| CODE |
| ---5.b The player statistics exactly 2 minutes and 30 from putting the data into history table  DECLARE @start\_time datetime2 = '2021-10-16 05:30:58.04'  -- Target time is 2.5 min from start time  DECLARE @target\_time datetime2 = DATEADD(second, 150, @start\_time)  SELECT \* from dbo.players FOR SYSTEM\_TIME AS OF @target\_time  GO |

* 1. The player statistics in the last minute of the period

Text

Description automatically generated with low confidence

|  |
| --- |
| CODE |
| --5.c Query for last 1 min of history  -- start time of history entry table  DECLARE @start\_history\_time datetime2 = '2021-10-16 05:30:58.04'  -- Add 4 mins to start of the time and store in between\_startTime variable  DECLARE @between\_startTime datetime2 = DATEADD(minute, 4, @start\_history\_time)  -- Add 5 mins to the start of time and store in @between\_end\_time variable  DECLARE @between\_end\_time datetime2 = DATEADD(minute, 5, @start\_history\_time)  -- Query the last min data  SELECT \* FROM dbo.players  FOR SYSTEM\_TIME BETWEEN @between\_startTime AND @between\_end\_time  ORDER BY player\_id ASC, shots\_made DESC;  GO |