```
-- This procedure takes EventID and OlympicID as input to obtain the Event Schedule for that particular Olympic --

GO
CREATE OR ALTER PROCEDURE [dbo].[EventSchedule]
@EventID INTEGER,
@OlympicID INTEGER,
@Olympic VARCHAR(50) OUTPUT,
@Event VARCHAR(50) OUTPUT,
@Sport VARCHAR(50) OUTPUT
AS

IF NOT EXISTS (SELECT * FROM Event WHERE EventID = @EventID)
BEGIN
   PRINT('Wrong EventID');
END

ELSE IF NOT EXISTS (SELECT * FROM Olympic WHERE OlympicID = @OlympicID)
BEGIN
   PRINT('Wrong OlympicID');
END

ELSE IF NOT EXISTS (SELECT * FROM EventMatch WHERE OlympicID = @OlympicID AND EventID = @EventID)
BEGIN
  PRINT('Olympic didnot had this Event');
END

ELSE

BEGIN

   SELECT EventPhase, CountryCode, Score, [Position], VenueName, MatchDate, OlympicDay FROM (MatchResult
   JOIN EventMatch ON MatchResult.MatchID = EventMatch.MatchID)
   JOIN Venue ON Venue.VenueID = EventMatch.VenueID
   JOIN [Group] ON [Group].GroupID = MatchResult.GroupID
   WHERE EventID = @EventID AND OlympicID = @OlympicID
   ORDER BY OlympicDay;

   SELECT  @Olympic = Season + ' ' + City + ' ' + Cast(Year([Year]) as varchar) FROM Olympic WHERE OlympicID = @OlympicID
   SELECT @Event = EventName FROM Event WHERE Event.EventID = @EventID
```

```
   SELECT @Sport = SportName FROM Sport JOIN Event ON Sport.SportID = Event.SportID
WHERE Event.EventID = @EventID

END

DECLARE @OlympicDetails VARCHAR(50)
DECLARE @EventDetails VARCHAR(50)
DECLARE @SportDetails VARCHAR(50)

EXEC EventSchedule @EventID=203, @OlympicID=12,
   @Olympic = @OlympicDetails OUTPUT,
   @Event = @EventDetails OUTPUT,
   @Sport =  @SportDetails OUTPUT;

SELECT @OlympicDetails AS 'Olympic', @SportDetails AS 'Sport', @EventDetails AS
'EventName';

-- This procedure takes EventID and OlympicID as input to obtain the Medal Winners List for
that particular Olympic and Event Combination --

GO
CREATE OR ALTER PROCEDURE [dbo].[EventWiseMedalWinners]
@EventID INTEGER,
@OlympicID INTEGER,
@Olympic VARCHAR(50) OUTPUT,
@Event VARCHAR(50) OUTPUT,
@Sport VARCHAR(50) OUTPUT
AS

IF NOT EXISTS (SELECT * FROM Event WHERE EventID = @EventID)
BEGIN
   PRINT('Wrong EventID');
END

ELSE IF NOT EXISTS (SELECT * FROM Olympic WHERE OlympicID = @OlympicID)
BEGIN
   PRINT('Wrong OlympicID');
END

ELSE IF NOT EXISTS (SELECT * FROM EventMatch WHERE OlympicID = @OlympicID AND EventID
= @EventID)
BEGIN
  PRINT('Olympic didnot had this Event');
END
```

```
ELSE

BEGIN

DECLARE @EventType VARCHAR(10)

SELECT @EventType = (SELECT EventType FROM Event WHERE EventID = @EventID)

IF @EventType = 'Individual'

   BEGIN

      SELECT Athlete.CountryCode, Athlete.AthleteName, MedalType  FROM ((Medal
      JOIN EventMatch ON EventMatch.MatchID = Medal.MatchID)
      JOIN GroupAthlete ON GroupAthlete.GroupID = Medal.GroupID)
      JOIN Athlete ON Athlete.AthleteID = GroupAthlete.AthleteID
      JOIN Olympic ON Olympic.OlympicID = EventMatch.OlympicID
      JOIN Event ON EventMatch.EventID = Event.EventID
      JOIN Sport ON Sport.SportID = Event.SportID
      WHERE EventMatch.EventID = @EventID AND Olympic.OlympicID = @OlympicID;

     SELECT  @Olympic = Season + ' ' + City + ' ' + Cast(Year([Year]) as varchar) FROM Olympic
WHERE OlympicID = @OlympicID
      SELECT @Event = EventName FROM Event WHERE Event.EventID = @EventID
      SELECT @Sport = SportName FROM Sport JOIN Event ON Sport.SportID = Event.SportID
WHERE Event.EventID = @EventID

   END

IF @EventType = 'Team'

   BEGIN

      SELECT [Group].CountryCode, MedalType FROM ((Medal
      JOIN EventMatch ON EventMatch.MatchID = Medal.MatchID )
      JOIN [Group] ON [Group].GroupID = Medal.GroupID)
      JOIN Event ON EventMatch.EventID = Event.EventID
      JOIN Olympic ON Olympic.OlympicID = EventMatch.OlympicID
      JOIN Sport ON Sport.SportID = Event.SportID
      WHERE EventMatch.EventID = @EventID AND Olympic.OlympicID = @OlympicID;

      SELECT  @Olympic = Season + ' ' + City + ' ' + Cast(Year([Year]) as varchar) FROM Olympic
WHERE OlympicID = @OlympicID
```

```
      SELECT @Event = EventName FROM Event WHERE Event.EventID = @EventID
      SELECT @Sport = SportName FROM Sport JOIN Event ON Sport.SportID = Event.SportID
WHERE Event.EventID = @EventID

   END

END

DECLARE @OlympicDetails VARCHAR(50)
DECLARE @EventDetails VARCHAR(50)
DECLARE @SportDetails VARCHAR(50)

EXEC EventWiseMedalWinners @EventID=122, @OlympicID=10,
   @Olympic = @OlympicDetails OUTPUT,
   @Event = @EventDetails OUTPUT,
   @Sport =  @SportDetails OUTPUT;

SELECT @OlympicDetails AS 'Olympic', @SportDetails AS 'Sport', @EventDetails AS
'EventName';


-- This procedure takes OlympicID as input to generate the Medal Tally for that Olympic --

GO
CREATE OR ALTER PROCEDURE [dbo].[OlypicMedalTable]
@OlympicID INTEGER
AS

IF NOT EXISTS (SELECT * FROM Olympic WHERE OlympicID = @OlympicID)
BEGIN
   PRINT('Wrong OlympicID');
END

ELSE

BEGIN

   IF  EXISTS (SELECT 1 FROM sys.objects WHERE object_id = OBJECT_ID(N'[dbo].[MedalTally]')
AND type in (N'U'))
   DROP TABLE [dbo].[MedalTally]

   CREATE Table MedalTally(
   CountryName VARCHAR(50),
   Gold INTEGER,
```

```
  Silver INTEGER,
  Bronze INTEGER,
  Total INTEGER);

  CREATE CLUSTERED INDEX MedalTally_Index
  ON MedalTally(Total DESC, Gold DESC, Silver DESC, Bronze DESC)

  DECLARE

     @CountryName VARCHAR(50),
     @CountryCode CHAR(3),
     @Gold INTEGER,
     @Silver INTEGER,
     @Bronze INTEGER,
     @Total INTEGER,
     @Count INTEGER

  SELECT @Count = (SELECT COUNT(*) from Country)
  DECLARE CountryCursor CURSOR for SELECT CountryCode FROM Country

  OPEN CountryCursor

  WHILE @Count > 0

    BEGIN

      FETCH CountryCursor INTO @CountryCode

      SELECT @CountryName = CountryName FROM Country WHERE CountryCode =
@CountryCode;
      SET @Gold = 0
      SET @Silver= 0
      SET @Bronze = 0
      SET @Total = 0

      DECLARE @GroupCount INTEGER

      SELECT @GroupCount = (SELECT COUNT(*) FROM [Group] WHERE CountryCode =
@CountryCode);

      DECLARE GroupCursor CURSOR for SELECT GroupID FROM [Group] WHERE CountryCode =
@CountryCode

      OPEN GroupCursor
```

```
    WHILE @GroupCount > 0

        BEGIN

        DECLARE @GroupID INTEGER

        FETCH GroupCursor INTO @GroupID

        DECLARE @MedalCount INTEGER

        SELECT @MedalCount = (SELECT COUNT(*) FROM Medal JOIN EventMatch ON
Medal.MatchID = EventMatch.MatchID
                    WHERE GroupID = @GroupID AND OlympicID = @OlympicID)

        DECLARE MedalCursor CURSOR for SELECT MedalID FROM Medal JOIN EventMatch ON
Medal.MatchID = EventMatch.MatchID
                        WHERE GroupID = @GroupID AND OlympicID = @OlympicID

        OPEN MedalCursor

        WHILE @MedalCount > 0

          BEGIN

          DECLARE @MedalID INTEGER

          FETCH MedalCursor INTO @MedalID

          DECLARE @MedalType VARCHAR(10)

          SELECT @MedalType = (SELECT MedalType FROM Medal WHERE MedalID =
@MedalID)

            IF @MedalType = 'Gold'
              BEGIN
                SET @Gold = @Gold + 1;
              END

            IF @MedalType = 'Silver'
            BEGIN
              SET @Silver = @Silver + 1;
            END;
```

```
            IF @MedalType = 'Bronze'
            BEGIN
               SET @Bronze = @Bronze + 1;
            END;

          SET @MedalCount = @MedalCount - 1

          END

        CLOSE MedalCursor

        DEALLOCATE MedalCursor

        SET @GroupCount = @GroupCount - 1

        END

      CLOSE GroupCursor

      DEALLOCATE GroupCursor

      SET @Total = @Gold + @Silver + @Bronze

      INSERT INTO MedalTally VALUES (@CountryName, @Gold, @Silver, @Bronze, @Total);

      SET @Count = @Count - 1

    END

  CLOSE CountryCursor

  DEALLOCATE CountryCursor

   SELECT * FROM MedalTally;

END

EXEC OlypicMedalTable @OlympicID=12;

-- Create View to Display Event Match Details

GO
CREATE OR ALTER VIEW [dbo].[EventMatchDetails] AS
```

```
SELECT SportName, EventName, EventType, EventSex, EventPhase, City, Year([Year]) as
Olympic_Year, VenueName
FROM Event JOIN Sport ON Event.SportID = Sport.SportID
JOIN EventMatch ON Event.EventID = EventMatch.EventID
JOIN Olympic ON EventMatch.OlympicID = Olympic.OlympicID
JOIN Venue ON Venue.VenueID = EventMatch.VenueID;
GO

SELECT * FROM EventMatchDetails

-- Create View to List all the Medal Winners for a particular Country in selected Olympic Year

GO
CREATE OR ALTER VIEW vw_MedalWinners AS
SELECT Athlete.AthleteName, Sport.SportName, Event.EventName, MedalType, Olympic.City,
Olympic.Season, Year(Olympic.Year) as Olympic_Year
FROM Medal JOIN EventMatch ON Medal.MatchID = EventMatch.MatchID
JOIN Event ON EventMatch.EventID = Event.EventID
JOIN Sport ON Event.SportID = Sport.SportID
JOIN GroupAthlete ON GroupAthlete.GroupID = Medal.GroupID
JOIN [Group] ON [Group].GroupID = GroupAthlete.GroupID
JOIN Athlete ON Athlete.AthleteID = GroupAthlete.AthleteID
JOIN Olympic ON Olympic.OlympicID = EventMatch.OlympicID
WHERE [Group].CountryCode = 'USA' AND Year(Olympic.Year) = '2016';

GO

SELECT * FROM vw_MedalWinners;

-- Create View to track all the Records broken in a particular Olympic Year

GO
CREATE OR ALTER VIEW vw_records AS
SELECT AthleteName, CountryCode, SportName, EventName, EventPhase, RecordType,
RecordScore, ScoreType, MatchDate
FROM Record JOIN EventMatch ON Record.MatchID = EventMatch.MatchID
JOIN GroupAthlete ON Record.GroupID = GroupAthlete.GroupID
JOIN Athlete ON Athlete.AthleteID = GroupAthlete.AthleteID
JOIN Event ON Event.EventID = EventMatch.EventID
JOIN Sport ON Event.SportID = Sport.SportID
WHERE YEAR(MatchDate) = '2016'
WITH CHECK OPTION;
GO
```

```
SELECT * FROM vw_records;



-- Create a trigger to avoid the conflict of Sport Name

GO
CREATE OR ALTER TRIGGER ConflictSportName
ON Sport
AFTER INSERT
AS
BEGIN
DECLARE @name VARCHAR(30)
SELECT @name = SportName FROM inserted

IF EXISTS (SELECT * FROM Sport WHERE SportName = @name)
BEGIN

   RAISERROR('Sport already exists. Please check',1,1)
   DELETE FROM Sport WHERE SportID = (SELECT SportID FROM inserted)

END

END

INSERT INTO Sport (SportName) VALUES ('Volleyball')
SELECT * FROM Sport

-- Create a trigger to capture changes on Medal Table and Update Medal Tally

GO
CREATE OR ALTER TRIGGER updateMedalTally
ON Medal
AFTER INSERT, UPDATE, DELETE
AS
BEGIN

   DECLARE @Action as CHAR(1);

   SET @Action = (CASE WHEN EXISTS(SELECT * FROM INSERTED)
                   AND EXISTS(SELECT * FROM DELETED)
                  THEN 'U'  -- Set Action to Updated.
                  WHEN EXISTS(SELECT * FROM INSERTED)
                  THEN 'I'  -- Set Action to Insert.
```

```
            WHEN EXISTS(SELECT * FROM DELETED)
            THEN 'D'  -- Set Action to Deleted.
          END)

   DECLARE @Olympic INTEGER

   IF @Action = 'I'
   BEGIN

      SELECT @Olympic = EventMatch.OlympicID FROM inserted JOIN EventMatch ON
inserted.MatchID = EventMatch.MatchID;
      EXEC OlypicMedalTable @OlympicID = @Olympic;

   END

   IF @Action = 'D'
   BEGIN

      SELECT @Olympic = EventMatch.OlympicID FROM deleted JOIN EventMatch ON
deleted.MatchID = EventMatch.MatchID;
      EXEC OlypicMedalTable @OlympicID = @Olympic;

   END

   IF @Action = 'U'
   BEGIN

      SELECT @Olympic = EventMatch.OlympicID FROM inserted JOIN EventMatch ON
inserted.MatchID = EventMatch.MatchID;
      EXEC OlypicMedalTable @OlympicID = @Olympic;

   END


END

INSERT INTO Medal (MedalType, MatchID, GroupID) VALUES ('Bronze', '49', '50');
UPDATE Medal SET MedalType='Gold' WHERE MedalID = 79;
DELETE FROM Medal WHERE MedalID = 79
```