**Assignment # 3**

**Ali Asif 22i-1019**

**Muhammad Taha 22i-0870**

**Q1**

-- Create all required tables (other than the 2 already shared) in SQL and then insert at least 20 dummy data into each table.

Note : The TEAM & PLAYER tables were created using import as instructed.

-- TEAM

-- Primary Key : TNAME

-- PLAYER

-- Primary Key : PLAYER\_ID

-- STADIUM

-- Primary Key : STADIUM\_ID

CREATE TABLE STADIUM

(

STADIUM\_ID NUMERIC(2) NOT NULL,

STADIUM\_NAME VARCHAR(50) NOT NULL,

LOC VARCHAR(20) NOT NULL,

CAPACITY NUMERIC(7) NOT NULL,

PRIMARY KEY (STADIUM\_ID)

);

INSERT INTO STADIUM

VALUES

(1, 'National Stadium', 'Karachi', 30000),

(2, 'Gaddafi Stadium', 'Lahore', 27000),

(3, 'Rawalpindi Cricket Stadium', 'Rawalpindi', 25000),

(4, 'Multan Cricket Stadium', 'Multan', 35000),

(5, 'Arbab Niaz Stadium', 'Peshawar', 35000),

(6, 'Bugti Stadium', 'Quetta', 35000);

-- MATCH

-- Primary Key : MATCH\_ID

-- Foreign Key : STADIUM\_ID, TNAME

CREATE TABLE MATCHES

(

MATCH\_ID NUMERIC(2) NOT NULL,

MATCH\_DATE DATE NOT NULL,

MATCH\_TIME TIME NOT NULL,

STADIUM\_ID NUMERIC(2) NOT NULL,

MATCH\_TYPE VARCHAR(20) NOT NULL,

TEAM1\_NAME NVARCHAR(20) NOT NULL,

TEAM2\_NAME NVARCHAR(20) NOT NULL,

PRIMARY KEY (MATCH\_ID),

FOREIGN KEY (STADIUM\_ID) REFERENCES STADIUM(STADIUM\_ID),

FOREIGN KEY (TEAM1\_NAME) REFERENCES TEAM(TNAME),

FOREIGN KEY (TEAM2\_NAME) REFERENCES TEAM(TNAME),

);

INSERT INTO MATCHES

VALUES

(1, '2024-03-25', '18:00:00', 1, 'Round Robin', 'Karachi Kings', 'Lahore Qalandars'),

(2, '2024-03-26', '18:00:00', 2, 'Round Robin', 'Islamabad United', 'Peshawar Zalmi'),

(3, '2024-03-27', '18:00:00', 3, 'Round Robin', 'Multan Sultans', 'Islamabad United'),

…

(25, '2024-04-10', '18:00:00', 4, 'Final', 'Islamabad United', 'Multan Sultans');

-- PERFORMANCE

-- Primary Key : PERFORMANCE\_ID

-- Foreign Key : MATCH\_ID, PLAYER\_ID

CREATE TABLE PERFORMANCE

(

PERFORMANCE\_ID NUMERIC(2) NOT NULL,

MATCH\_ID NUMERIC(2) NOT NULL,

PLAYER\_ID INT NOT NULL,

RUNS NUMERIC(4) NOT NULL,

WICKETS NUMERIC(2) NOT NULL,

CATCHES NUMERIC(2) NOT NULL,

PRIMARY KEY (PERFORMANCE\_ID),

FOREIGN KEY (MATCH\_ID) REFERENCES MATCHES(MATCH\_ID),

FOREIGN KEY (PLAYER\_ID) REFERENCES PLAYER(PLAYER\_ID),

)

INSERT INTO PERFORMANCE

VALUES

(1, 1, 1, 129, 1, 4),

(2, 1, 2, 63, 2, 2),

(3, 1, 3, 27, 5, 5),

…

(76, 25, 18, 4, 4, 2);

-- WINNER

-- Foreign Key : TNAME, MATCH\_ID

CREATE TABLE WINNER

(

FOREIGN KEY (MATCH\_ID) REFERENCES MATCHES(MATCH\_ID),

FOREIGN KEY (TNAME) REFERENCES TEAM(TNAME)

)

INSERT INTO WINNER

VALUES

(1, 'Karachi Kings'),

(2, 'Islamabad United'),

(3, 'Multan Sultans'),

…

(25, 'Islamabad United')

-- CALCULATE WINS OF EACH TEAM

-- Concept Used : Nested Query & Aggregate Functions

UPDATE TEAM

SET WINS = (

SELECT COUNT(\*)

FROM WINNER

WHERE WINNER.TNAME = TEAM.TNAME

)

-- CALCULATE LOSSES OF EACH TEAM

-- Concept Used : Nested Query & Aggregate Functions

UPDATE TEAM

SET LOSSES = (

SELECT COUNT(\*)

FROM MATCHES

WHERE (

(MATCHES.TEAM1\_NAME = TEAM.TNAME) OR (MATCHES.TEAM2\_NAME = TEAM.TNAME)

) AND MATCHES.MATCH\_ID NOT IN (

SELECT WINNER.MATCH\_ID

FROM WINNER

WHERE WINNER.TNAME = TEAM.TNAME

)

)

A screenshot of a sports stadium

Description automatically generated

**Q2**

-- Player with the highest number of Catches

-- Concept Used : Nested Query

SELECT NAME AS [HIGHEST CATCHES]

FROM PLAYER

WHERE (

SELECT COUNT(CATCHES)

FROM PERFORMANCE

WHERE PLAYER.PLAYER\_ID = PERFORMANCE.PLAYER\_ID

) = (

SELECT MAX(TOTAL\_CATCHES)

FROM (

SELECT COUNT(CATCHES) AS TOTAL\_CATCHES

FROM PERFORMANCE

GROUP BY PLAYER\_ID

) AS TOTAL\_CATCHES

)



**Q3**

--Team Names along with their Home Stadiums

SELECT TNAME, HOME\_STADIUM

FROM TEAM

****

**Q4**

-- Total capacity of a specific Stadium

SELECT STADIUM\_NAME, CAPACITY

FROM STADIUM

A screenshot of a sports schedule

Description automatically generated

**Q5**

--Average Runs scored by players in each team

--Concept Used : Join

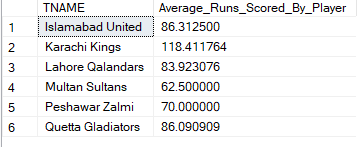
SELECT T.TNAME, AVG(PR.RUNS) AS Average\_Runs\_Scored\_By\_Player

FROM TEAM T

JOIN PLAYER P ON T.TNAME = P.TEAM\_NAME

JOIN PERFORMANCE PR ON P.PLAYER\_ID = PR.PLAYER\_ID

GROUP BY T.TNAME

****

**Q6**

-- Number of Matches played in a specific Stadium

-- Concept Used : Nested Query & Join

SELECT STADIUM\_NAME, [MATCHES PLAYED]

FROM STADIUM

JOIN (

SELECT STADIUM\_ID AS STD\_ID, COUNT(STADIUM\_ID) AS [MATCHES PLAYED]

FROM MATCHES

GROUP BY STADIUM\_ID

) AS STD\_ID

ON STADIUM.STADIUM\_ID = STD\_ID

A screenshot of a sports schedule

Description automatically generated

**Q7**

--Total Runs Scored by each player

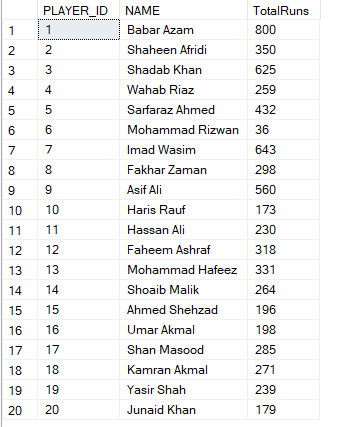
--Concept Used : Join

SELECT P.PLAYER\_ID, P.NAME, SUM(PR.RUNS) AS TotalRuns

FROM PLAYER P

JOIN PERFORMANCE PR ON P.PLAYER\_ID = PR.PLAYER\_ID

GROUP BY P.PLAYER\_ID, P.NAME

****

**Q8**

-- Teams with more than 3 Wins

SELECT TNAME, WINS

FROM TEAM

WHERE WINS > 3

A close-up of a list

Description automatically generated

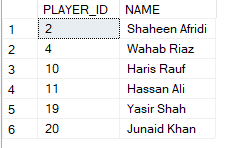
**Q9**

--Players that are bowlers

SELECT P.PLAYER\_ID, P.NAME

FROM PLAYER P

WHERE P.ROLE = 'Bowler'

****

**Q10**

-- List players who scored more than 50 runs and took at least 3 wickets in a single match

-- Concept Used : Nested Query & Join

SELECT NAME, [RUNS], [WICKETS]

FROM (

SELECT NAME, RUNS AS [RUNS], WICKETS AS [WICKETS]

FROM PLAYER

JOIN PERFORMANCE

ON PLAYER.PLAYER\_ID = PERFORMANCE.PLAYER\_ID

) AS NAME

WHERE [RUNS] > 50 AND [WICKETS] > 3

A table of numbers with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

**Q11**

--The team with the highest runs in the tournament

--Concept Used : Join

SELECT TOP 1 T.TNAME, SUM(PR.RUNS) AS Runs

FROM TEAM T

JOIN PLAYER P ON T.TNAME = P.TEAM\_NAME

JOIN PERFORMANCE PR ON P.PLAYER\_ID = PR.PLAYER\_ID

GROUP BY T.TNAME

ORDER BY RUNS DESC

****

**Q12**

-- Show matches along with the winning team name

-- Concept Used : Join

SELECT TNAME AS [WINNER], MATCHES.MATCH\_ID, MATCH\_DATE, MATCH\_TIME, STADIUM\_ID, MATCH\_TYPE

FROM MATCHES

JOIN WINNER

ON WINNER.MATCH\_ID = MATCHES.MATCH\_ID

A table with numbers and numbers

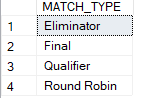
Description automatically generated

**Q13**

--All match types without duplicates

SELECT DISTINCT MATCH\_TYPE

FROM MATCHES

****

**Q14**

-- List stadiums and the number of matches hosted by each

-- Concept Used : Nested Query & Join

SELECT STADIUM\_NAME, [MATCHES HOSTED]

FROM STADIUM

JOIN (

SELECT STADIUM\_ID AS STD\_ID, COUNT(STADIUM\_ID) AS [MATCHES HOSTED]

FROM MATCHES

GROUP BY STADIUM\_ID

) AS STD\_ID

ON STADIUM.STADIUM\_ID = STD\_ID

A screenshot of a sports schedule

Description automatically generated

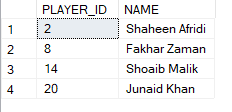
**Q15**

--All Players in a specific team

SELECT PLAYER\_ID, NAME

FROM PLAYER

WHERE TEAM\_NAME = 'Lahore Qalandars'

****

**Q16**

-- List the top 3 players with the most runs scored in final matches

-- Concept Used : Nested Query & Join

SELECT \*

FROM PLAYER

JOIN PERFORMANCE

ON PLAYER.PLAYER\_ID = PERFORMANCE.PLAYER\_ID

WHERE (PERFORMANCE.RUNS IN (

SELECT TOP 3 RUNS

FROM PERFORMANCE

WHERE MATCH\_ID = 25

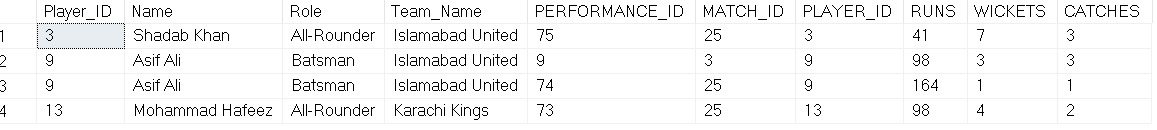
)) AND PLAYER.PLAYER\_ID IN (

SELECT PLAYER\_ID

FROM PERFORMANCE

WHERE MATCH\_ID = 25

) AND (MATCH\_ID = 25)



**Q17**

--The players who scored more than 50 runs in winning matches

--Concept Used : Join

SELECT DISTINCT P.PLAYER\_ID, P.TEAM\_NAME, PR.RUNS

FROM PERFORMANCE PR

JOIN (

SELECT MATCH\_ID

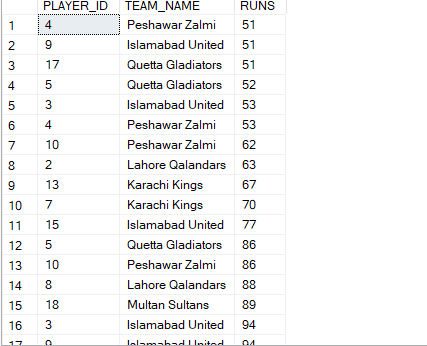
FROM WINNER

) AS W ON PR.MATCH\_ID = W.MATCH\_ID

JOIN Player P ON PR.PLAYER\_ID = P.PLAYER\_ID

JOIN Winner WIN ON P.TEAM\_NAME = WIN.TNAME

WHERE PR.RUNS > 50;

****

**Q18**

-- Determine the top 3 players with the highest aggregate runs scored in Qualifier, Eliminator, and Final matches

-- Concept Used : Nested Query, Join, Group By & Aggregate Functions

SELECT TOP 3 NAME AS [PLAYER], SUM(RUNS) AS [AGGREGATE RUNS]

FROM (

SELECT NAME, MATCH\_ID, RUNS

FROM PERFORMANCE

JOIN PLAYER

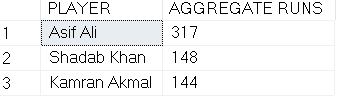
ON PLAYER.PLAYER\_ID = PERFORMANCE.PLAYER\_ID

WHERE MATCH\_ID IN (22, 23, 24, 25)

) AS RES

GROUP BY NAME

ORDER BY SUM(RUNS) DESC



**Q19**

--Average Runs and Wickets scored by each team per stadium

--Concept Used : Join

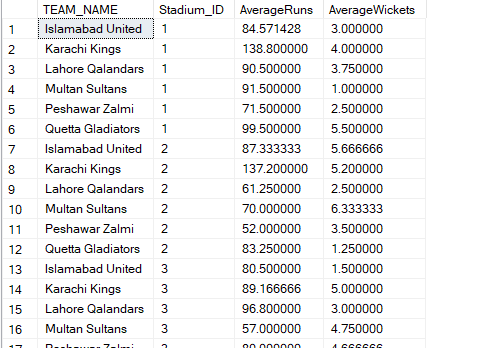
SELECT P.TEAM\_NAME, M.Stadium\_ID, AVG(PR.RUNS) AS AverageRuns, AVG(PR.WICKETS) AS AverageWickets

FROM MATCHES M

JOIN PERFORMANCE PR ON M.MATCH\_ID = PR.MATCH\_ID

JOIN PLAYER P ON PR.PLAYER\_ID = P.PLAYER\_ID

GROUP BY P.TEAM\_NAME, M.Stadium\_ID

****

**Q20**

-- Find the number of wins each team has achieved in their home stadium

-- Concept Used : Nested Query, Join, Group By & Aggregate Functions

SELECT [WINNER] AS [TEAM], STADIUM\_NAME AS [HOME STADIUM], RES2.[WINS]

FROM (

SELECT STADIUM\_NAME, [WINNER], COUNT(\*) AS [WINS]

FROM (

SELECT STADIUM\_NAME, [WINNER]

FROM STADIUM

JOIN (

SELECT STADIUM\_ID AS [STD\_ID], TNAME AS [WINNER]

FROM MATCHES

JOIN WINNER

ON MATCHES.MATCH\_ID = WINNER.MATCH\_ID

) AS STD\_ID

ON STADIUM.STADIUM\_ID = STD\_ID

) AS RES

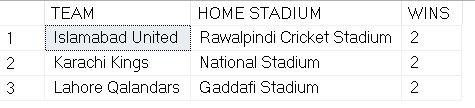
GROUP BY RES.STADIUM\_NAME, RES.WINNER

) AS RES2

JOIN TEAM

ON RES2.STADIUM\_NAME = TEAM.HOME\_STADIUM

WHERE TEAM.HOME\_STADIUM = RES2.STADIUM\_NAME AND [WINNER] = TEAM.TNAME



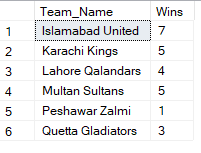
**Q21**

--Current number of wins for each team

SELECT W.TNAME AS Team\_Name, COUNT(\*) AS Wins

FROM WINNER W

GROUP BY W.TNAME

****

**Q22**

-- Identify players whose performance (runs scored, wickets taken, or catches) was pivotal in securing wins for their team

-- Concept Used : Nested Query & Join

SELECT PLAYER.NAME, MATCHES.MATCH\_ID, PERFORMANCE.RUNS, PERFORMANCE.WICKETS, PERFORMANCE.CATCHES, PLAYER.TEAM\_NAME

FROM PERFORMANCE

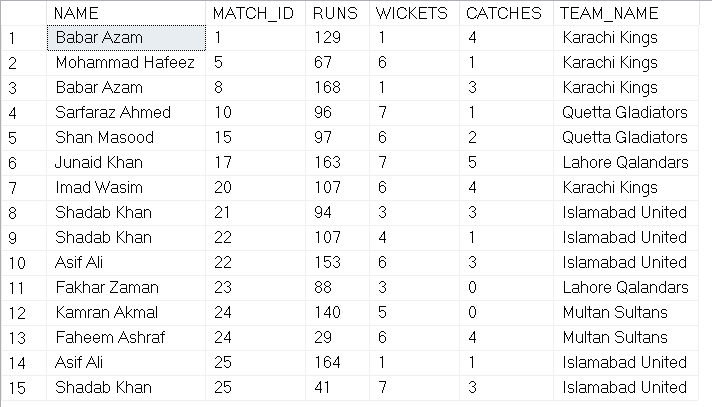
JOIN MATCHES ON MATCHES.MATCH\_ID = PERFORMANCE.MATCH\_ID

JOIN WINNER ON WINNER.MATCH\_ID = PERFORMANCE.MATCH\_ID

JOIN PLAYER ON PLAYER.PLAYER\_ID = PERFORMANCE.PLAYER\_ID

WHERE (MATCHES.TEAM1\_NAME = WINNER.TNAME OR MATCHES.TEAM2\_NAME = WINNER.TNAME) AND (PLAYER.TEAM\_NAME = WINNER.TNAME)

AND ((RUNS > 80) OR (CATCHES > 3) OR (WICKETS > 3))



**Q23**

--Correlated Nested Queries

--Nested Query where the inner query uses value from outer query

SELECT RUNS FROM PERFORMANCE

WHERE PLAYER\_ID IN ( SELECT PLAYER\_ID FROM PLAYER )

--Union

--Combines the values of a column from two different tables without duplicates

SELECT PLAYER\_ID FROM PLAYER

UNION

SELECT PLAYER\_ID FROM PERFORMANCE

--Group By

--Groups the rows that have the same values

SELECT STADIUM\_ID, COUNT(\*) AS Matches

FROM MATCHES

GROUP BY STADIUM\_ID

--Like

--Finds entries with matching substrings

SELECT NAME FROM PLAYER

WHERE NAME LIKE '%E%'

--Having

--Used instead of WHERE because WHERE cannot be used in aggregate

SELECT STADIUM\_ID, COUNT(\*) AS Matches

FROM MATCHES

GROUP BY STADIUM\_ID

HAVING COUNT(\*) > 7

