**Table of Contents**

[**Database Description**](#_h0g6ojts8fwz)[**2**](#_h0g6ojts8fwz)

[**E/R Diagram**](#_c27xu993lv80)[**3**](#_c27xu993lv80)

[**Tables**](#_xhsn5248fxhf)[**4**](#_xhsn5248fxhf)

[**Queries**](#_4jym3gm22awh)[**17**](#_4jym3gm22awh)

# Database Description

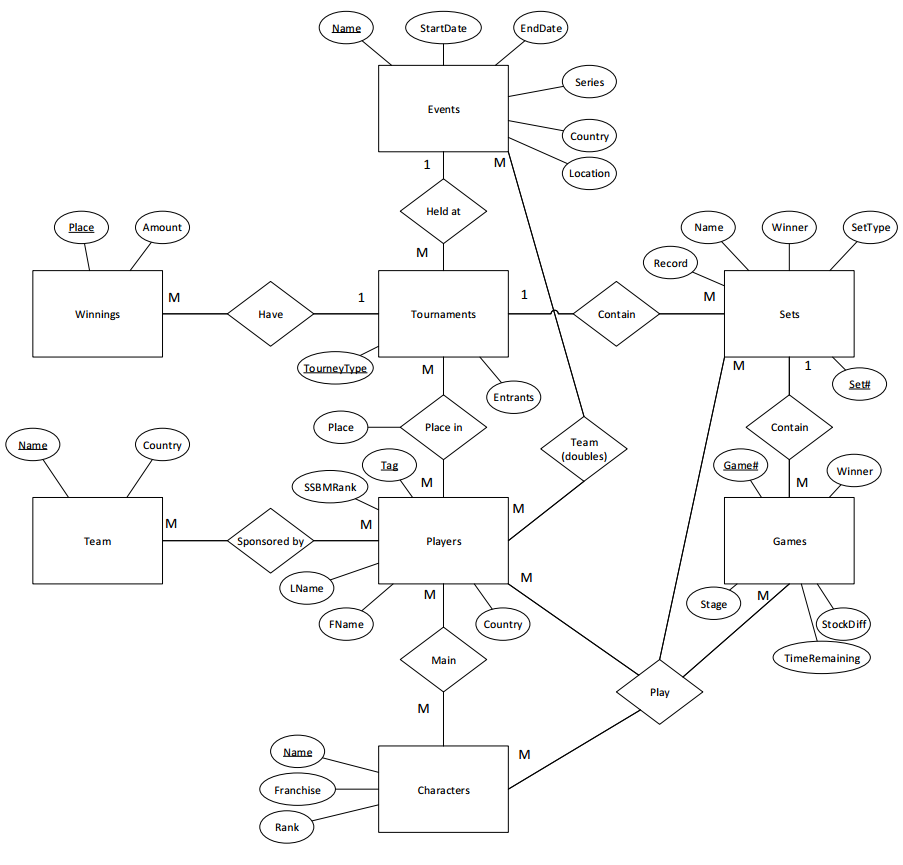
This database models data from the *Super Smash Brothers Melee* competitive scene. *Super Smash Brothers Melee* is a fighting game where each player plays a nintendo character and tries to eliminate all of their opponent's stocks (lives) by knocking them out of the stage. There is a total of 26 different characters, and the community has ranked each character in a tier list based on how good each character is. Each character also originates in another game franchise. The Database should list each character’s name, franchise, and rank.

Each player typically plays only one or two characters most of the time which are referred to as their mains. Players are typically referred to by their tag instead of their real name (ie. Adam Lindgren is called Armada). The top players are also ranked by MIOM (Melee It On Me) at the end of each year. Their ranking is referred to as their SSBMRank. Players can also be sponsored by teams. The database should record each player’s name, tag, SSBMrank (if applicable), age, country of origin, team, and mains. Teams can sponsor multiple players. The database will keep track of each team’s name, and country.

Tournaments are the main competitive events for Melee. Typically two types of tournaments are held at each event, singles are doubles. Also many The database should record the name, type, entrants, series, results, each player’s winnings, location, the start date, and end date of each tournament. The database will only record “Super-Majors.” (<http://wiki.teamliquid.net/smash/Premier_Tournaments>)

Each tournament set is either a best of 3 games or a best of 5 games. For each game, each player starts with 4 stocks, and there is a 8 minute time limit. Also each game is played on one of 6 stages. The database should record the winner of each game and set, the characters used in each game, the time remaining in each game, the stage for each game, and the stock difference of each game

# E/R Diagram



# Tables:

Characters

CREATE TABLE Characters

(

Name VARCHAR(30),

Franchise VARCHAR(30),

Rank INTEGER

);

ALTER TABLE Characters ADD PRIMARY KEY (Name);

Players can play any of the 26 characters in the game. Each character has a name of course and come from another gaming franchise. The Melee community has ranked each of the characters based on how good they are. Rank 1 being the best rank 26 being the worst.

Characters is in 1NF because it has a key (Name) and no repeating groups.

Characters is in 2NF because it is in 1NF and has no partial dependencies (not even possible since PK is not composite).

Characters is in 3NF because it is in 2NF and has no transitive dependencies.

DoublesTeams

CREATE TABLE DoublesTeams

(

TourneyName VARCHAR(30),

Tag VARCHAR(30),

Team VARCHAR(30)

);

ALTER TABLE DoublesTeams ADD PRIMARY KEY (TourneyName, Tag);

ALTER TABLE DoublesTeams ADD FOREIGN KEY (TourneyName) REFERENCES Events(Name);

ALTER TABLE DoublesTeams ADD FOREIGN KEY (Tag) REFERENCES Players;

In doubles tournaments, each player has a teammate. This table identifies who teamed up with who at certain events. TourneyName is the name of the tournament the players compete in referencing the Events table. Tag is the Player’s tag referencing the Players table. Team combines the tags of the teammates in the format ‘tagA/tagB’.

DoublesTeams is in 1NF because it has a key (TourneyName, Tag) and no repeating groups.

DoublesTeams is in 2NF because it is in 1NF and has no partial dependencies.

DoublesTeams is in 3NF because it is in 2NF and has no transitive dependencies.

Events

CREATE TABLE Events

(

Name VARCHAR(30),

Series VARCHAR(30),

Location VARCHAR(30),

Country VARCHAR(30),

StartDate DATE,

EndDate DATE

);

ALTER TABLE Events ADD PRIMARY KEY (Name);

Tournaments are held at events. These events are named and many are part of a series of events (ex. The Big House 6 is the 6th event of The Big House series). Location is the city where the event was held. Country was the country the event was held in. StartDate and EndDate indicate when the event started and finished.

Events is in 1NF because it has a key (Name) and no repeating groups.

Events is in 2NF because it is in 1NF and has no partial dependencies (not even possible since PK is not composite).

Events is in 3NF because it is in 2NF and has no transitive dependencies.

Games

CREATE TABLE Games

(

Set# INTEGER,

Game# INTEGER,

Winner VARCHAR(30),

StockDiff INTEGER,

Stage VARCHAR(30),

TimeRemaining CHAR(5)

);

ALTER TABLE Games ADD PRIMARY KEY (Set#, Game#);

ALTER TABLE Games ADD FOREIGN KEY (Set#) REFERENCES Sets;

Players play multiple games throughout sets. Set# indicates what set this game is in referencing Sets. Game # indicates which gameit is in the set (first is Game 1, Second is game 2, etc...). Winner indicates the winner of that game. For singles games, the winner equals the player’s tag. For doubles games, the winner equals the team name defined in the double’s teams table. StockDiff indicates how many stocks the winning side won by. Stage indicates what stage it was played on. TimeRemaining indicates how much time was left at the end of the game.

Games is in 1NF because it has a key (Set#, Game#) and no repeating groups.

Games is in 2NF because it is in 1NF and has no partial dependencies.

Games is in 3NF because it is in 2NF and has no transitive dependencies.

Mains

CREATE TABLE Mains

(

Player VARCHAR(30),

Character VARCHAR(30)

);

ALTER TABLE Mains ADD PRIMARY KEY (Player, Character);

ALTER TABLE Mains ADD FOREIGN KEY (Player) REFERENCES Players(Tag);

ALTER TABLE Mains ADD FOREIGN KEY (Character) REFERENCES Characters(Name);

Every player mains a character. Maining a character means that the player has spent a significant amount of time playing that character. Most players main only one character, but some main 2 or more. Player is the tag of the player referencing Players. Character is the name of the character referencing characters.

Mains is in 1NF because it has a key (Player, Character) and no repeating groups.

Mains is in 2NF because it is in 1NF and has no partial dependencies.

Mains is in 3NF because it is in 2NF and has no transitive dependencies.

Placings

CREATE TABLE Placings

(

Tag VARCHAR(30),

TourneyName VARCHAR(30),

TourneyType CHAR(7),

Place INTEGER

);

ALTER TABLE Placings ADD PRIMARY KEY (Tag, TourneyName, TourneyType);

ALTER TABLE Placings ADD FOREIGN KEY (Tag) REFERENCES Players;

ALTER TABLE Placings ADD FOREIGN KEY (TourneyName,TourneyType) REFERENCES Tournaments;

Players who compete will receive a place noting how far they’ve made it through the tournament compared to other players. For the purpose of this database ‘placing’ in a tournament mean they made it to top 32 in singles or top 16 in doubles, which will be recorded in the table. Tag is the tag of the player referencing Players. TourneyName and TourneyType references the tournament.

Placings is in 1NF because it has a key (Tag, TourneyName, TourneyType) and no repeating groups.

Placings is in 2NF because it is in 1NF and has no partial dependencies.

Placings is in 3NF because it is in 2NF and has no transitive dependencies.

Players

CREATE TABLE Players

(

Tag VARCHAR(30),

FName VARCHAR(30),

LName VARCHAR(30),

Country VARCHAR(30),

SSBMRank INTEGER

);

ALTER TABLE Players ADD PRIMARY KEY (Tag);

Players compete in the game. Tag is the alias these players go by. FName and LName is their Name of course. Country is what country they live in. SSBMRank is a rank a player is given based on how good they are. The top 100 players are ranked (1 is the best, 100 is the worst).

Players is in 1NF because it has a key (Tag) and no repeating groups.

Players is in 2NF because it is in 1NF and has no partial dependencies (not even possible since PK is not composite).

Players is in 3NF because it is in 2NF and has no transitive dependencies.

Plays

CREATE TABLE Plays

(

Tag VARCHAR(30),

Character VARCHAR(30),

Set# INTEGER,

Game# INTEGER

);

ALTER TABLE Plays ADD PRIMARY KEY (Tag, Set#, Game#);

ALTER TABLE Plays ADD FOREIGN KEY (Tag) REFERENCES Players;

ALTER TABLE Plays ADD FOREIGN KEY (Character) REFERENCES Characters(Name);

ALTER TABLE Plays ADD FOREIGN KEY (Set#,Game#) REFERENCES Games;

ALTER TABLE Plays ADD FOREIGN KEY (Set#) REFERENCES Sets;

Plays is an associative entity that show what player played what character in a certain game in a certain set. Tag is the tag of the player referencing Players. Character is the name of the character played referencing Characters. Set# and Game# indicates the game and set a record is associated with referencing Sets and Games.

Plays is in 1NF because it has a key (Tag, Set#, Game#) and no repeating groups.

Plays is in 2NF because it is in 1NF and has no partial dependencies.

Plays is in 3NF because it is in 2NF and has no transitive dependencies.

Sets

CREATE TABLE Sets

(

Set# INTEGER,

Winner VARCHAR(30),

Name VARCHAR(30),

SetType CHAR(3),

Record CHAR(5),

TourneyName VARCHAR(30),

TourneyType CHAR(7)

);

ALTER TABLE Sets ADD PRIMARY KEY (Set#);

ALTER TABLE Sets ADD FOREIGN KEY (TourneyName,TourneyType) REFERENCES Tournaments;

When players meet in a tournament they play a set of games to determine who moves on in the tournament. Set# is a unique identifier for each set. Winner indicates the winner of each set. For singles sets, the winner equals the player’s tag. For doubles sets, the winner equals the team name defined in the double’s teams table. Name is the name of the set. SetType can be either ‘bo3’ or ‘bo3’ indicating that the set is either a best of 3 or a best of 5 set. Record shows how many game each player/team won in the format of ‘winner - loser’. TourneyName and TourneyType references the tournament.

Sets is in 1NF because it has a key (Set#) and no repeating groups.

Sets is in 2NF because it is in 1NF and has no partial dependencies (not even possible since PK is not composite).

Sets is in 3NF because it is in 2NF and has no transitive dependencies.

SponsoredBy

CREATE TABLE SponsoredBy

(

Tag VARCHAR(30),

Team VARCHAR(30)

);

ALTER TABLE SponsoredBy ADD PRIMARY KEY (Tag, Team);

ALTER TABLE SponsoredBy ADD FOREIGN KEY (Tag) REFERENCES Players;

ALTER TABLE SponsoredBy ADD FOREIGN KEY (Team) REFERENCES Team(Name);

SponsoredBy is an associative entity that show what players are sponsored by which team. Tag indicates the tag of the player referencing Players. Team indicates the name of the team referencing Teams.

SponsoredBy is in 1NF because it has a key (Tag, Team) and no repeating groups.

SponsoredBy is in 2NF because it is in 1NF and has no partial dependencies.

SponsoredBy is in 3NF because it is in 2NF and has no transitive dependencies.

Team

CREATE TABLE Team

(

Name VARCHAR(30),

Country VARCHAR(30)

);

ALTER TABLE Team ADD PRIMARY KEY (Name);

Teams can sponsor players. Name is the name of the team. Country is the country the team originates in.

Team is in 1NF because it has a key (Name) and no repeating groups.

Team is in 2NF because it is in 1NF and has no partial dependencies (not even possible since PK is not composite).

Team is in 3NF because it is in 2NF and has no transitive dependencies.

Tournaments

CREATE TABLE Tournaments

(

TourneyName VARCHAR(30),

TourneyType CHAR(7),

Entrants INTEGER

);

ALTER TABLE Tournaments ADD PRIMARY KEY (TourneyName, TourneyType);

ALTER TABLE Tournaments ADD FOREIGN KEY (TourneyName) REFERENCES Events(Name);

Tournaments are what players compete in. TourneyName is the name of the Event the tournament is held at referencing Events. There are two types of tournaments, singles and doubles, which is indicated by TourneyType. Entrants indicates how many players entered the event.

Tournaments is in 1NF because it has a key (TourneyName, TourneyType) and no repeating groups.

Tournaments is in 2NF because it is in 1NF and has no partial dependencies.

Tournaments is in 3NF because it is in 2NF and has no transitive dependencies.

Winnings

CREATE TABLE Winnings

(

TourneyName VARCHAR(30),

TourneyType CHAR(7),

Place INTEGER,

Amount NUMBER(9,2)

);

ALTER TABLE Winnings ADD PRIMARY KEY (TourneyName, TourneyType, Place);

ALTER TABLE Winnings ADD FOREIGN KEY (TourneyName,TourneyType) REFERENCES Tournaments;

If a player places high enough in a tourney they are rewarded. TourneyName and TourneyType references the tournament. Place indicates what place a player must get to earn these winnings. Amount is how much money a player will get if they get that place in the tournament.

Winnings is in 1NF because it has a key (TourneyName, TourneyType, Place) and no repeating groups.

Winnings is in 2NF because it is in 1NF and has no partial dependencies.

Winnings is in 3NF because it is in 2NF and has no transitive dependencies.

# Queries:

Query 1

Name players who main every character from the 'Star Fox' franchise

SELECT p.tag, p.FName, p.LName

FROM Players p

WHERE NOT EXISTS

(SELECT \*

FROM Characters c

WHERE c.franchise = 'Star Fox'

AND NOT EXISTS

(SELECT \*

FROM Mains m

WHERE p.tag = m.player

AND m.character = c.name));



Cardinality: 2

Query 2

Name players who only main characters from the 'Mario' franchise

SELECT p.tag, p.FName, p.LName

FROM Players p

WHERE p.tag NOT IN

(SELECT m.player

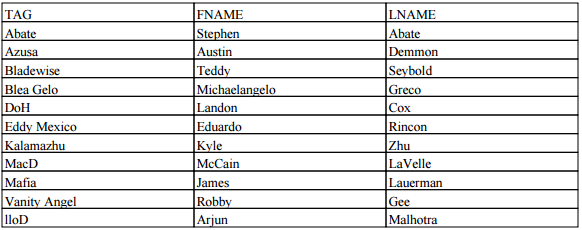
FROM Mains m

WHERE m.character NOT IN

(SELECT c.name

FROM Characters c

WHERE c.franchise = 'Mario'));



Cardinality: 11

Query 3

Name players who main none of the top 8 characters

SELECT p.tag, p.FName, p.LName

FROM Players p

WHERE p.tag NOT IN

(SELECT m.player

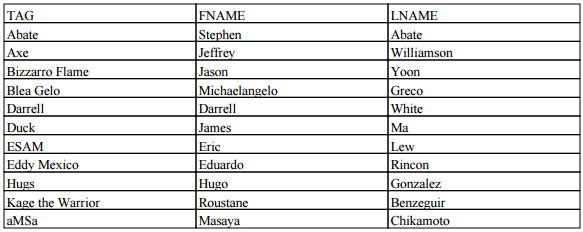
FROM Mains m

WHERE m.character IN

(SELECT c.name

FROM Characters c

WHERE c.rank <= 8));



Cardinality: 11

Query 4

List all Players and any team they have been sponsored by, and any team that is not currently sponsoring any players.(include the player's tag, the player's full name, the player's country, the team's name, and the team's country)

SELECT p.tag, p.FName, p.LName, p.country Player\_Country, t.name Team\_Name, t.country Team\_Country

FROM Players p FULL JOIN SponsoredBy sb ON p.tag = sb.tag

FULL JOIN Team t ON sb.team = t.name;

(result shown on attached Query4.pdf)

Cardinality: 110

Query 5

List all players who placed in The Big House 6 singles tournament and any winnings they've earned.

SELECT p.tag, p.place, w.amount

FROM Placings p LEFT JOIN Winnings w ON p.place = w.place AND p.tourneyName = w.tourneyName AND p.tourneyType = w.tourneyType

WHERE p.tourneyName = 'The Big House 6'

AND p.tourneyType = 'Singles';

(result shown on attached Query5.pdf)

Cardinality: 32

Query 6

List all characters and any players who main them.

SELECT m.player, c.name Character

FROM Mains m RIGHT JOIN Characters C ON c.name = m.character;

(result shown on attached Query6.pdf)

Cardinality: 127

Query 7

Name the players who competed in Losers Finals of the EVO 2016 singles tournament. List the results of the set, and the results each game. Order results by game number.

SELECT DISTINCT Sets.winner Set\_Winner, p1.tag Player1\_Tag, p2.tag Player2\_Tag, Games.game#, Games.winner Game\_Winner, Games.StockDiff, Games.stage, Games.TimeRemaining, Plays1.character Player1\_Character, Plays2.character Player2\_Character

FROM Players p1, Players p2, Sets, Plays Plays1, Plays Plays2, Games

WHERE Sets.name = 'Losers Finals'

AND Sets.tourneyName = 'EVO 2016'

AND Sets.tourneyType = 'Singles'

AND Sets.set# = Games.set#

AND Sets.set# = Plays1.set#

AND Sets.set# = Plays2.set#

AND Plays1.game# = Games.game#

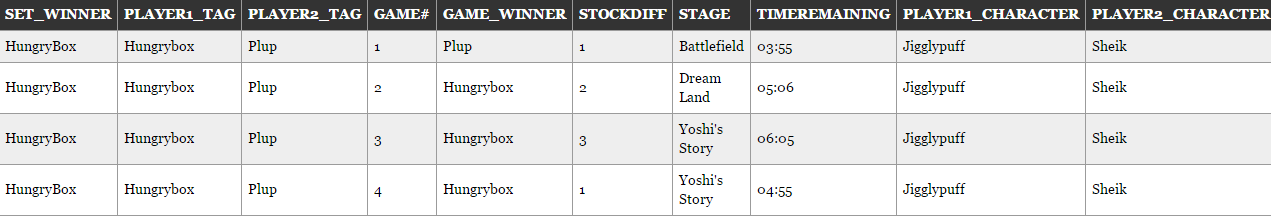
AND Plays2.game# = Games.game#

AND Plays1.tag = p1.tag

AND Plays2.tag = p2.tag

AND p1.tag < p2.tag

ORDER BY Games.Game#;



Cardinality: 4

Query 8

Name players who have played multiple characters in the same set.

SELECT DISTINCT p.tag, p.FName, p.LName

FROM Players p, Plays plays1, Plays plays2, Sets

WHERE p.tag = plays1.tag

AND p.tag = plays2.tag

AND plays1.set# = plays2.set#

AND plays1.character <> plays2.character;



Cardinality: 1

Query 9

List unranked players who have placed at a tourney with at least 1000 entrants in the USA.

SELECT DISTINCT p.tag, p.FName, p.LName, Placings.place, Placings.tourneyName, Placings.tourneyType

FROM Players p, Placings, Tournaments, Events

WHERE p.tag = Placings.tag

AND Placings.tourneyName = Tournaments.tourneyName

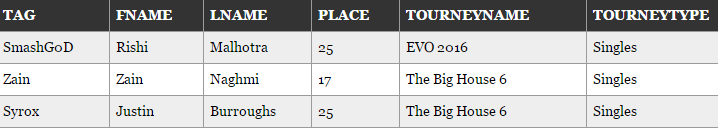
AND Placings.tourneyType = Tournaments.tourneyType

AND Tournaments.tourneyName = Events.Name

AND Tournaments.entrants >= 1000

AND Events.country = 'USA'

AND p.SSBMRank IS NULL;



Cardinality:3

Query 10

List every singles game where an American player won with at least 3 stocks remaining. (Include the tag of the opponent, both players characters, the set name, and the tourneyName)

SELECT p.tag, p.FName, p.LName, Sets.tourneyName, Sets.Name Set\_Name, Plays2.tag Opponent, g.game#, g.stage, g.stockDiff, Plays1.character Winners\_Character, Plays2.character Opponents\_Character

FROM Players p, Plays Plays1, Plays Plays2, Games g, Sets

WHERE p.tag = Plays1.tag

AND Plays1.set# = sets.set#

AND Plays1.game# = g.game#

AND sets.set# = g.set#

AND sets.set# = Plays2.set#

AND g.game# = Plays2.game#

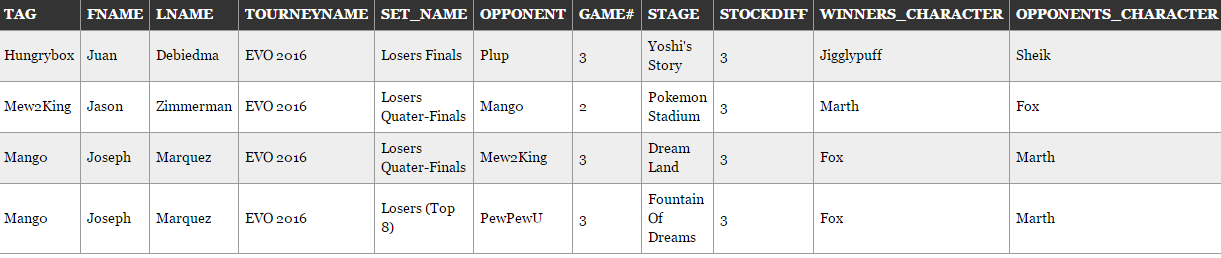
AND p.country = 'USA'

AND sets.tourneyType = 'Singles'

AND Plays1.tag = g.winner

AND g.stockDiff >= 3

AND Plays2.tag <> Plays1.tag;



Cardinality: 4

Query 11

List every doubles game where the winning team won with 5 or more minutes remaining. (Include the tag of each player, the characters they played, the set name, and the tourneyName)

SELECT g.winner, Sets.tourneyName, Sets.Name Set\_Name, g.game#, g.stage, g.stockDiff, g.timeRemaining, p1.tag Team1\_Player1, p1.character T1P1\_Character, p2.tag Team1\_Player2, p2.character T1P2\_Character, p3.tag Team2\_Player1, p3.character T2P1\_Character, p4.tag Team2\_Player2, p4.character T2P2\_Character

FROM Plays p1, Plays p2, Plays p3, Plays p4, Games g, Sets, DoublesTeams d1, DoublesTeams d2, DoublesTeams d3, DoublesTeams d4

WHERE sets.set# = g.set#

AND p1.game# = g.game#

AND p1.set# = sets.set#

AND p2.game# = g.game#

AND p2.set# = sets.set#

AND p3.game# = g.game#

AND p3.set# = sets.set#

AND p4.game# = g.game#

AND p4.set# = sets.set#

AND d1.tag = p1.tag

AND d1.TourneyName = sets.TourneyName

AND d2.tag = p2.tag

AND d2.TourneyName = sets.TourneyName

AND d3.tag = p3.tag

AND d3.TourneyName = sets.TourneyName

AND d4.tag = p4.tag

AND d4.TourneyName = sets.TourneyName

AND d1.team = d2.team

AND p1.tag < p2.tag

AND d3.team = d4.team

AND p3.tag < p4.tag

AND d1.team < d3.team

AND g.timeRemaining >= '05:00';

(results on next page)

Cardinality: 10

