RYERSON UNIVERSITY

Faculty of Engineering, Architecture and Science

Department of Electrical and Computer Engineering

Course Number	848
Course Title	Fundamentals of Data Engineering
Semester/Year	W2023

Instructor	Dr. Faezeh Ensan

Lab No.	3

Lab Title	Database Design

Submission Date	February 26th, 2023
Due Date	February 27th, 2023

Student Name	Student ID	Signature*
Abdulrehman Khan	500968727	A.K.

**By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your own work.*

TABLES CODE:

```
CREATE TABLE player (
 playerID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
 name VARCHAR(255),
 role VARCHAR(25),
 gamesPlayed INTEGER,
 avgKDA INTEGER,
 avgCSR INTEGER
);

CREATE TABLE champion (
championID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
name VARCHAR(255),
skin VARCHAR(255),
winrate INTEGER,
banrate INTEGER,
pickrate INTEGER,
lossrate INTEGER
);

CREATE TABLE maker (
makerID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
name VARCHAR(255)
);

CREATE TABLE team (
 teamID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
 name VARCHAR(255),
 region VARCHAR(255),
 gamesPlayed INTEGER,
 winrate INTEGER,
 lossrate INTEGER,
 avgGameDuration INTEGER,
 avgDragonsSlaughtered INTEGER
);

CREATE TABLE contest (
contestID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
name VARCHAR(255),
region VARCHAR(255),
numberOfGames INTEGER,
```

```
purse INTEGER
);

CREATE TABLE game (
gameID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
teamsInvolved VARCHAR(255),
duration INTEGER,
totalKills INTEGER,
totalDeaths INTEGER
);

CREATE TABLE player_plays_champion (
playID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
player_ID INTEGER,
champion_ID INTEGER,
CONSTRAINT fk_player_ID FOREIGN KEY (player_ID) REFERENCES player(playerID)
CONSTRAINT fk_champion_ID FOREIGN KEY (champion_ID) REFERENCES champion(championID)
);

CREATE TABLE maker_makes_champion (
makeID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
maker_ID INTEGER,
champion_ID INTEGER,
CONSTRAINT fk_designer_id FOREIGN KEY (maker_ID) REFERENCES maker(makerID)
CONSTRAINT fk_champion_id FOREIGN KEY (champion_ID) REFERENCES champion(championID)
);

CREATE TABLE team_has_player (
teamplayerID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
team_ID INTEGER,
player_ID INTEGER,
CONSTRAINT fk_team_ID FOREIGN KEY (team_ID) REFERENCES team(teamID),
CONSTRAINT fk_player_ID FOREIGN KEY (player_ID) REFERENCES player(playerID));

CREATE TABLE contest_has_team (
contestTeamID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
contest_ID INTEGER,
team_ID INTEGER,
CONSTRAINT fk_contest_ID FOREIGN KEY (contest_ID) REFERENCES contest(contestID),
CONSTRAINT fk_team_ID FOREIGN KEY (team_ID) REFERENCES team(teamID));

CREATE TABLE contest_holds_game (
```

```
contestGameID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
contest_ID INTEGER,
game_ID INTEGER,
CONSTRAINT fk_contest_ID FOREIGN KEY (contest_ID) REFERENCES contest(contestID),
CONSTRAINT fk_game_ID FOREIGN KEY (game_ID) REFERENCES game(gameID));
```

INSERTING AND DISPLAYING TABLES VIA SQLITE3:

```
sqlite> .tables
champion                game                    player_plays_champion
contest                 maker                   team
contest_has_team        maker_makes_champion    team_has_player
contest_holds_game      player
sqlite>
```

SQLITE3 DUMP COMMAND:

```
sqlite> .dump
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE player (
  playerID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  name VARCHAR(255),
  role VARCHAR(25),
  gamesPlayed INTEGER,
  avgKDA INTEGER,
  avgCSR INTEGER
);
CREATE TABLE champion (
championID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
name VARCHAR(255),
skin VARCHAR(255),
winrate INTEGER,
banrate INTEGER,
pickrate INTEGER,
lossrate INTEGER
);
CREATE TABLE maker (
makerID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
name VARCHAR(255)
);
CREATE TABLE team (
  teamID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  name VARCHAR(255),
  region VARCHAR(255),
  gamesPlayed INTEGER,
  winrate INTEGER,
  lossrate INTEGER,
  avgGameDuration INTEGER,
  avgDragonsSlaughtered INTEGER
);
CREATE TABLE contest (
 contestID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
 name VARCHAR(255),
 region VARCHAR(255),
 numberOfGames INTEGER,
 purse INTEGER
);
CREATE TABLE game (
 gameID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
 teamsInvolved VARCHAR(255),
 duration INTEGER,
 totalKills INTEGER,
 totalDeaths INTEGER
);
CREATE TABLE player_plays_champion (
playID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
player_ID INTEGER,
champion_ID INTEGER,
CONSTRAINT fk_player_ID FOREIGN KEY (player_ID) REFERENCES player(playerID)
CONSTRAINT fk_champion_ID FOREIGN KEY (champion_ID) REFERENCES champion(championID)
);
CREATE TABLE maker_makes_champion (
makeID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
maker_ID INTEGER,
champion_ID INTEGER,
CONSTRAINT fk_designer_id FOREIGN KEY (maker_ID) REFERENCES maker(makerID)
CONSTRAINT fk_champion_id FOREIGN KEY (champion_ID) REFERENCES champion(championID)
);
CREATE TABLE team_has_player (
```

```
CREATE TABLE maker (
makerID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
name VARCHAR(255)
);
CREATE TABLE team (
  teamID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  name VARCHAR(255),
  region VARCHAR(255),
  gamesPlayed INTEGER,
  winrate INTEGER,
  lossrate INTEGER,
  avgGameDuration INTEGER,
  avgDragonsSlaughtered INTEGER
);
CREATE TABLE contest (
 contestID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
 name VARCHAR(255),
 region VARCHAR(255),
 numberOfGames INTEGER,
 purse INTEGER
);
CREATE TABLE game (
 gameID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
 teamsInvolved VARCHAR(255),
 duration INTEGER,
 totalKills INTEGER,
 totalDeaths INTEGER
);
CREATE TABLE player_plays_champion (
playID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
player_ID INTEGER,
champion_ID INTEGER,
CONSTRAINT fk_player_ID FOREIGN KEY (player_ID) REFERENCES player(playerID)
CONSTRAINT fk_champion_ID FOREIGN KEY (champion_ID) REFERENCES champion(championID)
);
CREATE TABLE maker_makes_champion (
makeID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
maker_ID INTEGER,
champion_ID INTEGER,
CONSTRAINT fk_designer_id FOREIGN KEY (maker_ID) REFERENCES maker(makerID)
CONSTRAINT fk_champion_id FOREIGN KEY (champion_ID) REFERENCES champion(championID)
);
CREATE TABLE team_has_player (
teamplayerID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
team_ID INTEGER,
player_ID INTEGER,
CONSTRAINT fk_team_ID FOREIGN KEY (team_ID) REFERENCES team(teamID),
CONSTRAINT fk_player_ID FOREIGN KEY (player_ID) REFERENCES player(playerID));
CREATE TABLE contest_has_team (
contestTeamID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
contest_ID INTEGER,
team_ID INTEGER,
CONSTRAINT fk_contest_ID FOREIGN KEY (contest_ID) REFERENCES contest(contestID),
CONSTRAINT fk_team_ID FOREIGN KEY (team_ID) REFERENCES team(teamID));
CREATE TABLE contest_holds_game (
contestGameID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
contest_ID INTEGER,
game_ID INTEGER,
CONSTRAINT fk_contest_ID FOREIGN KEY (contest_ID) REFERENCES contest(contestID),
CONSTRAINT fk_game_ID FOREIGN KEY (game_ID) REFERENCES game(gameID));
DELETE FROM sqlite_sequence;
COMMIT;
sqlite>
```

TABLE DESCRIPTION:

TEAM ENTITY: Possesses the details pertaining to the team

team-name	Name of group
team-region	Region the team plays in
team win-rate	# of wins / Total # of games played
team loss-rate	Derived from 100% - # of wins
games-played	Total # of games played
average dragons slaughtered	Total dragons slaughtered per game / Total # of games played
average game-duration	Total duration of game / Total # of games played

PLAYER ENTITY: Possesses the individual's data

player-name	Individual's in-game name
average creep score ratio	Average creep score per minute

average kill-death/assist (KDA) ratio	Average kill-death/assist (KDA) ratio
games -played	Total # of games played
player-role	Lane the individual will play in

CONTEST ENTITY: Possesses the contest's statistics

contest-name	Name of the contest
games to be played	Total # of games to be played within the contest
contest-region	Region the contest will be held in

CHAMPION ENTITY: Possesses the character's statistics

champion-name	Name of character
champion-skin	Cosmetic costume of character
champion win-rate	# of wins / Total # of games played
champion loss-rate	Derived from 100% - win-rate
champion pick-rate	# of games champion is chosen / Total games
champion ban-rate	# of games champion is banned / Total games

GAME ENTITY: Possesses the game's details

game-duration	Total time the game lasts
game-kills	Total # of game kills performed per team
game-deaths	Derived from 100% - game-kills
teams-involved	Name of the teams that have participated in this game

MAKER ENTITY: Holds the details of the champion designer

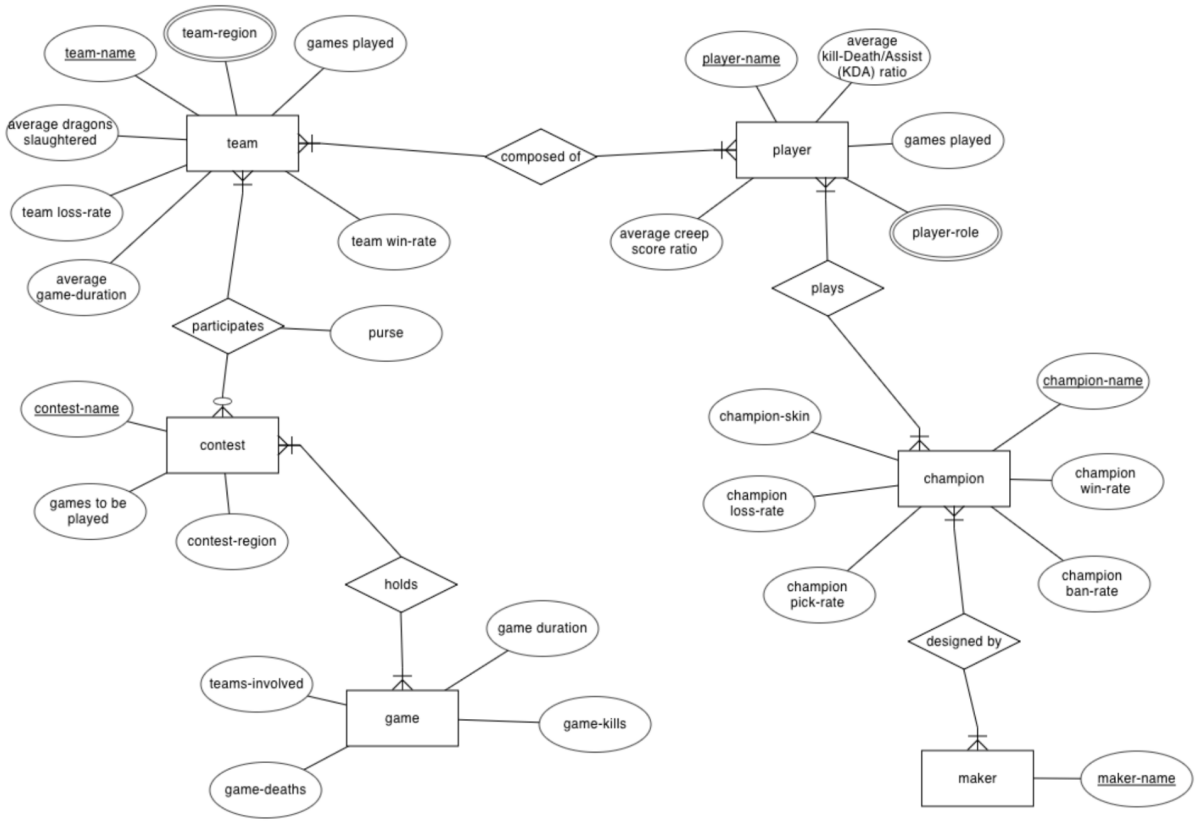
maker-name	Name of the champion designer

RELATIONSHIPS:

team — player	Many to Many	Each team will have many players and each player will be part of one team at a time.

		Players could have been part of different teams prior.
team — contest	Many to Many	It is optional for a team to participate in a contest and many teams can participate in many contests. But, each contest must have teams participating and can have many teams participating.
contest — game	Many to Many	Each contest can hold many games, and each game can be played in many contests.
player — champion	Many to Many	Each player can play many champions, and each champion can be played by different players.
champion — maker	Many to Many	Each champion can be designed by many designers and each designer can design numerous champions.

ER-DIAGRAM:


team-region
team-name
games played
average dragons slaughtered
team
team loss-rate
team win-rate
average game-duration
participates
purse
contest-name
contest
games to be played
contest-region
holds
game duration
teams-involved
game
game-kills
game-deaths
composed of
player-name
average kill-Death/Assist (KDA) ratio
player
games played
average creep score ratio
player-role
plays
champion-name
champion-skin
champion
champion win-rate
champion loss-rate
champion pick-rate
champion ban-rate
designed by
maker
maker-name