Project Team: Eric Studley; Jake Dec; Tom Diaz

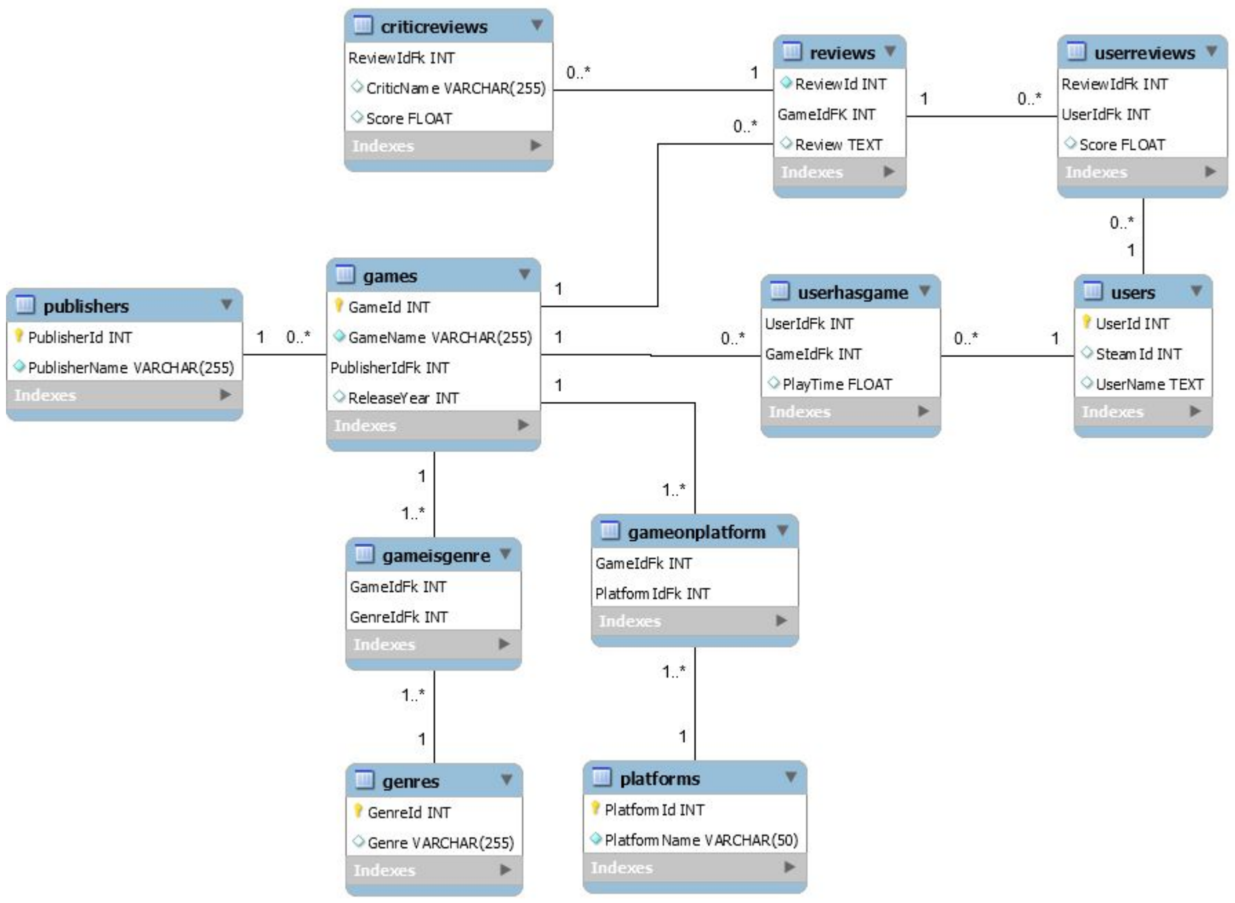
Team Name: Gold Team

CS5200 DBMS Project - Game Ranker

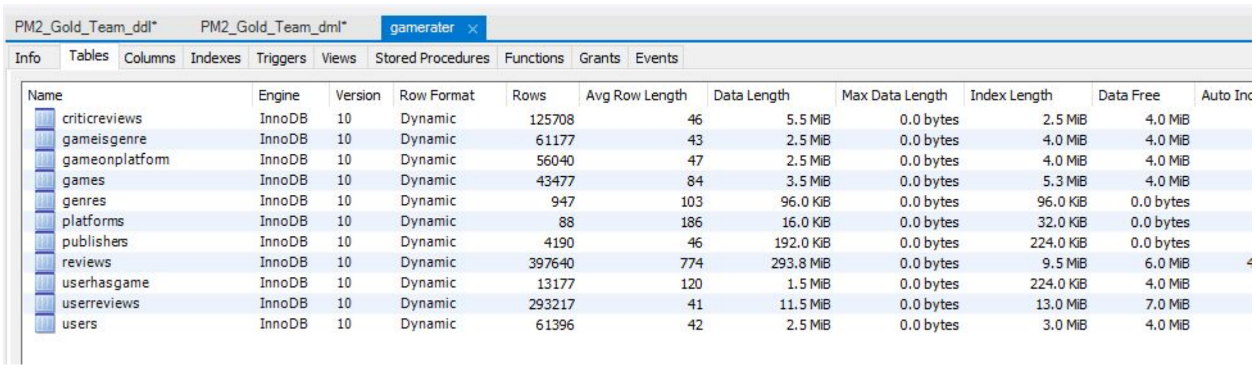
PM2: Relational Model and Data

Video: https://www.youtube.com/watch?v=pcHrC7zomBI&feature=youtu.be

Game Ranker UML



Game Ranker DB Aggregates



Name	Engine	Version	Row Format	Rows	Avg Row Length	Data Length	Max Data Length	Index Length	Data Free	Auto Inc
criticreviews	InnoDB	10	Dynamic	125708	46	5.5 MiB	0.0 bytes	2.5 MiB	4.0 MiB	
gameisgenre	InnoDB	10	Dynamic	61177	43	2.5 MiB	0.0 bytes	4.0 MiB	4.0 MiB	
gameonplatform	InnoDB	10	Dynamic	56040	47	2.5 MiB	0.0 bytes	4.0 MiB	4.0 MiB	
games	InnoDB	10	Dynamic	43477	84	3.5 MiB	0.0 bytes	5.3 MiB	4.0 MiB	
genres	InnoDB	10	Dynamic	947	103	96.0 KiB	0.0 bytes	96.0 KiB	0.0 bytes	
platforms	InnoDB	10	Dynamic	88	186	16.0 KiB	0.0 bytes	32.0 KiB	0.0 bytes	
publishers	InnoDB	10	Dynamic	4190	46	192.0 KiB	0.0 bytes	224.0 KiB	0.0 bytes	
reviews	InnoDB	10	Dynamic	397640	774	293.8 MiB	0.0 bytes	9.5 MiB	6.0 MiB	4
userhasgame	InnoDB	10	Dynamic	13177	120	1.5 MiB	0.0 bytes	224.0 KiB	4.0 MiB	
userreviews	InnoDB	10	Dynamic	293217	41	11.5 MiB	0.0 bytes	13.0 MiB	7.0 MiB	
users	InnoDB	10	Dynamic	61396	42	2.5 MiB	0.0 bytes	3.0 MiB	4.0 MiB	

DDL for Database

```
CREATE SCHEMA IF NOT EXISTS GameRater;
USE GameRater;

DROP TABLE IF EXISTS GameIsGenre;
DROP TABLE IF EXISTS GameOnPlatform;
DROP TABLE IF EXISTS UserHasGame;
DROP TABLE IF EXISTS CriticReviews;
DROP TABLE IF EXISTS UserReviews;
DROP TABLE IF EXISTS Reviews;
DROP TABLE IF EXISTS Games;
DROP TABLE IF EXISTS Users;
DROP TABLE IF EXISTS Platforms;
DROP TABLE IF EXISTS Publishers;
DROP TABLE IF EXISTS Genres;

CREATE TABLE Platforms (
  PlatformId INT NOT NULL UNIQUE AUTO_INCREMENT,
  PlatformName VARCHAR(50) NOT NULL UNIQUE,
  CONSTRAINT PlatformPk
    PRIMARY KEY (PlatformId)
) ENGINE = InnoDB;

CREATE TABLE Publishers (
  PublisherId INT NOT NULL UNIQUE AUTO_INCREMENT,
  PublisherName VARCHAR(255) NOT NULL UNIQUE,
  CONSTRAINT PublisherPk
    PRIMARY KEY (PublisherId)
) ENGINE = InnoDB;

CREATE TABLE Genres (
  GenreId INT NOT NULL UNIQUE AUTO_INCREMENT,
  Genre VARCHAR(255) UNIQUE,
  CONSTRAINT GenresPk
    PRIMARY KEY (GenreId)
) ENGINE = InnoDB;

CREATE TABLE Users (
  UserId INT NOT NULL UNIQUE AUTO_INCREMENT,
  SteamId INT UNIQUE,
  UserName TEXT DEFAULT NULL,
  CONSTRAINT UserPk
    PRIMARY KEY (UserId)
) ENGINE = InnoDB;

CREATE TABLE Games (
  GameId INT NOT NULL UNIQUE AUTO_INCREMENT,
  GameName VARCHAR(255) NOT NULL UNIQUE,
  PublisherIdFk INT,
  ReleaseYear INT,
  CONSTRAINT GamesPk
    PRIMARY KEY (GameId),
  CONSTRAINT GamesPublishersFk
    FOREIGN KEY (PublisherIdFk)
    REFERENCES Publishers (PublisherId)
    ON UPDATE CASCADE ON DELETE SET NULL
) ENGINE = InnoDB;

CREATE TABLE UserHasGame (
  UserIdFk INT NOT NULL,
  GameIdFk INT NOT NULL,
  PlayTime FLOAT,
  CONSTRAINT UserHasGamePk
    UNIQUE (UserIdFk, GameIdFk),
  CONSTRAINT UserHasGameUsersFk
    FOREIGN KEY (UserIdFk)
    REFERENCES Users (UserId)
    ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT UserHasGameGamesFk
    FOREIGN KEY (GameIdFk)
    REFERENCES Games (GameId)
    ON UPDATE CASCADE ON DELETE CASCADE
) ENGINE = InnoDB;
```

```
CREATE TABLE GameIsGenre (
  GameIdFk INT NOT NULL,
  GenreIdFk INT NOT NULL,
  CONSTRAINT GameIsGenreGamesFk
    FOREIGN KEY (GameIdFk)
    REFERENCES Games (GameId)
    ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT GameIsGenreGenresFk
    FOREIGN KEY (GenreIdFk)
    REFERENCES Genres (GenreId)
    ON UPDATE CASCADE ON DELETE CASCADE
) ENGINE = InnoDB;

CREATE TABLE GameOnPlatform (
  GameIdFk INT NOT NULL,
  PlatformIdFk INT NOT NULL,
  CONSTRAINT GameOnPlatformGamesFk
    FOREIGN KEY (GameIdFk)
    REFERENCES Games (GameId)
    ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT GameOnPlatformGenresFk
    FOREIGN KEY (PlatformIdFk)
    REFERENCES Platforms (PlatformId)
    ON UPDATE CASCADE ON DELETE CASCADE
) ENGINE = InnoDB;

CREATE TABLE Reviews (
  ReviewId INT NOT NULL UNIQUE AUTO_INCREMENT,
  GameIdFK INT NOT NULL,
  Review TEXT,
  CONSTRAINT ReviewsGamesFK
    FOREIGN KEY (GameIdFk)
    REFERENCES Games (GameId)
    ON UPDATE CASCADE ON DELETE CASCADE
) ENGINE = InnoDB;

CREATE TABLE UserReviews (
  ReviewIdFk INT NOT NULL,
  UserIdFk INT NOT NULL,
  Score FLOAT,
  CONSTRAINT UserReviewsReviewFk
    FOREIGN KEY (ReviewIdFk)
    REFERENCES Reviews (ReviewId)
    ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT UserReviewsUsersFk
    FOREIGN KEY (UserIdFk)
    REFERENCES Users (UserId)
    ON UPDATE CASCADE ON DELETE CASCADE
) ENGINE = InnoDB;

CREATE TABLE CriticReviews (
  ReviewIdFk INT NOT NULL,
  CriticName VARCHAR(255),
  Score FLOAT,
  CONSTRAINT CriticReviewsReviewFk
    FOREIGN KEY (ReviewIdFk)
    REFERENCES Reviews (ReviewId)
    ON UPDATE CASCADE ON DELETE CASCADE
) ENGINE = InnoDB;
```

Data Insertion Logic

```
USE GameRater;

DROP TABLE IF EXISTS rawSteamUserReviews;
DROP TABLE IF EXISTS rawidToUsername;
DROP TABLE IF EXISTS rawMetaCriticGame;
DROP TABLE IF EXISTS rawVgchartzGame;
DROP TABLE IF EXISTS rawMetaUserComments;
DROP TABLE IF EXISTS rawMetaUserCommentsFixed;
DROP TABLE IF EXISTS rawMetaCriticGameReviews;
DROP TABLE IF EXISTS rawMetaCriticGameReviewsFixed;

CREATE TABLE rawSteamUserReviews (
  UserId INT,
  GameName TEXT,
  PurchasedOrPlayed TEXT,
  PlayTime FLOAT,
  Zero INT
) ENGINE = InnoDB;

CREATE TABLE rawidToUsername (
  UserId INT,
  UserName varchar(255)
) ENGINE = InnoDB;

CREATE TABLE rawMetaCriticGame (
  Title varchar(255),
  TheYear INT NULL,
  Publisher varchar(255),
  Genre varchar(255),
  Platform varchar(255),
  Metascore INT NULL,
  AvgUserscore FLOAT NULL,
  Players varchar(255)
) ENGINE = InnoDB;
```

```
CREATE TABLE rawVgchartzGame (
  TheRank INT,
  TheName varchar(255),
  basename varchar(255),
  Genre varchar(255),
  ESRB_Rating varchar(255),
  Platform varchar(255),
  Publisher varchar(255),
  Developer varchar(255),
  VGChartz_Score FLOAT NULL,
  Critic_Score FLOAT NULL,
  User_Score FLOAT NULL,
  Total_Shipped FLOAT NULL,
  TheYear FLOAT NULL
) ENGINE = InnoDB;

CREATE TABLE rawMetaCriticGameReviews (
  CriticName varchar(255),
  Review TEXT,
  Game varchar(255),
  Platform varchar(255),
  Score FLOAT NULL,
  TheDate DATE
) ENGINE = InnoDB;

CREATE TABLE rawMetaCriticGameReviewsFixed (
  CriticName varchar(255),
  Review TEXT,
  Game INT,
  Score FLOAT NULL
) ENGINE = InnoDB;

DELIMITER $$

CREATE TRIGGER CriticReviewTrigger
AFTER INSERT
ON rawMetaCriticGameReviewsFixed FOR EACH ROW
```

```
BEGIN
   DECLARE review_id INT DEFAULT 0;

   INSERT INTO reviews(GameIdFK, Review)
   VALUES(NEW.Game, NEW.Review);

   -- get review_id
   SET review_id= LAST_INSERT_ID();

   INSERT INTO criticreviews(ReviewIdFk, CriticName, Score)
      VALUES(review_id, NEW.CriticName, NEW.Score);
END$$
DELIMITER ;


CREATE TABLE rawMetaUserComments (
CommentID INT,
GameName TEXT,
Platform VARCHAR(255),
UserScore FLOAT,
MetaComment TEXT,
UserName TEXT
) ENGINE = InnoDB;

CREATE TABLE rawMetaUserCommentsFixed (
GameId TEXT,
UserScore FLOAT,
Review TEXT,
UserId INT
) ENGINE = InnoDB;

DELIMITER $$
CREATE TRIGGER UserReviewTrigger
AFTER INSERT
ON rawMetaUserCommentsFixed FOR EACH ROW
BEGIN
   DECLARE review_id INT DEFAULT 0;
   INSERT INTO reviews(GameIdFK, Review)
```

```
    VALUES(NEW.GameId, NEW.Review);

    -- get review_id
    SET review_id= LAST_INSERT_ID();

    -- insert review into critic account
    INSERT INTO userreviews(ReviewIdFk, UserIdFk, Score)
        VALUES(review_id,NEW.UserId,NEW.UserScore);
END$$
DELIMITER ;

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Steam.csv'
IGNORE INTO TABLE rawSteamUserReviews
 FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
 LINES TERMINATED BY '\n';

 -- Load our steam games into the games table.
INSERT INTO Games (GameName) SELECT DISTINCT GameName FROM
rawSteamUserReviews;

-- Load SteamID to UserName mapping pulled from API
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/idToUsername.csv' INTO TABLE rawidToUsername
 FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
 LINES TERMINATED BY '\n'
 (UserName, UserId);

 -- Load our steam users from the Steam into a user table
INSERT INTO Users (SteamId, Username) SELECT DISTINCT UserId,UserName FROM
rawidToUsername;

-- Map users to games played
INSERT IGNORE INTO userhasgame (SELECT users.UserId, games.GameId,
playedGame.PlayTime
FROM users
INNER JOIN (SELECT * FROM rawsteamuserreviews WHERE PurchasedOrPlayed='play') AS
playedGame
  ON users.SteamId = playedGame.UserId
INNER JOIN games
  ON games.GameName = playedGame.GameName);

-- If the user has purchased a game, but has not played it, insert as gametime 0
INSERT IGNORE INTO userhasgame (Select users.UserId, games.GameId, 0
FROM users
INNER JOIN (SELECT * FROM rawsteamuserreviews WHERE
PurchasedOrPlayed='purchase') as purchasedGame
  ON users.SteamId = purchasedGame.UserId
INNER JOIN games
  ON games.GameName = purchasedGame.GameName);


LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/metacriticGameInfo.csv' INTO TABLE rawMetaCriticGame
 FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
 LINES TERMINATED BY '\n'
 IGNORE 1 LINES
 (@var,Title,@TheYear,Publisher,Genre,Platform,@Metascore,@score,Players)
 SET
   AvgUserscore = if(@score in ('tbd', 'not specified'), NULL, @score),
   TheYear = if(@TheYear in ('tbd', 'not specified'), NULL, @TheYear),
   MetaScore = if(@Metascore in ('tbd', 'not specified'), NULL, @Metascore);

-- Insert publisher and Platform information for games
INSERT IGNORE INTO Publishers (PublisherName) SELECT DISTINCT Publisher FROM
rawMetaCriticGame;
INSERT IGNORE INTO Platforms (PlatformName) SELECT DISTINCT Platform FROM
rawMetaCriticGame;
INSERT IGNORE INTO Genres (Genre) SELECT DISTINCT Genre FROM
rawMetaCriticGame;

-- Update and insert games that do not have steam reviews into games table.
INSERT INTO games (GameName, PublisherIdFk, ReleaseYear)
SELECT rawmetacriticgame.Title, publishers.PublisherId, rawmetacriticgame.TheYear
FROM rawmetacriticgame
INNER JOIN publishers
  ON publishers.PublisherName = rawmetacriticgame.Publisher
ON DUPLICATE KEY UPDATE
PublisherIdFk = publishers.PublisherId,
ReleaseYear = rawmetacriticgame.TheYear;

-- Map games to genres (games can have more than one genre)
INSERT IGNORE INTO gameisgenre (GameIdFk, GenreIdFk)  (SELECT games.GameId,
genres.GenreId
FROM games
INNER JOIN rawmetacriticgame
  ON games.GameName = rawmetacriticgame.Title
INNER JOIN genres
  ON rawmetacriticgame.Genre = genres.Genre);

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/vgchartz12042019.csv' INTO TABLE rawVgchartzGame
 FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
 LINES TERMINATED BY '\n'
 IGNORE 1 LINES

(TheRank,TheName,basename,Genre,ESRB_Rating,Platform,Publisher,Developer,@VGChartz_
Score,@Critic_Score,@User_Score,@Total_Shipped,@Global_Sales,@NA_Sales,@PAL_Sales
,@JP_Sales,@Other_Sales,@TheYear,@Last_Update,@url,@thestatus,@Vgchartzscore,@img_
url)
 SET
        VGChartz_Score = nullif(@VGChartz_Score,''),
        Critic_Score = nullif(@Critic_Score,''),
        User_Score = nullif(@User_Score,''),
        Total_Shipped = nullif(@Total_Shipped,''),
        TheYear = nullif(@TheYear,'');

-- Add a more exastive list of games, including older console games to the platforms and
publishers tables
INSERT IGNORE INTO Publishers (PublisherName) SELECT DISTINCT Publisher FROM
rawVgchartzGame;
INSERT IGNORE INTO Platforms (PlatformName) SELECT DISTINCT Platform FROM
rawVgchartzGame;
INSERT IGNORE INTO Genres (Genre) SELECT DISTINCT Genre FROM
rawVgchartzGame;
```

```
-- Update our games table with new entries
INSERT INTO games (GameName, PublisherIdFk, ReleaseYear)
SELECT rawvgchartzgame.TheName, publishers.PublisherId, rawvgchartzgame.TheYear
FROM rawvgchartzgame
INNER JOIN publishers
  ON publishers.PublisherName = rawvgchartzgame.Publisher
ON DUPLICATE KEY UPDATE
PublisherIdFk = publishers.PublisherId,
ReleaseYear = rawvgchartzgame.TheYear;

-- Fill in the platform information for games (games can be on multiple platforms
INSERT INTO GameOnPlatform SELECT games.GameId, platforms.PlatformId
FROM games
  INNER JOIN rawvgchartzgame
    ON games.GameName = rawvgchartzgame.TheName
  INNER JOIN platforms
    ON rawvgchartzgame.Platform = platforms.PlatformName;

-- Map games to genres (games can have more than one genre)
INSERT IGNORE INTO gameisgenre (GameIdFk, GenreIdFk)  (SELECT games.GameId,
genres.GenreId
FROM games
INNER JOIN rawvgchartzgame
  ON games.GameName = rawvgchartzgame.TheName
INNER JOIN genres
  ON rawvgchartzgame.Genre = genres.Genre);

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/metacriticUserComments_1.csv' IGNORE INTO TABLE rawMetaUserComments
 CHARACTER SET 'utf8'
 FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' ESCAPED BY "\\"
 LINES TERMINATED BY '\n'
 IGNORE 1 LINES;

INSERT IGNORE INTO Users (UserName) SELECT UserName FROM
rawMetaUserComments;
```

```
INSERT INTO rawMetaUserCommentsFixed (GameId,UserScore,Review,UserId)
  SELECT games.GameId, rawMetaUserComments.UserScore,
rawMetaUserComments.MetaComment, users.UserId
  FROM rawMetaUserComments
INNER JOIN games
  ON games.GameName = rawMetaUserComments.GameName
INNER JOIN users
  ON users.UserName = rawMetaUserComments.UserName;

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/metacriticCriticReviews.csv' IGNORE INTO TABLE rawMetaCriticGameReviews
 FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
 LINES TERMINATED BY '\n'
 IGNORE 1 LINES
 (CriticName,Review,Game,Platform,Score,@TheDate)
 SET
  TheDate = STR_TO_DATE(@TheDate, "%M %d, %Y");

INSERT IGNORE INTO Games (GameName) SELECT DISTINCT Game FROM
rawMetaCriticGameReviews;

INSERT INTO rawMetaCriticGameReviewsFixed (CriticName,Review,Game,Score)
  SELECT rawMetaCriticGameReviews.CriticName, rawMetaCriticGameReviews.Review,
games.GameId, rawMetaCriticGameReviews.Score
  FROM rawMetaCriticGameReviews
INNER JOIN games
  ON games.GameName = rawMetaCriticGameReviews.Game;

DROP TABLE IF EXISTS rawSteamUserReviews;
DROP TABLE IF EXISTS rawidToUsername;
DROP TABLE IF EXISTS rawMetaCriticGame;
DROP TABLE IF EXISTS rawVgchartzGame;
DROP TABLE IF EXISTS rawMetaUserComments;
DROP TABLE IF EXISTS rawMetaUserCommentsFixed;
DROP TABLE IF EXISTS rawMetaCriticGameReviews;
DROP TABLE IF EXISTS rawMetaCriticGameReviewsFixed;
```