**COMPETE**

**A Network to Promote Gaming Events in Your Area**

**Group 3**

**Daniel McCormick**

**Carver Sorensen**

**Nicholas Lewis**

**Bryan Diaz**



**Section 1:** Overview

**Motivation**

Gamers currently lack effective methods of locally organizing. Most gaming communities are distributed across the country or world and there is not an effective way of finding local gaming events and connecting with other gamers in the area. It is similarly hard for event organizers (campus organizations, city committees, etc.) to keep up with what games are popular in their area.

Our web app will address this problem by providing a platform for gamers to connect to other gamers in their area and host events for their community. It will provide information about the number of gamers in area, what games they are playing, and events that are being held.

**Major Functions**

Some major functions of our database are outlined below. Please note that some of these features are dependent on private user accounts, which would be managed through a secure log-in process that we have not implemented as part of this project.

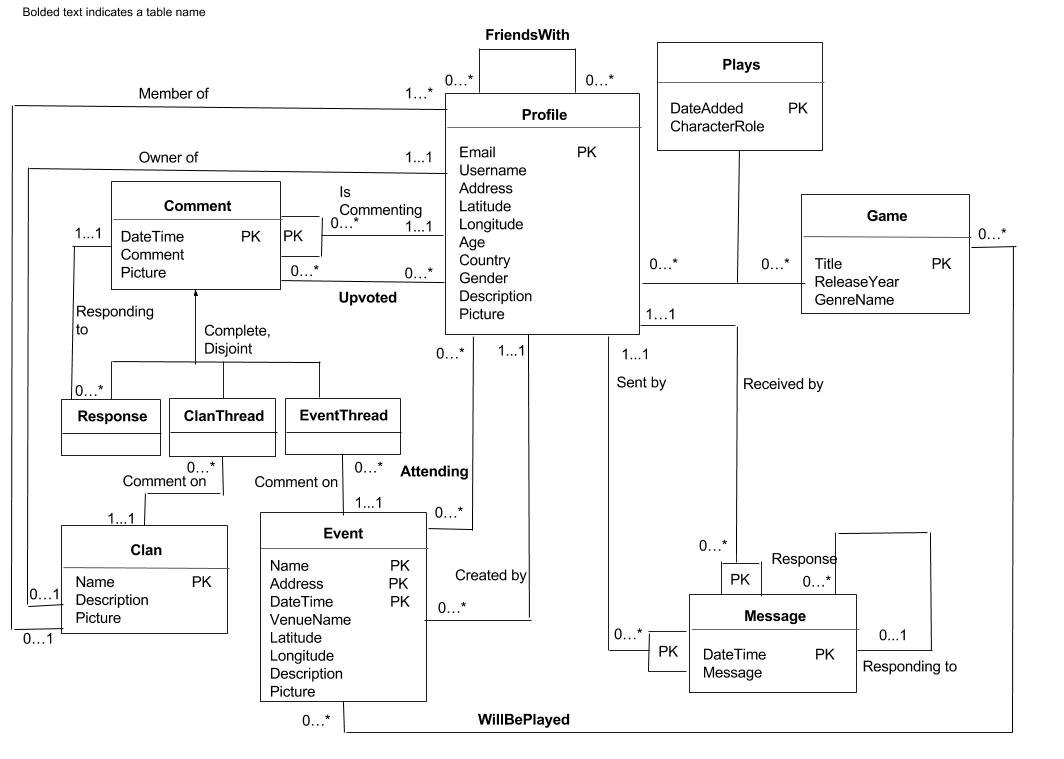
* Allows users to see all gaming events within x miles of their location
* Allows users to search for and create events in their area
* Allows users to add friends as connections, see their events, and join clans with them
* Allows users to create and upvote comments linked to events or clans
* Allows users to send and reply to messages
* Allows players to declare characters/roles they play dependent on game

**Implementation**

Our database stores information on user profiles, past and future gaming events, comment threads, and messages between users. All of the functionality is centered around the user profile, which is directly related to nearly every table in our database (see Section 2: UML). Once users create a profile, they can join, create, and comment on events and clans, add other users as friends, and send messages.

We used a subclassing for comments that allows storage of a rich history of user made comments. Comments can be at the top of a Clan or Event page, or a response to another comment. This allows (metaphorically) infinite levels of stored responses that can be viewed easily as a single thread.

**Section 2:** Comprehensive UML diagram of database



**Section 3:** Assertions

/\*  
\* Created by: Nick  
\* Reviewed by: Danny  
\*/  
/\* Ensure that our comment subclassing is demonstrating complete coverage  
\* i.e. Comments should either be responses, clan threads, or event threads  
\* A set of triggers is included to cover this by inserting a response item into any unassigned   
\* comments then removing that response when the actual subclass is added  
\*/   
/\*  
CREATE ASSERTION CommentCoverage  
 CHECK (  
 NOT EXISTS (SELECT \* FROM Comment  
 WHERE (Email, DateTime)   
 NOT IN ((SELECT Email, DateTime FROM Response   
 UNION SELECT Email, DateTime FROM ClanThread)  
 UNION SELECT Email, DateTime FROM EventThread)  
 )  
);\*/  
  
/\*  
\* Created by: Nick  
\* Reviewed by: Carver  
\*/  
/\* Ensure that our comment subclassing is demonstrating disjoint coverage  
\* i.e. Comments should be only one of: Response, ClanThread, EventThread  
\* This Assertion is covered in our Comment implementation because all  
\* comment subclasses share the same primary keys and are therefore inherently unique  
\* with no overlap  
\*/  
/\*  
CREATE ASSERTION CommentDisjoint  
 CHECK (  
 NOT EXISTS (SELECT \*   
FROM Response R, ClanThread CT, EventThread ET, Comment C  
WHERE R.ResponseEmail = C.Email AND R.ResponseDateTime = C.DateTime  
AND CT.ThreadEmail = C.Email AND CT.ThreadDateTime = C.DateTime  
AND ET.ThreadEmail = C.Email AND ET.ThreadDateTime = C.DateTime  
 GROUP BY C.Email, C.DateTime  
 HAVING count(\*) > 1  
);\*/  
  
/\*  
\* Created by: Nick  
\* Reviewed by: Bryan  
\*/  
/\* Enforces 1…\* cardinality for having at least one member (owner) in each clan at any time   
\* This assertion is covered in the implementation of clan, where owner is declared   
\* as NOT NULL and UNIQUE \*/  
/\*  
CREATE ASSERTION ClanNotEmpty  
 CHECK (  
 NOT EXISTS (SELECT \*   
 FROM Clan  
 WHERE Name NOT IN (SELECT ClanMember FROM Profile))  
);\*/

**Section 4:** Comprehensive Tables, Inserts, Deletes, Updates, Triggers

/\* Group 3 \*/

/\* Daniel McCormick, Nicholas Lewis, Bryan Diaz, Carver Sorensen \*/

/\*\*\*\*\*\*\*\*\*DROP TABLES\*\*\*/

DROP TABLE IF EXISTS UpVoted;

DROP TABLE IF EXISTS Attending;

DROP TABLE IF EXISTS WillBePlayed;

DROP TABLE IF EXISTS Clan;

DROP TABLE IF EXISTS Message;

DROP TABLE IF EXISTS FriendsWith;

DROP TABLE IF EXISTS ClanThread;

DROP TABLE IF EXISTS EventThread;

DROP TABLE IF EXISTS Response;

DROP TABLE IF EXISTS Comment;

DROP TABLE IF EXISTS Event;

DROP TABLE IF EXISTS Plays;

DROP TABLE IF EXISTS Game;

DROP TABLE IF EXISTS Profile;

DROP TABLE IF EXISTS GenreType;

DROP VIEW IF EXISTS RolesPlayed;

DROP VIEW IF EXISTS LargeAttendance;

DROP VIEW IF EXISTS MessageDisplay;

DROP VIEW IF EXISTS NumComments;

DROP TRIGGER IF EXISTS ClanCreated;

DROP TRIGGER IF EXISTS CommentEdited;

DROP TRIGGER IF EXISTS FriendsAdded;

DROP TRIGGER IF EXISTS ClanWelcome;

/\*\*\*\*\*\*\*\* TABLES \*\*\*\*\*\*\*/

/\*

\* Created by: Carver

\* Reviewed by: Bryan

\* Relation for the user profiles which includes basic user information

\* and whether they are a clan member.

\* We used SET NULL for delete because someone’s account shouldn’t be deleted just because

\* someone gets rid of their clan.

\*/

/\*\* Picture is a string to the image url. \*\*/

/\*\* Lat/Long are in Degrees Minutes Seconds format \*\*/

CREATE TABLE Profile (

Email VARCHAR(63) PRIMARY KEY,

Username VARCHAR(31) UNIQUE,

Address VARCHAR(127),

Latitude FLOAT,

Longitude FLOAT,

Age INT,

Country VARCHAR(63),

Gender VARCHAR(63),

Description VARCHAR(255),

Picture VARCHAR(255),

ClanMember VARCHAR(31),

FOREIGN KEY(ClanMember) REFERENCES Clan(Name)

ON UPDATE CASCADE

ON DELETE SET NULL

);

/\*

\* Created by: Danny

\* Reviewed by: Nick

\* Relation for games which include basic info about them

\*/

CREATE TABLE Game (

Title VARCHAR(63) PRIMARY KEY,

ReleaseYear INT,

GenreName VARCHAR(63) NOT NULL DEFAULT ('Not Selected') REFERENCES GenreType(Genre)

);

/\*

\* Created by: Bryan

\* Holds choices for genre for game table’s attribute “GenreName” to emulate enum type

\*/

CREATE TABLE GenreType (

Genre PRIMARY KEY

);

/\*

\* Created by: Danny

\* Reviewed by: Carver

\* Relation relating users to games.

\* The user with that email plays the game with title and uses the role when they play

\*/

CREATE TABLE Plays (

Email VARCHAR(63),

Title VARCHAR(63),

DateAdded DATETIME,

CharacterRole VARCHAR(255),

PRIMARY KEY(DateAdded, Email, Title),

FOREIGN KEY(Email) REFERENCES Profile(Email)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY(Title) REFERENCES Game(Title)

ON UPDATE CASCADE

ON DELETE CASCADE

);

/\*

\* Created by: Carver

\* Reviewed by: Nick

\* Relation that holds all of the basic information for an event such as

\* name, location, description, etc.

\* Also tracks the owners of each event

\*/

/\*\* Picture is a string to the image url. \*\*/

CREATE TABLE Event (

Name VARCHAR(31),

Address VARCHAR(127),

DateTime DATETIME,

VenueName VarChar(63),

Latitude FLOAT,

Longitude FLOAT,

Description VARCHAR(255),

Picture VARCHAR(255),

Creator VARCHAR(63) NOT NULL,

PRIMARY KEY(Name, Address, DateTime),

FOREIGN KEY(Creator) REFERENCES Profile(Email)

ON UPDATE CASCADE

ON DELETE CASCADE

);

/\*

\* Created by: Bryan

\* Reviewed by: Danny

\* Parent class structure for the Response, EventThread, and subclasses

\* Either begins a thread on an Event or Clan page or responds to another thread/response

\* Can include both text and/or a picture

\*/

/\*\* Picture is a string to the image url. \*\*/

CREATE TABLE Comment (

Email VARCHAR(63),

DateTime DATETIME,

Comment VARCHAR(255) NOT NULL,

Picture Varchar(255),

PRIMARY KEY(Email, DateTime),

FOREIGN KEY(Email) REFERENCES Profile(Email)

ON UPDATE CASCADE

ON DELETE CASCADE

);

/\*

\* Created by: Nick

\* Reviewed by: Carver

\* A special type of comment that is a response to an existing comment

\* These comments will inherit the thread location of their parent

\*/

CREATE TABLE Response(

ResponseEmail VARCHAR(63),

ResponseDateTime DATETIME,

RespondingToEmail VARCHAR(63),

RespondingToDateTime DATETIME,

PRIMARY KEY(ResponseEmail, ResponseDateTime),

FOREIGN KEY(ResponseEmail, ResponseDateTime) REFERENCES Comment(Email, DateTime)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY(RespondingToEmail, RespondingToDateTime) REFERENCES Comment(Email, DateTimel)

ON UPDATE CASCADE

ON DELETE CASCADE

);

/\*

\* Created by: Bryan

\* Reviewed by: Nick

\* Subclass of Comment

\* Creates a new thread on an Event page

\*/

CREATE TABLE EventThread(

EventName VARCHAR(31),

EventAddress VARCHAR(127),

EventDateTime DATETIME,

ThreadDateTime DATETIME,

ThreadEmail VARCHAR(63),

PRIMARY KEY(ThreadDateTime, ThreadEmail),

FOREIGN KEY(EventName, EventAddress, EventDateTime) REFERENCES Event(Name, Address, DateTime)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY(ThreadDateTime, ThreadEmail) REFERENCES Comment(DateTime, Email)

ON UPDATE CASCADE

ON DELETE CASCADE

UNIQUE (EventName, EventAddress, EventDateTime)

);

/\*

\* Created by: Nick

\* Reviewed by: Bryan

\* Creates a new top-level comment on a clan page

\*/

CREATE TABLE ClanThread(

ClanName VARCHAR(31),

ThreadDateTime DATETIME,

ThreadEmail VARCHAR(63),

PRIMARY KEY(ThreadDateTime, ThreadEmail),

FOREIGN KEY(ClanName) REFERENCES Clan(Name)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY(ThreadDateTime, ThreadEmail) REFERENCES Comment(DateTime, Email)

ON UPDATE CASCADE

ON DELETE CASCADE,

UNIQUE (ClanName)

);

/\*

\* Created by: Carver

\* Reviewed by: Nick

\* Self-Association relation of profile that records who are friends.

\* Check that friends cannot be friends with themselves.

\*/

CREATE TABLE FriendsWith (

Email1 VARCHAR(63),

Email2 VARCHAR(63),

PRIMARY KEY(Email1, Email2),

FOREIGN KEY(Email1) REFERENCES Profile(Email)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY(Email2) REFERENCES Profile(Email)

ON UPDATE CASCADE

ON DELETE CASCADE,

CHECK (Email1 <> Email2)

);

/\*

\* Created by: Danny

\* Reviewed by: Carver

\* Relation for messages between 2 profiles.

\* Identifies the profiles based on receivedBy and sentBy.

\* Also has an optional foreign key to itself specifying if it is a reply to a previous message

\* Sets NULL on delete to ensure that messages are not deleted if the profile to which it was

\* responding was deleted

\*/

CREATE TABLE Message (

ReceivedBy VARCHAR(63),

SentBy VARCHAR(63),

DateTime DATETIME,

Message VARCHAR(255),

ResponseTo VARCHAR(63),

ResponseFrom VARCHAR(63),

ResponseDate VARCHAR(63),

PRIMARY KEY(ReceivedBy, SentBy, DateTime),

FOREIGN KEY(ReceivedBy) REFERENCES Profile(Email)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY(SentBy) REFERENCES Profile(Email)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY(ResponseTo, ResponseFrom, ResponseDate) REFERENCES Message(ReceivedBy, SentBy, DateTime)

ON UPDATE CASCADE

ON DELETE SET NULL

);

/\*

\* Created by: Carver

\* Reviewed by: Danny

\* Relation to hold the information of a clan, which is a group of users with a common connection

\* Clans must have one and only one owner

\*/

/\*\* Picture is a string to the image url. \*\*/

CREATE TABLE Clan (

Name VARCHAR(31) PRIMARY KEY,

Description VARCHAR(255),

Picture VARCHAR(255),

Owner VARCHAR(63) NOT NULL,

FOREIGN KEY(Owner) REFERENCES Profile(Email)

ON UPDATE CASCADE

ON DELETE CASCADE,

UNIQUE (Owner)

);

/\*

\* Created by: Danny

\* Reviewed by: Bryan

\* Relates Events to Games based on whether

\* the Game will be played at the Event

\*/

CREATE TABLE WillBePlayed (

Name VARCHAR(31),

Address VARCHAR(127),

DateTime DATETIME,

Title VARCHAR(63),

PRIMARY KEY(Name, Address, DateTime, Title),

FOREIGN KEY(Name, Address, DateTime) REFERENCES Event(Name, Address, DateTime)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY(Title) REFERENCES Game(Title)

ON UPDATE CASCADE

ON DELETE CASCADE

);

/\*

\* Created by: Nick

\* Reviewed by: Bryan

\* Relates profiles to events when people sign-up as “attending”

\*/

CREATE TABLE Attending (

Email VARCHAR(63),

Name VARCHAR(31),

Address VARCHAR(127),

DateTime DATETIME,

PRIMARY KEY(Email, Name, Address, DateTime),

FOREIGN KEY(Email) REFERENCES Profile(Email)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY(Name, Address, DateTime) REFERENCES Event(Name, Address, DateTime)

ON UPDATE CASCADE

ON DELETE CASCADE

);

/\*

\* Created by: Bryan

\* Reviewed by: Danny

\* Records which Comment was upvoted and which Profile upvoted it

\*/

CREATE TABLE Upvoted (

ProfileEmail VARCHAR(63),

CommentEmail VARCHAR(63),

DateTime DATETIME,

PRIMARY KEY(ProfileEmail,CommentEmail, DateTime),

FOREIGN KEY(ProfileEmail) REFERENCES Profile(Email)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY (CommentEmail, DateTime) REFERENCES Comment(Email, DateTime)

ON UPDATE CASCADE

ON DELETE CASCADE

);

/\*\*\*\*\*\*\* TRIGGERS \*\*\*\*\*\*\*/

/\*

\* Created by: Danny

\* Reviewed by: Nicholas

\*/

/\*After a clan is created, set the creator to be in the clan\*/

CREATE TRIGGER ClanCreated

AFTER INSERT ON Clan

FOR EACH ROW

BEGIN

UPDATE Profile

SET ClanMember = New.Name

WHERE Email = New.Owner;

END;

/\*

\* Created by: Nick

\* Reviewed by: Danny

\*/

/\* When a comment is updated (edited), change the DateTime to the most recent edit \*/

CREATE TRIGGER CommentEdited

AFTER UPDATE ON Comment

FOR EACH ROW

BEGIN

UPDATE Comment

SET DateTime = 'now' /\* NOW() returns the current DATETIME \*/

/\* This checks if a comment or picture was changed but the comment creator

and DateTime were NOT changed, and automatically updates the DateTime\*/

WHERE New.Email = Old.Email AND New.DateTime = Old.DateTime AND

(New.Comment <> Old.Comment OR New.Picture <> Old.Picture);

END;

/\*

\* Created by: Carver

\* Reviewed by Bryan

\* When a friend is added, make sure that corresponding opposite friend is added\*/

CREATE TRIGGER FriendsAdded

AFTER INSERT ON FriendsWith

FOR EACH ROW

BEGIN

INSERT INTO FriendsWith(Email1, Email2)

VALUES

(New.Email2, New.Email1);

END;

/\*

\* Created by: Danny

\* Reviewed by Bryan

\* When someone is added to a clan, automatically send a message to them from the clan owner\*/

CREATE TRIGGER ClanWelcome

AFTER UPDATE ON Profile

FOR EACH ROW

WHEN Old.ClanMember <> New.ClanMember AND New.ClanMember IS NOT NULL

BEGIN

INSERT INTO Message(ReceivedBy, SentBy, DateTime,Message,ResponseTo,ResponseFrom,ResponseDate)

SELECT New.Email, C.Owner, 'now', 'Welcome to the clan!', null, null, null

FROM Clan C

WHERE C.Name = New.ClanMember;

END;

/\*

\* Created by: Bryan

\* When a user’s comment is responded to, send the user a message from the responding user

\*/

CREATE TRIGGER ResponseAlert

AFTER INSERT ON Response

FOR EACH ROW

BEGIN

INSERT INTO Message(ReceivedBy, SentBy, DateTime, Message, ResponseTo, ResponseFrom, ResponseDate)

VALUES

(New.RespondingToEmail, New.ResponseEmail, 'now', 'Your comment got a response!', NULL, NULL, NULL);

END;

/\*

\* Created by: Danny

\* The following sets of insertions/triggers enforce complete coverage of Comment\*/

INSERT INTO Profile(Email, Username, Address, Latitude, Longitude, Age, Country, Gender, Description, Picture, ClanMember)

VALUES

('invalidComment@comment.edu','Invalid','N/A',0,0,20,'USA','Male','This is used for invalid comments', NULL, NULL);

INSERT INTO Clan(Name, Description, Picture, Owner)

VALUES

('Empty Clan','This clan is for comments that dont belong anywhere', NULL, 'invalidComment@comment.edu');

INSERT INTO Comment(Email,DateTime,Comment,Picture)

VALUES

('invalidComment@comment.edu','1-1-2000','N/A',NULL);

INSERT INTO ClanThread(ClanName, ThreadDateTime, ThreadEmail)

VALUES

('Empty Clan', '1-1-2000', 'invalidComment@comment.edu');

CREATE TRIGGER CommentInsert

AFTER INSERT ON Comment

FOR EACH ROW

WHEN NOT EXISTS (SELECT R.ResponseEmail,R.ResponseDateTime FROM Response R WHERE R.ResponseEmail = New.Email AND R.ResponseDateTime = New.DateTime

UNION SELECT C.ThreadEmail, C.ThreadDateTime FROM ClanThread C WHERE C.ThreadEmail = New.Email AND C.ThreadDateTime = New.DateTime

UNION SELECT E.ThreadEmail, E.ThreadDateTime FROM EventThread E WHERE E.ThreadEmail = New.Email AND E.ThreadDateTime = New.DateTime)

BEGIN

INSERT INTO Response(ResponseEmail, ResponseDateTime, RespondingToEmail, RespondingToDateTime)

VALUES

(New.Email,New.DateTime, 'invalidComment@comment.edu','1-1-2000');

END;

CREATE TRIGGER ResponseInsert

BEFORE INSERT ON Response

FOR EACH ROW

BEGIN

DELETE FROM Response

WHERE ResponseEmail = New.ResponseEmail AND ResponseDateTime = New.ResponseDateTime

AND RespondingToEmail = 'invalidComment@comment.edu' AND RespondingToDateTime = '1-1-2000';

END;

CREATE TRIGGER ClanThreadInsert

BEFORE INSERT ON ClanThread

FOR EACH ROW

BEGIN

DELETE FROM Response

WHERE ResponseEmail = New.ThreadEmail AND ResponseDateTime = New.ThreadDateTime

AND RespondingToEmail = 'invalidComment@comment.edu' AND RespondingToDateTime = '1-1-2000';

END;

CREATE TRIGGER EventThreadInsert

BEFORE INSERT ON EventThread

FOR EACH ROW

BEGIN

DELETE FROM Response

WHERE ResponseEmail = New.ThreadEmail AND ResponseDateTime = New.ThreadDateTime

AND RespondingToEmail = 'invalidComment@comment.edu' AND RespondingToDateTime = '1-1-2000';

END;

/\*\*\*\*\*\*\* VIEWS \*\*\*\*\*\*\*/

/\*

\* Created by: Carver

\* Reviewed by: Bryan

\*/

/\*View all of the roles users have/will have played over all events

\* The motivation for this is that it allows you to see the information about a player’s

\* roles in a game without seeing personal information about the user

\* such as email, age, gender, location, etc.

\*/

CREATE VIEW RolesPlayed(Username,Title,Role,Name, Address, DateTime) AS

SELECT P.Username, G.Title, Pl.Role, E.Name, E.Address, E.DateTime

FROM Profile P, Game G, Event E, Plays Pl, WillBePlayed W, Attending A

WHERE P.Email = Pl.Email AND Pl.Title = G.Title AND G.Title = W.Title AND

E.Name = W.Name AND E.Address = W.Address AND E.DateTime = W.DateTime AND

P.Email = A.Email AND E.Name = A.Name AND E.Address = A.Address AND E.DateTime = A.DateTime;

/\*

\* Created by: Bryan

\* Reviewed by: Nicholas

\* View all events attended by X (in this case 500) or more users

\* The motivation for this is that it allow you to see a summary of event attendance

\* without seeing the personal information of the people who attended

\*/

CREATE VIEW LargeAttendance as

SELECT \*

FROM Event E, (SELECT COUNT(\*) FROM Attending A WHERE A.Name = E.Name AND A.Address = E.Address AND A.DateTime = E.DateTime) as ACount

WHERE ACount >= 500

GROUP BY E.Name, E.Address, E.DateTime;

/\*

\* Created By: Danny

\* Reviewed By: Nicholas

\* View that returns all messages along with the associated usernames of the sender/receiver

\* The motivation for this is that it hides personal information of users

\* such as email, age, gender, and address, but still allows administrative view of messages

\*/

CREATE VIEW MessageDisplay(Sender, Receiver, Message) AS

SELECT P1.Username, P2.Username, M.Message

FROM Profile P1, Profile P2, Message M

WHERE M.SentBy = P1.Email AND M.ReceivedBy = P2.Email;

/\*

\* Created By: Nicholas

\* Reviewed By: Danny

\* Returns the number of comments each user has made

\* This allows people to see the most active users in the community without revealing

\* personal information or comment contents

\*/

CREATE VIEW NumComments(Username, count) AS

SELECT P.Username, COUNT(\*) as count

FROM Profile P, Comment C

WHERE P.Email = C.Email

GROUP BY P.Username;

/\*\*\*\*\*\*\* ASSERTIONS \*\*\*\*\*\*\*/

/\*

\* Created by: Nick

\* Reviewed by: Danny

\*/

/\* Ensure that our comment subclassing is demonstrating complete coverage

\* i.e. Comments should either be responses, clan threads, or event threads

\* A set of triggers is included to cover this by inserting a response item into any unassigned

\* comments then removing that response when the actual subclass is added

\*/

/\*

CREATE ASSERTION CommentCoverage

CHECK (

NOT EXISTS (SELECT \* FROM Comment

WHERE (Email, DateTime)

NOT IN ((SELECT Email, DateTime FROM Response

UNION SELECT Email, DateTime FROM ClanThread)

UNION SELECT Email, DateTime FROM EventThread)

)

);\*/

/\*

\* Created by: Nick

\* Reviewed by: Carver

\*/

/\* Ensure that our comment subclassing is demonstrating disjoint coverage

\* i.e. Comments should be only one of: Response, ClanThread, EventThread

\* This Assertion is covered in our Comment implementation because all

\* comment subclasses share the same primary keys and are therefore inherently unique

\* with no overlap

\*/

/\*

CREATE ASSERTION CommentDisjoint

CHECK (

NOT EXISTS (SELECT \*

FROM Response R, ClanThread CT, EventThread ET, Comment C

WHERE R.ResponseEmail = C.Email AND R.ResponseDateTime = C.DateTime

AND CT.ThreadEmail = C.Email AND CT.ThreadDateTime = C.DateTime

AND ET.ThreadEmail = C.Email AND ET.ThreadDateTime = C.DateTime

GROUP BY C.Email, C.DateTime

HAVING count(\*) > 1

);\*/

/\*

\* Created by: Nick

\* Reviewed by: Bryan

\*/

/\* Enforces 1…\* cardinality for having at least one member (owner) in each clan at any time

\* This assertion is covered in the implementation of clan, where owner is declared

\* as NOT NULL and UNIQUE \*/

/\*

CREATE ASSERTION ClanNotEmpty

CHECK (

NOT EXISTS (SELECT \*

FROM Clan

WHERE Name NOT IN (SELECT ClanMember FROM Profile))

);\*/

/\*\*\*\*\*INSERTS\*\*\*\*\*\*\*/

/\*

\* Created by: Bryan

\* Populating GenreType with possible choices for GenreName

\*/

INSERT INTO GenreType(Genre) VALUES ('Not Selected');

INSERT INTO GenreType(Genre) VALUES ('Action');

INSERT INTO GenreType(Genre) VALUES ('Action-adventure');

INSERT INTO GenreType(Genre) VALUES ('Adventure');

INSERT INTO GenreType(Genre) VALUES ('Role-playing');

INSERT INTO GenreType(Genre) VALUES ('Simulation');

INSERT INTO GenreType(Genre) VALUES ('Strategy');

INSERT INTO GenreType(Genre) VALUES ('Sports');

/\*

\* Created by: Danny

\* Reviewed by: Carver

\*/

/\*Insert sample data into profile\*/

INSERT INTO Profile(Email, Username, Address, Latitude, Longitude, Age, Country, Gender, Description, Picture, ClanMember)

VALUES

('daniel.mccormick@vanderbilt.edu', 'Danny', '311 Orchard Ln', 15321.5, -145.32,20, 'USA', 'Male', 'Best gamer', 'http://www.smashbros.com/us/images/index/image/main-wiiu-logo.png', NULL);

/\* Inserted by: Bryan \*/

INSERT INTO Profile(Email, Username, Address, Latitude, Longitude, Age, Country, Gender, Description, Picture, ClanMember)

VALUES

('bryan.a.diaz@vanderbilt.edu', 'Bryan', '15332 SW 74th Pl', 25.627040, -80.314885,21, 'USA', 'Male', '1v1 me bro', 'https://i.ytimg.com/vi/807aebvKsmc/hqdefault.jpg', NULL);

/\* Inserted by: Bryan \*/

INSERT INTO Profile(Email, Username, Address, Latitude, Longitude, Age, Country, Gender, Description, Picture, ClanMember)

VALUES

('tigerlily123@gmail.com', 'Patty', '123 Clearwater Ct', 32.869042, -83.754402, 29, 'USA', 'Male', 'best candy crusher NA', 'https://s-media-cache-ak0.pinimg.com/736x/bb/77/2c/bb772c59f3ce9fd144a7d26103c2495c.jpg', NULL);

/\* Inserted by: Bryan \*/

INSERT INTO Event(Name, Address, DateTime, VenueName, Latitude, Longitude, Description, Picture, Creator)

VALUES

('Smash Rivalries by Yahoo Esports', 'Burbank, CA', '2017-04-08 12:00:00', 'Burbank', 34.180839, -118.308966, 'Yahoo Esports has reignited the East Coast vs. West Coast rivalry featuring 8 players from each coast, who will duke it out in doubles, singles, and a 8v8 crew battle. The event will consist of a prize pool of $15,000!', 'https://images.smash.gg/images/tournament/9742/image-83496d275b813abf374074c53316e235.png', 'bryan.a.diaz@vanderbilt.edu');

/\* Inserted by: Bryan \*/

INSERT INTO Event(Name, Address, DateTime, VenueName, Latitude, Longitude, Description, Picture, Creator)

VALUES

('Mobile Mayhem', '2500 West End Ave, Nashville, TN', '2017-04-09 8:00:00', 'Centennial Park', 36.149026, -86.811991, 'Come out and celebrate the magic of mobile gaming with us!', 'https://www.gameogre.com/wp-content/uploads/2016/11/mobilegames.jpg', 'tigerlily123@gmail.com');

/\* Inserted by: Bryan \*/

INSERT INTO Clan(Name, Description, Picture, Owner)

VALUES

('The Originals', 'Creators of COMPETE', 'https://static.independent.co.uk/s3fs-public/thumbnails/image/2016/02/10/18/pg-22-God.jpg', 'bryan.a.diaz@vanderbilt.edu');

/\* Inserted by: Bryan \*/

INSERT INTO Clan(Name, Description, Picture, Owner)

VALUES

('Candy Crushers', 'Mobile game enthusiasts', 'https://i.ytimg.com/vi/v9WF5D71P9o/hqdefault.jpg', 'tigerlily123@gmail.com');

/\* Inserted by: Bryan \*/

INSERT INTO Game(Title, ReleaseYear, GenreName)

VALUES

('Super Smash Bros. Melee', 2001, 'Action-Adventure');

/\* Inserted by: Bryan \*/

INSERT INTO Game(Title, ReleaseYear, GenreName)

VALUES

('Hearthstone', 2014, 'Strategy');

/\* Created by: Bryan \*/

INSERT INTO Plays(Email, Title, DateAdded, CharacterRole)

VALUES

('bryan.a.diaz@vanderbilt.edu', 'Super Smash Bros. Melee', 'now', 'Marth, Samus');

/\* Inserted by: Bryan \*/

INSERT INTO Plays(Email, Title, DateAdded, CharacterRole)

VALUES

('bryan.a.diaz@vanderbilt.edu', 'Hearthstone', 'now', 'Mage, Druid');

/\* Inserted by: Bryan \*/

INSERT INTO FriendsWith(Email1, Email2)

VALUES

('bryan.a.diaz@vanderbilt.edu', 'daniel.mccormick@vanderbilt.edu');

/\* Inserted by: Bryan \*/

INSERT INTO FriendsWith(Email1, Email2)

VALUES

('bryan.a.diaz@vanderbilt.edu', 'tigerlily123@gmail.com');

/\* Created by: Bryan \*/

INSERT INTO WillBePlayed(Name, Address, DateTime, Title)

VALUES

('Smash Rivalries by Yahoo Esports', 'Burbank, CA', '2017-04-08 12:00:00', 'Super Smash Bros. Melee');

/\* Inserted by: Bryan \*/

INSERT INTO Attending(Email, Name, Address, DateTime)

VALUES

('tigerlily123@gmail.com', 'Mobile Mayhem', '2500 West End Ave, Nashville, TN','2017-04-09 8:00:00');

/\* Inserted by: Bryan \*/

INSERT INTO Attending(Email, Name, Address, DateTime)

VALUES

('bryan.a.diaz@vanderbilt.edu', 'Smash Rivalries by Yahoo Esports', 'Burbank, CA','2017-04-08 12:00:00');

/\*Inserted by Bryan\*/

INSERT INTO Comment(Email,DateTime,Comment, Picture)

VALUES

('bryan.a.diaz@vanderbilt.edu', '2017-04-10 2:32:14', 'Our platform is getting a lot of traction. This is great!', NULL);

/\*Inserted by Bryan\*/

INSERT INTO

ClanThread(ClanName, ThreadDateTime, ThreadEmail)

VALUES

('The Originals', '2017-04-10 2:34:14', 'bryan.a.diaz@vanderbilt.edu');

/\*Inserted by: Bryan\*/

/\*\*\*INSERT INTO UpVoted\*\*\*/

/\*Inserted by Danny\*/

INSERT INTO Clan(Name, Description, Picture, Owner)

VALUES

('Dannys Clan', 'Best Clan', 'http://www.smashbros.com/us/images/index/image/main-wiiu-logo.png', 'daniel.mccormick@vanderbilt.edu');

/\*Inserted by Danny\*/

INSERT INTO Profile(Email, Username, Address, Latitude, Longitude, Age, Country, Gender, Description, Picture, ClanMember)

VALUES

('dannymccormick3@gmail.com', 'Daniel', '311 Orchard Ln', 15321.5, -145.32,20, 'USA', 'Male', '2nd account', 'http://www.smashbros.com/us/images/index/image/main-wiiu-logo.png', 'Dannys Clan');

/\*Inserted by Danny\*/

INSERT INTO Profile(Email, Username, Address, Latitude, Longitude, Age, Country, Gender, Description, Picture, ClanMember)

VALUES

('test@test.com', 'Test', '311 Test St', 0, 0, 63,'Mexico', 'Female', 'Test Gamer', 'http://www.smashbros.com/us/images/index/image/main-wiiu-logo.png', 'The Originals');

/\*Inserted by Danny\*/

INSERT INTO Profile(Email, Username, Address, Latitude, Longitude, Age, Country, Gender, Description, Picture, ClanMember)

VALUES

('albus@hogwarts.edu', 'Dumbledore', 'Hogwarts st', 10000, 10000,80, 'England', 'Male', 'Gaming wizard', 'http://www.smashbros.com/us/images/index/image/main-wiiu-logo.png', NULL);

/\*Inserted by Danny\*/

INSERT INTO FriendsWith(Email1, Email2)

VALUES

('daniel.mccormick@vanderbilt.edu','albus@hogwarts.edu');

/\*Inserted by Danny\*/

INSERT INTO FriendsWith(Email1, Email2)

VALUES

('albus@hogwarts.edu', 'test@test.com');

/\*Inserted by Danny\*/

INSERT INTO FriendsWith(Email1, Email2)

VALUES

('albus@hogwarts.edu', 'dannymccormick3@gmail.com');

/\*Inserted by Danny\*/

INSERT INTO Game(Title, ReleaseYear, GenreName)

VALUES

('Super Smash Bros. Brawl', 2008, "Action-Adventure");

/\*Inserted by Danny\*/

INSERT INTO Game(Title, ReleaseYear, GenreName)

VALUES

('Mario Kart Double Dash', 2001, "Action");

/\*Inserted by Danny\*/

INSERT INTO Game(Title, ReleaseYear, GenreName)

VALUES

('League of Legends', 2009, 'Role-playing');

/\*Inserted by Danny\*/

INSERT INTO Event(Name, Address,DateTime,VenueName,Latitude,Longitude,Description,Picture,Creator)

VALUES

('Cool hang out','311 Orchard Ln','5-1-2017','My house',15321.5, -145.32, 'Cool hang out with chill games','http://www.smashbros.com/us/images/index/image/main-wiiu-logo.png','dannymccormick3@gmail.com');

/\*Inserted by Danny\*/

INSERT INTO Event(Name, Address,DateTime,VenueName,Latitude,Longitude,Description,Picture,Creator)

VALUES

('hogwartsLAN','Hogwarts St','now','Hogwarts',10000, 10000, 'Itll be magical','http://www.smashbros.com/us/images/index/image/main-wiiu-logo.png','albus@hogwarts.edu');

/\*Inserted by Danny\*/

INSERT INTO WillBePlayed(Name,Address,DateTime,Title)

VALUES

('Cool hang out','311 Orchard Ln','5-1-2017','Mario Kart Double Dash');

/\*Inserted by Danny\*/

INSERT INTO Attending(Email,Name,Address,DateTime)

VALUES

('daniel.mccormick@vanderbilt.edu','Cool hang out', '311 Orchard Ln', '5-1-2017');

/\*Inserted by Danny\*/

INSERT INTO Attending(Email,Name,Address,DateTime)

VALUES

('albus@hogwarts.edu','Cool hang out', '311 Orchard Ln', '5-1-2017');

/\*Inserted by Danny\*/

INSERT INTO Attending(Email,Name,Address,DateTime)

VALUES

('test@test.com','Cool hang out', '311 Orchard Ln', '5-1-2017');

/\*Inserted by Danny\*/

INSERT INTO Attending(Email,Name,Address,DateTime)

VALUES

('dannymccormick3@gmail.com','Cool hang out', '311 Orchard Ln', '5-1-2017');

/\*Inserted by Danny\*/

INSERT INTO Attending(Email,Name,Address,DateTime)

VALUES

('bryan.a.diaz@vanderbilt.edu','Cool hang out', '311 Orchard Ln', '5-1-2017');

/\*Inserted by Danny\*/

INSERT INTO Plays(Email, Title, DateAdded, CharacterRole)

VALUES

('daniel.mccormick@vanderbilt.edu','Super Smash Bros Melee', 'now','Yoshi');

/\*Inserted by Danny\*/

INSERT INTO Plays(Email, Title, DateAdded, CharacterRole)

VALUES

('daniel.mccormick@vanderbilt.edu','Mario Kart Double Dash', 'now','Baby Mario and Bowser');

/\*Inserted by Danny\*/

INSERT INTO Comment(Email,DateTime,Comment, Picture)

VALUES

('daniel.mccormick@vanderbilt.edu','4-16-2017','Ill be there', NULL);

/\*Inserted by Danny\*/

INSERT INTO EventThread(EventName,EventAddress,EventDateTime,ThreadDateTime,ThreadEmail)

VALUES

('Cool hang out', '311 Orchard Ln', '5-1-2017', 'now', 'dannymccormick3@gmail.com');

/\*Inserted by Danny\*/

INSERT INTO Comment(Email,DateTime,Comment, Picture)

VALUES

('albus@hogwarts.edu','4-7-2017','Me too!', NULL);

/\*Inserted by Danny\*/

INSERT INTO Response(ResponseEmail, ResponseDateTime, RespondingToEmail, RespondingToDateTime)

VALUES

('albus@hogwarts.edu', '4-7-2017', 'daniel.mccormick@vanderbilt.edu', '4-16-2017');

/\*Inserted by Danny\*/

INSERT INTO Upvoted(ProfileEmail,CommentEmail,DateTime)

VALUES

('albus@hogwarts.edu','daniel.mccormick@vanderbilt.edu','4-16-2017');

/\*Inserted by Danny\*/

INSERT INTO Message(ReceivedBy,SentBy,DateTime,Message,ResponseTo,ResponseFrom,ResponseDate)

VALUES

('daniel.mccormick@vanderbilt.edu','dannymccormick3@gmail.com','4-6-2017','Im the real Danny',null,null,null);

/\*Inserted by Danny\*/

INSERT INTO Message(ReceivedBy,SentBy,DateTime,Message,ResponseTo,ResponseFrom,ResponseDate)

VALUES

('dannymccormick3@gmail.com','daniel.mccormick@vanderbilt.edu','4-7-2017','No I am','daniel.mccormick@vanderbilt.edu','dannymccormick3@gmail.com','4-6-2017');

/\* Inserted by: Nicholas \*/

INSERT INTO Profile(Email, Username, Address, Latitude, Longitude, Age, Country, Gender, Description, Picture, ClanMember)

VALUES

('nicholas.j.lewis@vanderbilt.edu', 'Hanyoma', '2301 Vanderbilt Place', 36.1627, -86.7816, 22, 'USA', 'Male', 'Vanderbilt Event Organizer', 'http://yifan-guo.github.io/Logo.jpg', 'The Originals');

/\* Inserted by: Nicholas \*/

INSERT INTO Game(Title, ReleaseYear, GenreName)

VALUES

('Sid Meier’s Civilization VI', 2016, 'Strategy');

/\* Inserted by: Nicholas \*/

INSERT INTO Event(Name, Address, DateTime, VenueName, Latitude, Longitude, Description, Picture, Creator)

VALUES

('Vandy\_LAN VII', '400 24th Ave S', '2017-03-25 13:00:00', 'Featheringill Hall', 36.1448, -86.8034, 'GET HYPED! Vanderbilts Premiere Gaming Event returns for the Spring 2017 semester! Join us in Featheringill Hall on Saturday, March 25th starting at 1 PM', 'http://nashville.carpediem.cd/data/afisha/bp/11/2e/112e3b64cd.jpg', 'nicholas.j.lewis@vanderbilt.edu');

/\* Inserted by: Nicholas \*/

INSERT INTO Plays(Email, Title, DateAdded, CharacterRole)

VALUES

('nicholas.j.lewis@vanderbilt.edu', 'Sid Meier’s Civilization VI', 'now' ,'India');

/\* Inserted by: Nicholas \*/

INSERT INTO Comment(Email, DateTime, Comment, Picture)

VALUES

('nicholas.j.lewis@vanderbilt.edu', 'now', 'How does this work?', NULL);

/\* Inserted by: Nicholas \*/

INSERT INTO Response(ResponseEmail, ResponseDateTime, RespondingToEmail, RespondingToDateTime)

VALUES

('nicholas.j.lewis@vanderbilt.edu', 'now', 'daniel.mccormick@vanderbilt.edu', '4-16-2017');

/\* Inserted by: Nicholas \*/

INSERT INTO EventThread(EventName, EventAddress, EventDateTime, ThreadDateTime,ThreadEmail)

VALUES

('Vandy\_LAN VII', '400 24th Ave S', '2017-03-25 13:00:00', '2017-03-25 12:59:99', 'nicholas.j.lewis@vanderbilt.edu');

/\* Inserted by: Nicholas \*/

INSERT INTO FriendsWith(Email1, Email2)

VALUES

('nicholas.j.lewis@vanderbilt.edu', 'test@test.com');

/\* Inserted by: Nicholas \*/

INSERT INTO FriendsWith(Email1, Email2)

VALUES

('nicholas.j.lewis@vanderbilt.edu', 'daniel.mccormick@vanderbilt.edu');

/\* Inserted by: Nicholas \*/

INSERT INTO Message(ReceivedBy,SentBy,DateTime,Message,ResponseTo,ResponseFrom,ResponseDate)

VALUES

('nicholas.j.lewis@vanderbilt.edu', 'dannymccormick3@gmail.com', '4-6-2017', 'No, I’M the real Danny', null, null, null);

/\* Inserted by: Nicholas \*/

INSERT INTO WillBePlayed(Name, Address, DateTime, Title)

VALUES

('Vandy\_LAN VII', '400 24th Ave S', '2017-03-25 13:00:00', 'Sid Meier’s Civilization VI');

/\* Inserted by: Nicholas \*/

INSERT INTO Attending(Email,Name,Address,DateTime)

VALUES

('nicholas.j.lewis@vanderbilt.edu', 'Vandy\_LAN VII', '400 24th Ave S', '2017-03-25 13:00:00');

/\* Inserted by: Nicholas \*/

INSERT INTO Attending(Email,Name,Address,DateTime)

VALUES

('nicholas.j.lewis@vanderbilt.edu', 'Cool hang out', '311 Orchard Ln', '5-1-2017');

/\* Inserted by: Nicholas \*/

INSERT INTO Upvoted(ProfileEmail, CommentEmail, DateTime)

VALUES

('nicholas.j.lewis@vanderbilt.edu', 'nicholas.j.lewis@vanderbilt.edu', '2017-03-25 12:59:99');

/\* Inserted by: Carver \*/

INSERT INTO Profile(Email, Username, Address, Latitude, Longitude, Age, Country, Gender, Description, Picture, ClanMember)

VALUES

('carver.d.sorensen@vanderbilt.edu', 'CarverAlpha', '1945 Port Carney', 36.1912, 86.7222, 21, 'USA', 'Male', 'Carvers Primary Account', 'http://www.everyeye.it/public/covers/23052016/logo.jpg', 'AlphaClan');

/\* Inserted by: Carver \*/

INSERT INTO Profile(Email, Username, Address, Latitude, Longitude, Age, Country, Gender, Description, Picture, ClanMember)

VALUES

('harry@hogwarts.edu', 'TheChosenOne', '1949 Port Carney', 36.1933, 86.6965, 21, 'England', 'Male', 'Gryffindor Rules', 'https://pbs.twimg.com/profile\_images/481181875528925184/RcRcC\_JD.jpeg', 'AlphaClan');

/\* Inserted by: Carver \*/

INSERT INTO Clan(Name, Description, Picture, Owner)

VALUES

('AlphaClan', 'Compete to Win', 'http://www.psibernetix.com/wp-content/uploads/2015/12/ALPHA\_Logo.png', 'carver.d.sorensen@vanderbilt.edu');

/\* Inserted by: Carver \*/

INSERT INTO Game(Title, ReleaseYear, GenreName)

VALUES

('Overwatch', 2016, 'Action');

/\* Inserted by: Carver \*/

INSERT INTO Game(Title, ReleaseYear, GenreName)

VALUES

('Skyrim', 2011, 'Role-playing');

/\* Inserted by: Carver \*/

INSERT INTO Plays(Email, Title, DateAdded, CharacterRole)

VALUES

('carver.d.sorensen@vanderbilt.edu', 'Overwatch', 'now', 'Zenyadda');

/\* Inserted by: Carver \*/

INSERT INTO Plays(Email, Title, DateAdded, CharacterRole)

VALUES

('harry@hogwarts.edu', 'Skyrim', 'now', 'Magic');

/\* Inserted by: Carver \*/

INSERT INTO Event(Name, Address, DateTime, VenueName, Latitude, Longitude, Description, Picture, Creator)

VALUES

('Overwatch and Chill', '1945 Port Carney', '2017-05-01 18:00:00', 'Carvers House', 36.1912, 86.7222, 'If you need some time to relax during finals, come Overwatch and chill!', 'http://www.everyeye.it/public/covers/23052016/logo.jpg', 'carver.d.sorensen@vanderbilt.edu');

/\* Inserted by: Carver \*/

INSERT INTO Event(Name, Address, DateTime, VenueName, Latitude, Longitude, Description, Picture, Creator)

VALUES

('Hogwarts Overwatch Tourney', '1 Gryffindor Way', '2017-05-05 20:00:00', 'Gryffindor Common Room', 10000, 10000, 'Using any spells on the console, controller, opponent, or yourself is frowned upon.', 'https://shop.wbstudiotour.co.uk/images/dynamic/660x660/6242b981.jpg', 'harry@hogwarts.edu');

/\* Created by: Carver \*/

INSERT INTO WillBePlayed(Name, Address, DateTime, Title)

VALUES

('Hogwarts Overwatch Tourney', '1 Gryffindor Way', '2017-05-05 20:00:00', 'Overwatch');

/\* Inserted by: Carver \*/

INSERT INTO Message(ReceivedBy,SentBy,DateTime,Message,ResponseTo,ResponseFrom,ResponseDate)

VALUES

('carver.d.sorensen@vanderbilt.edu','harry@hogwarts.edu','now','Do not invite Malfoy.',null,null,null);

/\* Inserted by: Carver \*/

INSERT INTO Attending(Email, Name, Address, DateTime)

VALUES

('harry@hogwarts.edu', 'Hogwarts Overwatch Tourney', '1 Gryffindor Way', '2017-05-05 20:00:00');

/\* Inserted by: Carver \*/

INSERT INTO Attending(Email, Name, Address, DateTime)

VALUES

('carver.d.sorensen@vanderbilt.edu', 'Hogwarts Overwatch Tourney', '1 Gryffindor Way', '2017-05-05 20:00:00');

/\* Inserted by: Carver \*/

INSERT INTO Comment(Email,DateTime,Comment, Picture)

VALUES

('harry@hogwarts.edu','2017-04-08 11:11:11', 'Will there be food?', NULL);

/\* Inserted by: Carver \*/

INSERT INTO EventThread(EventName,EventAddress,EventDateTime,ThreadDateTime,ThreadEmail)

VALUES

('Hogwarts Overwatch Tourney', '1 Gryffindor Way', '2017-05-05 20:00:00', '2017-04-08 11:11:11', 'harry@hogwarts.edu');

/\* Inserted by: Carver \*/

INSERT INTO Upvoted(ProfileEmail,CommentEmail,DateTime)

VALUES

('albus@hogwarts.edu','harry@hogwarts.edu','2017-04-08 11:11:11');

/\* Inserted by: Carver \*/

INSERT INTO FriendsWith(Email1, Email2)

VALUES

('harry@hogwarts.edu', 'albus@hogwarts.edu');

/\* Inserted by: Carver \*/

INSERT INTO FriendsWith(Email1, Email2)

VALUES

('carver.d.sorensen@vanderbilt.edu', 'harry@hogwarts.edu');

/\*\*\*\*\* QUERIES \*\*\*\*\*/

/\*

\* Created by: Danny

\* Reviewed by: Nicholas

\*/

/\*Get the total number of upvotes for a given comment\*/

SELECT COUNT(\*)

FROM UpVoted U

WHERE U.CommentEmail = 'daniel.mccormick@vanderbilt.edu'

AND U.DateTime = '4-16-2017';

/\*

\* Created by: Danny

\* Reviewed by: Bryan

\*/

/\*Get the most attended event on a given day and the number of people attending\*/

SELECT Tot.Name, Tot.Address, '2016-03-23', MAX(Tot.Num)

FROM (SELECT Name, Address, COUNT(Email) AS Num

FROM Attending A

WHERE A.DateTime = '2016-03-23'

GROUP BY Name,Address) Tot;

/\*

\* Created by: Bryan

\* Reviewed by: Carver

\* Finds how many of a profile’s friends play a game

\* In this case, find how many of Bryan’s play Super Smash Bros Melee

\*/

SELECT COUNT(\*)

FROM FriendsWith F, Plays P

WHERE F.Email1 = 'bryan.a.diaz@vanderbilt.edu' AND P.Email = F.Email2 AND P.Title = 'Super Smash Bros Melee';

/\*

\* Created by: Bryan

\* Reviewed by: Danny

\* Find the games played at all events created by a profile

\* ‘User’ is a placeholder for the profile we want to find out about

\*/

SELECT W.Title

FROM Event E, WillBePlayed W

WHERE E.Creator = 'User' AND E.Name = W.Name AND E.Address = W.Address AND E.DateTime = W.DateTime;

/\*

\* Created by: Bryan

\* Reviewed by: Nicholas

\* Find the top 5 largest clans and how many people are in each clan

\*/

SELECT CC.Name, CC.Count

FROM (SELECT C.Name as Name, COUNT(\*) as Count

FROM Profile P, Clan C

WHERE C.Name = P. ClanMember

GROUP BY C.Name) as CC

ORDER BY CC.Count desc

LIMIT 5;

/\*

\* Created by: Bryan

\* Find which role is the most played for a specified game

\*/

SELECT R.Role, R.Count

FROM (SELECT Pl.CharacterRole as Role, COUNT(\*) as Count

FROM Profile P, Game G, Plays Pl

WHERE P.Email = Pl. Email AND G.Title = 'Super Smash Bros. Melee'

ORDER BY Pl.CharacterRole) as R

LIMIT 1;

/\*

\* Created by: Bryan

\* Find mutual friends between two users

\*/

SELECT distinct F1.Email1, F2.Email1, F1.Email2

FROM FriendsWith F1, FriendsWith F2

WHERE F1.Email2 = F2.Email2 AND F1.Email1 < F2.Email1;

/\*

\* Created by: Carver

\* Reviewed by: Nicholas

\*/

/\*Find mutual friends between 2 given users. In this query, return mutual friends of Danny and Carver\*/

SELECT f2.Email2

FROM FriendsWith f1 JOIN FriendsWith f2 USING (Email2)

WHERE f1.Email1 = 'carver.d.sorensen@vanderbilt.edu' AND f2.Email1 = 'daniel.mccormick@vanderbilt.edu';

/\*

\* Created by: Carver

\* Reviewed by: Danny

\*/

/\*Get the 10 profiles that attended the most events\*/

SELECT P.Email

FROM (SELECT A.Email AS Email, Count(\*) AS Tot

FROM Attending A

GROUP BY A.Email) AS P

ORDER BY P.Tot DESC

LIMIT 10;

/\*

\* Created by: Carver

\* Reviewed by: Bryan

\*/

/\*Delete all messages that are at least 10 years old\*/

DELETE FROM Message

WHERE datetime('now', '-10 year') >= DateTime;

/\*

\* Created by: Nick

\* Reviewed by: Danny

\*/

/\* Find Venues that have held more than 5 events \*/

SELECT E.VenueName

FROM Event E

GROUP BY E.VenueName

HAVING COUNT(\*) > 5;

/\*

\* Created by: Carver

\* Reviewed by: Nick

\*/

/\* Find games that have been played (or will be played) at more than 10 tournaments \*/

SELECT G.Title

FROM WillBePlayed G

GROUP BY G.Title

HAVING COUNT(\*) > 10;

/\*

\* Created by: Nick

\* Reviewed by: Bryan

\*/

/\* Find the average age of male gamers for each country in the database \*/

SELECT P.Country, AVG(P.Age)

FROM Profile P

WHERE P.Gender = 'Male'

GROUP BY P.Country;

/\*

\* Created By: Danny

\* Reviewed By: Bryan

\* Get's all queries within approximately 50 miles of a given latitude/longitude (10020,10120 in this case)

\* Note that latitude and longitude are stored as seconds. 60 seconds (or 1 minute, 1/60 degrees) is approximately equal to 1 mile most places

\*/

SELECT E.\*

FROM Event E

WHERE ((E.latitude - 10020)\*(E.latitude - 10020) + (E.longitude - 10120)\*(E.longitude - 10120)) <= (50\*50\*60\*60);

/\* Created By: Danny

\* Reviewed By: Carver

\* Find the users with the most friends

\*/

SELECT F.Email1, COUNT(F.Email2)

FROM FriendsWith F

GROUP By F.Email1

HAVING COUNT (F.Email2) = (

SELECT MAX(myCount)

FROM (

SELECT F.Email1, COUNT(F.Email2) myCount

FROM FriendsWith F

GROUP BY F.Email1));

/\* Created By: Danny

\* Reviewed By: Nicholas

\* Gets the games with names containing smash

\*/

SELECT G.\*

FROM Game G

WHERE G.Title LIKE '%smash%';

/\* Created By: Danny

\* Reviewed By: Carver

\* Gets all people attending a given event ('Smash Rivalries by Yahoo Esports', 'Burbank, CA', '2017-04-08 12:00:00' as PK) in this case\*/

SELECT P.\*

FROM Profile P, Attending A

WHERE P.Email = A.Email AND A.Name = 'Smash Rivalries by Yahoo Esports'

AND A.Address = 'Burbank, CA' AND A.DateTime = '2017-04-08 12:00:00';

/\* Created By: Danny

\* Reviewed By: Carver

\* Find the most popular event (by attendance)\*/

SELECT A.Name, A.Address, A.DateTime, COUNT(\*)

FROM Attending A

GROUP BY A.Name, A.Address, A.DateTime

ORDER BY COUNT(\*) DESC

LIMIT 1;

/\* Created By: Danny

\* Reviewed By: Bryan

\* Find the player who has been to the most events\*/

SELECT P.\*, COUNT(\*)

FROM Attending A, Profile P

WHERE P.Email = A.Email

GROUP BY P.Email

ORDER BY COUNT(\*) DESC

LIMIT 1;

/\* Created By: Danny

\* Reviewed By: Bryan

\* Find the name of the biggest clan\*/

SELECT P.ClanMember, COUNT(\*)

FROM Profile P

GROUP BY P.ClanMember

ORDER BY COUNT(\*) DESC

LIMIT 1;

/\* Created By: Carver

\* Reviewed By: Danny

\* List average age of each game’s users \*/

SELECT Pl.Title, AVG(P.Age)

FROM Plays Pl, Profile P

WHERE P.Email = Pl.Email

GROUP BY Pl.Title;

/\* Created By: Bryan

\* Reviewed By: Nick

\* List games where the average player is over 21 \*/

SELECT Pl.Title

FROM Plays Pl, Profile P

WHERE P.Email = Pl.Email

GROUP BY Pl.Title

HAVING AVG(P.Age) > 21;

/\*\*\*\*\*\*\*\*\*DROP TABLES\*\*\*/

DROP TABLE IF EXISTS UpVoted;

DROP TABLE IF EXISTS Attending;

DROP TABLE IF EXISTS WillBePlayed;

DROP TABLE IF EXISTS Clan;

DROP TABLE IF EXISTS Message;

DROP TABLE IF EXISTS FriendsWith;

DROP TABLE IF EXISTS ClanThread;

DROP TABLE IF EXISTS EventThread;

DROP TABLE IF EXISTS Response;

DROP TABLE IF EXISTS Comment;

DROP TABLE IF EXISTS Event;

DROP TABLE IF EXISTS Plays;

DROP TABLE IF EXISTS Game;

DROP TABLE IF EXISTS Profile;

DROP TABLE IF EXISTS GenreType;

DROP VIEW IF EXISTS RolesPlayed;

DROP VIEW IF EXISTS LargeAttendance;

DROP VIEW IF EXISTS MessageDisplay;

DROP VIEW IF EXISTS NumComments;

DROP TRIGGER IF EXISTS ClanCreated;

DROP TRIGGER IF EXISTS CommentEdited;

DROP TRIGGER IF EXISTS FriendsAdded;

DROP TRIGGER IF EXISTS ClanWelcome;