Denormalization Example

We chose to add the email to each row of community_members. If we're querying the members in a community, it's likely that we're also looking to communicate with them after searching for them through email. This saves us a join as we don't need to join community_members with the users table. To accomplish this, we added a trigger that fires on insert to the community_members table. It adds the appropriate email for the user being added to that community.

Views

#List the number of players and the name of the game for all games

```
create view game_population_view as
   select game_name, count(*) as num
   from user_games
   group by game_name;
```

#List the community member and the games that they play

```
create view community_member_game_view as
   select com_name, user, email, user_name, game_name
   from community_members c
       inner join user_games ug on c.user = ug.user_name;
```

#List the community member and their current status

```
create view community_member_status_view as
   select com_name, user, status, state, country, c.email
   from community_members c inner join users u on c.user = u.name;
```

#List the game name, cost of the game, and genres for every game

```
create view game_cost_genre_view as
  select game_name, cost, genre_name
  from game_genres inner join games on game_genres.game_name = games.name;
```

#List the user, the game, and the cost of the game for every game that the user plays

```
create view user_game_cost_view as
   select user_name, game_name, cost
   from user_games ug inner join games g on ug.game_name = g.name;
```

Queries

Note: Some of the queries use views which account for the joins that are required for each query.

1.List the game_name, the cost of the game, and the average rating of the game with at least 3 ratings with an average rating greater than or equal to 2. (ajs)

```
select game_name, cost, avg(rating) as rating
from games inner join game_ratings on games.name = game_ratings.game_name
where game_name in
 (select game_name
 from game_ratings
 group by game_name
 having count(*) >= 3)
group by game_name
having avg(rating) >= 2;
```

1. Skyrim,40,2.6667
2. Terraria,4.99,3.0000

2.List the pair of users and their number of genre matches where the number of genre matches between the two players is at least 2. (raj)

```
select u1.user_name, u2.user_name, count(*) matches
from user_genres u1 left outer join user_genres u2 on u1.genre_name =
u2.genre_name
where u1.user_name < u2.user_name and u2.genre_name is not null
group by u1.user_name, u2.user_name
having count(*) >= 2;
```

1. UserA,UserC,2
2. UserA,UserB,3

3.List the pair of users in ‘CommunityA’ who play the same game along with the game that the users match on. (rj)

```
select cm1.user as user1, cm2.user as user2, cm1.game_name as game
from community_member_game_view cm1 inner join community_member_game_view cm2
using (game_name)
where cm1.com_name = cm2.com_name and cm1.user < cm2.user and cm1.com_name =
'CommunityA';
```

1. UserA,UserB,Skyrim
2. UserA,UserB,Terraria
3. UserA,UserC,Skyrim
4. UserA,UserC,Terraria
5. UserA,UserD,Skyrim
6. UserB,UserC,Skyrim
7. UserB,UserC,Terraria
8. UserB,UserD,Skyrim
9. UserC,UserD,Skyrim

4.List the pair of users in ‘CommunityA’ who live in different countries where the status of both users is ‘in-game’. Also include the country and status of each user (rj)

```
select c1.user, c1.country, c1.status, c2.user, c2.country, c2.status
from community_member_status_view c1 inner join community_member_status_view c2
using (com_name)
where c1.user < c2.user and c1.country <> c2.country and c1.status = 'in-game'
and c2.status = 'in-game';
```

1. UserA,USA,in-game,UserC,Japan,in-game
2. UserB,USA,in-game,UserC,Japan,in-game

5.List the community and the owner of the community with the largest number of forum posts along with the number of forum posts. (ajs)

```
select com_name, owner, count(*) as num
from forum_posts inner join communities on com_name = name
group by com_name, owner
having count(*) =
   (select max(result.num)
   from (select count(*) as num
        from forum_posts
        group by com_name) as result);
```

1. CommunityA,UserA,3

6.List the game, cost of the game, and the total number of players for the game with the total number of players being greater or equal to 3. (ajs)

```
select game_name, cost, count(*) as num
from user_game_cost_view
where game_name in
  (select game_name
  from game_population_view gp
  where num >= 3)
group by game_name;
```

1. Skyrim,40,5
2. Terraria,4.99,3

7.List the users who like the genre that is the most common genre among games, along with their email and the genre name. (ajs)

```
select distinct name, email, genre_name
from user_genres inner join users on user_genres.user_name = users.name
where genre_name in
  (select genre_name
  from game_genres
  group by genre_name
  having count(*) =
      (select max(result.num)
      from (select count(*) as num
          from game_genres
          group by genre_name) as result));
```

1. UserA,a@a.com,action
2. UserA,a@a.com,rpg
3. UserB,b@b.com,rpg
4. UserC,c@c.com,action

8.List the users who have left a rating of 3 on games that the user plays and the name of the game. (js)

```
select ug.user_name, ug.game_name
from user_games ug
   inner join
   (select user_name, game_name
   from game_ratings
   where rating = 3) as result on ug.user_name = result.user_name
where ug.game_name = result.game_name;
```

1. UserA,Skyrim
2. UserC,Skyrim
3. UserA,Terraria
4. UserB,Terraria
5. UserC,Terraria

9.List the owner of the team, the date of the tournament, and the name of the team whose team is participating in a tournament that will take place before the year 2021. (j)

```
select owner, start_date, team_name
from tournament_teams tt
   inner join tournaments t on tt.tournament_name = t.name
   inner join teams on tt.team_name = teams.name
where year(start_date) < 2021;
```

1. UserA,2012-07-10,TeamA
2. UserA,2012-07-11,TeamA
3. UserA,2012-07-10,TeamB

10.List the game and the number of people that play the game where the cost of the game is less than the average cost of all games (ajs)

```
select game_name, gp.num
from game_population_view gp inner join
   (select name
   from games
   where cost <
       (select avg(cost)
       from games)) as result on gp.game_name = result.name
```

1. Genshin Impact,1
2. Hollow Knight,1
3. Terraria,3

Some Definitions

Some classes are not completely identifiable by their name alone.

UserGame: Games that the user likes to play
UserGenre: Genres that the user likes for their games
GameRating: A rating from the user
TournamentTeam: A signup sheet that indicates a team's interest in participating in a tournament
Event: An event that is hosted by a community for other community members to participate in
GameGenre: The genres of a game.