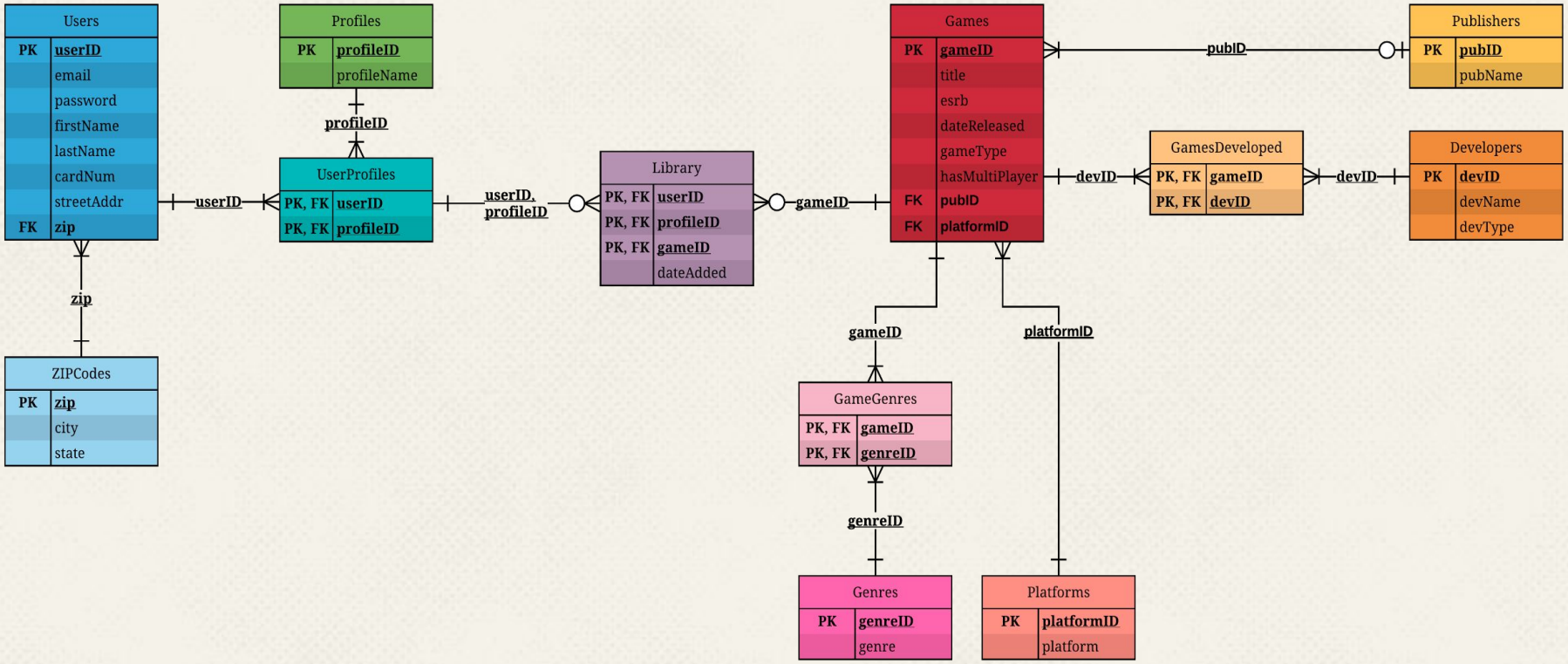
Games On Clouds

Table of Contents

Executive Summary

Game on Clouds is a Netflix-like alternative to gaming. It will be able to stream your favorite games without the need to have multiple consoles. After getting the agreement of various Publishers, Developers, and Console Makers, Game on Clouds will be able to stream games from your today's consoles such as the Playstation 4 or Xbox One

ER Diagram


Users
PK userID
email
password
firstName
lastName
cardNum
streetAddr
FK zip
Profiles
PK profileID
profileName
UserProfiles
PK, FK userID
PK, FK profileID
Library
PK, FK userID
PK, FK profileID
PK, FK gameID
dateAdded
Games
PK gameID
title
esrb
dateReleased
gameType
hasMultiPlayer
FK pubID
FK platformID
Publishers
PK pubID
pubName
GamesDeveloped
PK, FK gameID
PK, FK devID
Developers
PK devID
devName
devType
ZIPCodes
PK zip
city
state
GameGenres
PK, FK gameID
PK, FK genreID
Genres
PK genreID
genre
Platforms
PK platformID
platform
userID
profileID
userID, profileID
gameID
pubID
devID
devID
zip
zip
gameID
platformID
genreID

Tables

Users' Table

```
create table Users (
    userID        serial not null,
    email         text not null,
    password      text not null,
    firstName     text not null,
    lastName      text not null,
    cardNum       bigint not null check (cardNum <= 9999999999999999),    --Credit Card must be 16 numbers long
    streetAddr    text not null,
    zip           int not null check (zip <= 99999),   --ZIP codes must be 5 numbers long
    unique(userID, email, cardNum),
    primary key(userID),
    foreign key(zip) references ZIPCodes(zip)
);
```

Functional Dependencies:

userID → email, password, firstName, lastName, CardNum, streetAddr

Users' Table Example

userid integer	email text	password text	firstname text	lastname text	cardnum bigint	streetaddr text	zip integer
1	michael@marist.edu	dogs	Michael	Gutierrez	1234123412341234	3399 North Rd	12601
2	alan@labouseur.com	alpaca	Alan	Labouseur	1234432112344321	3399 North Rd	12601
3	bob@marist.edu	database	Bob	Smith	1234123412341231	93 Whitney Ave	6516

*Note that passwords will be encrypted on the server side and not on database itself

Games' Table

```
create table Games (
    gameID              serial not null,
    title               text not null,
    esrb                text not null,
    pubID               int,
    platformID          int not null,
    dateReleased        date not null,
    gameType            char(5) not null,
    hasMultiplayer      boolean not null,
    unique(gameID, title),
    primary key(gameID),
    foreign key(pubID)          references Publishers(pubID),
    foreign key(platformID) references Platforms(platformID)
);
```

Functional Dependencies:

gameID → title, esrb, dateReleased, gameType, hasMultiplayer

Games' Table Example

gameid integer	title text	esrb text	pubid integer	platformid integer	datereleased date	gametype character(5)	hasmultiplayer boolean
1	Final Fantasy XV	T	2	2	2016-11-29	AAA	f
2	Pokemon X	E	3	4	2013-10-12	AAA	t
3	Portal 2	E	6	1	2011-04-18	AAA	t
4	World of Warcraft	T	5	1	2004-11-23	AAA	t
5	Call of Duty: Modern Warfare 3	M	4	3	2011-11-08	AAA	t
6	The Last of Us	M	1	2	2013-06-14	AAA	t
7	No Man's Sky	T	<NULL>	1	2016-08-12	Indie	f
8	Counter-Strike: Source	M	2	1	2001-11-04	AAA	t

* Note that games can have one or no publishers

**Games are only streamed in one platform as streaming multiple platforms of the same game is not a wise decision

Profiles' & UserProfiles Table

```
create table Profiles (
    profileID       serial not null,
    profileName   text not null,
    unique(profileID),
    primary key(profileID)
);
```

Functional Dependencies:

profileID → profileName

```
create table UserProfiles (
    userID      int not null,
    profileID  int not null,
    primary key(userID, profileID),
    foreign key (userID)    references Users(userID),
    foreign key (profileID) references Profiles(profileID)
);
```

Functional Dependencies:

No functional dependencies

Profiles' & UserProfiles Table Examples

profileid integer	profilename text
1	Mike
2	Minion
3	Alpaca
4	Mom
5	Friend1
6	Cousin

Profiles

userid integer	profileid integer
1	1
2	2
2	3
1	4
2	5
1	6

UserProfiles

* Note that games can have one or no publishers

**Games are only streamed in one platform as streaming multiple platforms of the same game is not a wise decision

Library Table

```
create table Library (
    userID         int not null,
    profileID      int not null,
    gameID         int not null,
    dateAdded  date default current_date,
    primary key(userID, profileID, gameID),
    foreign key (userID, profileID) references UserProfiles(userID, profileID),
    foreign key (gameID)              references Games(gameID)
);
```

Functional Dependencies:

userID, profileID, gameID → dateAdded

Library Table Example

userid integer	profileid integer	gameid integer	dateadded date
1	1	1	2017-05-02
1	1	2	2017-05-02
1	1	3	2017-05-02
1	1	4	2017-05-02
1	1	5	2017-05-02
1	1	6	2017-05-02
1	1	7	2017-05-02
1	4	3	2017-05-02
1	6	5	2017-05-02
1	6	6	2017-05-02
2	2	2	2017-05-02
2	2	3	2017-05-02
2	2	4	2017-05-02
2	3	3	2017-05-02
2	3	4	2017-05-02
2	5	1	2017-05-02
2	5	6	2017-05-02
2	5	7	2017-05-02

*An auto timestamp keeps track of when users added games to their profile libraries

Genres' & GameGenres' Table

```
create tableGenres (
    genreID   serial not null,
    genre      text not null,
    unique(genreID, genre),
    primary key(genreID)
);
```

Functional Dependencies:

genreID → genre

```
create table GameGenres (
    gameID    int not null,
    genreID    int not null,
    primary key (gameID, genreID),
    foreign key (gameID)  references Games(gameID),
    foreign key (genreID)  references Genres(genreID)
);
```

Functional Dependencies:

No functional dependencies

Genres' & GameGenres' Table Examples

genreid integer	genre text
1	First Person
2	Role Playing
3	Massively Multiplayer Online
4	Puzzle
5	Adventure
6	Action
7	Shooter
8	Survival
9	Horror
10	Fantasy
11	Platformer

Genres

gameid integer	genreid integer
1	6
1	2
2	2
3	1
3	4
3	11
4	2
4	3
5	1
5	7
6	5
6	6
6	9
6	10
7	5
7	6
7	8
8	1
8	7

GameGenres

* Games can have multiple genres

Developers' & GamesDeveloped Table

```
create table Developers (
    devID         serial not null,
    devName    text not null,
    devType      char(2) not null,
    unique(devID, devName),
    primary key(devID)
);
```

Functional Dependencies:

devID → devName, devType

```
create table GamesDeveloped (
    gameID   int not null,
    devID      int not null,
    primary key(gameID, devID),
    foreign key (gameID) references Games(gameID),
    foreign key (devID)    references Developers(devID)
);
```

Functional Dependencies:

No functional dependencies

Developers' & GamesDeveloped Table

devid integer	devname text	devtype character(2)
1	Square Enix	3P
2	Valve Corporation	1P
3	Blizzard Entertainment	3P
4	Infinity Ward	3P
5	Naughty Dog	1P
6	Hello Games	ID
7	Game Freak	2P
8	Sledgehammer Games	3P

Developers

gameid integer	devid integer
1	1
2	7
3	2
4	3
5	4
5	8
6	5
7	6
8	2

GameGenres

* Games can have multiple developers

** Note that P stands for party, ID stands for Indie

Views

AllGames View

```
create or replace view AllGamesV as
        select distinct games.gameid,
                                title,
                                devname as "developer",
                                pubname as "publisher",
                                platform,
                                esrb,
                                gametype,
                                datereleased
        from Games left outer join Publishers on Games.pubid = Publishers.pubid
                        inner join Platforms  on Games.platformid = Platforms.platformid
                        inner join GamesDeveloped  on Games.gameid = GamesDeveloped.gameid
                        inner join Developers  on GamesDeveloped.devid = Developers.devid
        order by gameid asc;
```

AllGames View Example

gameid integer	title text	developer text	publisher text	platform text	esrb text	gametype character(5)	datereleased date
1	Final Fantasy XV	Square Enix	Square Enix	Playstation 4	T	AAA	2016-11-29
2	Pokemon X	Game Freak	Nintendo	Nintendo 3DS	E	AAA	2013-10-12
3	Portal 2	Valve Corporation	Valve Corporation	PC	E	AAA	2011-04-18
4	World of Warcraft	Blizzard Entertainment	Blizzard Entertainment	PC	T	AAA	2004-11-23
5	Call of Duty: Modern Warfare 3	Infinity Ward	Activision	Xbox One	M	AAA	2011-11-08
5	Call of Duty: Modern Warfare 3	Sledgehammer Games	Activision	Xbox One	M	AAA	2011-11-08
6	The Last of Us	Naughty Dog	Sony Interactive Entertainment	Playstation 4	M	AAA	2013-06-14
7	No Man's Sky	Hello Games	<NULL>	PC	T	Indie	2016-08-12
8	Counter-Strike: Source	Valve Corporation	Square Enix	PC	M	AAA	2001-11-04

* Note that games that are repeated more than once have multiple developers

**Games can have no pubishers

AllUsers View

```
create or replace view AllUsersV as
        select userid,
                firstname,
                lastname,
                email,
                city,
                state
        from Users inner join ZIPCodes on Users.zip = ZIPCodes.zip;
```

* This view simply lists all the users

AllUsers View Example

userid integer	firstname text	lastname text	email text	streetaddr text	city text	state character(2)
1	Michael	Gutierrez	michael@marist.edu	3399 North Rd	Poughkeepsie	NY
2	Alan	Labouseur	alan@labouseur.com	3399 North Rd	Poughkeepsie	NY
3	Bob	Smith	bob@marist.edu	93 Whitney Ave	West Haven	CT

* Note that passwords and credit card numbers are excluded as it is sensitive data

PopularGames View

```
create or replace view PopularGamesV as
        select Games.gameid, title,
                (select count(library.gameid)
                 from library
                 where library.gameid = games.gameid) as "TimesAdded"
        from Games
        order by "TimesAdded" desc;
```

* This view shows you the ranking of games by popularity (times added to library)

PopularGames View Example

gameid integer	title text	TimesAdded bigint
3	Portal 2	4
6	The Last of Us	3
4	World of Warcraft	3
7	No Man's Sky	2
1	Final Fantasy XV	2
2	Pokemon X	2
5	Call of Duty: Modern Warfare 3	2
8	Counter-Strike: Source	0

* Note that in this example Counter-Strike was never added to any profile's library

Stored Procedures

SearchByGenre Stored Procedure

```
create or replace function SearchByGenre(desiredGenre text, resultset refcursor)
returns refcursor as $$
declare
  desiredGenre     text := $1;
  resultset    refcursor := $2;
begin
open resultset for
        select  Games.gameid, title --,genre <--can include to see if correct genre
        from Games inner join GameGenres  on Games.gameid = GameGenres.gameid
                        inner join Genres  on GameGenres.genreid = Genres.genreid
        where genre = desiredGenre
        order by Games.gameid asc;
 return resultset;
end;
$$ language plpgsql
```

* This procedure allows you to search games by a specified genre

SearchByGenre
Stored Procedure Example

gameid integer	title text
1	Final Fantasy XV
6	The Last of Us
7	No Man's Sky

```
select
SearchByGenre('Action','results');

fetch all in results;
```

* This shows us action games

SearchByDev
Stored Procedure

```
create or replace function SearchByDev(desiredDev text, resultset refcursor)
returns refcursor as $$
declare
  desiredDev       text := $1;
  resultset    refcursor := $2;
begin
open resultset for
        select  Games.gameid, title--, devName as "Developer" <-- can include to check if correct developer
        from Games inner join GamesDeveloped  on Games.gameid = GamesDeveloped.gameid
                        inner join Developers  on GamesDeveloped.devid = Developers.devid
        where devName = desiredDev
        order by Games.gameid asc;
return resultset;
end;
$$ language plpgsql
```

* This procedure allows you to search games by a specific developer

SearchByDev
Stored Procedure Example

gameid integer	title text
3	Portal 2
8	Counter-Strike: Source

```
select SearchByDev('Valve Corporation','results');
fetch all in results;
```

* This shows us games made by Valve Corporation

SearchByPublisher Stored Procedure

```
create or replace function SearchByPublisher(desiredPublisher text, resultset refcursor)
returns refcursor as $$
declare
  desiredPublisher     text := $1;
  resultset         refcursor := $2;
begin
  open resultset for
        select  Games.gameid, title--, pubisher <-- can include to check if correct developer
        from Games inner join Publishers  on Games.pubID = Publishers.pubID
        where pubName = desiredPublisher
        order by Games.gameid asc;
  return resultset;
end;
$$ language plpgsql
```

**This procedure allows you to search games by a specific published

SearchByPublisher
Stored Procedure Example

gameid integer	title text
5	Call of Duty: Modern Warfare 3

```
select SearchByPublisher('Activision','results');
fetch all in results;
```

* This shows us games published by Activision

Reports

TotalGames Report

```
select count(*) as "TotalGames"
from games;
```

*This shows you the total number of games offered

TotalGames bigint
8

UserLocation Report

```
select state, count(state) as "NumOfUsers"
from Users inner join ZIPCodes on users.zip = ZIPCodes.zip
group by state
order by count(state) desc;
```

*This shows you the distribution of users per state

state character(2)	NumOfUsers bigint
NY	2
CT	1

TopUserState Report

```
select state, count(state) as "NumOfUsers"
from Users inner join ZIPCodes on users.zip = ZIPCodes.zip
group by state
order by count(state) desc
limit 1;
```

*This shows you which state has the most users

state character(2)	NumOfUsers bigint
NY	2

TopGame Report

```
select Games.gameid, title,
        (select count(library.gameid)
         from library
        where library.gameid = games.gameid) as "TimesAddedToLib"
from Games
order by "TimesAddedToLib" desc
limit 1;
```

*This shows you the most popular game (game that has been added to libraries the most)

gameid integer	title text	TimesAddedToLib bigint
3	Portal 2	4

Triggers

esrbCheck Report

```
create or replace function esrbCheck()
returns trigger as $$
declare esrbCheck varchar(1);
begin
   select esrb into esrbCheck
   from Games
   where games.gameid = NEW.gameid;

   if esrbCheck       != 'E'    and
          esrbCheck  != 'E10' and
          esrbCheck  != 'T'    and
          esrbCheck  != 'M'
     then raise exception 'Unexpected esrb';
   end if;
   return NEW;
end;
$$ language plpgsql;
```

```
-- Testing esrbCheck trigger

create trigger esrbCheck after update or insert on Games for each row execute procedure
esrbCheck();


insert into Games (title, esrb, dateReleased, gameType, hasMultiPlayer, pubID, platformID)
          values('Final Fantasy','D', '11/29/2016','AAA', false, 2, 2);
```

The sql above gives us this error

```
ERROR:  Unexpected esrb
CONTEXT:  PL/pgSQL function esrbcheck() line 12 at RAIS

********** Error **********

ERROR: Unexpected esrb
SQL state: P0001
Context: PL/pgSQL function esrbcheck() line 12 at RAISE
```

Security

Administrators & Advertisers Roles

```
create role admin;

grant all

on tables in schema public

to admin;
```

```
-- Administrators have the most access

-- They are allowed to insert update or delete, and are carefully chosen
```

```
create role advertisers;

grant select

on tables AllUsersV

to advertisers;
```

```
-- Advertisers can only see some User information

-- What they really want is your email so that they can send you marketing material
```

Implementation & Problems

Implementation

- This database was created under a few assumptions
  - It only supports Users in the United States
  - It is receiving a valid credit card number
- Games
  - Can only have one publisher
  - Can have multiple developers
  - Can have multiple genres
  - Are only offered on one platform
    - No need to stream same game on multiple platforms
- Profiles
  - Users can have many profiles

Problems To Solve in the Future

- When using AllGamesV (view) it will list games with multiple developers more than once
    - Also applies when adding genre column to the query
- Users can have unlimited profiles
- Users can enter anything for their email