INST327 0201 - Group 1 - Final Project

Jonathan Obenland, Peter Schultz, Alexander Janzen, Yosef Solomon, Jacob Krause

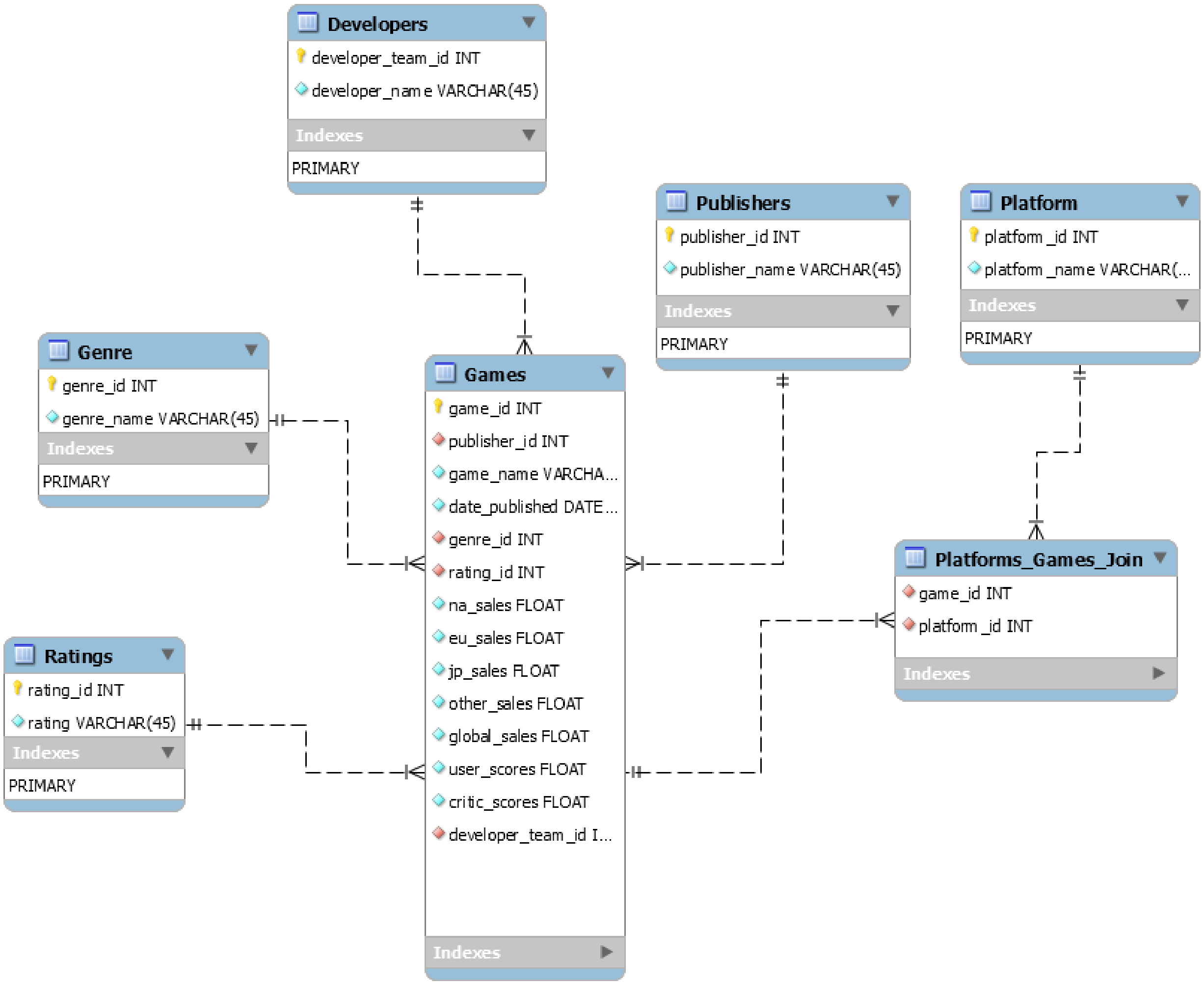
Introduction:

Our final database is a representation of information about video games, which includes genres, ratings, development teams, publishers, platforms and games themselves. Developers may not be sure of regions to release their games, or what genres to develop to be the most popular. After the creation of the database, we can solve some of these issues by adding these data points to a database and then querying it to see trends that we can report to potential clients.

The dataset is a list of sales of video games that have managed to sell more than a hundred thousand copies. Out of the 16,000 observations in the CSV, the AMP advised up to lower down to 100 data points because it was more in the scope that was required for the project. Along with the data, we are still going with the initial video game CSV. Out of the observations, we have data for the publisher, developers, genre, sales, games, ratings, and platform. Since the CSV file has been limited, we will include a data point from all the categories including Action, Adventure, Fighting, Misc, Platform, Puzzle, Racing, Role-Playing, Shooter, Simulation, Sports and Strategy in the form of a genre id. Because video games are a popular topic in today's world, we can answer many questions describing the data in our database to aid developers to target their products to a larger audience.

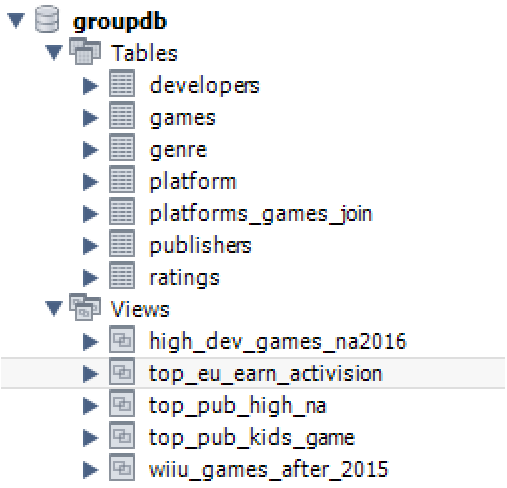
Logical Design:

As shown below, our ERD contains seven tables with there only being one join table. During our normalization process, we split the data into groups which were determined by one-to-many relationships. The included the Genre, Rating, Developers, Publishers and Platform tables. The reason why sales does not have its own table is because the dataset is a record of games from a specific date in time, meaning that there is only one sales value for every game causing a one-to-one relationship. The same also applies to user and critic scores as they have a one-to-one relationship with each game.

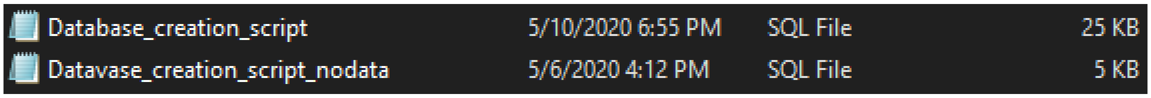


Physical Database:

All of the information that was included on the csv was split into corresponding table names based off of their column. Some columns such as date_published, did not require a seperate table to reference as each date was seemingly unique. The views were then generated by our queries using shorthand names to know which one was which. We then packaged up the data (created a backup) and sent it to group members to ensure that the database was created successfully.

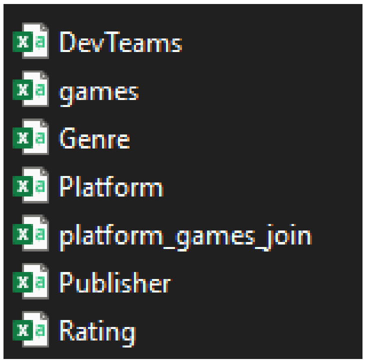


- **groupdb**
  - Tables
    - developers
    - games
    - genre
    - platform
    - platforms_games_join
    - publishers
    - ratings
  - Views
    - high_dev_games_na2016
    - top_eu_earn_activision
    - top_pub_high_na
    - top_pub_kids_game
    - wiiu_games_after_2015



```
Database_creation_script         5/10/2020 6:55 PM    SQL File        25 KB
Datavase_creation_script_nodata  5/6/2020 4:12 PM     SQL File         5 KB
```

Sample Data:



```
DevTeams
games
Genre
Platform
platform_games_join
Publisher
Rating
```

Each of our data in our database was limited and cleaned using R and then we split each individual file into its correlating database and then entered it into the actual database tables. Most of our values in-terms of naming convention are realistic; there is no inconsistency; each name of the video games, publisher companies and genres are exactly the same if you were to search it manually via online for comparison.

Views/Queries:

View Name:	**Req.A**	**Req. B**	**Req.C**	**Req.D**	**Req. E**
top_eu_earn_activision	**X**	**X**	**X**		**X**
top_pub_high_na	**X**		**X**		
high_dev_games_na2016	**X**	**X**			
top_pub_kids_game	**X**	**X**	**X**		
wiiu_games_after_2015	**X**	**X**		**X**	

Changes from the original design:

After analyzing the relationships between our database objects we concluded that we “cross referenced” a lot of tables creating foreign and primary keys in places that they did not belong. Some of the tables that we created used a linking table as if they represented a many to many relationship when in reality, it was a simple one to many design. Furthermore, we corrected the games table to include critic and user scores as they did not belong in the “ratings” table in which they previously resided. Our final model had games as the center database with all other tables referencing their primary keys, as foreign keys, in the main games table. The only table that had a linking table was the platforms table, which held information regarding what system the game was released on. Since any one game can have multiple platforms and any platform can have multiple games, a linking table was required to include this information on the database.

One of the main problems that we ran into when creating this database was the representation of the developer teams. Although this could have gotten split into multiple portions with each developer name having their own developer id, we found this to be difficult and instead made each team a "developer team" without having to perform regular expressions to get each individual developer.

Lessons Learned:

We learned that it was easier to reduce the amount of observations in our dataset to at least 100 and to remove any empty or null values. By doing this, the data was much easier to manage and write queries for.

We also learned how to better identify relationships between tables in that we had one-to-one relationships in place of one-to-many relationships, which were pointed out by Professor Duffy. Moving forward, our group knows how to identify these types of logical errors and how to correct them.

Overall, one of the main things that we learned about was importing the data into a database. It's one thing to learn about how to use SQL with the datafiles that we were given, but a whole new learning curve is needed when actually importing your own CSV into respective tables to store the data in a "custom" way. We can use these skills in our future jobs when our employers may want to store their data in an organized way, in somewhere other than a simple CSV.

Lastly, one of the more important lessons learned was about team work / team dynamics. While most teams go through stages where they initially form, dispute, get used to each other's work habits, and eventually become a solid working team, our team worked well from our initial meeting. We did have issues communicating, but we all worked well together from our initial proposal up through our final project. Our work schedules did conflict, but we overcame obstacles when they appeared and were able to successfully get together and do quality work done in an efficient manner.

Potential Future Work:

We are certain that our database will be useful for those seeking information regarding video game statistics. For example, video game journalists could use this information to create an article on "Most popular video games ranked by publishers from years X - Y" and use the data that we have indexed as a reference. Publishers could also use our database to help predict how they can better advertise to their demographics, since platforms, ratings, and genres are influencing factors which they may consider during a game's development cycle. A Japanese Indie game company working on their newest puzzle game might want to consider publishing their game internationally if our database indicates that past puzzle games (or any game genre of interest) were successes in foreign countries. In essence, our database would be an extremely useful asset for video game companies to utilize.

As far as adding future work, we could add some more data to the tables we already have and make our analysis even more accurate that it already is. Furthermore, an app can be created to query the data and display it in a graphical way that would allow non technical users to still be able to read, understand, and analyze the data in our database.