CIS 451/551 Final Project

# Fall 2021

## Name: Graham Whealan

## Project Title: The Game Collection Database

## Connection Information:

### -

## Project URL: <https://ix.cs.uoregon.edu/~gwhealan/home.html>

# Summary:

In this project I set out to create a games database for old consoles such as the MSX, NES, SNES, and Gameboy Series of consoles. My main goal in pursuing this project was to create a website that could organize games by regional release, console, genre, and series data with competent search tools that do not require much prior knowledge. Because of this I chose the search table.

# Logical Design:

# Physical Design:

* **Company:** This table stores company information on game developers, publishers (titled producers in the table), and console designers. First is a unique id – “cID” – which is a 4-character string that represents the companies’ stock id. As it is the key value it cannot be null. Next, we have the company name which is a 32-character string. Following this we store when the company was founded and when it closed. Obviously since not all companies have shut down the closing date can be null. This is followed by the address and parent identifier. The address is stored as text since addresses can vary in length greatly depending on the province or country; and the parent identifier – parent\_cID – stores a reference to the parent company.
* **Console:** This table stores console information such as the console name, release date, and the manufacturer/developer. Console’s primary ID is a 3-character string which holds a common abbreviation for the console (ex: SFC for the Super Nintendo). As previously stated, “developer” is a foreign key reference to company, storing the consoles main developer / manufacturer.
* **Tag:** This table stores genre and theme information as both a short and long name. If the short name is long enough to hold the full string no long name is required. For efficiency, the tags are stored with an auto incremented primary key called tID.
* **Series:** Series stores information on game franchises. As such it stores a name, and an about section. It also uses an auto incremented primary key for efficiency.
* **Region:** Region stores data on the different regions games are released in. Its primary key is a common abbreviation for the region. It also stores a full-length region name.
* **Meta\_Game:** Meta Game stores basic information that is consistent between game releases and consoles such as expected player count. It also stores an optional reference to series so the two can be linked. Meta Game uses an auto-incremented primary key.
* **Game\_Port:** Game port stores development information on releases and remakes. As such it’s primary keys are the game Id referenced from Meta\_game, and a console id referenced from console. The table also stores 2 foreign keys to company for game developers and producers.
* **Game\_release:** Game release stores regional specific data like the regional release name and release date. It is dependent on region and game\_port, which defines it’s primary key
* **Game\_tag:** A many-to-many table between Meta\_Game and Tag.

# List of Applications:

1. **Dynamically sized checkbox:** The checkboxes on the main search page are dynamically built from table references. Only consoles, tags and regions that are referenced will appear in the list. These link console and game\_port; region and game\_release; and tag and game\_tag respectively to glean the information.
2. **Search Script:** The search script links all tables together to allow one to specify each and every category listed. Only region, console, and tag are not linked as that information can already be gotten from the foreign keys.
3. **Series – First and Final release:** This script finds the max and min dates of games in a series. To do so it links series, meta\_game, and game\_port to connect the game\_release date to the series.
4. **Series List:** This script links series to meta\_game, game\_port, game\_release, and company to get all information on the games in the series.

# User Guide

Most of the project is implemented through php scripts, as such when you visit the home page you will find it pretty bare. Simply click the “enter” button and you will be taken to the main database. On the left side you will find some categories from which you can specify your search. Console, and region preform a “OR” search while tags will search for all cases. In the search section you will find search bars for game, series, producer, and developer. These searches use the “LIKE” keyword so have fun with some wild searches. Lastly, in the series column you can click the reference to see more information.

# Contents of tables

I will be submitting a dump of the tables along with this document.

# Implementation Code

The PHP scripts will also be included in the submission. I also attached some test and import sql scripts I was working on.

# Conclusion

Ultimately, I have mixed feelings on what I have put into this project. Although I was able to create a pretty capable system with expandability in mind, I wanted to go further. The major thing I wanted to design was an upload script to make adding new games, series, companies, tags, etc. significantly easier however that proved difficult with my limited PHP knowledge. If I were to continue this project further, and I probably will, I’d start by building scripts for publishers and developers and implement more interactivity between the separate pages. I’d also like to introduce an image table so specific pages can references game covers, logos, screenshots, etc. Lastly, and most significantly, I’d like to move away from PHP as I feel JavaScript is a much more versatile tool for adaptive websites.