**Introduction**

For this project we have been tasked with creating an SQL database for any form of system of our choosing. I have decided to make my database based off a fictional video game, like that of Team Fortress 2 or Counter Strike: Global Offensive.

**Scenario**

* This database is inspired by how I would assume the data systems for Team Fortress 2, of course simplified but still holding to the same ideas and structures.
* This database stores information on many major parts of a game’s data infrastructure, in this case being the platforms and marketplaces the game is sold on, the servers which run instances of the game for players to play on, the players themselves with their profiles, and the items which players can own within the game.
* A platform has its platform id as its primary key, which is typically a shortened variation on the platforms name, but it doesn’t need to be mandatorily.  
  These platforms have a name, an owning company or person, a price which the game is sold on said platform and the number of sales the game has had on that platform.
* A server is where players play the game together, each has a unique 12-digit server id as its primary key and has the platform id of the platform the server caters to, and the player id of the player who owns the server as its foreign keys. Each server also has a name and location, which are shown to players to best pick a server which fits what they are looking for.
* A player account is rather simple, with its player id as its primary key and platform id as its foreign key, showing what platform the player bought the game from. They also have their username which is what is seen by other players.
* Lastly items work by having an item id as a primary key to reference a specific instance of an item, and its owner’s player id which acts as the foreign key. An item also has an item type, which is an identifier used by the game to know what kind of item it is, and as such allow it to render correctly in-game, and items can also have an item name, this name can be changed by the item’s owner and will show to other players who view the owner’s inventory. They also store the time they are created in unix time.

**Entity Relationship Diagram**

Diagram

Description automatically generated

**Sample Data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **platforms** | | | | |
| **platform\_id (PK)** | **name** | **owner** | **pricing** | **sales** |
| WIIU | Wii U | Nintendo | 40 | 1000 |
| SWITCH | Nintendo Switch | Nintendo | 60 | 25000 |
| PS3 | PlayStation 3 | Sony | 40 | 9000 |
| PS4 | PlayStation 4 | Sony | 60 | 23000 |
| PS5 | PlayStation 5 | Sony | 60 | 21000 |
| XB360 | Xbox 360 | Microsoft | 40 | 12000 |
| XB1 | Xbox One | Microsoft | 60 | 26000 |
| XBSX | Xbox Series X | Microsoft | 60 | 19000 |
| STEAM | Steam | Valve | 50 | 150000 |
| EPIC | Epic Games Store | Epic Games | 50 | 90000 |

|  |  |  |
| --- | --- | --- |
| **players** | | |
| **player\_id (PK)** | **platform\_id (FK)** | **name** |
| 547432520446 | STEAM | AllisonPlays |
| 594720282446 | XBSX | DisguisedKitten |
| 594722333496 | SWITCH | thomasmatthews2005 |
| 594722502646 | XB1 | Emality |
| 594722522446 | STEAM | Allielife |
| 594722523446 | EPIC | Garrisonior |
| 594722982446 | STEAM | papaya |
| 594725528446 | WIIU | Niamh Murphy |
| 599527132446 | STEAM | festive noisemaker |
| 694702577446 | PS5 | MaddyClimbs |

**Sample Data (cont.)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **servers** | | | | |
| **server\_id (PK)** | **platform\_id (FK)** | **player\_id (FK)** | **name** | **location** |
| 000000000329 | STEAM | 594722502646 | Official PC Server 3 | EU |
| 000000001686 | STEAM | 594722502646 | Official PC Server 4 | EU |
| 000000001943 | STEAM | 594722502646 | Official PC Server 8 | ASIA |
| 000000005122 | STEAM | 594722502646 | Official PC Server 11 | US-CENTRAL |
| 000000005142 | STEAM | 594722502646 | Official PC Server 13 | US-WEST |
| 000000005906 | STEAM | 594720282446 | the funny zone | EU |
| 000000006573 | STEAM | 594722522446 | Allison's Alley NA | US-CENTRAL |
| 000000007318 | STEAM | 594722522446 | Allison's Alley EU | EU |
| 000000008438 | SWITCH | 594722502646 | Official Switch Server 2 | ASIA |
| 000000009094 | PS5 | 594722502646 | Official PS5 Server 4 | EU |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **items** | | | | |
| **item\_id (PK)** | **item\_type** | **creation\_date** | **item\_name** | **player\_id (FK**) |
| 189225318711 | cowering\_hat | 1616419341 | Scaredy Hat | 594722522446 |
| 206336227507 | demeaning\_showl | 1578671440 |  | 594722982446 |
| 291384798235 | rotating\_skirt | 1649304763 | the spinner | 594722502646 |
| 496789188508 | overbearing\_overcoat | 1530423839 | thiccums | 594720282446 |
| 598748252189 | inexpensive\_gest | 1535352421 |  | 594722333496 |
| 632356448941 | rotating\_skirt | 1485626278 |  | 594722982446 |
| 831610863354 | suspicious\_stare | 1623862380 | the pretender | 599527132446 |
| 850259986450 | jumping\_jacket | 1554106868 | The Speedster | 594722333496 |
| 860763853015 | hysterical\_handbag | 1549315259 |  | 547432520446 |
| 948206669579 | notorious\_knot | 1603282848 | the funny hat | 594720282446 |

**SQL Test Statements**

SQL Test Statements are included in the .sql file.

**Normalisation**

For my tables, they were designed from the beginning with object-oriented principals in mind, in such a way that from the beginning the tables were capable of reaching the third normal form.

For instance, each table has one primary key which is the only thing which over tables reference, for example items will refer to their owners using their primary key, which is the player\_id. This is the same for servers, keeping the player\_id of their owner and using that as their only reference.

All information for any data entry contains only unique data, with as few repeats as possible. As it stands the only data which could be split into its own table is the regions used to keep track of the locations of servers, but all other data points are unique and non-repeating. The only piece of data which could be seen as redundant is the sales figures, as in full application, you could find this by counting the number of players on any given platform, however in such a small example this is left in due to the small size of the sample data.

None of the data in the tables is redundant, it is possible to generate some data such as the total profits on any given platform, by multiplying the total sales by the sale price, however there is no need to have this in the database unless it was information accessed very often.

Of the four tables included, each focuses on a major part of the game and its services. The platforms table is important for any game which is playable on multiple platforms, and keeps the sales and prices separated, this could be removed if the game only existed on one platform.