

**Database Documentation**

**Cover**

****

**Game Overview**

Our game is a turn-based, 2-player, browser board game, where the hardware components of two PCs that were merged, fight each other until only one of the PCs is left.

Before the game starts, players buy any number and combination of characters using 16 Bits, a currency. There are 5 different characters to choose from.

During the game itself, players can place, move, or attack with their characters, all of which end their turn.

A player wins when all the enemy characters or CPU are destroyed.

**Database Dictionary**

The database of Hardware Warfare has 7 tables that make the game work!

**Table player**

The first table is the player, and it is where the base information of the player is located at such as the player id that helps to make connections to other tables.

|  |  |  |
| --- | --- | --- |
| Name | Type | Comments |
| player\_id | int not null auto\_increment | Primary Key. |
| player\_wins | int not null | The number of wins a certain player has |
| player\_losses | int not null | The number of losses a certain player has |
| player\_username | VARCHAR(45) not null | The username of the player |
| player\_email | VARCHAR(45) not null | The email of the player |
| player\_password | VARCHAR(45) not null | The password of the player |

**Table character\_**

The next table is character\_ where all the characters names, stats and cost show up. These values are always the same throughout the game. This table helps make connections with the match\_player\_character table.

|  |  |  |
| --- | --- | --- |
| Name | Type | Comments |
| cha\_id | int not null auto\_increment | Primary Key. |
| cha\_name | VARCHAR(45) not null | The name of every character. |
| cha\_hp | int not null | The health percentage that the characters (cha) start with. |
| cha\_atk | int not null | The amount of damage the cha deals onto other cha |
| cha\_spd | int not null | The amount of tiles that a cha can move in one turn |
| cha\_range | int not null | The amount of tiles that a cha can attack from |
| cha\_target | int not null | There’re two types of targets, singular attack, only attacks one cha at a time (0) and area attack, attacks more then one cha at a time |
| cha\_cost | int not null | The cost the each cha cost |

**Table match\_state**

The next table is the match\_state, and it is where the different states of the game are. There are different types of states such as, “Buying”, “Waiting”, “Gameplay”, etc.

|  |  |  |
| --- | --- | --- |
| Name | Type | Comments |
| match\_state\_id | int not null auto\_increment | Primary Key. |
| match\_state\_name | VARCHAR(45) not null | The name to differentiate the different states |

**Table match\_**

The next table is the match\_, and it is where the base information of the match is located at. This table helps, in the future make connections with the player, character and inventory.

|  |  |  |
| --- | --- | --- |
| Name | Type | Comments |
| match\_id | int not null auto\_increment | Primary Key. |
| match\_player1\_id | int not null | The id of the player 1 |
| match\_player2\_id | int not null | The id of the player 2 |
| match\_winner | int not null | Once the games finishes, the id of the winner will be here |
| match\_turn | int not null | The current turn the player is on |
| match\_state\_id | int not null | The current state the player is on, a foreign key from match\_state |

**Table inventory**

The next table is the inventory, It is the inventory of one player of one certain match. After the game ends, players have to create once again, by buying characters. This table will have connection with the match\_player and match\_player\_character to place the characters in the board (yet to be implemented)

|  |  |  |
| --- | --- | --- |
| Name | Type | Comments |
| inv\_id | int not null auto\_increment | Primary Key. |
| inv\_match\_id | VARCHAR(45) not null | A foreign key to get the match |
| inv\_player\_id | int not null | A foreign key to get the player |
| bits | int not null | The amount of bits the player has starting always at 16 |
| n\_rambow | int not null | The amount of Rambow’s the player has bought |
| n\_elventito | int not null | The amount of El Ventito’s the player has bought |
| n\_gipio | int not null | The amount of Gipio’s the player has bought |
| n\_decibelle | int not null | The amount of Decibelle’s the player has bought |
| n\_rommy | int not null | The amount of Rommy’s the player has bought |

**Table match\_player**

The next table is the match\_player, and it is where the connection of the match with player happens and with the inventory.

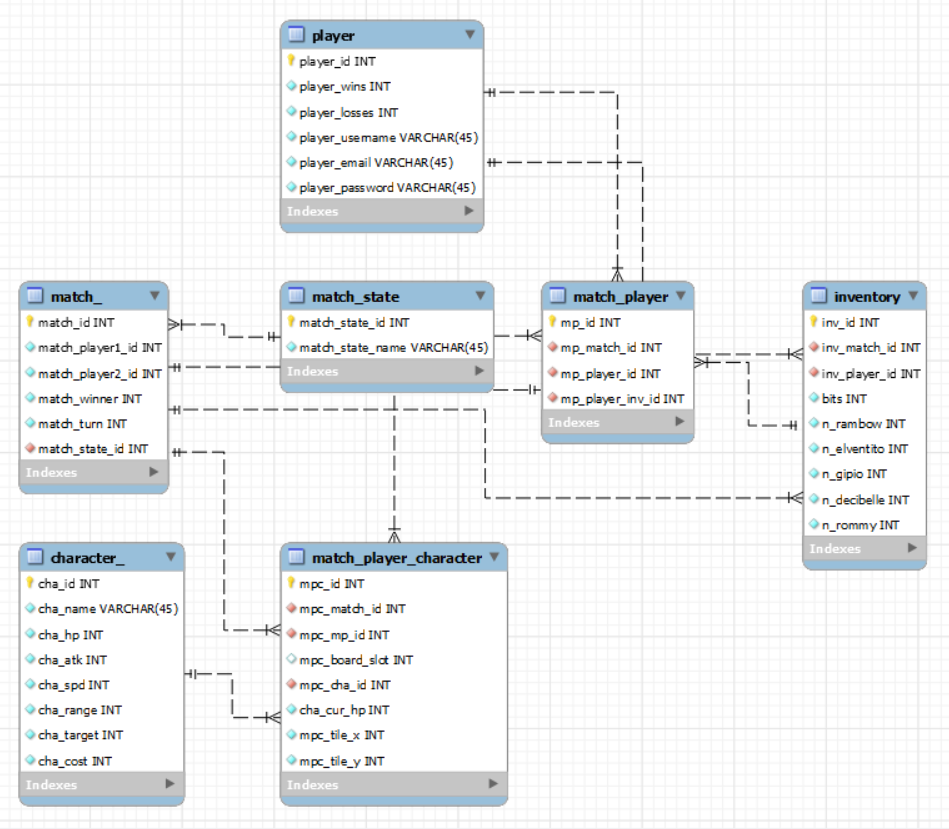
|  |  |  |
| --- | --- | --- |
| Name | Type | Comments |
| mp\_id | int not null auto\_increment | Primary Key. |
| mp\_match\_id | int not null | The foreign key that connects with the match\_ and gets the match |
| mp\_player\_id | int not null | The foreign key that connects with the player and gets the player id |
| mp\_\_player\_inv\_id | int not null | The foreign key that connects with the inventory and get the inventory of one player |

**Table match\_player\_character**

The next table is the match\_player\_character, and it is where the connection of the match\_, match\_player and characters happen. Here is where the characters from the inventory will go to to be placed on the board (mpc\_board\_slot, mpc\_tile\_x and mpc\_tile\_y) and also the number and type of characters and the current health.

|  |  |  |
| --- | --- | --- |
| Name | Type | Comments |
| mpc\_id | int not null auto\_increment | Primary Key. |
| mpc\_match\_id | int not null | The foreign key that connects with match\_ and gets the match |
| mpc\_mp\_id | int not null | The foreign key that connects with the match\_player and get the match\_player id |
| mpc\_board\_slot | int not null | Order by which characters are placed on the board by each player, from 1 to 5. |
| mpc\_cha\_id | int not null | Gets the character id of one character on the board |
| mpc\_cur\_hp | int not null | The current health that a character has |
| mpc\_tile\_x | int not null | The position on the x-axis of one character from 1 to 9 |
| mpc\_tile\_y | int not null | The position on the y-axis of one character from 1 to 7 |

**Database Model**

****

**Conclusion**

To summarize, the database of Hardware Warfare (HW) consists of 7 tables that are connected by different types of connections, such as “one to one” or “one to many” and it’s the base of the entire game, it’s a playable, ugly, “game” just using the database!

Still, there’re some things we still want to implement in HW, such as the placement of characters from the inventory into the board, and when a character dies, not only lose all the hp but also disappear, die.