**ark\_mmorpg Database**

**Description:**

ark\_mmorpg 2019 MLDM193-01 (V1.A) Database Project

When running the database please run in the following order:

1. ark\_mmorpg database creation (1)
2. ark\_mmorpg triggers (2)
3. ark\_mmorpg stored procedures (3)
4. The backup file is also included in a zipped folder.

**Tables:**

**player**

* Primary key: playerID (autogenerated number)
* Other attributes:
  + firstName: Players first name
  + lastName: Players last name
  + email: Players email address (Needs to be unique)
* Sample data: 25 lines

**playerAccount**

* Primary key: playerTag (Chosen by the player, however it needs to be unique in order to be accepted).
* Foreign key: playerRef (References playerID from the player table)
* Other attributes:
  + accountStatus: Current account status (blocked/open)
  + paymentDate: Date of last payment
  + expiryDate: When they account expires/ will expire
* Sample data: 25 lines

**characters**

* Primary key: characterID
* Foreign key: playerIden (References playerTag from the playerAccount table)
* Other attributes:
  + characterName: player chosen name for their character
  + skillLevel: character level. All new characters start on level one.
  + team: The team/ guild that the character belongs to
* Sample data: 20 lines

**Items**

* Primary key: itemID (autogenerated numbers)
* Other attributes:
  + itemName
* Sample data: 36 lines

**inventorySlots (intersecting entity)**

* Primary key: Composite Primary key formed by: itemRef and characterRef
* Foreign key:
  + characterRef (References characterID from the characters table)
  + itemRef (References itemID from the items table)
* Other attributes:
  + slotID
  + stackQuantity
* Sample data: 25 lines

**errorLog**

* Primary key: errorID (autogenerated numbers)
* Other attributes:
  + errorType
  + errorMessage
* Sample data: 5 lines

**Indexes:**

Indexes were created on the following tables:

* player table: Indexes created on playerID and email (indx\_player)
* playerAccount table: Indexes created on playerTag and accountStatus (indx\_playerAccount)
* characters table: Indexes created on characterID and skillLevel (indx\_character)
* items table: Indexes created on itemID and itemName (indx\_items)
* inventorySlots table: Indexes created on slotID, itemRef and stackQuantity (indx\_slots)
* errorLog table: Indexes created on errorID and errorMessage (indx\_errorLog)

**Views:**

* vwBlockedAccounts: This view will search and display all accounts that are currently blocked using the playerAccount table.
* vwTopSkill: This view will search and display the top 20 characters with the highest skill levels and their account details using the player, characters and playerAccount tables.
* vwTopStackedItems: This view will display the top 20 most stacked items and in which characters inventory they are stacked using the characters, items and inventorySlots tables.
* vwPopItems: This view will search and display the top 5 most popular items contained inside a character’s inventory using the characters, items and inventorySlots tables.

**Triggers:**

* player table: After insert trigger created that will display ‘New player inserted successfully’ when a new player has been added to the table. (tr\_notify\_player)
* items table: Instead of insert, delete, update trigger created that will display ‘You cannot edit the information in that table',5,6’. Which will then be inserted into the errorLog table with the following information:
  + 001,'USER TRIED TO EDIT TABLE ITEMS. TIME:' + CAST(GETDATE() AS VARCHAR))
  + This lets us know what time and when as well as which user created the error by trying to edit data in this table.

**Stored Procedures:**

Please run this one stored procedure at a time.

* spRegister: Registers a new player account and checks to make sure the user does not already exist within the database
* SpAddtime: Adds play time to the selected players account. It also checks the user exists before doing so
* spAddItem: Adds an item to a character’s inventory. It first checks to see if there is space in the inventory and whether the item can be stacked or not. If there is no space, an error message will pop up warning the player that they do not have space for more items.
* spAddChar: Adds a character to a players account. It first checks that the playerAccount exists and then adds the new character with a skill level of one.
* spSendLetter: Prints out a letter for each account with their play time left as well as the option to subscribe to a newsletter. This was done using a cursor instead or iteration.