Software Design Document

Puzzles and Dragons Database

Author: William Gale

Date: 2017-05-16

Version: 1.3

Organization: McKenzie SWTS1104

# Change History

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Version | Description |
| **William Gale** | 2017-05-11 | 1.0 | Initial Scribing and layout |
| **William Gale** | 2017-05-12 | 1.1 | Added Logical Viewpoint - WIP |
| **William Gale** | 2017-05-15 | 1.2 | Added Dependency viewpoint, First iteration of logical viewpoint complete. |
| **William Gale** | 2017-05-16 | 1.3 | Added more viewpoint, estimated resource usage. |
|  |  |  |  |

# Table of Contents

[1 Change History 2](#_Toc482689319)

[2 Table of Contents 3](#_Toc482689320)

[2.1 Figure Index 5](#_Toc482689321)

[2.2 Table Index 5](#_Toc482689322)

[3 Purpose 6](#_Toc482689323)

[4 Scope 6](#_Toc482689324)

[5 Context 6](#_Toc482689325)

[6 Summary 6](#_Toc482689326)

[7 References 7](#_Toc482689327)

[8 Glossary 7](#_Toc482689328)

[9 Identified Stakeholders and Design Concerns 8](#_Toc482689329)

[10 Design Viewpoints 8](#_Toc482689330)

[10.1 Introduction 8](#_Toc482689331)

[10.2 Context Viewpoint 9](#_Toc482689332)

[10.2.1 User Use Case Diagram 9](#_Toc482689333)

[10.3 Composition Viewpoint 10](#_Toc482689334)

[10.3.1 Deployment Diagram 10](#_Toc482689335)

[10.4 Logical Viewpoint 11](#_Toc482689336)

[10.4.1 MonsterClass Table 11](#_Toc482689337)

[10.4.2 Attribute Table 13](#_Toc482689338)

[10.4.3 MonsterType Table 13](#_Toc482689339)

[10.4.4 Growth Table 14](#_Toc482689340)

[10.4.5 ActiveSkill Table 14](#_Toc482689341)

[10.4.6 LeaderSkill Table 15](#_Toc482689342)

[10.4.7 AwokenSkill Table 15](#_Toc482689343)

[10.4.8 AwokenSkillList Table 16](#_Toc482689344)

[10.4.9 EvolutionTree Table 16](#_Toc482689345)

[10.4.10 LatentSkill Table 18](#_Toc482689346)

[10.4.11 Player Table 18](#_Toc482689347)

[10.4.12 Badge Table 19](#_Toc482689348)

[10.4.13 LatentSkillList Table 19](#_Toc482689349)

[10.4.14 MonsterInstance Table 20](#_Toc482689350)

[10.4.15 Team Table 21](#_Toc482689351)

[10.5 Dependency Viewpoint 22](#_Toc482689352)

[10.5.1 Table Creation Order 22](#_Toc482689353)

[10.5.2 Views 23](#_Toc482689354)

[10.6 Information Viewpoint 26](#_Toc482689355)

[10.6.1 Information gathering 26](#_Toc482689356)

[10.7 Patterns Viewpoint 26](#_Toc482689357)

[10.8 Interface Viewpoint 26](#_Toc482689358)

[10.9 Structure Viewpoint 27](#_Toc482689359)

[10.10 Interactions Viewpoint 27](#_Toc482689360)

[10.11 State Dynamics Viewpoint 28](#_Toc482689361)

[10.12 Algorithm Viewpoint 28](#_Toc482689362)

[10.13 Resources 28](#_Toc482689363)

## Figure Index

[Figure 1: Use Case Diagram 9](#_Toc482689364)

[Figure 2: Use Case Diagram 9](https://d.docs.live.net/7a0973039df06ae5/Documents/School%20documents/SWTS1104/Assignments/PADDatabaseSDD.docx#_Toc482689365)

[Figure 3: MonsterClass Table UML 11](#_Toc482689366)

[Figure 4: Attribute Table UML 13](#_Toc482689367)

[Figure 5: MonsterType Table UML 13](#_Toc482689368)

[Figure 6: Growth Table UML 14](#_Toc482689369)

[Figure 7: ActiveSkill Table UML 14](#_Toc482689370)

[Figure 8: LeaderSkill Table UML 15](#_Toc482689371)

[Figure 9: AwokenSkill Table UML 15](#_Toc482689372)

[Figure 10: AwokenSkillList Table UML 16](#_Toc482689373)

[Figure 11: EvolutionTree Table UML 17](#_Toc482689374)

[Figure 12: LatentSkill Table UML 18](#_Toc482689375)

[Figure 13: Player Table UML 18](#_Toc482689376)

[Figure 14: Badge Table UML 19](#_Toc482689377)

[Figure 15: LatentSkillList Table UML 19](#_Toc482689378)

[Figure 16: MonsterInstance Table UML 20](#_Toc482689379)

[Figure 17: Team Table UML 21](#_Toc482689380)

[Figure 18: DBShema 27](#_Toc482689381)

[Figure 19: Interaction viewpoint 27](#_Toc482689382)

## Table Index

[Table 1: MonsterClass Breakdown 12](#_Toc482689383)

[Table 2: Attribute Breakdown 13](#_Toc482689384)

[Table 3: MonsterType Breakdown 13](#_Toc482689385)

[Table 4: Growth Breakdown 14](#_Toc482689386)

[Table 5: ActiveSkill Breakdown 14](#_Toc482689387)

[Table 6: LeaderSkill Breakdown 15](#_Toc482689388)

[Table 7: AwokenSkill Breakdown 15](#_Toc482689389)

[Table 8: AwokenSkill Breakdown 16](#_Toc482689390)

[Table 9: EvolutionTree Breakdown 17](#_Toc482689391)

[Table 10: LatentSkill Breakdown 18](#_Toc482689392)

[Table 11: Player Breakdown 18](#_Toc482689393)

[Table 12: Badge Breakdown 19](#_Toc482689394)

[Table 13: LatentSkillList Breakdown 19](#_Toc482689395)

[Table 14: MonsterInstance Breakdown 20](#_Toc482689396)

[Table 15: Team Breakdown 21](#_Toc482689397)

# Purpose

To Delineate the design of the Puzzle and Dragons Database in a detailed manner.

# Scope

This Document covers the Datatype requirements, relations between information and technologies needed to complete the Puzzle and Dragons Database.

Users will have the ability to record ‘puzzles and dragons’ information within the database, relevant to monsters and teams, with the user as the designated owner.

# Context

This database will not have direct access by the user. Access will be through an application designed to utilize said Database.

# Summary

In summary, the puzzles and dragons Database will be made of 15 tables and 3 – 5 Calculated views.

# References

* <http://puzzledragonx.com> – Third Party Monster Information
* [https://www.microsoft.com/en-us/sql-server/sql-server-2016 - Sql Server 2016](https://www.microsoft.com/en-us/sql-server/sql-server-2016%20-%20Sql%20Server%202016) – Microsoft SQL server
* <http://www.gunghoonline.com/games/puzzle-dragons/> - Puzzles and Dragons Website

# Glossary

* SQL – Structured Query Language
* MS SQL – Microsoft Structured Query Language
* T – SQL – Transact – Structured Query Language
* Microsoft SQL Server – T – SQL Database server
* MySQL – Open-source SQL Service
* Database – a structured set of data held in a computer, especially one that is accessible in many ways.
* DB Schema – Database Relations and table designated

# Identified Stakeholders and Design Concerns

|  |  |
| --- | --- |
| Stakeholders | Design Concerns |
| Database Developers | Which Database service to use.  Tables, Fields, Keys.  Table Relationships |
| Database Administrators | Database Memory Requirements  Updating Base Database information  User Database backup and restore Process |
| Application designers | Database Access  Query language  User Authentications  Formulas Needed |

# Design Viewpoints

## Introduction

In this part design viewpoints will be described in the following order:

* Context viewpoint
* Composition viewpoint
* Logical Viewpoint
* Dependency Viewpoint
* Information Viewpoint
* Pattern Viewpoint
* Interface Viewpoint
* Structure Viewpoint
* Interaction Viewpoint
* State Dynamics Viewpoint
* Algorithm Viewpoint
* Resources

## Context Viewpoint

The Puzzle and Dragons Database Will Allow Users to Store and track Monster collections and teams. There are two main functionalities which are Creating of a User/Player and Creating instances of Monsters owned by said User/Player, Sub functionalities Include Creation of teams with said Monsters and storing history of Monsters.

### User Use Case Diagram

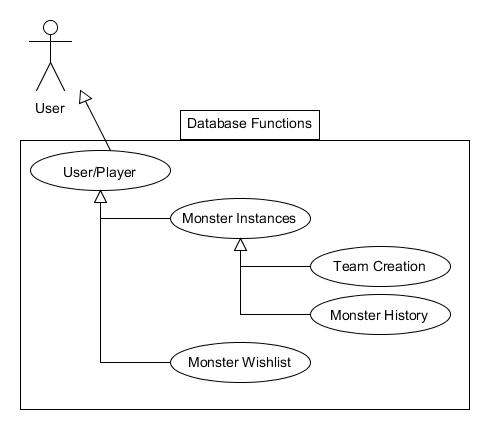


Figure : Use Case Diagram

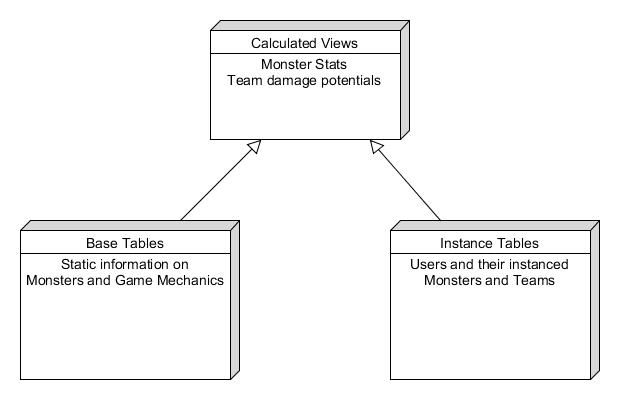
Figure : Use Case Diagram

## Composition Viewpoint

Our Database will be Composed of Various base tables and Instance tables. Information displayed will be a view with calculated fields from information provided in both Tables. Updates to base tables should not affect Instances

### Deployment Diagram

Basic Diagram of our Database:

  
Figure 2: Deployment Diagram

## Logical Viewpoint

In this part, we shall go into further detail of The Database Entities explaining Tables, Their fields and the relationship between these Tables

There are a Total Of 11 base tables and 4 instance tables

Base Tables – MonsterClass, Attribute, MonsterType, ActiveSkill, LeaderSkill, Growth, AwokenSkill, AwokenSkillList, EvolutionTree, LatentSkill, Badge.

Instance Tables – Player, MonsterInstance, Team, LatentSkillList

1. In this Section, these Tables will be Explained Separately.

### MonsterClass Table

MonsterClass Will Hold All Base Monster Information used to calculate stats of Instanced Monsters.

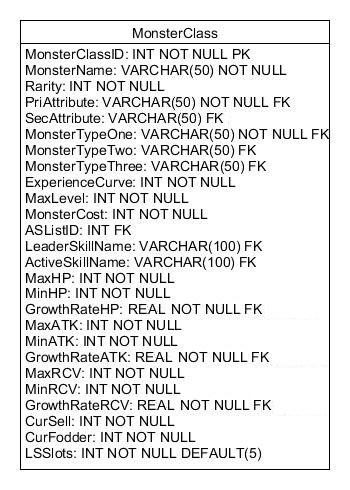


Figure : MonsterClass Table UML

Table : MonsterClass Breakdown

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Description | Foreign Key Relation |
| MonsterClassID | INT NOT NULL PK | Identification number of a monster |  |
| MonsterName | VARCHAR(50) NOT NULL | String Identification of a monster |  |
| Rarity | INT NOT NULL | A Classification of worth of a monster |  |
| PriAttribute | VARCHAR(50) NOT NULL FK | Primary Attribute of a monster | Attribute.AtrributeName |
| SecAttribute | VARCHAR(50) FK | Secondary Attribute of a monster | Attribute.AtrributeName |
| MonsterTypeOne | VARCHAR(50) NOT NULL FK | Primary Type of the Monster | MonsterType.MonsterTypeName |
| MonsterTypeTwo | VARCHAR(50) FK | Secondary Type of the Monster | MonsterType.MonsterTypeName |
| MonsterTypeThree | VARCHAR(50) FK | Trinary Type of the Monster | MonsterType.MonsterTypeName |
| ExperienceCurve | INT NOT NULL | Determines experience needed to level and Max Experience. |  |
| MaxLevel | INT NOT NULL | Maximum level a monster can hit |  |
| MonsterCost | INT NOT NULL | Team Point Cost to assign to a team |  |
| ASListID | INT FK | Awoken Skill List ID | AwokenSkillList.ASListID |
| LeaderSkillName | VARCHAR(100) FK | Name of the Monsters Leader Skill | LeaderSkill.LeaderSkillName |
| ActiveSkillName | VARCHAR(100) FK | Name of the Monsters Active Skill | ActiveSkill.ActiveSkillName |
| MaxHP | INT NOT NULL | Maximum base HP for the Monster |  |
| MinHP | INT NOT NULL | Minimum base HP for the Monster |  |
| GrowthRateHP | REAL NOT NULL FK | Rate at which HP increases when leveling | Growth.GrowthRate |
| MaxATK | INT NOT NULL | Maximum base ATK for the Monster |  |
| MinATK | INT NOT NULL | Minimum base ATK for the Monster |  |
| GrowthRateATK | REAL NOT NULL FK | Rate at which ATK increases when leveling | Growth.GrowthRate |
| MaxRCV | INT NOT NULL | Maximum base RCV for the Monster |  |
| MinRCV | INT NOT NULL | Minimum base HP for the Monster |  |
| GrowthRateRCV | REAL NOT NULL FK | Rate at which RCV increases when leveling | Growth.GrowthRate |
| CurSell | INT NOT NULL | Used to Calculate coin value at specific level |  |
| CurFodder | INT NOT NULL | Used to Calculate Experience value at specific level |  |
| LSSlots | INT NOT NULL DEFAULT(5) | Number of latent skill slots a monster has. |  |

### Attribute Table

The Attribute Table hold the attributes in Puzzles and Dragons.

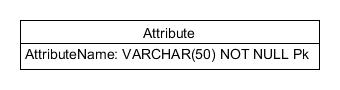


Figure : Attribute Table UML

Table : Attribute Breakdown

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| AttributeName | VARCHAR(50) NOT NULL PK | The Name of the Attribute |

### MonsterType Table

The Monster Type Table holds the Classification of monster in Puzzles and Dragons

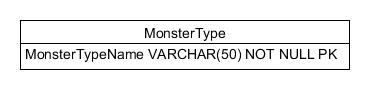


Figure : MonsterType Table UML

Table : MonsterType Breakdown

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| MonsterTypeName | VARCHAR(50) NOT NULL PK | The Name of the Monster Type |

### Growth Table

The Growth Table Holds the Growth Rates of stats in Puzzles and Dragons.

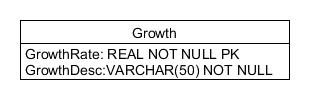


Figure : Growth Table UML

Table : Growth Breakdown

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| GrowthRate | REAL NOT NULL PK | Rate of growth for monster stats when leveling |
| GrowthDesc | VARCHAR(50) NOT NULL | Description of Growth Rate |

### ActiveSkill Table

That ActiveSkill Table holds all Active Skills within Puzzles and Dragons.

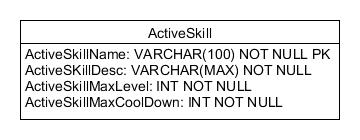


Figure : ActiveSkill Table UML

Table : ActiveSkill Breakdown

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| ActiveSkillName | VARCHAR(100) NOT NULL PK | Name of the Active Skill |
| ActiveSkillDesc | VARCHAR(MAX) NOT NULL | Description of the Active Skill |
| ActiveSkillMaxLevel | INT NOT NULL | Maximum level for the active skill |
| ActiveSkillMaxCooldown | INT NOT NULL | Starting Cooldown of Active Skill |

### LeaderSkill Table

The Leader Skill Table Holds all the Leader Skill Information from Puzzles and Dragons.

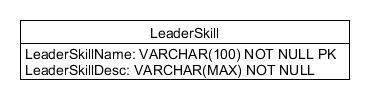


Figure : LeaderSkill Table UML

Table : LeaderSkill Breakdown

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| LeaderSkillName | VARCHAR(100) NOT NULL | Name of the Leader Skill |
| LeaderSkillDesc | VARCHAR(MAX) NOT NULL | Description of the leader skill |

### AwokenSkill Table

The Awoken Skill Table Holds all the Awoken Skills within Puzzles and Dragons.

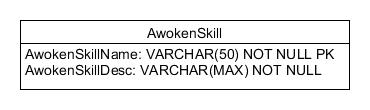


Figure : AwokenSkill Table UML

Table : AwokenSkill Breakdown

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| AwokenSkillName | VARCHAR(50) NOT NULL PK | Name of the Awoken Skill |
| AwokenSkillDesc | VARCHAR(MAX) NOT NULL | Description of the Awoken skill |

### AwokenSkillList Table

The AwokenSkillList Table holds the list of Awoken Skills that monsters have in puzzles and dragons.

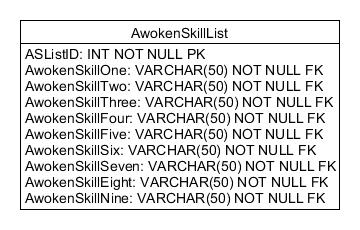


Figure : AwokenSkillList Table UML

Table : AwokenSkill Breakdown

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Description | Foreign Key Relation |
| ASListID | INT NOT NULL PK | ID number for Awoken skill list, Corresponds with MonsterClassID |  |
| AwokenSkillOne | VARCHAR(50) NOT NULL FK | Name of first Awoken skill | AwokenSkill.AwokenSkillName |
| AwokenSkillTwo | VARCHAR(50) NOT NULL FK | Name of second Awoken skill | AwokenSkill.AwokenSkillName |
| AwokenSkillThree | VARCHAR(50) NOT NULL FK | Name of third Awoken skill | AwokenSkill.AwokenSkillName |
| AwokenSkillFour | VARCHAR(50) NOT NULL FK | Name of forth Awoken skill | AwokenSkill.AwokenSkillName |
| AwokenSkillFive | VARCHAR(50) NOT NULL FK | Name of fifth Awoken skill | AwokenSkill.AwokenSkillName |
| AwokenSkillSix | VARCHAR(50) NOT NULL FK | Name of sixth Awoken skill | AwokenSkill.AwokenSkillName |
| AwokenSkillSeven | VARCHAR(50) NOT NULL FK | Name of seventh Awoken skill | AwokenSkill.AwokenSkillName |
| AwokenSkillEight | VARCHAR(50) NOT NULL FK | Name of eighth Awoken skill | AwokenSkill.AwokenSkillName |
| AwokenSkillNine | VARCHAR(50) NOT NULL FK | Name of ninth Awoken skill | AwokenSkill.AwokenSkillName |

### EvolutionTree Table

The EvolutionTree Table holds the information for the order of evolution for monsters, whether it is de-evolvable, and if it resets level.

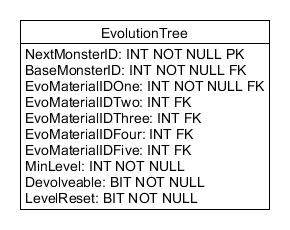


Figure : EvolutionTree Table UML

Table : EvolutionTree Breakdown

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Description | Foreign Key Relation |
| NextMonsterID | INT NOT NULL PK | The MonsterClassID of the monster it will evolve into. |  |
| BaseMonsterID | INT NOT NULL FK | The MonsterClassID of the Monster you need in order to evolve. | MonsterClass.MonsterClassID |
| EvoMaterialIDOne | INT NOT NULL FK | The MonsterClassID of the First Evolution Material Needed. | MonsterClass.MonsterClassID |
| EvoMaterialIDTwo | INT FK | The MonsterClassID of the Second Evolution Material Needed. | MonsterClass.MonsterClassID |
| EvoMaterialIDThree | INT FK | The MonsterClassID of the Third Evolution Material Needed. | MonsterClass.MonsterClassID |
| EvoMaterialIDFour | INT FK | The MonsterClassID of the Four Evolution Material Needed. | MonsterClass.MonsterClassID |
| EvoMaterialIDFive | INT FK | The MonsterClassID of the Five Evolution Material Needed. | MonsterClass.MonsterClassID |
| MinLevel | INT NOT NULL | Minimum Level of the Base Monster to Evolve. |  |
| Devolveable | BIT NOT NULL | Whether the monster can Return to a previous Evolution |  |
| LevelReset | BIT NOT NULL | Whether the Evolution process resets the level of your monster to 1 |  |

### LatentSkill Table

The Latent Skill Table holds the information of the available identified by name.

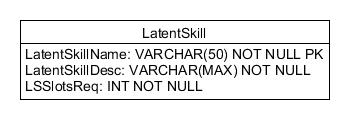


Figure : LatentSkill Table UML

Table : LatentSkill Breakdown

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| LatentSkillName | VARCHAR(50) NOT NULL PK | Name of the Latent Skill |
| LatentSkillDesc | VARCHAR(MAX) NOT NULL | Description of the Latent Skill |
| LSSlotsReq | INT NOT NULL | Number of Latent Skill Slots Required to use the Latent Skill |

### Player Table

The Player Table Holds Information Specific to the User.

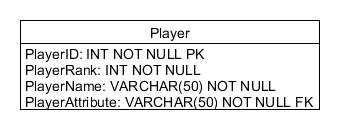


Figure : Player Table UML

Table : Player Breakdown

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Description | Foreign Key Relation |
| PlayerID | INT NOT NULL PK | Identification number of the player in Puzzles and Dragons. |  |
| PlayerRank | INT NOT NULL | Players Current Rank within Puzzles and Dragons. |  |
| PlayerName | VARCHAR(50) NOT NULL | Players In game name. |  |
| PlayerAttribute | VARCHAR(50) NOT NULL FK | Players Starting Attribute. | Attribute.AttributeName |

### Badge Table

The Badge Table holds the information of the team badges in puzzles and dragons.

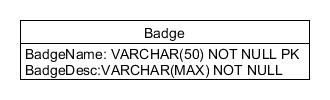


Figure : Badge Table UML

Table : Badge Breakdown

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| BadgeName | VARCHAR(50) NOT NULL PK | Name of the team badge |
| BadgeDesc | VARCHAR(MAX) NOT NULL | Desicription of what affect the team badge has. |

### LatentSkillList Table

The LatentSkillList Table Holds the Lists of Latent skills that a Monster instance has applied to it.

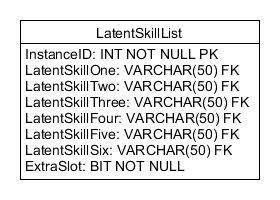


Figure : LatentSkillList Table UML

Table : LatentSkillList Breakdown

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Description | Foreign Key Relation |
| InstanceID | INT NOT NULL PK | Identification number that corresponds with Monster Instance ID. |  |
| LatentSkillOne | VARCHAR(50) FK | Name of First Latent Skill | LatentSkill.LatentSkillName |
| LatentSkillTwo | VARCHAR(50) FK | Name of second Latent Skill | LatentSkill.LatentSkillName |
| LatentSkillThree | VARCHAR(50) FK | Name of third Latent Skill | LatentSkill.LatentSkillName |
| LatentSkillFour | VARCHAR(50) FK | Name of forth Latent Skill | LatentSkill.LatentSkillName |
| LatentSkillFive | VARCHAR(50) FK | Name of Fifth Latent Skill | LatentSkill.LatentSkillName |
| LatentSkillSix | VARCHAR(50) FK | Name of sixth Latent Skill | LatentSkill.LatentSkillName |
| ExtraSlot | BIT NOT NULL | Whether or not the six slot is unlocked |  |

### MonsterInstance Table

The MonsterInstance Table holds monster instances from monster created by the user.

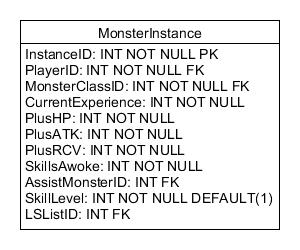


Figure : MonsterInstance Table UML

Table : MonsterInstance Breakdown

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Description | Foreign Key Relation |
| InstanceID | INT NOT NULL PK | Identification of the monster instance |  |
| PlayerID | INT NOT NULL FK | Identification of the player that owns the monster instance. | Player.PlayerID |
| MonsterClassID | INT NOT NULL FK | Identification of Base Monster information from MonsterClass Table | MonsterClass.MonsterClassID |
| CurrentExperience | INT NOT NULL | Amount of experience the monster has obtained. |  |
| PlusHP | INT NOT NULL | Amount of obtained Plus HP Stats. |  |
| PlusATK | INT NOT NULL | Amount of obtained Plus ATK Stat. |  |
| PlusRCV | INT NOT NULL | Amount of obtained Plus RCV Stat. |  |
| SkillsAwoke | INT NOT NULL | Amount of awoken skills that have been awoken |  |
| AssistMonsterID | INT FK | Monster Instance ID of Monster assisting | MonsterInstance.InstanceID |
| SkillLevel | INT NOT NULL | Level of the ActiveSkill of the Monster Instance |  |
| LSListID | INT FK | Identification number of the latent skill list record in LatentSkillList Table. | LatentSkillList.InstanceID |

### Team Table

The Team Table holds the identification numbers of 1 to 5 Monster instances, and the Player Id of the owner of the team.

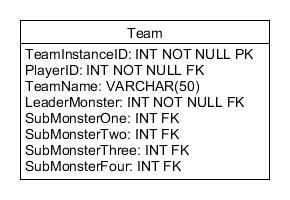


Figure : Team Table UML

Table : Team Breakdown

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Description | Foreign Key Relation |
| TeamInstanceID | INT NOT NULL PK | Identification of the Team instance |  |
| PlayerID | INT NOT NULL FK | Identification of the player that owns the Team instance. | Player.PlayerID |
| TeamName | VARCHAR(50) | String Identifier of the Team |  |
| LeaderMonster | INT NOT NULL FK | MonsterInstanceID of monster to be used as Leader | MonsterInstance.InstanceID |
| SubMonsterOne | INT FK | MonsterInstanceID of Monster to be used as sub | MonsterInstance.InstanceID |
| SubMonsterTwo | INT FK | MonsterInstanceID of Monster to be used as sub | MonsterInstance.InstanceID |
| SubMonsterThree | INT FK | MonsterInstanceID of Monster to be used as sub | MonsterInstance.InstanceID |
| SubMonsterFour | INT FK | MonsterInstanceID of Monster to be used as sub | MonsterInstance.InstanceID |

## Dependency Viewpoint

In this viewpoint, the relationships between tables and the corresponding views with Calculated fields will be explained in detail.

The Pad Database Works with Base Game Information Tables and User Information Tables to create views that display the corresponding statistics for Monster and Teams. The Benefits of this approach include:

* Ease if Update – Base Game Information can be updated and directly affect the information shown to the user
* Ease of Use – User will not have to enter more information then would be needed to calculate the statistics of their instances
* Less Data Stored – Having information calculated means not having to store all information possible about instances.

### Table Creation Order

Table Creation Order from least dependent to most dependent.

|  |  |
| --- | --- |
| Order | Tables |
| 1st | Attribute, MonsterType, Growth, Badge, LeaderSkill, ActiveSkill, AwokenSkill, LatentSkill |
| 2nd | AwokenSkillList, LatentSkillList |
| 3rd | Player, MonsterClass |
| 4th | MonsterInstance, EvolutionTree |
| 5th | Team |

### Views

#### User Collection View

This simple View will return monster instances owned by the PlayerID by Joining **Player**, **MonsterInstance, MonsterClass**

|  |  |  |
| --- | --- | --- |
| MonsterCollection | | |
| Field | Calculations | Condition |
| MonsterInstance.MonsterInstanceID |  | WHERE Player.PlayerID = MonsterInstance.PlayerID |
| MonsterClass.MonsterName |  | WHERE MonsterInstance.MonsterClassID = MonsterClass.MonsterClassID |

#### User Teams View

This View displays the teams owned by a PlayerID by joining **Player** and **Team** Tables by PlayerID.

|  |  |  |
| --- | --- | --- |
| TeamCollection | | |
| Field | Calculations | Condition |
| Team.TeamInstanceID |  | WHERE Player.PlayerID = Table.PlayerID |
| Team.TeamName |  |  |

#### Monster Instance View

This is a more complete view of monster instances, by joining **MonsterInstance**, **MonsterClass**, **AwokenSkillList**, **LatentSkillList**, **LeaderSkill**, and **ActiveSkill** so that a Program will have all the information needed to calculated the fields for stats of the monster’s level.

SQL JOIN SNIPPET:

MonsterInstance INNER JOIN

MonsterClass ON MonsterInstance.MonsterClassID = MonsterClass.MonsterClassID LEFT JOIN

AwokenSkillList ON MonsterClass.ASListID = AwokenSkillList.ASListID LEFT JOIN

AwokenSkill ON AwokenSkillList.AwokenSkillOne = AwokenSkill.AwokenSkillName AND AwokenSkillList.AwokenSkillTwo = AwokenSkill.AwokenSkillName AND

AwokenSkillList.AwokenSkillThree = AwokenSkill.AwokenSkillName AND AwokenSkillList.AwokenSkillFour = AwokenSkill.AwokenSkillName AND

AwokenSkillList.AwokenSkillFive = AwokenSkill.AwokenSkillName AND AwokenSkillList.AwokenSkillSix = AwokenSkill.AwokenSkillName AND

AwokenSkillList.AwokenSkillSeven = AwokenSkill.AwokenSkillName AND AwokenSkillList.AwokenSkillEight = AwokenSkill.AwokenSkillName AND

AwokenSkillList.AwokenSkillNine = AwokenSkill.AwokenSkillName LEFT JOIN

LatentSkillList ON MonsterInstance.LSListID = LatentSkillList.InstanceID LEFT JOIN

LatentSkill ON LatentSkillList.LatentSkillOne = LatentSkill.LatentSkillName AND LatentSkillList.LatentSkillTwo = LatentSkill.LatentSkillName AND

LatentSkillList.LatentSkillThree = LatentSkill.LatentSkillName AND LatentSkillList.LatentSkillFour = LatentSkill.LatentSkillName AND

LatentSkillList.LatentSkillFive = LatentSkill.LatentSkillName AND LatentSkillList.LatentSkillSix = LatentSkill.LatentSkillName LEFT JOIN

ActiveSkill ON MonsterClass.ActiveSkillName = ActiveSkill.ActiveSkillName LEFT JOIN

LeaderSkill ON MonsterClass.LeaderSkillName = LeaderSkill.LeaderSkillName

#### Team View

Displays the calculated stats of a TeamInstance Owned by a PlayerID by Joining Team Table with MonsterView View for Each Monster

Where MonsterInstance.InstanceID = (Select \* From Team Where Team.TeamInstanceID = TeamInstanceID).LeaderMonster or

MonsterInstance.InstanceID = (Select \* From Team Where Team.TeamInstanceID = TeamInstanceID).SubMonsterOne or

MonsterInstance.InstanceID = (Select \* From Team Where Team.TeamInstanceID = TeamInstanceID).SubMonsterTwo or

MonsterInstance.InstanceID = (Select \* From Team Where Team.TeamInstanceID = TeamInstanceID).SubMonsterThree or

MonsterInstance.InstanceID = (Select \* From Team Where Team.TeamInstanceID = TeamInstanceID).SubMonsterFour

Another Method Would the have a class to represent monsters within the application, Then Using a MonsterView Query for each monster in the team to construct and object for that monster.

## Information Viewpoint

### Information gathering

Information needed for the Puzzle and Dragons Database, is parsed from a Third-Party Resource as direct access to Puzzles and Dragons information is limited to viewing instanced monsters, with all pertinent information hidden from sight.

We use <http://puzzledragonx.com> as our third-party resource.

There are a multiple was to gather information:

* Analyzing each monster, Active Skill, Leader Skill page and manually gathering all the information.
* Web scraping the information automatically with a script
* There are also JSON files on the website which hold much of the information.

With there being 3500+ Monsters and more being added as time goes on, the manual method seems impossible, the web scraping or accessing the JSON files directly seems to be the most prudent choice.

## Patterns Viewpoint

N/A?

## Interface Viewpoint

Applications Developed for this database are required for Service access.

Applications would use SQL Queries To retrieve, Create, update and deleting Entries.

## Structure Viewpoint

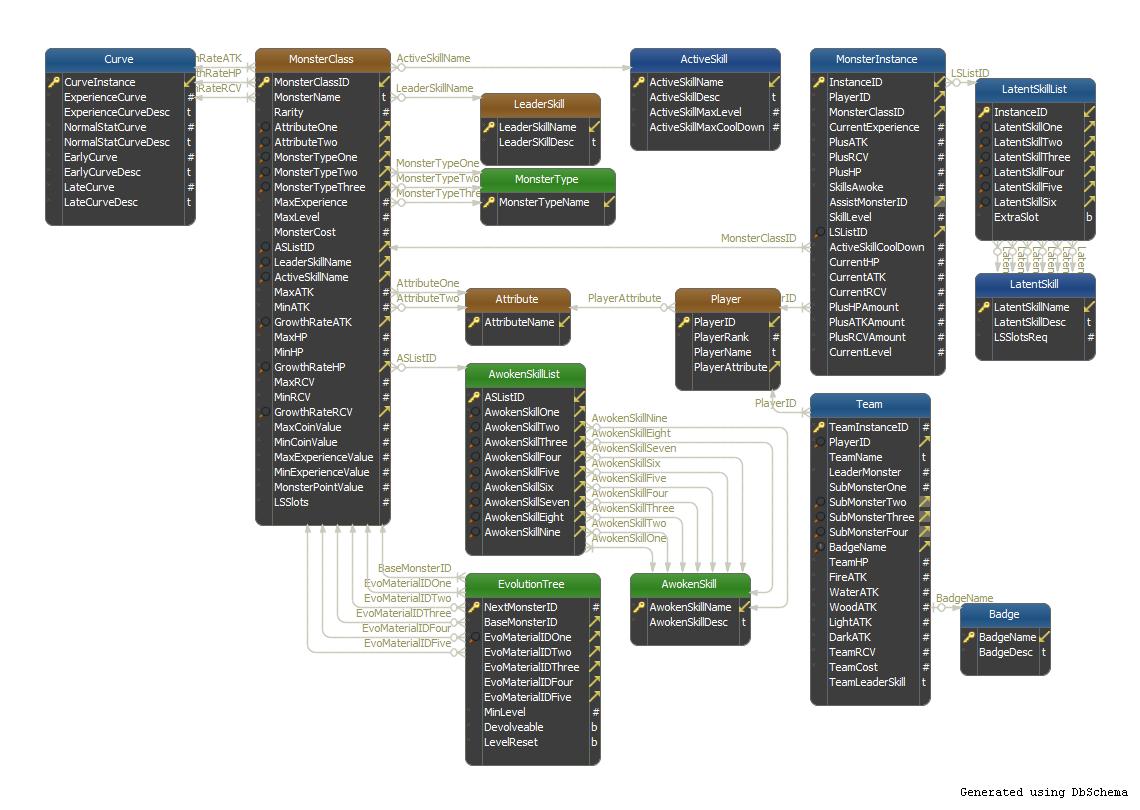


Figure : DBShema

## Interactions Viewpoint

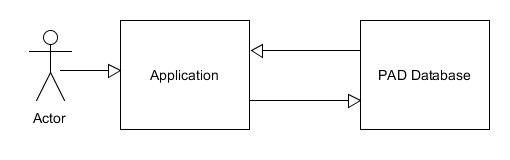


Figure : Interaction viewpoint

## State Dynamics Viewpoint

N/A?

## Algorithm Viewpoint

Algorithms needed for calculated information are best placed within the application, utilizing the information placed in the Database, here are the formulas for the monster instance information:

* Monsters
  + Level = Floor((CurrentExperience/float(ExperienceCurve)) ^ (1/2.5) \* MaxLevel -1 +1
  + StatValue = MinStat + (MaxStat – MinStat) \* ((level - 1) / (MaxLevel – 1)) ^GrowthRate
* Teams
  + TeamHP = All Monster TotalHP
  + TeamRCV = All Monster TotalRC
  + TeamAttATK = All Monster attack with same PriAttribute + 1/10of ATK with same SecAttribute ? PriAttribute == SecAttribute else 1/3 of ATK

## Resources

Base information on the Puzzles and Dragons database, Amounts to around 10-12MBs of Data storage.

Monster Instances take ~88bytes of storage

Team entries take ~ 106Bytes of storage

Player entries take 116Bytes of storage

LatentSkillLists entries take 309Bytes

A User with 1000 Instances each with LatentSkillList and 100 Teams Will Take ~408KB of storage

100,000 Users will take ~40Gb of storage.

A single tb harddrive will support a user base of ~2,450,980,392

Microsoft SQL Server 2016 Express support a maximum database size of 10GB, limiting userbase to ~25,000

MySQL Database limit on an Win32 NTFS system is ~2TB supporting a userbase of ~4.9 Billion