System design document

Subject: Puzzles and dragons database

Version 1.3

Author: Elie Godbout

May 16, 2017

McKenzie College

ourse: SWTS1104

Elie Godbout

eliegodbout@gmail.com

# Version History

|  |  |  |  |
| --- | --- | --- | --- |
| Version: | Editor: | Description: | Date: |
| Version 1.0 | Elie Godbout | Initial document for puzzles and dragons design | 05/10/2017 |
| Version 1.1 | Elie Godbout | Revision of the PAD database SDD: include added and updated sections | 05/11/2017 |
| Version 1.2 | Elie Godbout | 2nd revision of the PAD database SDD. updated sections: Glossary, context, design views and design viewpoints | 05/15/2017 |
| Version 1.3 | Elie Godbout | 3rd revision of the PAD database SDD. Updated structural viewpoint and Glossary. Added list of tables and new figures. | 05/16/2017 |

Table of Contents

[**Version History 2**](#_Toc482622078)

[**1 Introduction 6**](#_Toc482622079)

[1.1 Purpose 6](#_Toc482622080)

[1.2 Scope 6](#_Toc482622081)

[1.3 Context 6](#_Toc482622082)

[1.4 Summary 6](#_Toc482622083)

[**2 References 7**](#_Toc482622084)

[**3 Glossary 7**](#_Toc482622085)

[**4 Body 8**](#_Toc482622086)

[4.1 Identified stakeholders and design concerns 8](#_Toc482622087)

[4.1.1 Stakeholders 8](#_Toc482622088)

[4.1.2 Design concerns 8](#_Toc482622089)

[4.2 Design view 8](#_Toc482622090)

[4.3 Design viewpoints 9](#_Toc482622091)

[4.3.1 User monster instance input 9](#_Toc482622092)

[4.3.2 User monster deletion 9](#_Toc482622093)

[4.3.3 User monster instance update 9](#_Toc482622094)

[4.3.4 Team create 9](#_Toc482622095)

[4.3.5 Team delete 9](#_Toc482622096)

[4.3.6 Player sharing 9](#_Toc482622097)

[4.3.7 Create, update and delete monster class table 9](#_Toc482622098)

[4.3.8 Create, update and delete Evolution tree table 9](#_Toc482622099)

[4.3.9 Create, update and delete badge table 9](#_Toc482622100)

[4.3.10 Create full database back-up 9](#_Toc482622101)

[4.3.11 Restore database 9](#_Toc482622102)

[4.4 Context Design 10](#_Toc482622103)

[4.4.1 User perspective 10](#_Toc482622104)

[4.4.2 Admin: Create MonsterClass table 10](#_Toc482622105)

[4.4.3 Admin: Update MonsterClass table 10](#_Toc482622106)

[4.4.4 Admin: Delete MonsterClass table 10](#_Toc482622107)

[4.4.5 Admin: Create EvolutionTree table 10](#_Toc482622108)

[4.4.6 Admin: Update EvolutionTree table 10](#_Toc482622109)

[4.4.8 Admin: Create Badge table 10](#_Toc482622110)

[4.4.9 Admin: Update Badge table 10](#_Toc482622111)

[4.4.10 Admin: Delete Badge table 10](#_Toc482622112)

[4.4.11 Admin: Create full database back-up and update back-up 10](#_Toc482622113)

[4.4.12 Admin: Restore database 10](#_Toc482622114)

[4.5 Composition Design 11](#_Toc482622115)

[4.6 Design overlays 12](#_Toc482622116)

[4.7 Structural viewpoint 14](#_Toc482622117)

[4.8 Logical design 22](#_Toc482622118)

[4.9 Dependency viewpoint 23](#_Toc482622119)

[4.10 Design language 24](#_Toc482622120)

[**Appendix A: To be determined 24**](#_Toc482622121)

[**Appendix B: Future enhancements 24**](#_Toc482622122)

List of Figures

[Figure 1: PAD database function explanation 6](#_Toc482623380)

[Figure 2: Schema of the PAD database visual representation 12](#_Toc482623381)

[Figure 3: PAD monster screenshot 12](file:///C:\Users\elieg\Documents\WordDocs\May%202017\SDD%20subject%20PAD%20database%20v.1.3.docx#_Toc482623382)

[Figure 4: PAD team screenshot 13](file:///C:\Users\elieg\Documents\WordDocs\May%202017\SDD%20subject%20PAD%20database%20v.1.3.docx#_Toc482623383)

[Figure 5: PAD Badge select screenshot 13](file:///C:\Users\elieg\Documents\WordDocs\May%202017\SDD%20subject%20PAD%20database%20v.1.3.docx#_Toc482623384)

[Figure 6: PAD Player info screenshot 13](#_Toc482623385)

[Figure 7: PAD Player ID screenshot 13](#_Toc482623386)

List of Tables

[Table 1:MonsterAttribute 14](#_Toc482623387)

[Table 2:MonsterType 14](#_Toc482623388)

[Table 3:ActiveSkill 14](#_Toc482623389)

[Table 4:LeaderSkill 14](#_Toc482623390)

[Table 5:AwokenSkill 15](#_Toc482623391)

[Table 6:AwokenSkillList 15](#_Toc482623392)

[Table 7:LatentSkill 16](#_Toc482623393)

[Table 8:LatentSkillList 16](#_Toc482623394)

[Table 9:MonsterClass 18](#_Toc482623395)

[Table 10:EvolutionTree 19](#_Toc482623396)

[Table 11:Player 19](#_Toc482623397)

[Table 12:MonsterInstance 20](#_Toc482623398)

[Table 13:Team 21](#_Toc482623399)

[Table 14:Badge 22](#_Toc482623400)

[Table 15:Curve 22](#_Toc482623401)

# 1 Introduction

## 1.1 Purpose

The purpose of this document is too give detail on the design aspect of the PAD database and how we hope to achieve the designs by the end of the product.

## 1.2 Scope

The PAD database will be an interactive tool for players of the Puzzles and dragons mobile video-game where they will be able to simulate actions performed in the game in order to enhanced the player experience.

## 1.3 Context

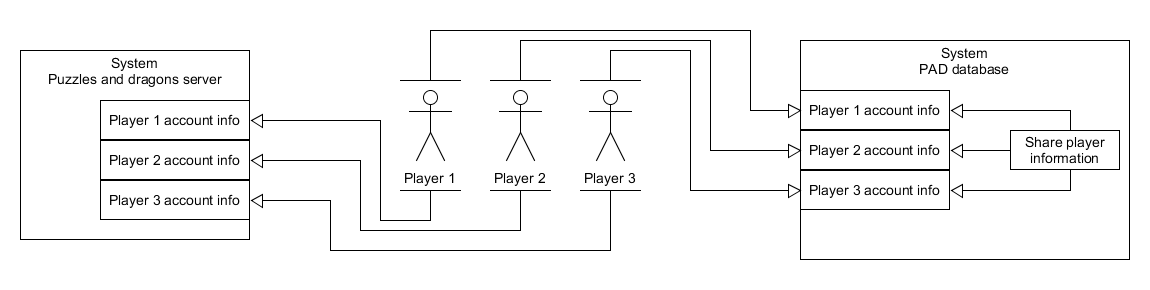
The PAD database is aimed at users of the mobile video-game Puzzles and dragons, a world-wide service free to anyone whom wishes to play, test, plan and strategize team builds to then possibly use the experience from the product within the game. The PAD database takes the information of the monsters belonging to players from puzzles and dragons and results in presenting that player with a team like the one found in puzzles and dragons. This product will benefit the players by introducing seemingly endless possibilities of team combinations as well as presenting teams of other puzzles and dragons players from around the world.

Figure 1: PAD database function explanation

In figure 1, we see how the PAD database is planned to be used. While each player’s monster information is handled by the puzzles and dragons server, the PAD database aims to not only have each player able to input their monster information, but also to share the information with other players.

## 1.4 Summary

NOTE: The PAD database is still currently in development.

The PAD database will be a free product used to simulate team building functionality found in puzzles and dragons in order to give players a way to experiment and test multiple different builds without compromising their in-game teams as well as taking a look at what other players are ding with their own teams.

# 2 References

* SDD IEEE 1016–2009.pdf
  + Software design document template
* CMS system design document
  + SDD template
* <http://www.gunghoonline.com/games/puzzle-dragons/>
  + Puzzles and dragons’ official website
* <https://play.google.com/store/apps/details?id=jp.gungho.padEN&hl=en>
  + Puzzles and dragons google play store page
* Puzzles and dragons video game (On android device: Samsung galaxy S5)

# 3 Glossary

**Puzzles and dragons:** *title of the mobile video game*

**PAD:** *acronym for Puzzles and dragons*

**Monsters:** *Characters in Puzzles and dragons*

**Monster instance:** *Specific monster belonging to a player*

**Monster box:** *area in puzzles and dragons where a player’s monsters are located*

**Team**: *Group of monsters used in puzzles and dragons’ combat/***:** *referring to the name of a table found in PAD database*

**Leader:** *monster set in a leading position within the team*

**Sub-Monsters**: *Other monsters within the team*

**Attribute:** *defining property of a monster*

**Monster type**: *defining property of a monster*

**ATK:** *(Attack) status of a monster defining its combat potential*

**HP*:*** *(Health Points) status of a monster defining its strength*

**RCV*:*** *(Recovery) status of a monster defining its potential to recover lost HP*

**Active Skills:** *Special power used in gameplay by a monster*

**Leader Skills:** *Special passive power only available when a monster is placed in the team’s leader position*

**Awoken skills**: *special bonuses used to enhance a monster permanently*

**Latent Skills**: *removable special bonuses used to enhance a monster*

**Evolution:** *Process in which a monster becomes a new/different monster*

**Badge:** *Special bonus added to the team/referring to the name of a table found in PAD database*

**Growth rate/growth curve:** *numeric value used in an equation to determine a monster’s stats and level*

**CRUD:** *acronym formed of; Create, update and delete used in database design*

**PK:** *acronym; Primary Key. Used in databases to uniquely identify tables*

**MonsterClass:** *referring to the name of a table found in PAD database*

**MonsterInstance:** *referring to the name of a table found in PAD database*

**Player:** *referring to players of puzzles and dragons/referring to the name of a table found in PAD database*

**MonsterAttributes:** *referring to the name of a table found in PAD database*

**MonsterType:** *referring to the name of a table found in PAD database*

**ActiveSkill:** *referring to the name of a table found in PAD database*

**LeaderSkill:** *referring to the name of a table found in PAD database*

**AwokenSkill:** *referring to the name of a table found in PAD database*

**LatenSkill:** *referring to the name of a table found in PAD database*

**EvolutionTree:** *referring to the name of a table found in PAD database*

**Player:** *also, referred to as “User” these are the target audience of the product*

**Admin:** *abbreviation of administrator. Personnel in charge of the PAD database*

**Management:** *Personnel in charge of upkeeping the Pad database after launch*

# 4 Body

## 4.1 Identified stakeholders and design concerns

### 4.1.1 Stakeholders

* Puzzles and dragons players
* McKenzie College personnel Class SWTS1104 (developers)

### 4.1.2 Design concerns

* *Completion of the PAD database*
  + - AS of this SDD version 1.0, the PAD database is currently unfinished. Completion of the PAD database is undetermined. There a re currently no plans for a release of the PAD database. There exist no documents detailing the use of the PAD database.
* *User interface design*
  + - AS of this SDD version 1.0, the PAD database is currently unfinished. Completion of the PAD database is undetermined. There a re currently no plans for a release of the PAD database. There exist no documents detailing the use of the PAD database.
* *Ease of use*
  + - The PAD database currently does not have any plans for the user interface. Users of the PAD database will require ways to interact with the information so they can input their monsters and receive results. String content will be required as well as images.

## 4.2 Design view

* *Users perspective*
  + Monster instance input
  + Monster instance deletion
  + Team create
  + Team delete
  + Player sharing
* *Admin and/or management perspective*
  + Create, update and delete monster class table
  + Create, update and delete Evolution tree table
  + Create, update and delete badge table
  + Create full database back-up
  + Restore database

## 4.3 Design viewpoints

**User perspective**

### 4.3.1 User monster instance input

Players will add instances of their monsters into the PAD database in the MonsterInstance table. Information stored within the MonsterClass table will identify the monster’s properties and the calculated fields of the MonsterInstance table will match the monster’s in-game stats.

### 4.3.2 User monster deletion

Players will delete monsters they have inputted into the PAD database in the MonsterInstance table. Areas affected will be the MonsterInstance table and team table.

### 4.3.3 User monster instance update

Players will update the information on an existing monster they have inputted. This will affect the MonsterInstance table and Team table. (This may also affect the LatentSkill table)

### 4.3.4 Team create

Players will create a team comprised of their inputted monsters. This encompasses the MonsterInstance table and Team table.

### 4.3.5 Team delete

Players will delete teams they have assembled. This will affect the Team table.

### 4.3.6 Player sharing

Players will share their team information they have created and monster instance information they have inputted to a server for other players to view.

**Admin/management perspective**

### 4.3.7 Create, update and delete monster class table

Administrators will create, update or delete the MonsterClass table when necessary. The MonsterClass table requires the following tables: MonsterAttribute, MonsterType, ActiveSkill, LeaderSkill, AwokenSkillList and AwokenSkill.

### 4.3.8 Create, update and delete Evolution tree table

Administrators will create, update or delete the EvolutionTree table when necessary. The EvolutionTree table requires the MonsterClass table.

### 4.3.9 Create, update and delete badge table

Administrators will create, update or delete the Badge table when necessary.

### 4.3.10 Create full database back-up

Administrators will create a back-up of the entire PAD database.

### 4.3.11 Restore database

Administrators will restore the PAD database with the back-up they will have created if the situation ever arises.

## 4.4 Context Design

### 4.4.1 User perspective

As of version 1.2 of this document, there is no user interface or plans for user interfaces and so is impossible to determine the route a user would take to use the product.

### 4.4.2 Admin: Create MonsterClass table

Administrators are required to Create the MonsterClass table as well as populate the table so the PAD database may function. The MonsterClass table is comprised of these other tables; MonsterAttribute, MonsterType, ActiveSkill, LeaderSkill, AwokenSkillList and AwokenSkill. Meaning these tables will need to be created before MonsterClass.

### 4.4.3 Admin: Update MonsterClass table

Administrators are required to update the contents of the MonsterClass table whenever necessary. This may also include all related tables as well.

### 4.4.4 Admin: Delete MonsterClass table

Administrators may need to delete the MonsterClass table as a result of unexpected or problematic behavior or other unknown reasons.

### 4.4.5 Admin: Create EvolutionTree table

Administrators are required to create the EvolutionTree table as well as populate the EvolutionTree table. The EvolutionTree table is used by players to view which monster can change into what monster.

### 4.4.6 Admin: Update EvolutionTree table

Administrators are required to update the EvolutionTree table whenever necessary.

### 4.4.8 Admin: Create Badge table

Administrators are required to create to create the Badge table as well as populate the Badge table. The badge table is required for the PAD database to function properly.

### 4.4.9 Admin: Update Badge table

Administrators are required to update the Badge table whenever necessary.

### 4.4.10 Admin: Delete Badge table

Administrators may need to delete the Badge table as a result of unexpected or problematic behavior or other unknown reasons.

### 4.4.11 Admin: Create full database back-up and update back-up

Administrators are required to create a back-up of all records in the PAD database in order to preserve all information in case of an emergency. If the PAD database should ever be updated, administrators are required to update the back-up as well.

### 4.4.12 Admin: Restore database

Administrators may need to restore the PAD database in the aftermath of an emergency using the back-up creating beforehand.

## 4.5 Composition Design

The PAD database is comprised of 15 tables and was built using Microsoft SQL server.

* *MonsterClass*
  + This table holds all the records of each monster in puzzles and dragons and is used to identify which monster is being used when a player enters their own into the database.
* *Attribute*
  + This table holds the all the attributes associate to a monster in puzzles and dragons
* *MonsterType*
  + This table holds all Types associated to the monsters in puzzles and dragons
* *Curve*
  + This table holds the numeric information concerning the growth rate for the monsters’ stats
* *ActiveSkill*
  + This table holds all the records of each active skill in puzzles and dragons
* *LeaderSkill*
  + This table holds all the records of each leader skill in puzzles and dragons
* *AwokenSkillList*
  + This table holds all the records of each awoken skill associate to each monster in the game
* *AwokenSkill*
  + This table holds all the records of each awoken skill in puzzles and dragons
* *LatentSkillList*
  + This table contains the possible slots for a monster to have any given latent awoken skill
* *LatentSkill*
  + This table holds all the records of each latent awoken skill in puzzles and dragons
* *EvolutionTree*
  + This table contains all the possible evolutions from one monster to another along with the required monsters used to perform a specific evolution
* *Player*
  + This table holds the given information of a player being; Player ID, Player Name, Player Rank and Player Attribute
* *MonsterInstance*
  + This table holds the specific monsters owned by players
* *Team*
  + This table contains the calculated fields for simulating the team builds found in puzzles and dragons
* *Badge*
  + This table holds all the records of each badge in puzzles and dragons

## 4.6 Design overlays

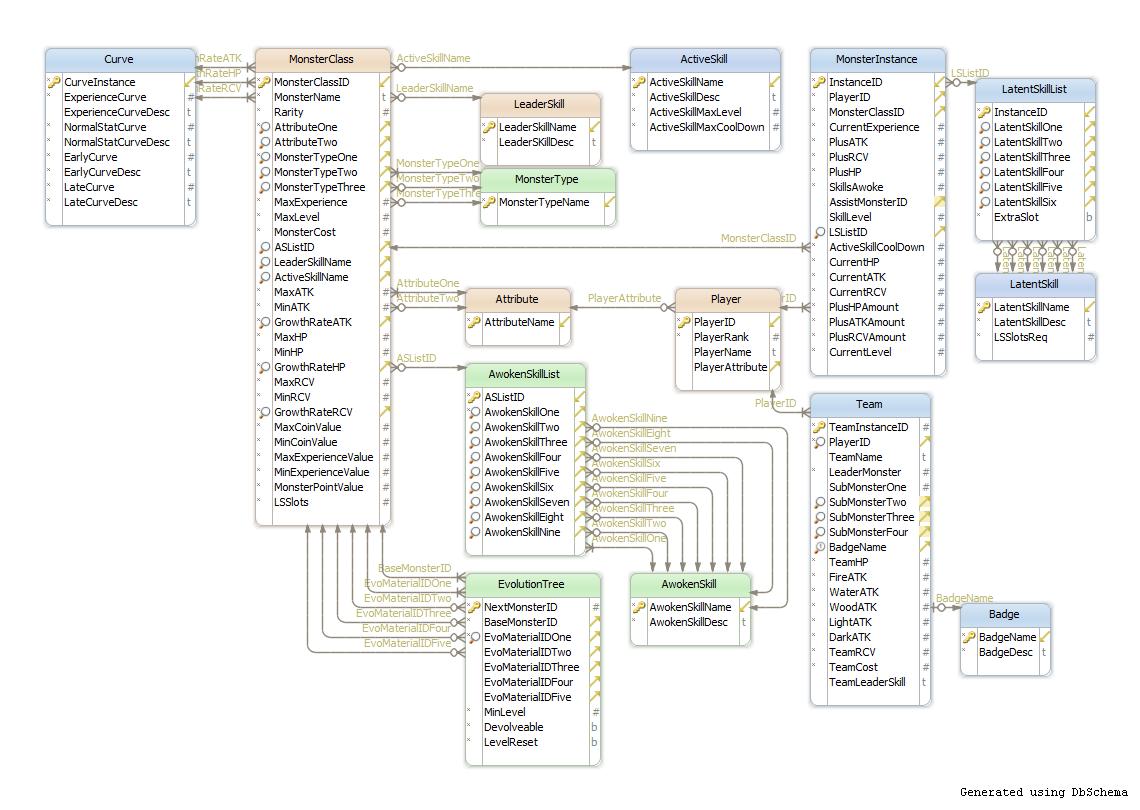
This is a schema representation of the PAD database. Here we can observe the components of all tables in the PAD database and how each table relates to one another.

Figure 2: Schema of the PAD database visual representation 

Figure 3 is a screenshot taken from Puzzles and dragons. Figure 3 represents the player with all information about the selected monster. These are all elements needed in order to simulate them in the PAD database.

Figure 3: PAD monster screenshot

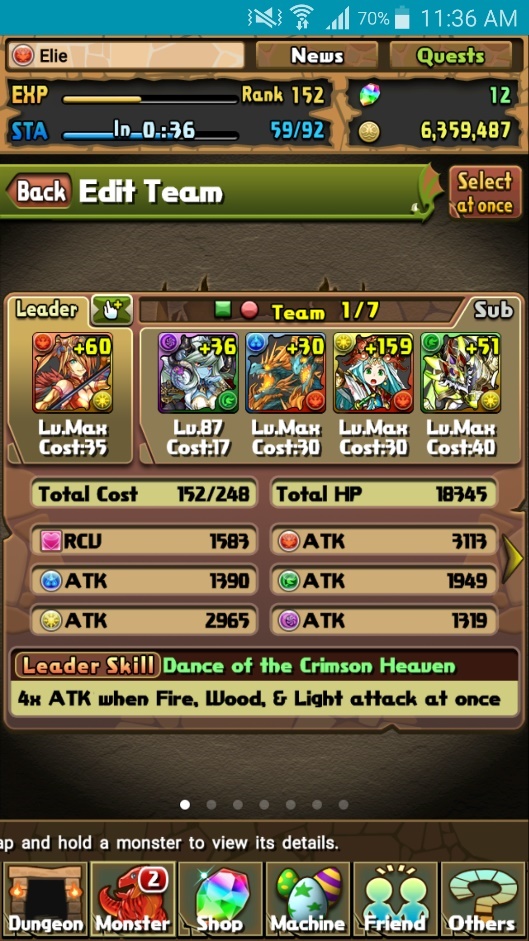
These are screenshots taken from Puzzles and dragons. In Figure 4 (on the left), we have what the result of a complete team built by players looks like. The goal of the PAD database is to replicate this function and expand upon its initial design. And in Figure 5 (on the right), we have the select screen for the badges that are currently unlocked and able to be equipped to each team.

Figure 4: PAD team screenshot

Figure 5: PAD Badge select screenshot

These are screenshots taken from Puzzles and dragons. In Figure 6 (top image), we can see player information displayed on the top of the screen. This is always present except when in the gameplay portion of the game (these are known as dungeons.) In Figure 7 (bottom image), we find the player’s unique ID. These pieces of information are crucial to the development of the PAD database.



Figure 6: PAD Player info screenshot



Figure 7: PAD Player ID screenshot

## 4.7 Structural viewpoint

In this section, we will identify and define each component of the PAD database in detail.

**MonsterAttribute table**

Holds the 5 attributes assigned to each monster in PAD

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| AttributeName | Identifies the name of the attribute as a string. | Varchar (50) - PK | MonsterClass:  AttributeOne  AttributeTwo |

Table 1:MonsterAttribute

**MonsterType table**

Holds the 12 types assigned to each monster in PAD

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| MonsterTypeName | Identifies the name of the types as a string. | Varchar (50) - PK | MonsterClass:  MonsterTypeOne  MonsterTypeTwo  MonsterTypeThree |

Table 2:MonsterType

**ActiveSkill table**

Holds all Active skills associated to monsters in PAD

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| ActiveSkillName | Identifies the name of an Active skill as a string. | Varchar (100) - PK | MonsterClass:  ActiveSkillName |
| ActiveSkillDesc | Describes the function of an Active skill using string content. | Varchar (MAX) – NOT NULL |  |
| ActiveSkillMaxLevel | Identifies the maximum numeric value in which an Active skill can be upgraded. | Int – NOT NULL |  |
| ActiveSkillMaxCoolDown | Identifies the initial numeric value needed before an Active skill can be used by a player. | Int – NOT NULL |  |

Table 3:ActiveSkill

**LeaderSkill table**

Holds all Leader skills associated to monsters in PAD

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| LeaderSkillName | Identifies the name of a Leader skill as a string. | Varchar (100) - PK | MonsterClass:  LeaderSkillName |
| LeaderSKillDesc | Describes the function of a Leader skill using string content. | Varchar (MAX) – NOT NULL |  |

Table 4:LeaderSkill

**AwokenSkill table**

Holds all Awoken skills found in PAD

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| AwokenSkillName | Identifies the name of an Awoken skill as a string. | Varchar (100) - PK | AwokenSkillList:  AwokenSkillOne  AwokenSkillTwo  AwokenSkillThree  AwokenSkillFour  AwokenSkillFive  AwokenSkillSix  AwokenSkillSeven  AwokenSkillEight  AwokenSkillNine |
| AwokenSkillDesc | Describes the function of an Awoken skill using string content. | Varchar (MAX) – NOT NULL |  |

Table 5:AwokenSkill

**AwokenSkillList table**

Holds all composed Awoken skills in order designed for a specific monster in PAD

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| ASListID | Identifies a given Awoken skill list to its destined monster as an integer value. | Int - PK | MonsterClass:  ASListID |
| AwokenSkillOne | Identifies the monsters’ Awoken skill using string content. | Varchar (100) – NOT NULL | AwokenSkill:  AwokenSkillName |
| AwokenSkillTwo | Identifies the monsters’ Awoken skill using string content. | Varchar (100) – NULL | AwokenSkill:  AwokenSkillName |
| AwokenSkillThree | Identifies the monsters’ Awoken skill using string content. | Varchar (100) – NULL | AwokenSkill:  AwokenSkillName |
| AwokenSkillFour | Identifies the monsters’ Awoken skill using string content. | Varchar (100) – NULL | AwokenSkill:  AwokenSkillName |
| AwokenSkillFive | Identifies the monsters’ Awoken skill using string content. | Varchar (100) – NULL | AwokenSkill:  AwokenSkillName |
| AwokenSkillSix | Identifies the monsters’ Awoken skill using string content. | Varchar (100) – NULL | AwokenSkill:  AwokenSkillName |
| AwokenSkillSeven | Identifies the monsters’ Awoken skill using string content. | Varchar (100) – NULL | AwokenSkill:  AwokenSkillName |
| AwokenSkillEight | Identifies the monsters’ Awoken skill using string content. | Varchar (100) – NULL | AwokenSkill:  AwokenSkillName |
| AwokenSkillNine | Identifies the monsters’ Awoken skill using string content. | Varchar (100) – NULL | AwokenSkill:  AwokenSkillName |

Table 6:AwokenSkillList

**LatentSkill table**

Holds all Latent skills found in PAD

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| LatentSkillName | Identifies the name of a Latent skill as a string. | Varchar (50) - PK | LatentSkillList:  LatentSkillOne  LatentSkillTwo  LatentSkillThree  LatentSkillFour  LatentSkillFive  LatentSkillSix |
| LatentSkillDesc | Describes the function of a Latent skill using string content. | Varchar (MAX) – NOT NULL |  |
| LSSlotsReq | Identifies the numeric value required for a monster to equip a Latent skill. | Int – NOT NULL |  |

Table 7:LatentSkill

**LatentSkillList table**

Determines each monsters’ available and used latent skill slots

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| InstanceID | Identifies the monster when checking its slots as an integer value. | Int- PK | MonsterInstance:  LSListID |
| LatentSkillOne | Identifies if the slot is available/Identifies a monsters’ Latent skill as a string. | Varchar (50) – NULL | LatentSkill:  LatentSkillName |
| LatentSkillTwo | Identifies if the slot is available/Identifies a monsters’ Latent skill as a string. | Varchar (50) – NULL | LatentSkill:  LatentSkillName |
| LatentSkillThree | Identifies if the slot is available/Identifies a monsters’ Latent skill as a string. | Varchar (50) – NULL | LatentSkill:  LatentSkillName |
| LatentSkillFour | Identifies if the slot is available/Identifies a monsters’ Latent skill as a string. | Varchar (50) – NULL | LatentSkill:  LatentSkillName |
| LatentSkillFive | Identifies if the slot is available/Identifies a monsters’ Latent skill as a string. | Varchar (50) – NULL | LatentSkill:  LatentSkillName |
| LatentSkillSix | Identifies if the slot is available/Identifies a monsters’ Latent skill as a string. | Varchar (50) – NULL | LatentSkill:  LatentSkillName |
| ExtraSlot | Determines if a monster has access to a sixth Latent skill slot | Bit – NOT NULL – default: 0 | LatentSkill:  LatentSkillName |

Table 8:LatentSkillList

**MonsterClass table**

Holds records of every monster in PAD (from the North American release)

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| MonsterClassID | Identifies a monster via an integer value. | Int - PK | MonsterInstance:  MonsterClassID  AssistMonsterID  EvolutionTree:  NextMonsterID  BaseMonsterID  EvoMaterialIDOne  EvoMaterialIDTwo  EvoMaterialIDThree  EvoMaterialIDFour  EvoMaterialIDFive |
| MonsterName | Identifies the monsters’ name as string content. | Varchar (100) – NOT NULL |  |
| Rarity | In-game representation of a monster’s value as an integer value. | Int – NOT NULL |  |
| AttributeOne | Identifies a monster’s primary attribute using a string. | Varchar (50) – NOT NULL | MonsterAttribute:  AttributeName |
| AttributeTwo | Identifies a monster’s secondary attribute using a string. | Varchar (50) – NULL | MonsterAttribute:  AttributeName |
| MonsterTypeOne | Identifies one of the monster’s 3 possible type as a string. | Varchar (50) – NOT NULL | MonsterType:  MonsterTypeName |
| MonsterTypeTwo | Identifies one of the monster’s 3 possible type as a string. | Varchar (50) – NULL | MonsterType:  MonsterTypeName |
| MonsterTypeThree | Identifies one of the monster’s 3 possible type as a string. | Varchar (50) – NULL | MonsterType:  MonsterTypeName |
| MaxExperience | Identifies a monster’s highest point reachable in experience as an integer value. | Int – NOT NULL |  |
| MaxLevel | Identifies a monster’s highest point reachable in level as an integer value. | Int – NOT NULL |  |
| MonsterCost | Identifies a monster’s integer value used to determine how teams are assembled. | Int – NOT NULL |  |
| ASListID | Confirms whether or not a monster has an AwokenSkill list. | Int - NULL | AwokenSkillList:  ASListID |
| LeaderSkillName | Identifies a monster’s Leader skill as a string. | Varchar (100) – NULL | LeaderSkill:  LeaderSkillName |
| ActiveSkillName | Identifies a monster’s Active skill as a string. | Varchar (100) – NULL | ActiveSkill:  ActiveSkillName |
| MaxATK | Identifies a monster’s highest point reachable in his attack stat as an integer value. | Int – NOT NULL |  |
| MinATK | Identifies a monster’s lowest point reachable in his attack stat as an integer value. | Int – NOT NULL |  |
| GrowthRateATK | Identifies how a monster’s attack stat will be influence as it continues to grow. | Int – NOT NULL | Curve:  CurveInstance |
| MaxHP | Identifies a monster’s highest point reachable in his health point stat as an integer value. | Int – NOT NULL |  |
| MinHP | Identifies a monster’s lowest point reachable in his health point stat as an integer value. | Int – NOT NULL |  |
| GrowthRateHP | Identifies how a monster’s health point stat will be influence as it continues to grow. | Int – NOT NULL | Curve:  CurveInstance |
| MaxRCV | Identifies a monster’s highest point reachable in his recovery stat as an integer value. | Int – NOT NULL |  |
| MinRCV | Identifies a monster’s lowest point reachable in his recovery stat as an integer value. | Int – NOT NULL |  |
| GrowthRateRCV | Identifies how a monster’s recovery stat will be influence as it continues to grow. | Int – NOT NULL | Curve:  CurveInstance |
| MaxCoinValue | Identifies the highest value a given monster can be sold as an integer value. | Int – NOT NULL |  |
| MinCoinValue | Identifies the lowest value a given monster can be sold as an integer value. | Int – NOT NULL |  |
| MaxExperienceValue | Identifies the highest integer value possible when using a given monster to enhance another. | Int – NOT NULL |  |
| MinExperienceValue | Identifies the lowest integer value possible when using a given monster to enhance another. | Int – NOT NULL |  |
| MonsterPointValue | Identifies the monster points received when selling a given monster as an integer value. | Int – NOT NULL |  |
| LSSlots | Identifies a monster’s available Latent skill slots. | Int – NOT NULL |  |

Table 9:MonsterClass

**EvolutionTree table**

Holds all records for each monster’s required material in order to change into different monsters

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| NextMonsterID | Identification number for a monster after the evolution. | Int - PK | MonsterClass:  MonsterClassID |
| BaseMonsterID | Identification number for a monster before the evolution. | Int – NOT NULL | MonsterClass:  MonsterClassID |
| EvoMaterialIDOne | Identification number for a monster required to evolve another monster. | Int – NOT NULL | MonsterClass:  MonsterClassID |
| EvoMaterialIDTwo | Identification number for a monster required to evolve another monster. | Int - NULL | MonsterClass:  MonsterClassID |
| EvoMaterialIDThree | Identification number for a monster required to evolve another monster. | Int - NULL | MonsterClass:  MonsterClassID |
| EvoMaterialIDFour | Identification number for a monster required to evolve another monster. | Int - NULL | MonsterClass:  MonsterClassID |
| EvoMaterialIDFive | Identification number for a monster required to evolve another monster. | Int - NULL | MonsterClass:  MonsterClassID |
| MinLevel | Minimum required level for a given monster in order to perform evolution. | Int – NOT NULL |  |
| Devolveable | Identifies whether or not a monster is able to return to a previous monster. | Bit – NOT NULL |  |
| LevelReset | Identifies whether or not a monster’s level will stay the same or return to the minimum level. | Bit – NOT NULL |  |

Table 10:EvolutionTree

**Player table**

Holds the important records of a player needed for identifying monsters and team building.

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| PlayerID | Identifies a player as an integer value. | Int - PK | MonsterInstance:  PlayerID |
| PlayerRank | Identifies a player’s progression in the game as an integer value. | Int- NOT NULL |  |
| PlayerName | Identifies a player’s chosen name as a string. | Varchar (50) –NULL |  |
| PlayerAttribute | Identifies a player’s chosen attribute as a string. | Varchar (50) –NULL | MonsterAttribute:  AttributeName |

Table 11:Player

**MonsterInstance table**

Holds the records of specific monsters owned by players of PAD.

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| InstanceID | Identifies a specific monster entered into the database. | Int - PK | Team:  LeaderMonster  SubMonsterOne  SubMonsterTwo  SubMonsterFour |
| PlayerID | Identifies the player related to his monsters in the database. | Int – NOT NULL | Player:  PlayerID |
| MonsterClassID | Identifies the monster instance to an existing monster in PAD. | Int – NOT NULL | MonsterClass:  MonsterClassID |
| CurrentExperience | Identifies a monster’s current integer value representing its experience points. | Int- NOT NULL |  |
| PlusATK | Identifies a monster’s additional integer value added to the attack stat. | Int- NOT NULL |  |
| PlusRCV | Identifies a monster’s additional integer value added to the recovery stat. | Int- NOT NULL |  |
| PlusHP | Identifies a monster’s additional integer value added to the health point stat. | Int- NOT NULL |  |
| SkillsAwoke | Identifies the monster’s current active Awoken skills. | Int- NOT NULL |  |
| AssistMonsterID | Identifies a monster’s assist representative monster by integer value. | Int – NULL | MonsterClass:  MonsterClassID |
| SkillLevel | Identifies a monster’s current Active skill integer value. | Int – NULL |  |
| LSListID | Identifies a monster’s current Latent skill equip. | Int – NULL | LatentSkillList:  LSListID |
| ActiveSkillCoolDown | Identifies a monster’s current integer value representing the number needed before use of the Active skill. | Int – NOT NULL - default 0 |  |
| CurrentHP | Identifies a monster’s current health points stat. | Int – NOT NULL -default 0 |  |
| CurrentATK | Identifies a monster’s current attack stat. | Int – NOT NULL - default 0 |  |
| CurrentRCV | Identifies a monster’s current recovery stat. | Int – NOT NULL - default 0 |  |
| PlusHPAmount | Identifies a monster’s additional health point stat bonus as an integer value. | Int – NOT NULL -default 0 |  |
| PlusATKAmount | Identifies a monster’s additional attack stat bonus as an integer value. | Int – NOT NULL -default 0 |  |
| PlusRCVAmount | Identifies a monster’s additional recovery stat bonus as an integer value. | Int – NOT NULL -default 0 |  |
| CurrentLevel | Identifies a monster’s current integer value representing its level. | Int – NOT NULL -default 0 |  |

Table 12:MonsterInstance

**Team table**

Contains the calculated fields needed to simulate the team building in PAD.

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| TeamInstanceID | Unique integer value used to identify a built team. | Int - PK |  |
| PlayerID | Identifies the player related to his team in the database. | Int – NOT NULL | Player:  PlayerID |
| TeamName | Identifies the team as a string. | Varchar (50) - NULL |  |
| LeaderMonster | Identifies a monster set in a team’s leader slot. | Int- NOT NULL | MonsterInstance:  InstanceID |
| SubMonsterOne | Identifies a monster set in a team’s sub slot. | Int- NULL | MonsterInstance:  InstanceID |
| SubMonsterTwo | Identifies a monster set in a team’s sub slot. | Int- NULL | MonsterInstance:  InstanceID |
| SubMonsterThree | Identifies a monster set in a team’s sub slot. | Int- NULL | MonsterInstance:  InstanceID |
| SubMonsterFour | Identifies a monster set in a team’s sub slot. | Int- NULL | MonsterInstance:  InstanceID |
| BadgeName | Identifies a Team’s selected Badge as a string. | Varchar (50) – NULL | Badge:  BadgeName |
| TeamHP | Calculated field identifying a team’s combined health points into one pool. | Int – NOT NULL- default 0 |  |
| FireATK | Calculated field identifying a team’s combined Fire attribute specific attack points into one pool. | Int – NOT NULL -default 0 |  |
| WaterATK | Calculated field identifying a team’s combined Water attribute specific attack points into one pool. | Int – NOT NULL -default 0 |  |
| WoodATK | Calculated field identifying a team’s combined Wood attribute specific attack points into one pool. | Int – NOT NULL- default 0 |  |
| LightATK | Calculated field identifying a team’s combined Light attribute specific attack points into one pool. | Int – NOT NULL -default 0 |  |
| DarkATK | Calculated field identifying a team’s combined Dark attribute specific attack points into one pool. | Int – NOT NULL -default 0 |  |
| TeamRCV | Calculated field identifying a team’s combined recovery points into one pool. | Int – NOT NULL -default 0 |  |
| TeamCost | Calculated field identifying a team’s combined cost points into one pool. | Int – NOT NULL- default 0 |  |
| TeamLeaderSkill | Identifies the monster set in the team’s Leader Slot’s Leader skill as a string. | Varchar (100) – NULL |  |

Table 13:Team

**Badge table**

Holds all Badges found in PAD.

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| BadgeName | Identifies the badge as a sting. | Varchar (50) - PK | Team:  BadgeName |
| BadgeDesc | Description of the badges’ effects. | Varchar (MAX) – NOT NULL |  |

Table 14:Badge

**Curve table**

Contains the numeric values used to influence the progression of a monster’s stats and experience.

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Properties:** | **Relationship:** |
| CurveInstance | Identifies the specific curve as an integer value. | Int - PK | MonsterClass:  GrowthRateATK  GrowthRateHP  GrowthRateRCV |
| ExperienceCurve | Identifies the floating-point value of the curve used to determine a monster’s experience growth. | Float – NOT NULL -default: 0 |  |
| ExperienceCurveDesc | Description of the experience curve’s purpose. | Varchar(MAX) – NOT NULL |  |
| NormalStatCurve | Identifies the floating-point value of the curve used to determine a monster’s stats growth. | Float – NOT NULL -default: 0 |  |
| NormalStatCurveDesc | Description of the Normal curve’s purpose. | Varchar(MAX) – NOT NULL |  |
| EarlyCurve | Identifies the floating-point value of the curve used to determine a monster’s stats growth. | Float – NOT NULL -default: 0 |  |
| EarlyCurveDesc | Description of the Early curve’s purpose. | Varchar(MAX) – NOT NULL |  |
| LateCurve | Identifies the floating-point value of the curve used to determine a monster’s stats growth. | Float – NOT NULL- default: 0 |  |
| LateCurveDesc | Description of the Late curve’s purpose. | Varchar(MAX) – NOT NULL |  |

Table 15:Curve

## 4.8 Logical design

The PAD database is designed in Microsoft SQL server in a single database. As of version 1.2 of the PAD database SDD, there is no network or server linked to the database. All relationships are between tables found in the database itself. (See figure 2: PAD database visual representation)

## 4.9 Dependency viewpoint

As of version 1.2 of the PAD database SDD, there exists only the contents of the database alone. However, since each table must be populated once created, it is integral that specific tables come before others. This applies to both creation and deletion of the tables.

* *Create tables in this order*:
  + LatentSkill
  + MonsterType
  + MonsterAttribute
  + ActiveSkill
  + LeaderSkill
  + AwokenSkill
  + AwokenSkillList
  + MonsterClass
  + EvolutionTree
  + Player
  + LatentList
  + MonsterInstance
  + Badge
  + Team
  + Curve
* *Delete tables in this order*:
  + Team
  + Badge
  + MonsterInstance
  + LatentSkillList
  + Player
  + EvolutionTree
  + MonsterClass
  + Curve
  + MonsterType
  + MonsterAttribute
  + ActiveSkill
  + LeaderSkill
  + AwokenSkillList
  + AwokenSkill
  + LatentSkill

## 4.10 Design language

During the production of the PAD database, we made use of these following design languages:

* DB(Database) schema
  + For documentation regarding the visual representation of the database relationships and for the creation queries for all tables.
  + Forms a relationship with Microsoft SQL server through queries.
* UML (unified modeling language)
  + Used to visualise the design for the PAD database.
  + Used to represent context design in the form of Use cases.

# Appendix A: To be determined

* PAD database release plan
* PAD database user interface
* PAD database networking
* Record management for database

# Appendix B: Future enhancements

As of version 1.0 of the SDD document about the PAD database, the project is still in development.

* Present each monster’s active skill along with the leader skill so players get a better idea of how to synergize their skills.
* User interface design
* PAD database performance