UNIVERSITATEA POLITEHNICA BUCURESTI
FACULTATEA DE INGINERIE IN LIMBI STRAINE

Databases

"Videogame World"

Marinescu Radu Mihai

1231B

February 2018

Contents

1. Description of the Universe

In this project, I will present the structure of an open-world online game, starting from geographical information and ending with how different items can be used by player characters.

A zone is a region inside a continent. An instance is a scenario with special encounters situated inside a zone, usually accessed from a portal, meaning it is not in the open world. An instance can have multiple difficulties. Every instance has enemy encounters. Instances can be one of two types: dungeon (for parties of 5 players) and raid (for parties of 10 or more players).

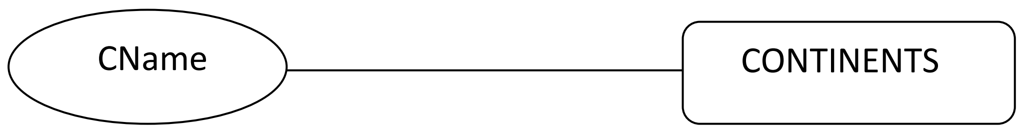
A "boss" is a powerful enemy that yields items to the player, either weapons or armor. A boss only gives one item when defeated, out of multiple possible items. In some cases, one item can be obtained from more than one boss. Weapons and armor can both be of multiple types, but only armor can have slots (gloves, chest plates, leg armor, boots, helmets, etc) as only one weapon can be equipped by a player.

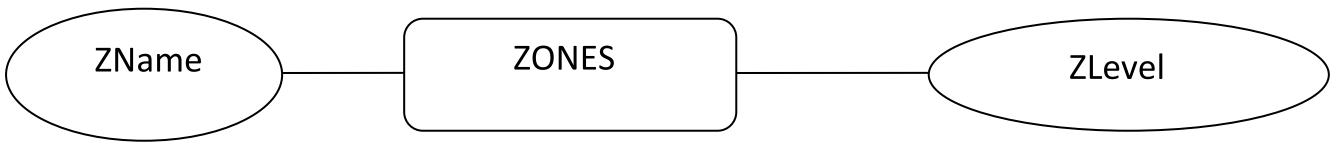
Some weapons and armor can only be used by specific classes. Classes are defined by their role in the world. For example, some bows may only be used by hunter characters, some robes may only be available to priest characters.

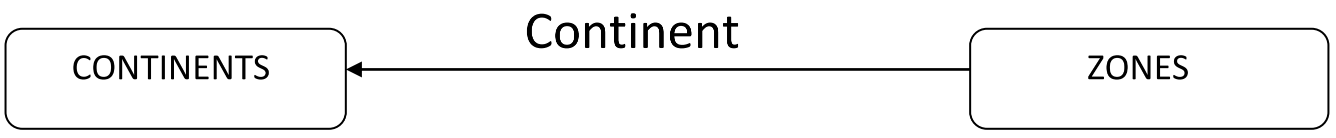
2. Restrictions

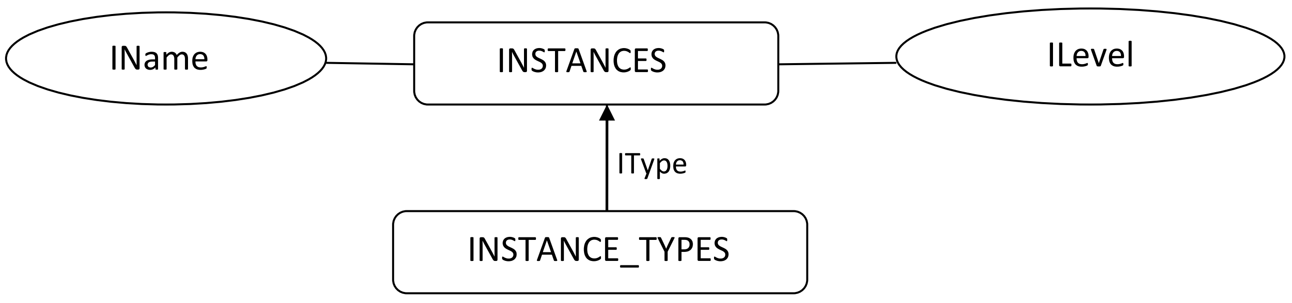
- R1: All continents must have different names.
- R2: All zones in a continent must have different names.
- R3: All instances must have different names.
- R4: All instance difficulties must have a different name.
- R5: No instance can be of more than one type.
- R6: Every instance must have at least one difficulty option.
- R7: All bosses of one instance must have different names.
- R8: One boss must yield at least 1 piece of armor or 1 weapon.
- R9: A weapon cannot be of two types.
- R10: A piece of armor occupies only one slot.
- R11: A piece of armor cannot be of two types.
- R12: Weapon damage values cannot be negative.
- R13: A piece of armor must be used by at least one class.
- R14: A weapon must be used by at least one class.

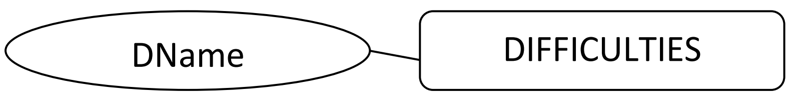
3. Entity-Relationship Diagrams

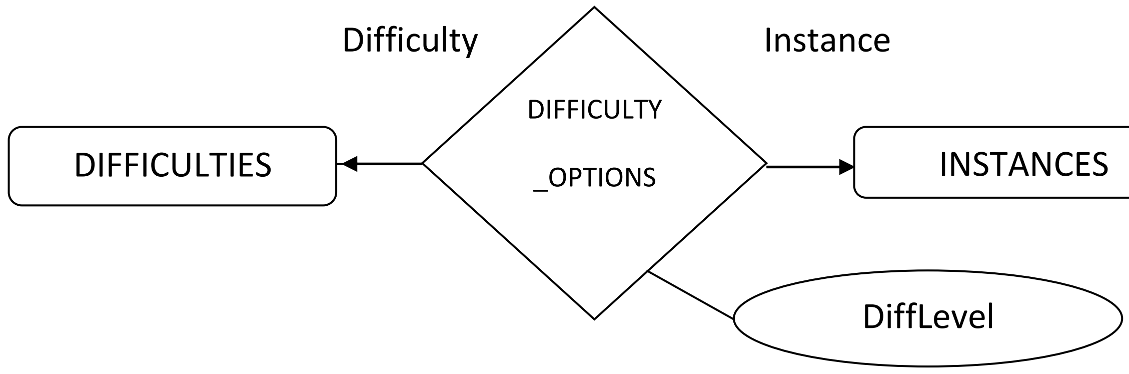


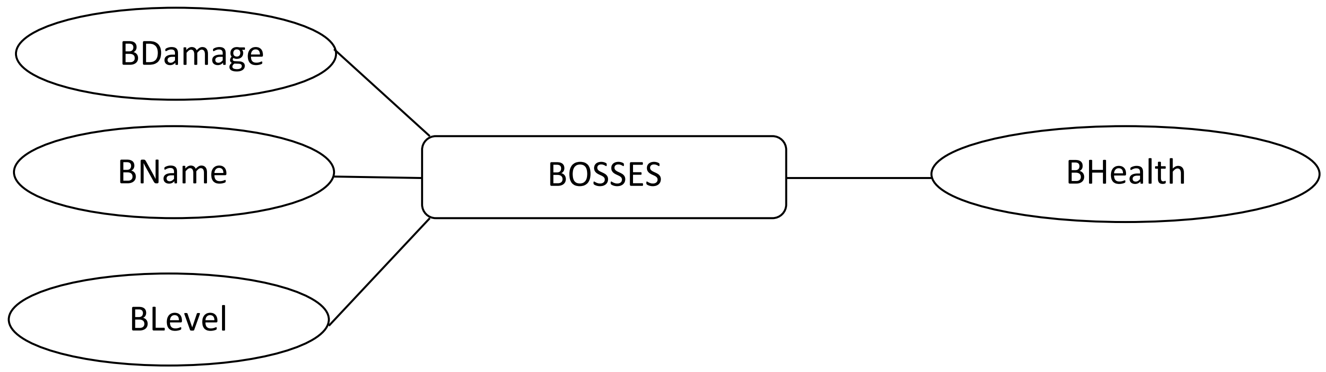


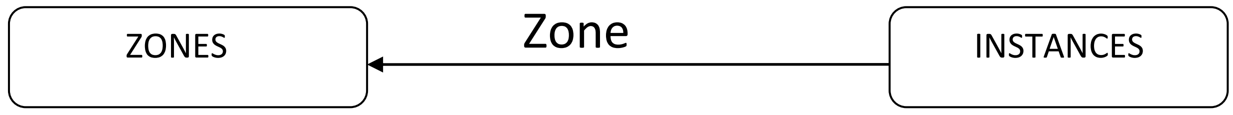


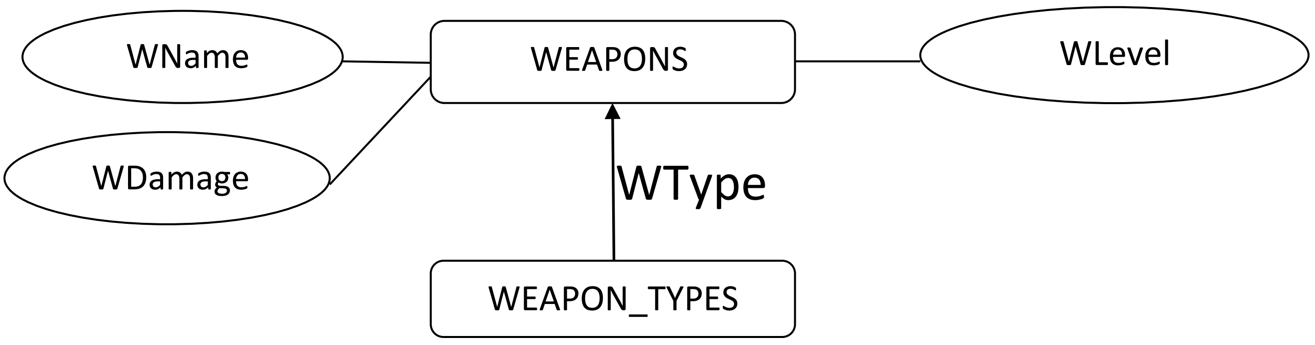


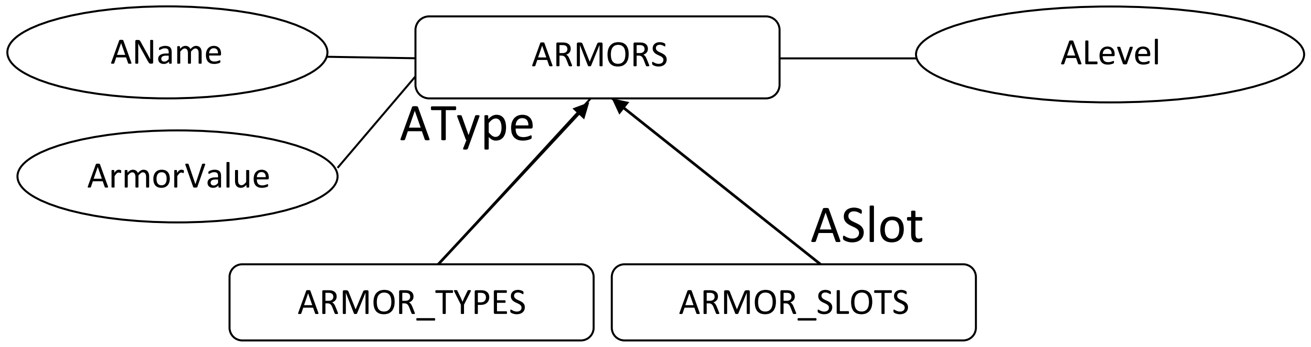


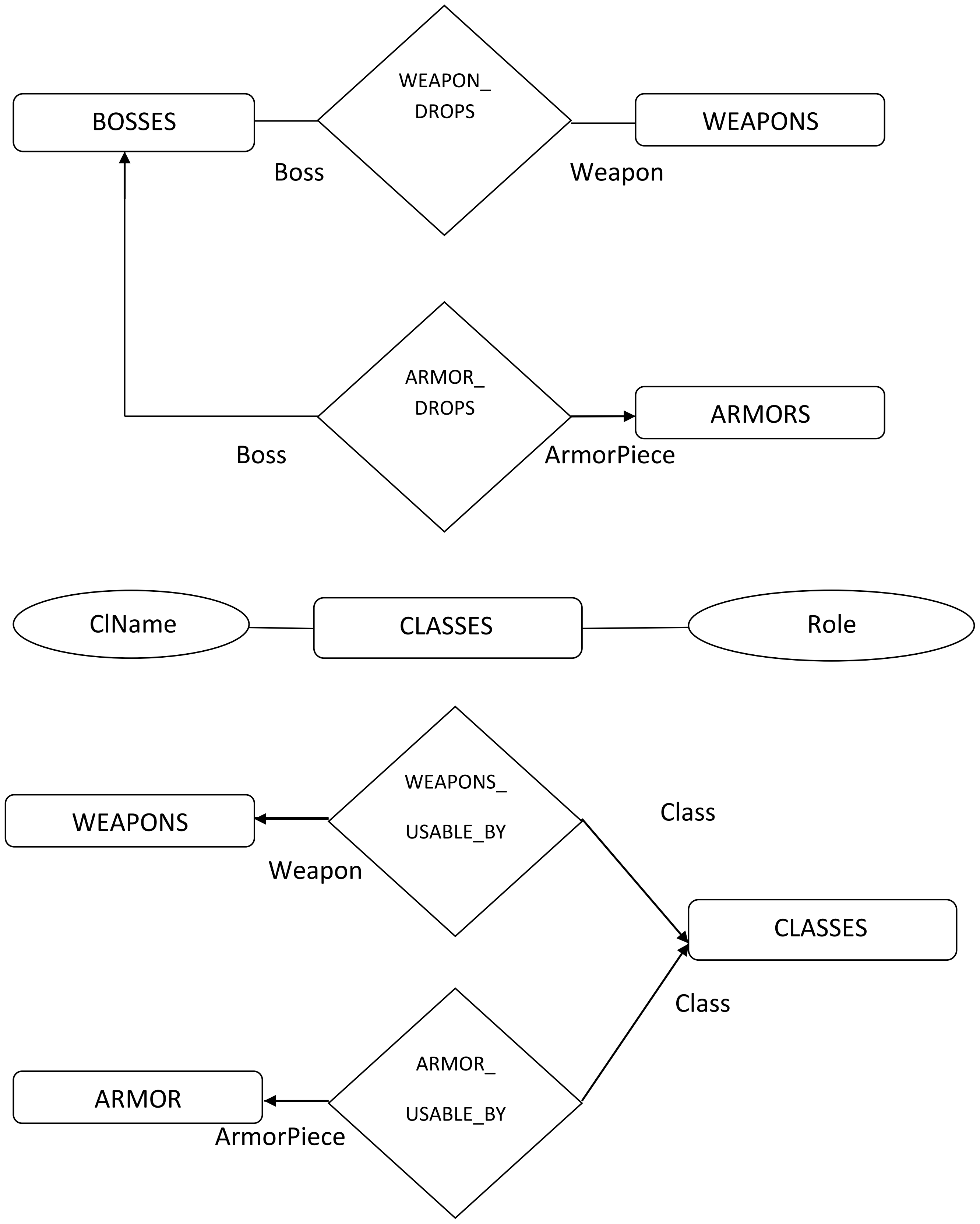

Difficulty
Instance
DIFFICULTIES
DIFFICULTY _OPTIONS
INSTANCES
DiffLevel

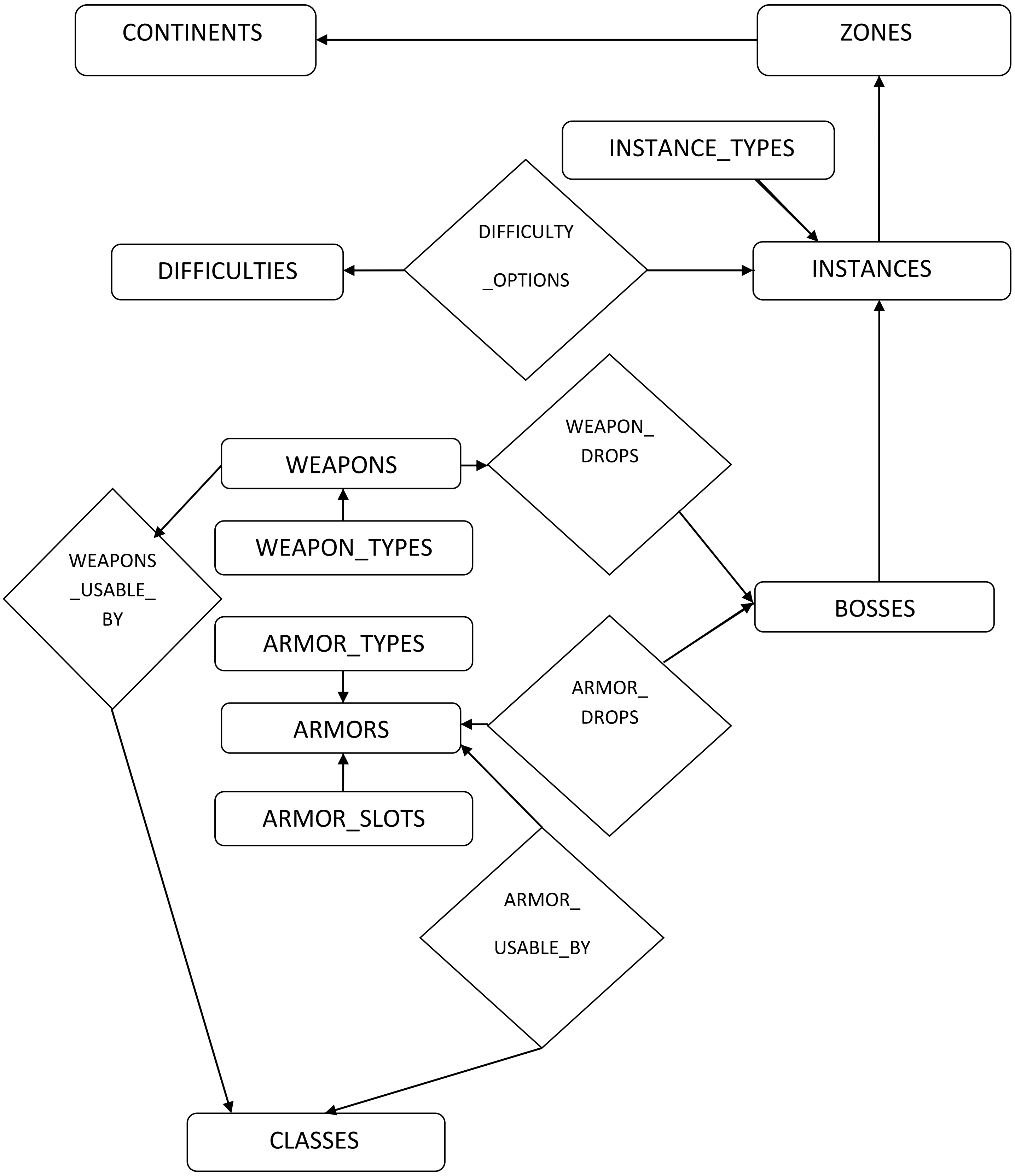

BDamage
BName
BLevel
BOSSES
BHealth


ZONES
Zone
INSTANCES


WName
WDamage
WEAPONS
WLevel
WType
WEAPON_TYPES


AName
ArmorValue
ARMORS
ALevel
AType
ASlot
ARMOR_TYPES
ARMOR_SLOTS


BOSSES
WEAPON_
DROPS
WEAPONS
Boss
Weapon
ARMOR_
DROPS
ARMORS
Boss
ArmorPiece
ClName
CLASSES
Role
WEAPONS_
USABLE_BY
Class
WEAPONS
Weapon
CLASSES
Class
ARMOR_
USABLE_BY
ARMOR
ArmorPiece


CONTINENTS
ZONES
INSTANCE_TYPES
DIFFICULTY
_OPTIONS
DIFFICULTIES
INSTANCES
WEAPON_
DROPS
WEAPONS
WEAPON_TYPES
WEAPONS
USABLE
BY
BOSSES
ARMOR_TYPES
ARMOR_
DROPS
ARMORS
ARMOR_SLOTS
ARMOR_
USABLE_BY
CLASSES

4. Associated Restrictions List

4.1. CONTINENTS (the set of continents in the world)
a. Cardinality: max(card(CONTINENTS))=20
b. Data Ranges:
    CName: ASCII(64)
c. Compulsory data: CName
d. Uniqueness: CName (there may not be two continents with the same name)

4.2. ZONES (the set of zones in the world)

a. Cardinality: max(card(ZONES))=1,000
b. Data Ranges:
    ZName: ASCII (128)
    ZLevel: [1, 110]
c. Compulsory data: ZName, Continent
d. Uniqueness ZName●Continent (there may not be two zones with the same name in the same continent)

4.3. INSTANCES (the set of instances in the world)

a. Cardinality: max(card(INSTANCES))=1,000
b. Data Ranges:
    IName: ASCII(255)
    ILevel: [15,110]
c. Compulsory data: IName, Zone, IType
d. Uniqueness: IName (there may not be two instances with the same name), IName●IType (the same instance cannot be of two types)

4.4. INSTANCE_TYPES (the set of instance types)

a. Cardinality: max(card(INSTANCE_TYPES))=3
b. Data Ranges:
    ITypeName: ASCII(128)
c. Compulsory data: ITypeName
d. Uniqueness: ITypeName (there may not be two instance types with the same name)

4.5. DIFFICULTIES (the set of difficulty choices an instance has)

a. Cardinality: max(card(DIFFICULTIES))=10
b. Data Ranges:
    DName: ASCII(128)
c. Compulsory data: DName
d. Uniqueness: DName (there may not be two difficulties with the same name)

4.6. DIFFICULTY_OPTIONS (the set of pairs <i, d> that store the difficulty options d of instances i)

a. Cardinality: max(card(DIFFICULTY_OPTIONS))=10,000
b. Compulsory data: Instance, Difficulty, DiffLevel
c. Uniqueness: the same pair <i, d> cannot appear more than once.

4.7. BOSSES (the set of bosses in every instance)

a. Cardinality: max(card(BOSSES))=20,000
b. Data Ranges:
    BName: ASCII(128)
    BLevel: [15, 113]
    BDamage: [100, 10,000,000]
    BHealth: [5,000, 10,000,000,000]
c. Compulsory data: BName, Instance
d. Uniqueness: BName●Instance (there may not be two bosses with the same name in the same instance)

4.8. WEAPONS (the set of available weapons in the world)

a. Cardinality: max(card(WEAPONS))=10,000
b. Data Ranges:
    WName: ASCII(128)
    WLevel: [1, 110]
    WDamage: [1, 10,000]
c. Compulsory data: WName, WType
d. Uniqueness: WName (there may not be two weapons with the same name)

4.9. WEAPON_TYPES (the set of weapon types)

a. Cardinality: max(card(WEAPON_TYPES))=20
b. Data Ranges:
    WTypeName: ASCII(128)
c. Compulsory data: WTypeName
d. Uniqueness: WTypeName (the same weapon type may not appear twice)

4.10. ARMORS (the set of available armor pieces in the game)

a. Cardinality: max(card(ARMOR))=100,000
b. Data Ranges:
    AName: ASCII(128)
    ALevel: [1, 110]
    ArmorValue: [1, 5000]
c. Compulsory data: AName, ASlot, AType
d. Uniqueness: AName (there may not be two armor pieces with the same name)

4.11. ARMOR_TYPES

a. Cardinality: max(card(ARMOR_TYPES))=20
b. Data Ranges:
    ATypeName: ASCII(128)
c. Compulsory data: ATypeName
d. Uniqueness: ATypeName (the same armor type may not appear twice)

4.12. ARMOR_SLOTS

a. Cardinality: max(card(ARMOR_SLOTS))=20
b. Data Ranges:
    ASlotName: ASCII(128)
c. Compulsory data: ASlotName
d. Uniqueness: ASlotName (the same armor slot may not appear twice)

4.13. WEAPON_DROPS (the set of pairs <b, w> that store which weapons w are yielded by boss b)

a. Cardinality: max(card(WEAPON_DROPS))=100,000
b. Compulsory data: Boss, Weapon
c. Uniqueness: the same pair <b, w> cannot appear more than once.

4.14. ARMOR_DROPS (the set of pairs <b, a> that store which armor pieces a are yielded by boss b)

a. Cardinality: max(card(ARMOR_DROPS))=1,000,000
b. Compulsory data: Boss, ArmorPiece
c. Uniqueness: the same pair <b, a> cannot appear more than once.

4.15. CLASSES (the set of playable classes in the game)

a. Cardinality: max(card(CLASSES))=15
b. Data Ranges:
    ClName: ASCII(128)
    Role: ASCII(128)
c. Compulsory data: ClName
d. Uniqueness: ClName (the same class cannot appear twice)

4.16. WEAPONS_USABLE_BY (the set of pairs <c, w> where the weapons w can be used by the class c)

a. Cardinality: max(card(WEAPONS_USABLE_BY))=150,000
b. Compulsory data: Class, Weapon
c. Uniqueness: the same pair <c, w> cannot appear twice.

4.17. ARMOR_USABLE_BY (the set of pairs <c, a> where the armor pieces a can be used by the class c)

a. Cardinality: max(card(ARMOR_USABLE_BY))=1,500,000
b. Compulsory data: Class, ArmorPiece
c. Uniqueness: the same pair <c, a> cannot appear twice.

5. Initial Mathematical Scheme

By translating the ERDs and restriction lists into mathematical schemes, we get the following scheme:

5.1. CONTINENTS

    x - NAT(2), total
    CName - ASCII(64), total, key (R1

5.2. ZONES

    x - NAT(3), total
    ZName - ASCII(128), total, key
    ZLevel - [1, 110]
    Continent: ZONES → CONTINENTS, total

ZName●Continent key (R2)
Any combination containing ZName is a key.
ZLevel NOT a key (two zones can have the same level)
Continent NOT a key (two zones can be in the same continent)
ZLevel●Continent NOT a key (two zones of the same level can be in the same continent)

5.3. INSTANCES

    x - NAT(3), total
    IName - ASCII(255), total, key (R3)
    ILevel - [15, 110]
    Zone - INSTANCES → ZONES, total
    IType - INSTANCES → INSTANCE_TYPES, total

IName●IType key (R5)
Any combination containing IName is a key.
ILevel NOT a key (two instances can be at the same level)
Zone NOT a key (more than one instance can be in one zone)
IType NOT a key (more than one instance can be of one type)
ILevel●IType NOT a key (multiple instances of the same level can be of the same type)
ILevel● Zone NOT a key (multiple instances of the same level can be in the same zone)
IType● Zone NOT a key (multiple instances of the same type can be in the same zone)
ILevel● Zone ●IType NOT a key (multiple instances of the same level can be of the same type and be in the same zone)

5.4. INSTANCE_TYPES

    x - NAT(1), total
    ITypeName - ASCII(128), total, key

5.5. DIFFICULTIES

    x - NAT(2), total
    DName - ASCII(128), total, key (R4)

5.6. DIFFICULTY_OPTIONS = (INSTANCES, DIFFICULTIES)

    x - NAT(3), total
    DiffLevel - [15, 110]

5.7. BOSSES

    x - NAT(5), total
    BName: ASCII(128), total
    BLevel: [15, 113]

BDamage: [100, 10,000,000]
BHealth: [5,000, 10,000,000,000]
Instance: BOSSES → INSTANCES, total

BName●Instance key (R7)
BLevel NOT a key (two bosses can have different levels)
BDamage, BHealth NOT keys (two bosses can have different health and damage values)
Instance NOT a key (a boss can be in two different instances)
BLevel●BHealth, BLevel●BDamage NOT keys (bosses of the same level can have the same damage and health values)
BName●BLevel NOT a key (two bosses with the same name can have the same level)
BName●BHealth, BName●BDamage NOT keys (bosses with the same name can have the same damage and health values)
Instance●BHealth, Instance●BDamage NOT keys (bosses in the same instance can have the same damage and health values)
Instance●BLevel NOT a key (bosses in the same instance can have the same level)

5.8. WEAPONS
x - NAT(5), total
WName: ASCII(128), total, key
WLevel: [1, 110]
WDamage: [1, 10,000]
WType: WEAPONS → WEAPON_TYPES, total

WName●WType key (R9)
Any combination containing WName is a key.
WLevel NOT a key (two weapons can have the same level)
WDamage NOT a key (two weapons can have the same damage values)
WType NOT a key (two weapons can be of the same type)
WLevel●WDamage NOT a key (two weapons of the same level can have the same damage values)
WLevel●WType NOT a key (two weapons of the same level can be of the same type)
WDamage●WType NOT a key (two weapons with the same damage values can be of the same type)
WLevel●WDamage●WType NOT a key (two weapons of the same level can have the same damage values and type)

5.9. WEAPON_TYPES
x - NAT(2), total
WTypeName: ASCII(128), total, key

5.10. ARMORS
x - NAT(5), total
AName: ASCII(128), total, key
ALevel: [1, 110]
ArmorValue : [1, 5000]
ASlot - ARMOR → AMOR_SLOTS, total
AType - ARMOR → ARMOR_TYPES, total

AName●ASlot key (R10), AName●AType key (R11)
Any combination containing AName is a key.
ALevel NOT a key (two armor pieces can have the same level)
AType NOT a key (two armor pieces can be of the same type)
ArmorValue NOT a key (two armor pieces can have the same armor values)
ASlot NOT a key (two armor pieces can occupy the same armor slot)
ALevel●ArmorValue NOT a key (two armor pieces of the same level can have the same armor values)

ALevel●AType NOT a key (two armor pieces of the same level can be of the same type)
ALevel●ASlot NOT a key (two armor pieces of the same level can occupy the same slot)
ArmorValue●AType NOT a key (two armor pieces with the same armor values can be of the same type)
ArmorValue●ASlot NOT a key (two armor pieces with the same armor values can occupy the same slot)
ASlot●AType NOT a key (two armor pieces of the same type can occupy the same slot)
ALevel●ArmorValue●AType NOT a key (two armor pieces of the same level can have the same armor values and type)
ALevel●ArmorValue●ASlot NOT a key (two armor pieces of the same level can have the same armor values and occupy the same slot)
ArmorValue●AType●ASlot NOT a key (two armor pieces of the same type can have the same armor values and occupy the same slot)
ALevel●AType●ASlot●ArmorValue NOT a key (two armor pieces can be of the same level, same type, same armor value and occupy the same slot)

5.11. ARMOR_TYPES
x - NAT(2), total
ATypeName: ASCII(128), total, key

5.12. ARMOR_SLOTS
x - NAT(2), total
ASlotName: ASCII(128), key

5.13. WEAPON_DROPS = (BOSSES, WEAPONS)
x - NAT(5), total

5.14. ARMOR_DROPS = (BOSSES, ARMOR)
x - NAT(6), total

5.15. CLASSES
x - NAT(2), total
ClName: ASCII(128), total, key
Role: ASCII(128)

5.16. WEAPONS_USABLE_BY = (CLASSES, WEAPONS)
x - NAT(6), total

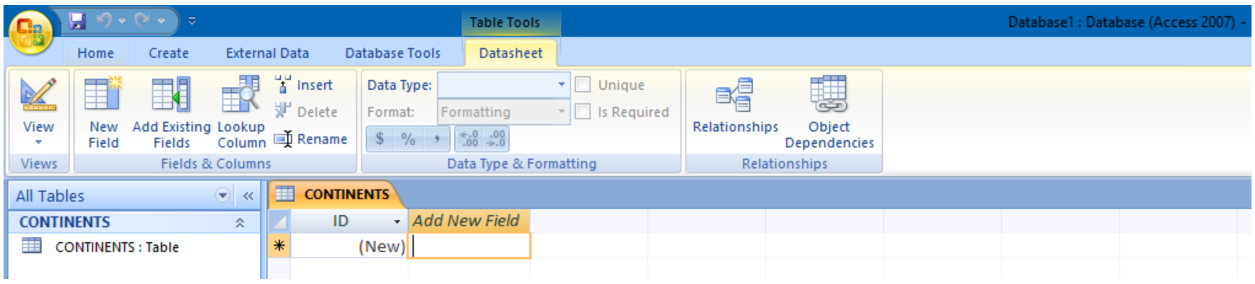
5.17. ARMOR_USABLE_BY = (CLASSES, ARMOR)
x - NAT(8), total

6. DB Implementation in MS Access 2007

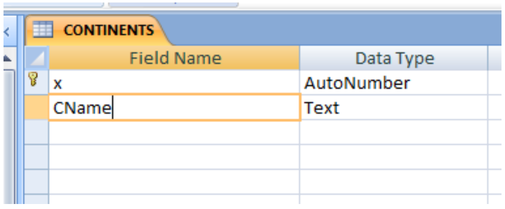
6.1. Creating the database

Open MS Access 2007 and click on the "Blank Database" button shown on the right.

Using Access' menu, I created and saved the first set, called CONTINENTS.



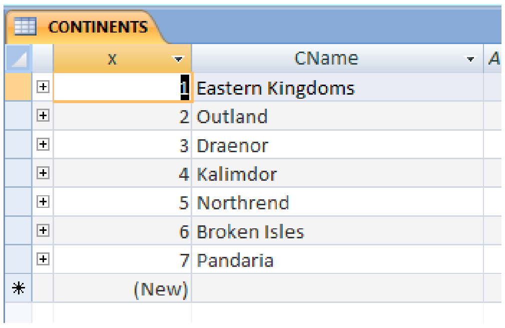
In the "Design Mode" of this table, I entered the corresponding fields, according to the mathematical scheme. x is the Primary Key and CName is the continent's name (also a key). Both fields have the "Required" attribute set to "Yes".



Field Name	Data Type
x	AutoNumber
CName	Text

The following sets are similar: all have an ID "x" and a name, but some also contain special links to other sets using SQL commands. For example, in the ZONES set, the Continent attribute is linked to the CONTINENTS set using the following command:

```
SELECT CONTINENTS.[x],[CONTINENTS].[CName]
FROM CONTINENTS ORDER BY [CName];
```

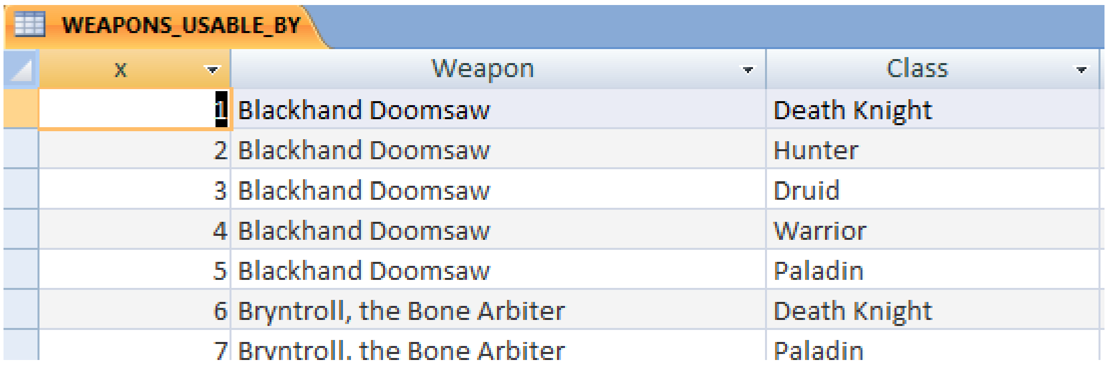


x	CName
1	Eastern Kingdoms
2	Outland
3	Draenor
4	Kalimdor
5	Northrend
6	Broken Isles
7	Pandaria
(New)	

The process is similar for all sets, except the ones which eliminate many to many relationships, such as ARMOR_DROPS or WEAPONS_USABLE_BY. These sets contain pairs of instances of other sets, such as (Boss, Armor) and (Weapon, Class) for the examples given above. They are obtained with SQL commands.

```
SELECT [WEAPONS].[x], [WEAPONS].[WName] FROM WEAPONS ORDER BY [WName];
SELECT [CLASSES].[x], [CLASSES].[ClName] FROM CLASSES ORDER BY [ClName];
```

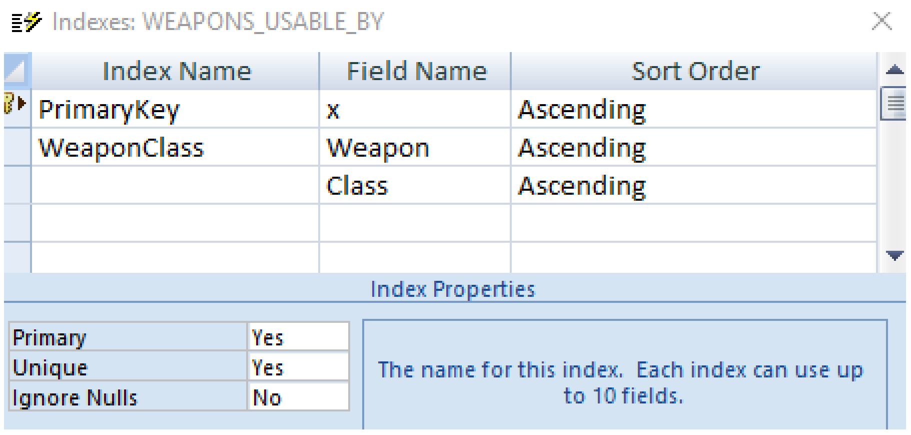
These commands give us:



WEAPONS_USABLE_BY

x	Weapon	Class
1	Blackhand Doomsaw	Death Knight
2	Blackhand Doomsaw	Hunter
3	Blackhand Doomsaw	Druid
4	Blackhand Doomsaw	Warrior
5	Blackhand Doomsaw	Paladin
6	Bryntroll, the Bone Arbiter	Death Knight
7	Bryntroll, the Bone Arbiter	Paladin

To enforce constraints and keys, we enter "Design Mode" and click on the "Indexes" button. For the WEAPONS_USABLE_BY set, the indexes are:



Indexes: WEAPONS_USABLE_BY

Index Name	Field Name	Sort Order
PrimaryKey	x	Ascending
WeaponClass	Weapon	Ascending
	Class	Ascending

Index Properties

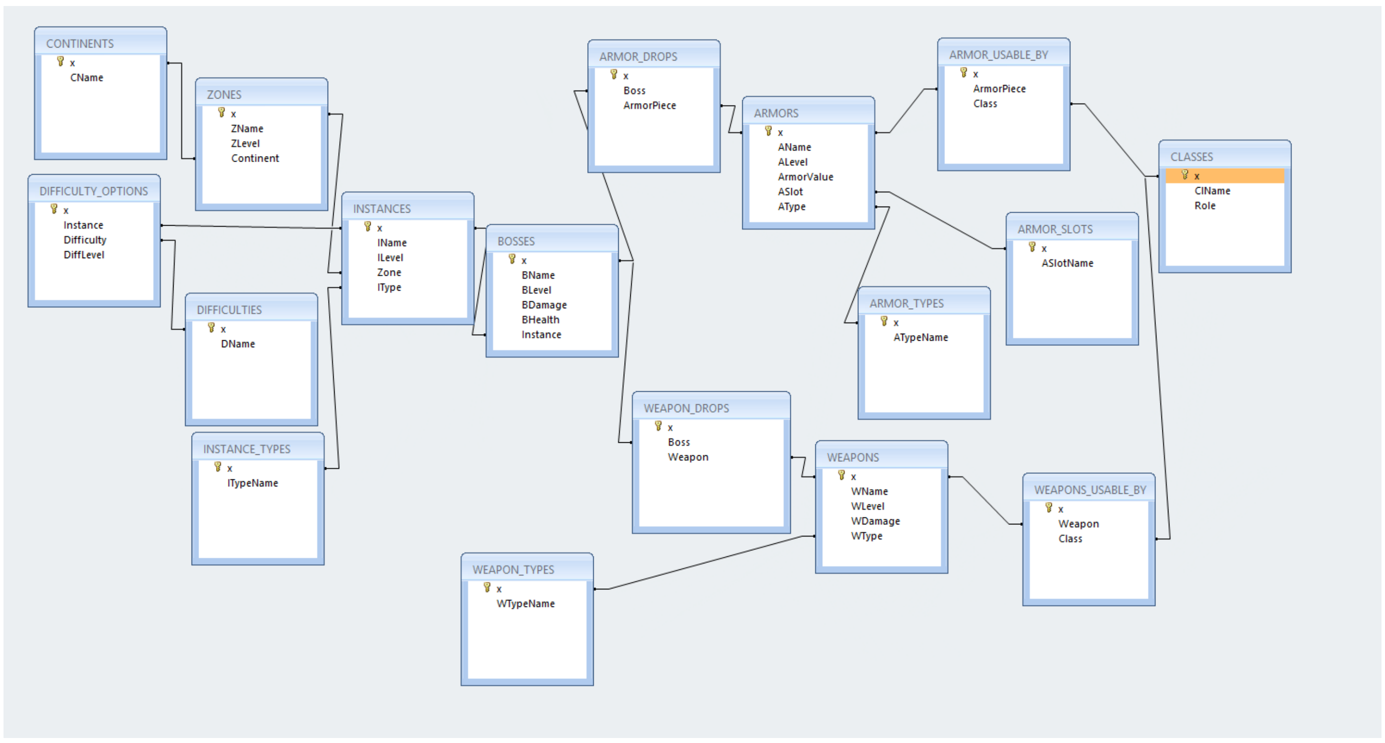
Primary	Yes
Unique	Yes
Ignore Nulls	No

The name for this index. Each index can use up to 10 fields.

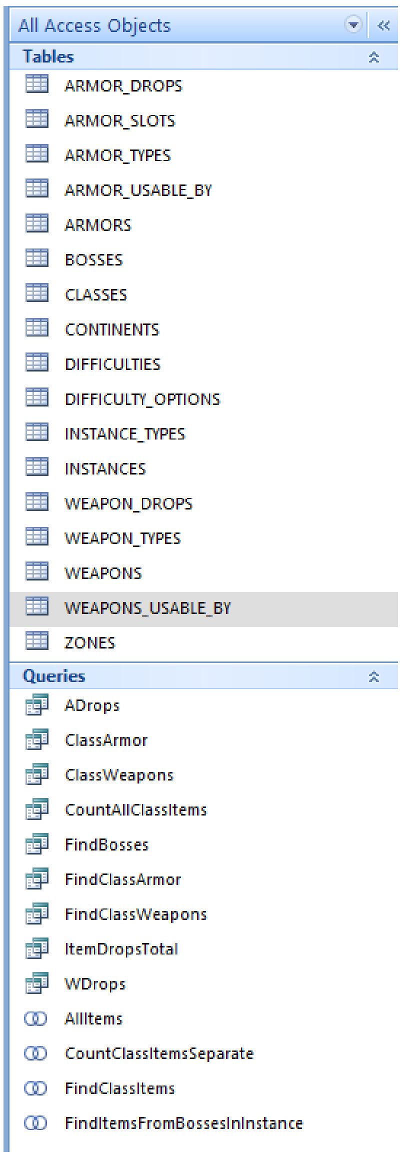
The Weapon●Class pair's "Unique" attribute is set to "Yes" because two the same pair may not appear twice, enforcing the uniqueness rule defined in 4.16.c.

Other sets have similar-looking indexes.

The final relationship scheme is:



7. Queries



All Access Objects

Tables

- ARMOR_DROPS
- ARMOR_SLOTS
- ARMOR_TYPES
- ARMOR_USABLE_BY
- ARMORS
- BOSSES
- CLASSES
- CONTINENTS
- DIFFICULTIES
- DIFFICULTY_OPTIONS
- INSTANCE_TYPES
- INSTANCES
- WEAPON_DROPS
- WEAPON_TYPES
- WEAPONS
- WEAPONS_USABLE_BY
- ZONES

Queries

- ADrops
- ClassArmor
- ClassWeapons
- CountAllClassItems
- FindBosses
- FindClassArmor
- FindClassWeapons
- ItemDropsTotal
- WDrops
- AllItems
- CountClassItemsSeparate
- FindClassItems
- FindItemsFromBossesInInstance

There are 4 queries which work with the aid of subqueries:

7.1.

CountAllClassItems sums up the number of armor pieces and weapons and displays it next to the name of its corresponding class. Its subquery is CountClassItemsSeparate, which creates a UNION of two tables generated by two other subqueries: ClassArmor (the counted number of armor pieces per class) and ClassWeapons (the counted number of weapons per class).

7.1.1. ClassArmor

```
SELECT Classes.ClName,
Count(ARMOR_USABLE_BY.ArmorPiece) AS A
FROM CLASSES INNER JOIN ARMOR_USABLE_BY ON
CLASSES.x=ARMOR_USABLE_BY.Class
GROUP BY ClName;
```

7.1.2. ClassWeapons

```
SELECT Classes.ClName,
Count(WEAPONS_USABLE_BY.Weapon) AS W
FROM CLASSES INNER JOIN WEAPONS_USABLE_BY ON
CLASSES.x=WEAPONS_USABLE_BY.Class
GROUP BY ClName;
```

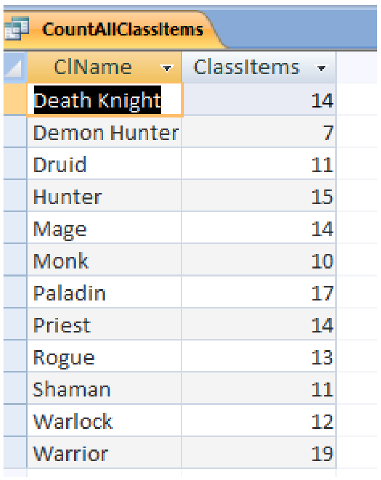
7.1.3. CountClassItemsSeparate

```
(SELECT CLASSES.ClName, ClassWeapons.W FROM
ClassWeapons)
UNION ALL
(SELECT CLASSES.ClName, ClassArmor.A FROM ClassArmor);
```

7.1.4. CountAllClassItems

```
SELECT     CLASSES.ClName,     SUM([CountClassItemsSeparate].W)     AS
ClassItems
FROM CountClassItemsSeparate
GROUP BY CLASSES.ClName;
```

The final result for the current data values is displayed in the image:



CountAllClassItems

ClName	ClassItems
Death Knight	14
Demon Hunter	7
Druid	11
Hunter	15
Mage	14
Monk	10
Paladin	17
Priest	14
Rogue	13
Shaman	11
Warlock	12
Warrior	19

7.2.

ItemDropsTotal works similar to CountAllClassItems in that it sums up the number of armor pieces and weapons, but it displays that number next to a column of bosses. Its subquery, AllItems allows for duplicate values of bosses, because it is the UNION of the tables created by ADrops (counted number of armor pieces yielded per boss) and WDrops (counted number of weapons yielded per boss).

7.2.1. ADrops

```
SELECT BOSSES.BName, Count(ARMOR_DROPS.ArmorPiece) AS A
FROM BOSSES INNER JOIN ARMOR_DROPS ON BOSSES.x=ARMOR_DROPS.Boss
GROUP BY BName;
```

7.2.2. WDrops

```
SELECT BOSSES.BName, Count(WEAPON_DROPS.Weapon) AS W
FROM BOSSES INNER JOIN WEAPON_DROPS ON BOSSES.x=WEAPON_DROPS.Boss
GROUP BY BOSSES.BName;
```

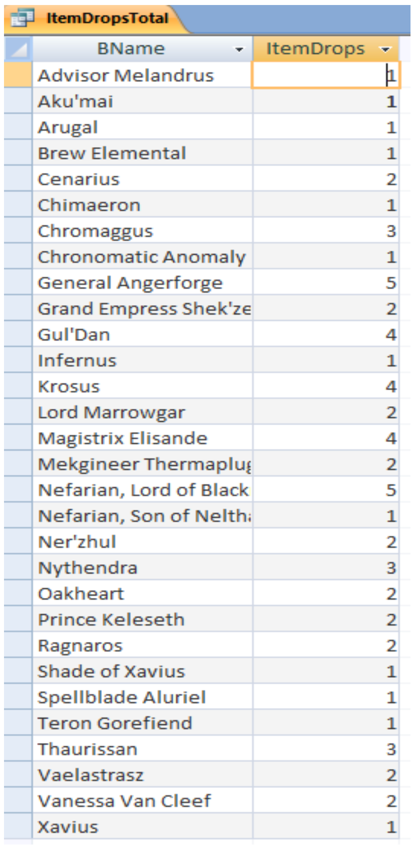
7.2.3. AllItems

```
(SELECT BOSSES.BName, WDrops.W FROM WDrops)
UNION ALL
(SELECT BOSSES.BName, ADrops.A FROM ADrops);
```

7.2.4. ItemDropsTotal

```
SELECT AllItems.BOSSES.BName, Sum(AllItems.W) AS ItemDrops
FROM AllItems
GROUP BY AllItems.BOSSES.BName;
```

The final result for the current data values is displayed in the image:



ItemDropsTotal

BName	ItemDrops
Advisor Melandrus	1
Aku'mai	1
Arugal	1
Brew Elemental	1
Cenarius	2
Chimaeron	1
Chromaggus	3
Chronomatic Anomaly	1
General Angerforge	5
Grand Empress Shek'ze	2
Gul'Dan	4
Infernus	1
Krosus	4
Lord Marrowgar	2
Magistrix Elisande	4
Mekgineer Thermaplug	2
Nefarian, Lord of Black	5
Nefarian, Son of Nelth	1
Ner'zhul	2
Nythendra	3
Oakheart	2
Prince Keleseth	2
Ragnaros	2
Shade of Xavius	1
Spellblade Aluriel	1
Teron Gorefiend	1
Thaurissan	3
Vaelastrasz	2
Vanessa Van Cleef	2
Xavius	1

7.3.

FindClassItems is a query which accepts keyboard input. The user types the name of the class he/she wants to be displayed by the query. This query is a UNION between two other subqueries, FindClassArmor (displays armor pieces usable by a class and their corresponding equipment slot) and FindClassWeapons (displays weapons usable by a class and their type). All three of these queries can be used individually, depending on the user's preference to find class-specific armor, weapons or both at the same time.

7.3.1. FindClassArmor

```
SELECT ARMORS.AName AS ItemName, ARMOR_SLOTS.ASlotName AS ItemType
FROM (ARMORS INNER JOIN ARMOR_USABLE_BY ON
ARMORS.x=ARMOR_USABLE_BY.ArmorPiece) INNER JOIN ARMOR_SLOTS ON
ARMOR_SLOTS.x=ARMORS.ASlot
WHERE Class=DLookup("x","CLASSES","ClName='" & [Enter Class Name] &
"'")
GROUP BY AName, ASlotName;
```

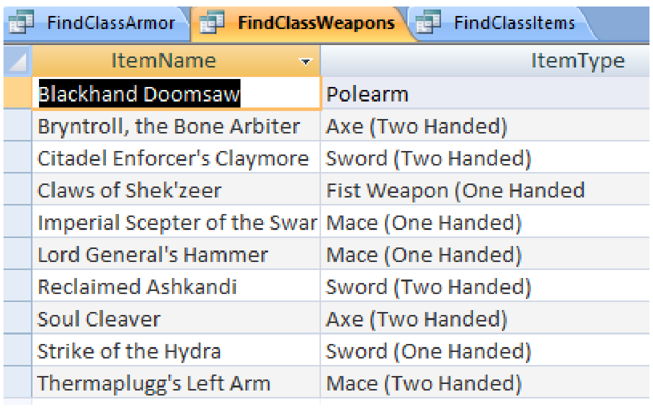
7.3.2. FindClassWeapons

```
SELECT WEAPONS.WName AS ItemName, WEAPON_TYPES.WTypeName AS ItemType
FROM (WEAPONS INNER JOIN WEAPONS_USABLE_BY ON
WEAPONS.x=WEAPONS_USABLE_BY.Weapon) INNER JOIN WEAPON_TYPES ON
WEAPON_TYPES.x=WEAPONS.WType
WHERE Class=DLookup("x","CLASSES","ClName='" & [Enter Class Name] &
"'")
GROUP BY WName, WTypeName;
```

7.3.3. FindClassItems

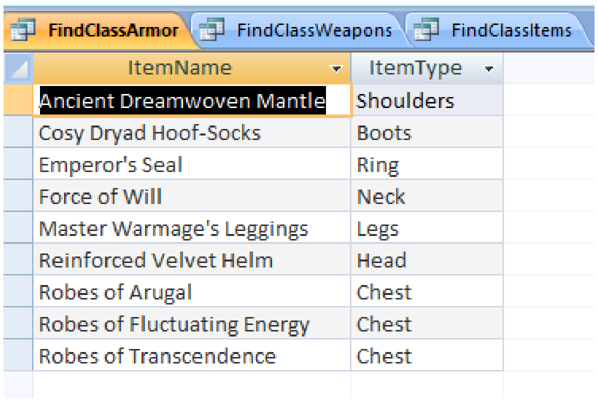
```
(SELECT FindClassWeapons.ItemName, FindClassWeapons.ItemType FROM
FindClassWeapons)
UNION (SELECT FindClassArmor.ItemName, FindClassArmor.ItemType FROM
FindClassArmor);
```

FindClassWeapons for the input: Warrior



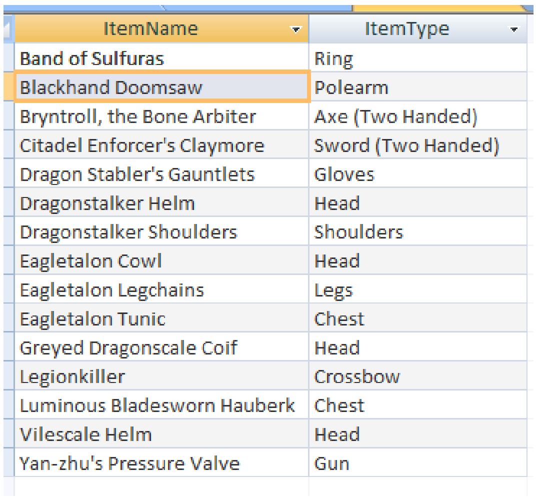
ItemName	ItemType
Blackhand Doomsaw	Polearm
Bryntroll, the Bone Arbiter	Axe (Two Handed)
Citadel Enforcer's Claymore	Sword (Two Handed)
Claws of Shek'zeer	Fist Weapon (One Handed
Imperial Scepter of the Swar	Mace (One Handed)
Lord General's Hammer	Mace (One Handed)
Reclaimed Ashkandi	Sword (Two Handed)
Soul Cleaver	Axe (Two Handed)
Strike of the Hydra	Sword (One Handed)
Thermaplugg's Left Arm	Mace (Two Handed)

FindClassArmor for the input: Priest



ItemName	ItemType
Ancient Dreamwoven Mantle	Shoulders
Cosy Dryad Hoof-Socks	Boots
Emperor's Seal	Ring
Force of Will	Neck
Master Warmage's Leggings	Legs
Reinforced Velvet Helm	Head
Robes of Arugal	Chest
Robes of Fluctuating Energy	Chest
Robes of Transcendence	Chest

FindClassItems for the input: Hunter



ItemName	ItemType
Band of Sulfuras	Ring
Blackhand Doomsaw	Polearm
Bryntroll, the Bone Arbiter	Axe (Two Handed)
Citadel Enforcer's Claymore	Sword (Two Handed)
Dragon Stabler's Gauntlets	Gloves
Dragonstalker Helm	Head
Dragonstalker Shoulders	Shoulders
Eagletalon Cowl	Head
Eagletalon Legchains	Legs
Eagletalon Tunic	Chest
Greyed Dragonscale Coif	Head
Legionkiller	Crossbow
Luminous Bladesworn Hauberk	Chest
Vilescale Helm	Head
Yan-zhu's Pressure Valve	Gun

7.4.

FindItemsFromBossesInInstance is a subquery which displays every armor piece and weapon from every boss in the instance whose name is received as a keyboard input. Its subquery, FindBosses displays only the bosses in the inputted instance.

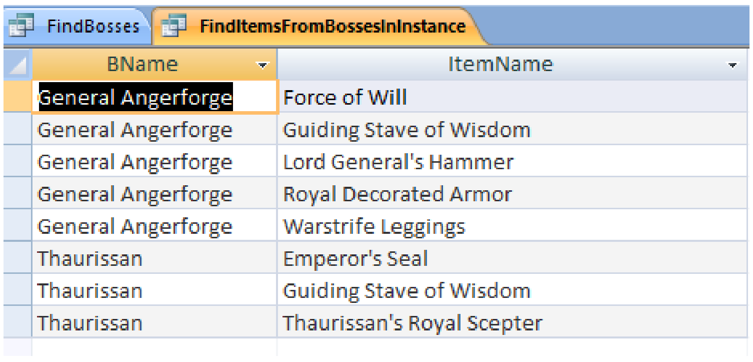
7.4.1. FindBosses

```
SELECT BOSSES.x AS Boss, BOSSES.BName
FROM BOSSES
WHERE BOSSES.Instance=DLookup("x","INSTANCES","IName='" & [Enter
Instance Name] & "'");
```

7.4.2. FindItemsFromBossesInInstance

```
(SELECT FindBosses.BName, ARMORS.AName AS ItemName
FROM (FindBosses INNER JOIN ARMOR_DROPS ON
ARMOR_DROPS.Boss=FindBosses.Boss) INNER JOIN ARMORS ON
ARMORS.x=ARMOR_DROPS.ArmorPiece)
UNION
(SELECT FindBosses.BName, WEAPONS.WName AS ItemName
FROM (FindBosses INNER JOIN WEAPON_DROPS ON
WEAPON_DROPS.Boss=FindBosses.Boss)  INNER JOIN WEAPONS ON
WEAPONS.x=WEAPON_DROPS.Weapon);
```

The final result for input: Blackrock Depths



BName	ItemName
General Angerforge	Force of Will
General Angerforge	Guiding Stave of Wisdom
General Angerforge	Lord General's Hammer
General Angerforge	Royal Decorated Armor
General Angerforge	Warstrife Leggings
Thaurissan	Emperor's Seal
Thaurissan	Guiding Stave of Wisdom
Thaurissan	Thaurissan's Royal Scepter

8. Conclusions

The universe which manages item attainment in a video game contains a database with sets which can reproduce the functionality of the in-game item system, with the help of the properties and restrictions I have enforced upon these sets. Of course, the database only handles the problem of "which items correspond to which boss". The queries I created serve to easily find items usable by a class of the user's choice and items found in a certain instance of the user's choice.