CS4416 Database project

MH4U Swords

Spring 2016

By

David Sims

Jason Fahy

Adam Doherty

**Group and Work Division**

David Sims – 14138409  
Created and worked on the Triggers and, Stored Procedures and Functions.

Jason Fahy –   
Created and worked on the Queries and Views.

Adam Doherty –  
Created and worked on the database schemas.

We all helped each other with each part and came together and wrote the report as a group.

**Database Overview**

The data in the tables we have created is based on Swords and Monsters in the game Monster Hunters 4 Ultimate (MH4U). The tables show a sample of swords and their attributes, a sample of Monsters and their attributes and the loot (Swords) which are dropped from the Monsters.

The Sample of swords which we are using have certain attributes. These attributes are ID, name, attack, special, sharpness and affinity. The swords are then broken up into three types, which are Bone, Iron, and Vile. Attack, special and affinity are all number based values, whereas sharpness is shown by two colours with a slash in the middle e.g. Green/Blue. The colours used are red, orange, yellow, green, blue, white, and purple.

C:\Users\David Sims\Downloads\Untitled Diagram (1).pngThe Sample of Monsters which we are using also have certain attributes. These attributes are ID, name, element and weakness. Element and weakness are shown by different elemental types in the game. These are fire, water, thunder, ice, dragon, paralysis, poison, sleep, and blast blight. Some of the Monsters have a chance of dropping certain swords.

This is the Entity Relation diagram for the database which we are doing. It consists of 6 tables. These are Swords, Iron, Bone, Vile, Monsters, and Drops.

Iron, Bone, and Vile all have a one to one relationship with Swords, as there cannot be duplicates.

Swords and Monsters both have a One to many or none relationship with Drops, as many monsters can drop many swords.

**Table Samples 5.**

Below are sample of the 6 tables in our database. These tables only show a sample of the tables used as the full tables. Primary Keys are underlined and in red.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Swords | |  |  | | Monsters | | | |
| Sid | Sname |  | |  | Monid | Mname | Melement | Mweakness |
| 1 | Iron Sword |  | |  | AA | Jaggi |  | Fire |
| 2 | Iron Sword+ |  | |  | AB | Uragaan | Fire | Dragon |
| 3 | Buster Sword |  | |  | AC | Khezu | Thunder | Fire |
| 4 | Buster Sword+ |  | |  | AD | Desert Seltas Queen | Water | Water |
| 5 | Ravager Blade |  | |  | AE | Rathian | Fire | Thunder |
| 6 | Ravager Blade+ |  | |  | AF | Kirin | Thunder | Fire |
| 7 | Lacerator Blade |  | |  | AG | Teostra | Fire | Water |
| 8 | Eviscerator Blade |  | |  |  |  |  |  |
| 9 | Chrome Razor |  | |  | Drops | | |  |
| 10 | Chrome Death Razor |  | |  | Sid | Monid | DropChance |  |
| 11 | Khezu Shock Sword |  | |  | 21 | AA | 60 |  |
| 12 | Khezu Shock Blade |  | |  | 20 | AB | 20 |  |
| 13 | Khezu Shock Full |  | |  | 38 | AC | 40 |  |
| 14 | Crude Cleaver |  | |  | 51 | AE | 35 |  |
| 15 | Fish Cleaver |  | |  | 11 | AG | 12 |  |
| 16 | Toad Cleaver |  | |  |  |  |  |  |
| 17 | Wild Toad Cleaer |  | |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Iron | | | | |  | Vile | | | | |
| Sid | Satt | Sspe | Sshar | Saff |  | Sid | Satt | Sspe | Sshar | Saff |
| 1 | 288 |  | Yellow/Green | 0 |  | 66 | 480 |  | Yellow/Green | 0 |
| 2 | 336 |  | Yelow/Green | 0 |  | 67 | 528 |  | Green/Green | 0 |
| 3 | 384 |  | Green/Green | 0 |  | 68 | 576 | Poisin | Green/Blue | 0 |
| 4 | 480 |  | Green/Green | 0 |  | 69 | 672 | Poisin | Blue/Blue | 0 |
| 5 | 528 |  | Green/Green | 0 |  | 70 | 720 | Poisin | Blue/Blue | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bone | | | | |
| Sid | Satt | Sspe | Sshar | Saff |
| 36 | 384 |  | Orange/Yellow | 0 |
| 37 | 432 |  | Yellow/Yellow | 0 |
| 38 | 480 |  | Yellow/Yellow | 0 |
| 39 | 528 |  | Yellow/Green | 0 |
| 40 | 576 |  | Green/Green | 0 |

Swords({Sid} → {Sname})  
Monsters({Monid} → {Mname}  
{Monid , Mname} → {Melement , Mweakness})  
Drops({Sid} → {Monid}  
{Sid , Monid} → {DropChance})  
Iron, Bone, Vile({Sid} → {Satt, Sspe, Sshar, Saff})

These tables are in 3nf because of many things. First thing in that in each table there is no duplicate data in a single row. A second reason is that there are no subsets that are recurring multiple times. A last reason is that there isn’t any data in a table that isn’t directly related to the key of the table.

**Queries and Views 7.**

All of the views that we wrote can be used for an app we are also creating for android which is using this database.

Below is the SQL for the creation of a view. The view it creates is of the swords that are dropped by monsters. This is could be used a lot especially for people looking up what weapons can be gotten from the dropped from certain monsters.

CREATE VIEW AllData AS

SELECT Sname, Sid, Mname, Monid

FROM

(

SELECT \*

FROM Swords s

RIGHT JOIN (

SELECT \*

FROM Drops d

LEFT JOIN Monsters m

WHERE d.Monid = m.Monid

) j

WHERE s.Sid = j.Sid

)

This is a query written to show the sword with the highest attack and all the swords with the same value of. This is also useful for users of the app we are creating to see what the best weapons are for them to use in the game.

SELECT Sname, Satt

FROM

(

SELECT \*

FROM Swords a

INNER JOIN Bone b

ON a.Sid = b.Sid

UNION

SELECT a.Sname i.Satt

FROM Swords a

INNER JOIN Iron i

ON a.Sid = i.Sid

UNION

SELECT a.Sname, v.Satt

FROM Swords a

INNER JOIN Vile v

ON a.Sid = v.Sid

)

HAVING Satt = MAX(Satt);

This outputs the count of the number of sords which are each sharpness. This can be used in the app the show how many of certain sharpness swords there are.

SELECT Sshar,COUNT(Sshar) AS NumberOfSharpness

FROM (

SELECT \* FROM Iron

UNION ALL

SELECT \* FROM Bone

UNION ALL

SELECT \* FROM Vile

)

GROUP BY Sshar;

The below code shows the number of monstors that don’t drop swords. Can be used for game statistics.

SELECT Count(Mname)

FROM Monsters

WHERE Monid NOT IN (SELECT Monid FROM Drops);

**Triggers and, Stored Procedures and Functions**

**Triggers**

This trigger is used for when a new drops entry is created and the sword for it isn’t in the sword table so null data is added to it. It keeps all the data in relation to each other and doesn’t have empty references.

CREATE TRIGGER NewSword

AFTER INSERT ON Drops

FOR EACH ROW BEGIN

IF NEW.Sid NOT IN (SELECT Sid FROM Swords) THEN

INSERT INTO Swords(Sid) Values(NEW.Sid);

END IF;

END;//

This trigger is used for when a sword is deleted from the main Sword table. It then precedes to delete all the data related to the sword so that there is no extra useless data on the tables.

CREATE TRIGGER DeleteSword

AFTER DELETE ON Swords

FOR EACH ROW BEGIN

IF (OLD.Sid IN (SELECT Sid FROM Iron)) THEN

DELETE FROM Iron WHERE Sid = OLD.Sid;

END IF;

IF (OLD.Sid IN (SELECT Sid FROM Bone)) THEN

DELETE FROM Bone WHERE Sid = OLD.Sid;

END IF;

IF (OLD.Sid IN (SELECT Sid FROM Vile)) THEN

DELETE FROM Vile WHERE Sid = OLD.Sid;

END IF;

IF (OLD.Sid IN (SELECT Sid FROM Drops)) THEN

DELETE FROM Drops WHERE Sid = OLD.Sid;

END IF;

End;//

**Stored Procedures and Functions**

The following procedure is used to add a new Sword to the database or edit a sword which is already in the database. It can be used to quickly keep the database up to date with swords. It is used mainly for the programmers use instead of the users.

CREATE PROCEDURE AddOrEdit (

IN SwordID INT,

IN SwordName VARCHAR(45),

IN SwordAtt INT,

IN SwordSpe VARCHAR(45),

IN SwordShar VARCHAR(45),

IN SwordAff INT

IN SwordType VARCHAR(45))

MODIFIES SQL DATA

BEGIN

DECLARE id Integer;

SELECT Sid INTO id

FROM Swords

WHERE Sid = SwordID;

START TRANSACTION

IF(id IS NULL) THEN -- SwordID doesnt already exist in system so is Added with given ID.

IF(SwordID IS NULL) THEN

INSERT INTO Swords (Sid, Sname) VALUES(LAST(Sid) +1, SwordName);

ELSE

INSERT INTO Swords (Sid, Sname) VALUES (SwordID, SwordName);

END IF;

IF(SwordType = 'Iron') THEN

INSERT INTO Iron (Sid, Satt, Sspe, Sshar, Saff) VALUES (SwordID, SwordAtt, SwordSpe, SwordShar, SwordAff);

ELSIF(SwordType = 'Bone') THEN

INSERT INTO Bone (Sid, Satt, Sspe, Sshar, Saff) VALUES (SwordID, SwordAtt, SwordSpe, SwordShar, SwordAff);

ELSE

INSERT INTO Vile (Sid, Satt, Sspe, Sshar, Saff) VALUES (SwordID, SwordAtt, SwordSpe, SwordShar, SwordAff);

END IF;

ELSE -- SwordID exists so data is updated.

UPDATE Sword

SET Sname = SwordName

WHERE Sid = SwordID;

UPDATE Iron

SET Satt = SwordID,Sspe = SwordSpe, Sshar = SwordSpe, Saff = SwordAff

WHERE Sid = SwordID;

UPDATE Bone

SET Satt = SwordID,Sspe = SwordSpe, Sshar = SwordSpe, Saff = SwordAff

WHERE Sid = SwordID;

UPDATE Vile

SET SET Satt = SwordID,Sspe = SwordSpe, Sshar = SwordSpe, Saff = SwordAff = SwordName

WHERE Sid = SwordID;

END IF;

COMMIT;

END;//

The function below is used to return a Boolean number(1= True, 0=Flase) if the given sword is dropped from the given monster. This can be used by a user of an app to quickly check this info when needed.

CREATE FUNCTION CheckDrops (

SwordID INT,

MonsterID INT,)

RETURNS TINYINT(1) DEFAULT 0

READS SQL DATA

BEGIN

DECLARE x INTEGER;

SELECT Sid INTO x

FROM Drops

WHERE Sid = SwordID AND Monid = MonsterID;

IF (x IS NULL) THEN

RETURN 0;

ELSE

RETURN 1;

END IF;

end;//