BMÜ329 Veri Tabanı Sistemleri Dersi Dönem Projesi Gereksinimleri ve E-R Diyagramı

Proje Başlığı: MMO Oyun Sistemleri İçin Veri Tabanı Mimarisi

Proje Ekibindeki kişiler: Barış Geçer , Ali Baran , Mehmet Karagülle

1- Proje Gereksinimleri

Bu proje, bir MMO (Massively Multiplayer Online) oyun için geliştirilen veritabanı modelini tanımlamaktadır. Veritabanı, oyuncuların hesap bilgilerini, karakter özelliklerini, görevleri, oyun içi varlıkları ve diğer oyun mekaniklerini organize etmek amacıyla tasarlanmıştır.

Varlıklar ve Nitelikleri

1. **Membership_Account (Üyelik Hesabı)**
   Oyuncuların hesap bilgilerini içeren bu varlık, her bir oyuncuya ait kullanıcı adı, e-posta adresi, şifre gibi bilgileri saklar. Bu bilgiler, oyuna giriş yapmak ve hesapla ilgili işlemleri gerçekleştirmek için kullanılır.

2. **Player_Character (Oyuncu Karakteri)**
   Oyuncuların oyun içinde oluşturduğu karakterlere dair bilgileri barındıran varlıktır. Her bir "Player_Character" varlığı, karakterin benzersiz bir kimliğini (CharacterID) ve adını (CharacterName) , karakterin seviyesi (Character_Level) , karakterin seviye atlaması için gereken deneyim puanı miktarı (Character_Req_Xp) , karakterin mevcut tecrübe puanı (Character_Xp) niteliklerini içerir.

3. **Attributes (Özellikler)**
   Oyuncu karakterinin çeşitli oyun içi özelliklerini (örneğin, sağlık, mana, saldırı gücü gibi) içerir. "Player_Character" ile 1:n şeklinde ilişkilendirilmiştir; böylece bir karakterin bir nitelik tablosu olabilir ve aynı değerlere sahip özellik tablosu birden fazla karakterde bulunabilir.

4. **Skills (Yetenekler)**
   Oyuncuların öğrenebileceği yetenekler burada tanımlanır. Bir "Player_Character" birden fazla yeteneğe sahip olabilir ve bu yetenekler "Skills" varlığında tutulur. "Player_Character" ve "Skills" arasında n:m ilişkisi bulunmaktadır.

5. **Proficiency (Uzmanlık)**
   Oyuncu karakterlerinin belirli alanlardaki ustalık seviyelerini gösterir. Bu varlık, oyuncuların zamanla belirli alanlarda uzmanlaşmasını sağlar ve bu yeteneklerin gelişimini takip eder. Her "Player_Character" bir veya birden fazla "Proficiency" ile ilişkilendirilmiştir (n:m ilişkisi).

6. **Inventory (Envanter)**
   Oyuncuların oyun içinde sahip oldukları eşyaları organize eden "Inventory" varlığı oyuncuların taşıyabileceği eşya sayısına limit koyar (Capacity) Her "Player_Character" bir envantere ve bu envanterde bir veya birden fazla eşyaya sahip olabilir. "Player_Character" ve Inventory arasında 1:1 ilişkisi vardır, böylece bir oyuncu karakteri sadece bir envantere sahip olur

7. **Items (Eşyalar)**
Oyuncuların oyun içerisinde sahip oldukları eşyaları temsil eder. Oyuncuların eşyaları kullabilme yetisine seviye kısıtlaması getirilmiştir (Req_Level) oyun içindeki farklı eşya tipleri birbirinden kendine özgü olan numarası ile ayrılır (Item_ID) ve eşyalar bu numaraya göre sınıflanır örneğin iki farlı çeşit kılıç biri 1.1 iken diğeri 1.2 olarak numaralandırılır burada 1 kılıçların numarası noktadan sonrası ise eşyanın ayırt edicisidir Envanter ve Eşyalar n:n ilişkisine sahiptir

8. **Quests (Görevler)**
   Oyuncuların oyunda alabileceği görevleri temsil eder. Her görev, görev adı (Quest_Name), açıklaması(Decription), görev ödülü (Reward_Amount) gibi nitelikleri içerir. "Player_Character" ve "Quests" arasında n:m
ilişkisi bulunmaktadır; yani bir görev birden fazla karakter tarafından yapılabilir ve bir karakter birden fazla görevi üstlenebilir.

9. **Monsters (Canavarlar)** ve **NPCs (Oyuncu Olmayan Karakterler)**
   Oyun dünyasında bulunan canavarlar ve NPC'ler (Non-Player Characters) bu varlıklarla temsil edilir. Her bir NPC, oyunun hikayesi veya oyuncuların ilerlemesi için belirli işlevlere sahiptir. "Quests" ile "NPC's" arasında 1:n ilişkisi vardır .Bu sayede bir görevde bir NPC yer alırken bir görev de birden fazla NPC yer alabilir.Canavarlar ise oyuncunun görev sırasında karşılaşacığı zorluklardır eşyalarda olan sınıflandırma sistemi canavarlar için de geçerlidir

10. **Factions (Gruplar)**
    Oyun içindeki grupları ve oyuncuların bu gruplara katılımlarını tanımlayan varlıkdır. "Factions" ile "Player_Character" arasında n:1 ilişkisi vardır; yani her oyuncu yalnızca bir gruba katılabilir ve her grup birden fazla karaktere sahip olabilir.

11. **Achievements (Başarımlar)**
    Oyuncuların oyun boyunca kazandıkları ödülleri ve başarıları içerir. "Membership_Account" ve "Achievements" arasında n:m

ilişki vardır; her üyelik hesabında birden fazla başarıma sahip olabilir.Bir başarım birden fazla üyelik hesabında olabilir.

12. **Authority(Yetki)**

Oyunlardaki oyuncuları denetleyen duruma göre oyuna müdahale eden yetkili görevli.”Authority” yetkisine sahip kişilier geçici sürelik yasaklama olsun,oyundan atma olsun o tarz yetkilerin bulunduğu bir veritabanı tablosudur.”Authority” ile “Membership_Account” arasında (1:n) ilişki vardır bir yetkilinin birden fazla hesabı olabilir ama bir hesabın tek yetkisi olabilir

13.**Server(Sunucu)**

Oynanan oyundaki üye hesaplarının yer aldığı çok fazla kişinin aynı ortamda bulunup oynadığı sistematik yapıya sunucu deriz.Sunucuların kendisi ”server_id” leri “server_ip “leri “server_location “ gibi nitelikleri bulunur .Server ile Membership_Account arasında da (1:n) ilişki vardır bir sunucuda birden fazla üye hesabı olurken bir üye hesabı tek server da yer alır .

14.**Appearance(Dış Görünüş)**

Bir karakterin dıştan görünümü sağlayan özellikleri dış görünüş tablosundaki nitelikler ile belirliyoruz.Bir karakterin saç rengi,göz rengi,kas tipi gibi nitelikleri bulunur.Player_Character ile Appearance arasında (n:1)ilişki vardır. Bir karakterin bir dış görünüş olabilir bir dış görünüş birden fazla karakterde olabilir

İlişkiler ve Sayısal Kısıtlamalar

- **Membership_Account - Player_Character**: 1:n

Bir "Player_Account" (oyuncu hesabı) birden fazla "Player_Character" (oyuncu karakteri) oluşturabilir, ancak her karakter yalnızca bir hesap ile ilişkilidir.

- **Attributes – Player_Character**: 1:n

Bir "Player_Character" bir "Attributes" (özellik tablosu) değerine sahip olabilir, aynı zamanda bir özellik tablosu birden fazla karakterde olabilir.

- **Player_Character - Skills**: n:n

Bir "Player_Character" birden fazla "Skills" (yetenek) öğrenebilir ve aynı yetenek birden fazla karakter tarafından kullanılabilir.

- **Player_Character - Proficiency**: n:n

Her "Player_Character" birden fazla "Proficiency" (uzmanlık) alanında beceri sahibi olabilir.Bir uzmanlık alanında da birden fazla karakter sahip olabilir.

- **Player_Character - Inventory - Items**: 1:1 | 1:n

Her "Player_Character" bir envantere sahiptir ve bu envanterde birden fazla "Items" (eşya) bulunabilir

- **Player_Character - Quests**: n:m

Bir "Player_Character" birden fazla "Quests" (görev) alabilir ve aynı görev birden fazla karakter tarafından yapılabilir.

- **Quests - NPCs**: n:1

Her görevde bir "NPCs" (oyuncu olmayan karakter) olurken her NPC ise birden fazla görev verebilir

- **Player_Character – Factions:** n:1

Bir "Player_Character" birden fazla "Factions" (grup) üyesi olamaz ve aynı grup birden fazla karaktere sahip olabilir.

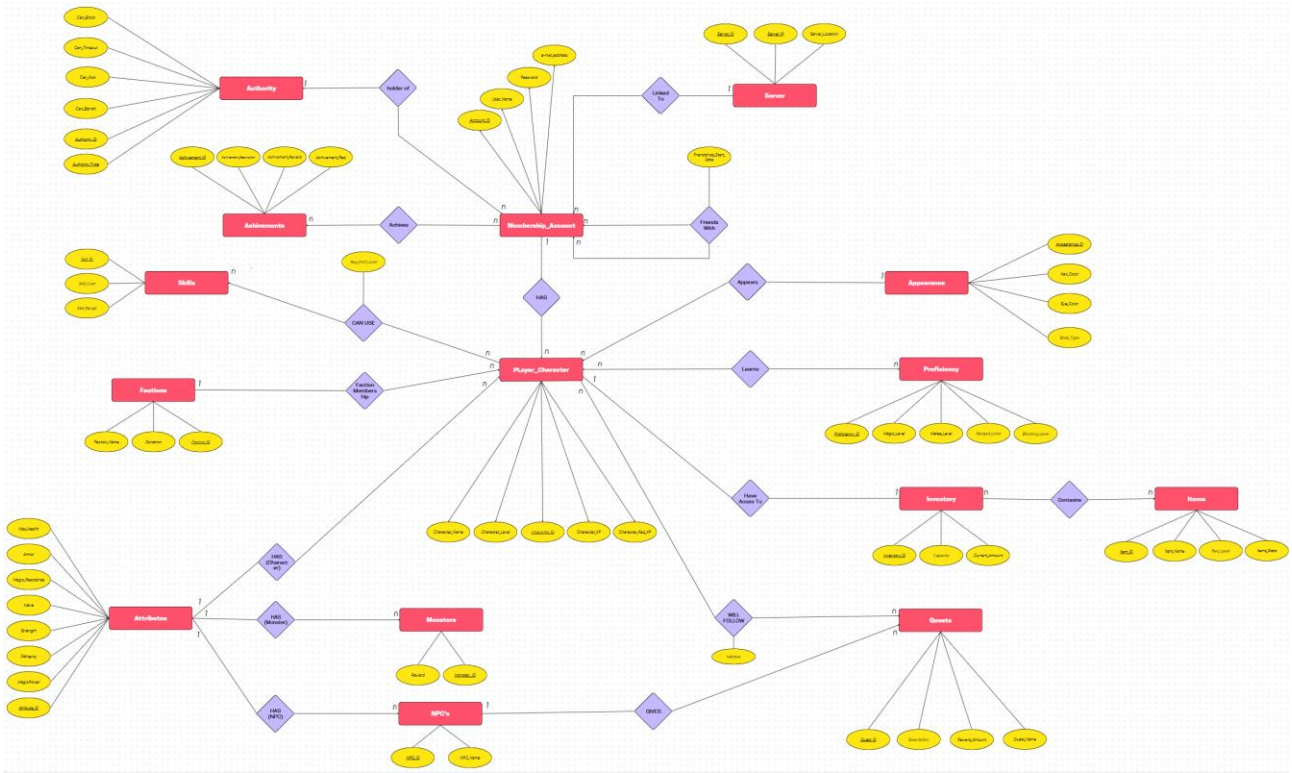
- **Membership_Account - Membership_Account**: n:m

Bir "Player_Account" (oyuncu hesabı) bir veya birden fazla "Friends" (arkadaş) ilişkisine sahip olabilir.

- **Membership_Account - Achievements**: m:n

Her "Player_Account" birden fazla "Achievements" (başarım) kazanabilir, ancak her başarı bir hesaba aittir

2 – E-R Diyagramı



// TABLOLARI OLUŞTURMA

```
--Table: Appearance  <--Appears--> Player_Character
Create Table Appearance(
Appearance_ID int PRIMARY KEY  not null,
Hair_Color NVARCHAR(30) default 'Black' NOT NULL,
Eye_Color NVARCHAR(30) default 'Brown' NOT NULL,
Body_Type NVARCHAR(50) NOT NULL,
CHECK(Body_Type = 'Masculine' or Body_Type = 'Feminine')
);

--Table: Inventory  <--Have Acces To--> Player_Character
create Table Inventory(
Inventory_ID int PRIMARY KEY NOT NULL,
Capacity int default 50 not null,
Current_Amount int not null
CHECK (Capacity > 0)
);

--Table: Items
create Table Items(
Item_ID int PRIMARY KEY NOT NULL,
Item_Name NVARCHAR(255) NOT NULL,
Req_Level int default 0 NOT NULL,
Item_Stats int default 0 NOT NULL,

CHECK( Req_Level <= 255)
);

--Table : Items <-Contains-> Inventory
create Table Contains_(
Inventory_ID int not null,
FOREIGN KEY (Inventory_ID) REFERENCES Inventory(Inventory_ID),
Item_ID int not null,
FOREIGN KEY (Item_ID) references Items(Item_ID),
PRIMARY KEY(Inventory_ID,Item_ID),
);

--TABLE: Attributes
create Table Attributes(
Attribute_ID int PRIMARY KEY  not null,
MagicPower int default 10 not null,
Dexterity int default 10 not null,
Strength int default 10 not null,
Mana int default 10 not null,
Magic_Resistanse int default 10 not null,
Armor int  default 10 not null,
Max_Health int default 100 not null,
);

--TABLE : FACTIONS

create Table Factions(
Faction_ID int PRIMARY KEY NOT NULL,
Donation int not null,
Faction_Name NVARCHAR(255) NOT NULL,
);

--TABLE:AUTHORİTY
Create Table Authority(
Authority_Type NVARCHAR(10) NOT NULL,
Authority_ID int not null,
```

```
PRIMARY KEY (Authority_Type,Authority_ID),
Can_Banish BIT default 0 NOT NULL,
Can_Kick BIT default 0 NOT NULL,
Can_Timeout BIT default 0 NOT NULL,
Can_Block BIT default 0 NOT NULL,
CHECK(Authority_Type = 'Player' or Authority_Type = 'GM' or Authority_Type = 'Admin' or
Authority_Type = 'Executive')
);

--TABLE:SERVER
create Table Server(
Server_ID int  NOT NULL,
Server_IP varchar(20) Not NULL,
PRIMARY KEY(Server_ID,Server_IP),
Server_Location NVARCHAR(255) NOT NULL,
);


--TABLE:Membership_Account
create Table Membership_Account(
Account_ID int PRIMARY KEY NOT NULL,
User_Name NVARCHAR(255) NOT NULL,
Password NVARCHAR(255) NOT NULL,
email_Address NVARCHAR(255) NOT NULL,

Authority_Type NVARCHAR(10) not null,
Authority_ID int not null,
FOREIGN KEY(Authority_Type,Authority_ID) REFERENCES Authority(Authority_Type,Authority_ID),

Server_ID int  NOT NULL,
Server_IP varchar(20) Not NULL,
FOREIGN KEY (Server_ID,Server_IP) REFERENCES Server(Server_ID,Server_IP),

);

-- Membership_Account <--- Friends_with ---> Membership_Account
CREATE TABLE Friends_with (
    account_id_1 INT,
    account_id_2 INT,
       Friendship_Start_Date date,
    PRIMARY KEY (account_id_1, account_id_2),
    FOREIGN KEY (account_id_1) REFERENCES membership_account(Account_ID),
    FOREIGN KEY (account_id_2) REFERENCES membership_account(Account_ID),
    CHECK (account_id_1 != account_id_2)  -- Prevents self-friendship
);

-- Main TABLE: Player_Character
Create Table Player_Character(
 Character_ID int PRIMARY KEY Not null,
 Character_Name NVARCHAR(32) NOT NULL,
 Character_Level int default 0 Not Null,
 Character_XP int not null,
 Character_Req_XP int not null,

 Appearance_ID int not null,
 FOREIGN KEY (Appearance_ID) REFERENCES Appearance(Appearance_ID),
 Inventory_ID int not null,
 FOREIGN KEY (Inventory_ID) REFERENCES Inventory(Inventory_ID),
 Attribute_ID int not null,
 FOREIGN KEY (Attribute_ID) REFERENCES Attributes(Attribute_ID),
 Faction_ID int,
 FOREIGN KEY (Faction_ID) REFERENCES Factions(Faction_ID),
 Account_ID int not null,
 FOREIGN KEY (Account_ID) REFERENCES Membership_Account(Account_ID),
```

```
 CHECK(Character_Level <= 255 and Character_Level >= 0 )
);

-- Table: Proficiency
Create Table Proficiency(
Proficiency_ID int PRIMARY KEY NOT NULL,
Magic_Level int default 0 Not null,
Melee_Level int default 0 Not null,
Ranged_Level int default 0 Not null,
Blocking_Level int default 0 Not null

CHECK(Magic_Level <= 100 and Melee_Level <= 100 and Ranged_Level <= 100 and Blocking_Level
<= 100 and
              Magic_Level >= 0 and Melee_Level >= 0 and Ranged_Level >= 0 and
Blocking_Level >= 0)
);

--Table: Player_Character <---Learns---> Proficiency
Create Table Learns(
Character_ID int not null,
FOREIGN KEY(Character_ID) REFERENCES Player_Character(Character_ID),
Proficiency_ID int not null,
FOREIGN KEY(Proficiency_ID) REFERENCES Proficiency(Proficiency_ID),
PRIMARY KEY(Character_ID,Proficiency_ID)
);

--Table : NPC
create Table NPC(
NPC_ID int PRIMARY KEY NOT NULL,
NPC_Name NVARCHAR(64),
Attribute_ID int not null,
FOREIGN KEY(Attribute_ID) REFERENCES Attributes(Attribute_ID),
);

--Table: Quests
create Table Quests(
Quest_ID int PRIMARY KEY NOT NULL,
Description NVARCHAR(255) default NULL,
Reward_Amount int NOT NULL,
Quest_Name NVARCHAR(255) NOT NULL,

NPC_ID int not null,
FOREIGN KEY (NPC_ID) REFERENCES NPC(NPC_ID),
);

--Table: Player_Character <-WILL_FOLLOW-> Quests
create Table WILL_FOLLOW(
Character_ID int not null,
FOREIGN KEY (Character_ID) references Player_Character(Character_ID),
Quest_ID int NOT NULL,
FOREIGN KEY (Quest_ID) REFERENCES Quests(Quest_ID),
PRIMARY KEY (Character_ID,Quest_ID),
isActive BIT default 0 not null,
);

--Table : Monsters
create Table Monsters(
Monster_ID int PRIMARY KEY NOT NULL,
Monster_Reward int not null,

Attribute_ID int not null,
FOREIGN KEY(Attribute_ID) REFERENCES Attributes(Attribute_ID),
);

--TABLE : Skills
```

```
create Table Skills(
Skills_ID int PRIMARY KEY NOT NULL,
Skill_Cost int not null,
Skill_Range  int default 1 not null,
);

--Table : Skills <--CAN USE--> Player_Character
create Table CanUse(
Skills_ID int not null,
FOREIGN KEY (Skills_ID) REFERENCES Skills(Skills_ID),
Character_ID int not null,
FOREIGN KEY (Character_ID) REFERENCES Player_Character(Character_ID),
Req_Prof_Level int,
PRIMARY KEY(Character_ID,Skills_ID),
);

--TABLE:ACHİVEMENTS

create Table Achivements(
Achivement_ID int PRIMARY KEY NOT NULL,
Achivement_Description NVARCHAR(255) default NULL,
Achivement_Reward int not null,
Achivement_Req int not null,
);

--Table: Achivements <--Achives--> Membership_Account

create Table Achives(
Achivement_ID int NOT NULL,
FOREIGN KEY(Achivement_ID) REFERENCES Achivements(Achivement_ID),
Account_ID int not null,
FOREIGN KEY(Account_ID) REFERENCES Membership_Account(Account_ID),
PRIMARY KEY (Achivement_ID,Account_ID),
);
```

//VERİLERİ YERLEŞTİRME

```
insert into
Achivements(Achivement_ID,Achivement_Description,Achivement_Reward,Achivement_Req)
values (1,'Play for the first time',1455,1300)
insert into Achivements values(2,'defeat a monster',1275,481)
insert into Achivements values(3,'complete a quest',1320,1100)
insert into Achivements values(4,'Level up for the first time',1469,890)
insert into Achivements values(5,'use a skill 100 times',1890,440)
insert into Achivements values(6,'defat a moster without other noticing',2100,560)
insert into Achivements values(7,'Demolish bandit hideout',2300,570)
```

```
insert into Achives (Achivement_ID,Account_ID)
values(2,1)
insert into Achives values(2,2)
insert into Achives values(2,3)
insert into Achives values(5,4)
insert into Achives values(6,4)
insert into Achives values(7,4)
```

```
-------------------------------------------------------------
insert into Appearance(Appearance_ID,Hair_Color,Eye_Color,Body_Type)
values(16,'Brown','Blue','Masculine')
insert into Appearance values(17,'Black','Black','Masculine')
insert into Appearance values(18,'Black','Brown','Feminine')
-------------------------------------------------------------



-- pirmary key = row value + first digit of each attribute
insert into Attributes(Attribute_ID , MagicPower , Dexterity , Strength , Mana ,
Magic_Resistanse , Armor , Max_Health)
values(04241564,45,23,41,150,57,61,495)
insert into attributes values (14241513,45,23,41,150,54,10,395)
insert into attributes values (21111214,10,14,13,123,23,14,400)
insert into attributes values (32431452,23,43,32,112,43,56,298)
insert into attributes values (42149153,20,13,41,90,10,52,345)
insert into attributes values (51147333,13,14,43,78,34,32,330)
insert into attributes values (61451533,16,43,51,139,50,37,397)
insert into attributes values (73334422,35,31,32,49,43,28,289)
insert into attributes values (82434424,24,42,31,43,47,27,425)
insert into attributes values (91142133,12,11,45,240,18,36,390)
insert into attributes values (103142311,32,10,41,200,35,19,145)
-------------------------------------------------------------


insert into Authority(Authority_Type , Authority_ID , Can_Banish , Can_Kick , Can_Timeout ,
Can_Block)
values ('Admin',0,1,1,1,1)
insert into Authority values('Player',1,0,0,0,1)
insert into Authority values('GM',2,0,1,1,1)
insert into Authority values('Player',3,0,1,1,1)
insert into Authority values('Executive',4,1,1,1,1)
-------------------------------------------------------------


insert into CanUse (Skills_ID,Character_ID,Req_Prof_Level)
values(8,14,23)
insert into CanUse values(7,15,12)
-------------------------------------------------------------


insert into Contains_ (Inventory_ID,Item_ID)
values(2,1232)
insert into Contains_ values(2,1342)
insert into Contains_ values(2,2556)
insert into Contains_ values(4,3234)
insert into Contains_ values(6,5042)
insert into Contains_ values(3,3112)

-------------------------------------------------------------

insert into Factions(Faction_ID,Donation,Faction_Name)
values(0,0,'DRAGONX')
insert into Factions values(1,7300,'GPEX')

-------------------------------------------------------------

insert into Friends_with (account_id_1,account_id_2,Friendship_Start_Date)
values(4,3,GETDATE())

-------------------------------------------------------------

insert into Inventory (Inventory_ID,Capacity,Current_Amount)
```

```
values (0,50,15)
insert into Inventory values(1,50,50)
insert into Inventory values(2,50,14)
insert into Inventory values(3,50,23)
insert into Inventory values(4,50,24)
insert into Inventory values(5,50,31)
insert into Inventory values(6,60,42)

-----------------------------------------------------------------

-- primary key = starts with 1 if melee , 2 if ranged , 3 if shield , 4 if consumable , 5 if
pet , 6 if armor
-- + type number  + item place

-- melee types
-- 0 if sword , 1 if greatsword , 2 if polearm , 3 if dagger , 4 if mace , 5 if hammer
-- ranged types
-- 0 if short-bows , 1 if long-bows , 2 if crossbows , 3 if pistols , 4 if wands , 5 if
staff , 6 if darts
-- shield types
-- 0 if kite-shield , 1 if buckler , 2 if tower-shield, 3 if parrying dagger
-- consumable types
-- 0 if Healt-Potions , 1 if Mana-Potions , 2 if buff-potions , 3 if scrolls , 4 if food
-- pet types
-- 0 if mount , 1 if not
-- armor types
-- 0 if medium-grade , 1 if light-grade , 2 if heavy-grade ,3 if superheavy-grade ,4 if
superlight-grade
insert into Items (Item_ID,Item_Name,Req_Level,Item_Stats)
values(100,'Novice Blade',25,16)
insert into Items values(101,'Rusted Sword',27,19)
insert into Items values(1232,'glaive of the bandit lord',27,19)
insert into Items values(1342,'fang of the mountain lord',27,19)
insert into Items values(1426,'skull crusher',27,19)
insert into Items values(1574,'valley creator',27,19)
insert into Items values(2017,'short bow of the mounted warior',27,19)
insert into Items values(2556,'staff of necromancy',27,19)
insert into Items values(2127,'WindWisperer',27,19)
insert into Items values(222,'standart crossbow',27,19)
insert into Items values(3112,'steel buckler',27,19)
insert into Items values(3234,'BoneMass',27,19)
insert into Items values(333,'dull dagger',27,19)
insert into Items values(402,'lesser healing potion',27,19)
insert into Items values(405,'greater healing potion',27,19)
insert into Items values(412,'lesser mana potion',27,19)
insert into Items values(432,'Scroll of Idetification',27,19)
insert into Items values(500,'Standart War Horse',27,19)
insert into Items values(5042,'flash the sky soverign',27,19)
insert into Items values(5112,'Hunter Dof',27,19)
insert into Items values(600,'standart issue chain mail',27,19)
insert into Items values(626,'iron plate armor',27,19)
insert into Items values(6345,'Drake Scale armor',27,19)


-----------------------------------------------------------------

insert into Learns (Character_ID,Proficiency_ID)
values(14,11)

insert into Learns values(15,12)

-----------------------------------------------------------------

insert into Membership_Account(Account_ID , User_Name , Password , email_Address ,
Authority_Type , Authority_ID , Server_ID , Server_IP)
```

```
values(0,'Yasaki','295113','aktar25@gmail.com','Player',1,0,'192.168.1.1')
insert into Membership_Account
values(1,'Karina_32','A!_ammfae','asd@gmail.com','Player',1,0,'192.168.1.1')
insert into Membership_Account
values(2,'MASKLESS','B%SNgAc','abc@gmail.com','Player',1,0,'192.168.1.1')
insert into Membership_Account
values(3,'DarkKnight','Jkjncw10','jkl@gmail.com','Admin',0,0,'192.168.1.1')
insert into Membership_Account
values(4,'DE4thIsC0mming','9eEfnv!#','xyz@gmail.com','GM',3,0,'192.168.1.1')

-----------------------------------------------------------------


insert into Monsters(Monster_ID,Monster_Reward,Attribute_ID)
values(16,210,21111214)
insert into Monsters values(17,3200,51147333)

-----------------------------------------------------------------

insert into NPC (NPC_ID,NPC_Name,Attribute_ID)
values(0,'Fisherman',82434424)
insert into NPC values(1,'blacksmith',91142133)

-----------------------------------------------------------------

insert into
Player_Character(Character_ID,Character_Name,Character_Level,Character_XP,Character_Req_XP,A
ppearance_ID,Inventory_ID,Attribute_ID,Faction_ID,Account_ID)
values(14,'Black Widow',26,5000,7500,16,0,91142133,NULL,3)
insert into Player_Character values(15,'Dare Devil',72,3000,4500,17,1,82434424,1,2)
insert into Player_Character values(16,'BigGuy',46,3000,4500,17,1,91142133,NULL,2)
insert into Player_Character values(17,'MisterHammer',52,3000,4500,18,2,14241513,NULL,5)

-----------------------------------------------------------------

insert into Proficiency (Proficiency_ID,Magic_Level,Melee_Level,Ranged_Level,Blocking_Level)
values(11,23,25,40,30)
insert into Proficiency values(12,24,16,17,35)

-----------------------------------------------------------------


insert into Quests(Quest_ID,Description,Reward_Amount,Quest_Name,NPC_ID)
values(20,NULL,2460,'Avalanche',0)

insert into Quests values(21,NULL,2970,'journey throug sea',1)

-----------------------------------------------------------------

insert into Server(Server_ID,Server_IP,Server_Location)
values(0,'192.168.1.1','E WEST')
insert into Server values(1,'192.168.1.1','Turkey')

-----------------------------------------------------------------

insert into Skills(Skills_ID,Skill_Cost,Skill_Range)
values(9,173,54)
insert into Skills values(10,85,19)

-----------------------------------------------------------------

insert into WILL_FOLLOW(Character_ID,Quest_ID,isActive)
values(14,20,1)
insert into WILL_FOLLOW values(15,21,0)
```

//SAKLI YORDAMLAR

```
create procedure isQuestActive
@Quest_ID varchar(50),
@isactive BIT
AS
Begin
select*from Quests where  Quest_ID = @Quest_ID and isActive=@isactive;
End;
```

```
CREATE PROCEDURE GetCharacterInventory
    @id VARCHAR(10)
AS
BEGIN

    SELECT C.Item_ID, I.Item_Name,Req_Level,Item_Stats
    FROM Contains_ C
    INNER JOIN Items I ON C.Item_ID = I.Item_ID
    WHERE C.Inventory_ID = @id;
END;
```

```
CREATE PROCEDURE GetBetweenTwoLevels
    @lwl1 INT,
    @lwl2 INT
AS
BEGIN
    SELECT
              pc.Character_ID,
                 pc.Character_Name,
              p.Proficiency_ID,
           p.Magic_Level,
           p.Melee_Level,
           p.Ranged_Level,
           p.Blocking_Level
    FROM Learns l
    INNER JOIN Proficiency p ON l.Proficiency_ID = p.Proficiency_ID
    INNER JOIN Player_Character pc ON l.Character_ID = pc.Character_ID
      where pc.Character_Level between @lwl1 and @lwl2
      End;
```

```
CREATE PROCEDURE GetMemberCount
    @Faction_id INT -- Aranacak klan ID'si
AS
BEGIN
    DECLARE @PlayerCount INT;
```

```
    SELECT @PlayerCount = COUNT(*)
    FROM Player_Character
    WHERE Faction_ID = @Faction_id;


    RETURN @PlayerCount;
END;
```

```
create procedure GetMembersFromFactionID
@Faction_Name varchar(25)
as
begin
select f.Faction_ID, f.Faction_Name, f.Donation,p.Account_ID, p.Character_Name,
p.Character_Level, p.Appearance_ID, p.Attribute_ID, p.Character_ID
from  Factions f inner join Player_Character p on f.Faction_ID = p.Faction_ID
where f.Faction_Name = @Faction_Name;
End;
```

// AKTARIMLAR

```
CREATE PROCEDURE LevelUpCharacter
    @CharacterID INT
AS
BEGIN
    BEGIN TRANSACTION;

    BEGIN TRY

        DECLARE @CurrentXP INT, @RequiredXP INT, @Level INT;
        DECLARE @AttributeID INT;

        -- Karakter verilerini al
        SELECT Character_XP, Character_Req_XP, Character_Level, Attribute_ID
        INTO @CurrentXP, @RequiredXP, @Level, @AttributeID
        FROM Player_Character
        WHERE Character_ID = @CharacterID;

             if @CurrentXP < @RequiredXP
        BEGIN
            PRINT 'Character does not have enough XP to level up.';
        END;

             else
             BEGIN
        -- Level atladı mı?
                    WHILE @CurrentXP >= @RequiredXP
                    BEGIN
                           @LevelUpAmount = @LevelUpAmount + 1;
                           SET @CurrentXP = @CurrentXP - @RequiredXP; -- XP'yi gerekli
miktar kadar azalt
                           SET @Level = @Level + 1;                   -- level artışı
                           SET @RequiredXP = @RequiredXP + (@RequiredXP / 2); -- Gereken XP
yi 1.5 katına çıkar

                           -- ödül olarak nitelikleri güncelle
                           UPDATE Attributes
                           SET
```

```
                            Max_Health = Max_Health + 10),
                        Strength = Strength + 2,
                    MagicPower = MagicPower + 2,
                               Armor = Armor + 1
                        WHERE Attribute_ID = @AttributeID;
                        END
             END
        -- Karakter bilgilerini güncelle
        UPDATE Player_Character
        SET
            Character_XP = @CurrentXP,
            Character_Level = @Level,
            Character_Req_XP = @RequiredXP
        WHERE Character_ID = @CharacterID;

   -- Enhance attributes as a reward for leveling up
            UPDATE Attributes

            SET
                Max_Health = Max_Health + 100,
                MagicPower = Magic_Resistanse + 2,
                           Dexterity = Dexterity + 2,
                           Strength = Strength + 2,
                           mana = mana + 10

                    WHERE Attribute_ID = @AttributeID;

            PRINT 'Karakter başarıyla seviye atladı';

        -- Commit the transaction if all operations succeed
        COMMIT TRANSACTION;

    CATCH

    BEGIN CATCH
        -- Rollback the transaction in case of any errors
        ROLLBACK TRANSACTION;
        PRINT 'seviye atlarken bir hata gerçekleşti';

    END CATCH
END;



-----------------------------------------------

CREATE PROCEDURE CompleteQuestAndAwardXP
    @CharacterID INT,
    @QuestID INT
AS
BEGIN
    BEGIN TRANSACTION;

    BEGIN TRY

        DECLARE @CurrentXP INT, @RequiredXP INT, @Level INT, @RewardXP INT;
        DECLARE @AttributeID INT;

        --Görev ödülünü çek
        SELECT Reward_Amount
        INTO @RewardXP
        FROM Quests
        WHERE Quest_ID = @QuestID;

        IF @RewardXP IS NULL
```

```
        BEGIN
            PRINT 'Görev bulunamadı';
        END

        -- Karakter bilgilerini çek
        SELECT Character_XP, Character_Req_XP, Character_Level, Attribute_ID
        INTO @CurrentXP, @RequiredXP, @Level, @AttributeID
        FROM Player_Character
        WHERE Character_ID = @CharacterID;

        IF @CurrentXP IS NULL
        BEGIN
            PRINT 'Karakter bulunamadı';
        END

        -- Karakter XP sini güncelle
        SET @CurrentXP = @CurrentXP + @RewardXP;

        -- Level atladı mı?
        WHILE @CurrentXP >= @RequiredXP
        BEGIN
            SET @CurrentXP = @CurrentXP - @RequiredXP; -- XP'yi gerekli miktar kadar azalt
            SET @Level = @Level + 1;                   -- level artışı
            SET @RequiredXP = @RequiredXP + (@RequiredXP / 2); -- Gereken XP yi 1.5 katına
çıkar

            -- ödül olarak nitelikleri güncelle
            UPDATE Attributes
            SET
                Max_Health = Max_Health + 10,
                Strength = Strength + 2,
                MagicPower = MagicPower + 2,
                Armor = Armor + 1
            WHERE Attribute_ID = @AttributeID;
        END

        -- Karakter bilgilerini güncelle
        UPDATE Player_Character
        SET
            Character_XP = @CurrentXP,
            Character_Level = @Level,
            Character_Req_XP = @RequiredXP
        WHERE Character_ID = @CharacterID;

        -- görevi tamamlanmış olarak işaretle
        UPDATE WİLL_FOLLOW
        SET isActive = 0
        WHERE Character_ID = @CharacterID AND Quest_ID = @QuestID;

        -- Her şey başarılı ise değişiklikleri gerçekleştir
        COMMIT TRANSACTION;

        PRINT 'XP başarıyla güncellendi';
    CATCH
    BEGIN CATCH
        -- Hata olursa eski durumuna dön
        ROLLBACK TRANSACTION;

        PRINT 'Ödülü verirken bir sorun ile karşılaşıldı';

    END CATCH
END;
```

```
------------------------------------------------------------

CREATE PROCEDURE RemoveItemFromInventory
    @InventoryID INT,
    @ItemID INT,
    @QuantityToRemove INT
AS
BEGIN
    BEGIN TRANSACTION;

    BEGIN TRY

          DECLARE @CurrentAmount INT;

        -- Item envanterde var mı?
        SELECT Current_Amount
        INTO @CurrentAmount
        FROM Contasins
        WHERE Inventory_ID = @InventoryID AND Item_ID = @ItemID;

        IF @CurrentAmount IS NULL
        BEGIN
            PRINT 'Item envanterde bulunamadı';
        END

        -- Kaldırmak için yeterli sayıda item var mı?
        IF @QuantityToRemove > @CurrentAmount
        BEGIN
            PRINT 'Çıkarmak için gereken sayıda eşya yok';
        END

        -- Envanterden Eşya kaldır veya güncelle
        IF @QuantityToRemove = @CurrentAmount
        BEGIN
            -- eşya sayısı kaldırılacak ile aynı ise eşyayı envanterden kaldır
            DELETE FROM Contasins
            WHERE Inventory_ID = @InventoryID AND Item_ID = @ItemID;
        END
        ELSE
        BEGIN
            -- Item sayısını azalt
            UPDATE Contasins
            SET Current_Amount = @CurrentAmount - @QuantityToRemove
            WHERE Inventory_ID = @InventoryID AND Item_ID = @ItemID;
        END

        -- Mevcut miktarı güncelle
        UPDATE Inventory
        SET Current_Amount = Current_Amount - @QuantityToRemove
        WHERE Inventory_ID = @InventoryID;

        -- Aktarımı gerçekleştir
        COMMIT TRANSACTION;

        PRINT 'Item başarıyla envanterden kaldırıldı';
    CATCH
    BEGIN CATCH
        -- Hata olduysa eski haline geri getir
        ROLLBACK TRANSACTION;

          PRINT 'Itemi envanterden kaldırırken bir hatayla karşılaşıldı';

      END CATCH
```

```
END;
-----------------------------------------------------------------------

CREATE PROCEDURE UnlockSkillForCharacter
    @CharacterID INT,
    @SkillID INT
AS
BEGIN
    BEGIN TRANSACTION;

    BEGIN TRY

        DECLARE @ProficiencyID INT, @CurrentProficiencyLevel INT, @RequiredProficiencyLevel
INT;

        -- skill için gereken levli çek
        SELECT Req_Prof_Level
        INTO @RequiredProficiencyLevel
        FROM Skills
        WHERE Skills_ID = @SkillID;

        IF @RequiredProficiencyLevel IS NULL
        BEGIN
            PRINT 'Böyle bir skill yok';
        END

        -- Karakterin Ustalık seviyesini çek
        SELECT Learns.Proficiency_ID, Proficiency.Magic_Level
        INTO @ProficiencyID, @CurrentProficiencyLevel
        FROM Learns
        INNER JOIN Proficiency ON Learns.Proficiency_ID = Proficiency.Proficiency_ID
        WHERE Learns.Character_ID = @CharacterID;

        IF @CurrentProficiencyLevel IS NULL
        BEGIN
            PRINT 'Karakter gereken bilgilere sahip değil';
        END

        -- Karakter gereklilikleri karşılıyor mu?
        IF @CurrentProficiencyLevel < @RequiredProficiencyLevel
        BEGIN
            PRINT 'Karakter gereken yeterlilik seviyesini karşılamıyor';
        END

        -- skill hali hazırda açık mı
        IF EXISTS (SELECT 1 FROM CanUse WHERE Character_ID = @CharacterID AND Skills_ID =
@SkillID)
        BEGIN
            PRINT 'Bu skill hali hazırda bu karakter için açık';
        END

        -- skillin kilidini aç
              INSERT INTO CanUse (Skills_ID, Character_ID, Req_Prof_Level)
        VALUES (@SkillID, @CharacterID, @RequiredProficiencyLevel);

        -- Değişiklikleri gerçekleştir
        COMMIT TRANSACTION;

        PRINT 'Skill başarıyla açıldı';
    CATCH
    BEGIN CATCH
        -- Hata olursa eski durumuna dön
        ROLLBACK TRANSACTION;

    END CATCH
```

```
END;
```

// TETİKLEMELER

```
---TRİGGER -------
---ESKİ VE YENİ VERİLERİ TUTMAK İÇİN TABLO AÇIYORUZZ
CREATE TABLE Appearance_Change_Log (
    Log_ID INT PRIMARY KEY IDENTITY(1,1),
    Old_Hair_Color NVARCHAR(30),
    New_Hair_Color NVARCHAR(30),
    Old_Eye_Color NVARCHAR(30),
    New_Eye_Color NVARCHAR(30),
    Old_Body_Type NVARCHAR(50),
    New_Body_Type NVARCHAR(50),
    Change_Date DATETIME DEFAULT GETDATE(),
    Appearance_ID INT
);

GO
CREATE TRIGGER trg_AfterAppearanceUpdate
ON Appearance
AFTER UPDATE
AS
BEGIN
    -- Eski ve yeni verileri almak için tablo bazlı işlemler
    SELECT
        d.Appearance_ID AS [Appearance ID],
        d.Hair_Color AS [Old Hair Color],
        i.Hair_Color AS [New Hair Color],
        d.Eye_Color AS [Old Eye Color],
        i.Eye_Color AS [New Eye Color],
        d.Body_Type AS [Old Body Type],
        i.Body_Type AS [New Body Type]
    FROM
        DELETED d
    INNER JOIN
        INSERTED i ON d.Appearance_ID = i.Appearance_ID;

    -- Mesaj yazdır
    PRINT 'Update işlemi tamamlandı. Değişiklikler yukarıda listelenmiştir.';
END;
GO

go
CREATE TRIGGER trg_UpdateInventory
ON Contasins
AFTER INSERT
AS
BEGIN
    DECLARE @Inventory_ID INT;
    DECLARE @Current_Amount INT;
    DECLARE @Capacity INT;

    -- İlgili Inventory_ID'yi al
    SELECT @Inventory_ID = inserted.Inventory_ID
    FROM inserted;

    -- Mevcut Current_Amount ve Capacity değerlerini al
    SELECT @Current_Amount = Inventory.Current_Amount,
           @Capacity = Inventory.Capacity
    FROM Inventory
    WHERE Inventory.Inventory_ID = @Inventory_ID;
```

```
    -- Capacity'yi aşmamak şartıyla Current_Amount'u arttır
    IF @Current_Amount + 1 <= @Capacity
    BEGIN
        UPDATE Inventory
        SET Current_Amount = @Current_Amount + 1
        WHERE Inventory.Inventory_ID = @Inventory_ID;
    END
    ELSE
    BEGIN
        RAISERROR('Inventory capacity exceeded.', 16, 1);
        ROLLBACK TRANSACTION;
    END
END;
go

go
CREATE TRIGGER trg_UpdateInventoryOnDelete
ON Contasins
AFTER DELETE
AS
BEGIN
    DECLARE @Inventory_ID INT;
    DECLARE @Current_Amount INT;

    -- İlgili Inventory_ID'yi al
    SELECT @Inventory_ID = deleted.Inventory_ID
    FROM deleted;

    -- Mevcut Current_Amount değerini al
    SELECT @Current_Amount = Inventory.Current_Amount
    FROM Inventory
    WHERE Inventory.Inventory_ID = @Inventory_ID;

    -- Current_Amount'u 0'dan küçük olmamak şartıyla azalt
    IF @Current_Amount - 1 >= 0
    BEGIN
        UPDATE Inventory
        SET Current_Amount = @Current_Amount - 1
        WHERE Inventory.Inventory_ID = @Inventory_ID;
    END
    ELSE
    BEGIN
        RAISERROR('Inventory current amount cannot be negative.', 16, 1);
        ROLLBACK TRANSACTION;
    END
END;

go
```