CS 3743 – Project Phase 2

Due: Friday, October 23rd by the end of the day (11:59pm)

Upload a PDF to Blackboard.com

Total possible points: 25 points

Project: Phase 2 - PART 2

Based on your updated ***ER diagram, relational schema and report*** write the SQL queries required to create the tables and implement the functionalities you are looking for. Please consider the following:

- **For each table** in your relational model you need to provide the **create table** command with all the required constraints (**Ex**. values should be unique, setting primary and foreign keys, adding check constraints, Not Null constraint, default values if any, address referential integrity violation ,etc.)

```
CREATE TABLE Account(
      AccountName VARCHAR(20),
      Password VARCHAR(10) NOT NULL,
      Email VARCHAR(30),
      PRIMARY KEY(AccountName)
);

CREATE TABLE Owns(
      AccountName VARCHAR(20),
      CharacterName VARCHAR(20),
      FOREIGN KEY (AccountName) REFERENCES Account (AccountName) ON DELETE
CASCADE,
      FOREIGN KEY (CharacterName) REFERENCES Character (CharacterName) ON DELETE
CASADE
);

CREATE TABLE Character(
      CharacterName VARCHAR(20),
      PRIMARY KEY(CharacterName)
);

CREATE TABLE Equip(
      CharacterName VARCHAR(20),
      ItemNum INT,
      ItemName VARCHAR(20),
      FOREIGN KEY (CharacterName) REFERENCES Character ON DELETE CASCADE,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE Have(
      CharacterName VARCHAR(20),
      Label VARCHAR(10),
      FOREIGN KEY (CharacterName) REFERENCES Character ON DELETE CASCADE,
      FOREIGN KEY (Label) REFERENCES Storage
);
```

```
CREATE TABLE Storage(
      CharacterName VARCHAR(20),
      Label VARCHAR(10) NOT NULL,
      FOREIGN KEY (CharacterName) REFERENCES Character ON DELETE CASCADE,
      PRIMARY KEY (CharacterName, Label)
);

CREATE TABLE Stores(
      CharacterName VARCHAR(20),
      Label VARCHAR(10),
      ItemNum INT,
      ItemName VARCHAR(20),
      FOREIGN KEY (CharacterName) REFERENCES Character ON DELETE CASCADE,
      FOREIGN KEY (Label) REFERENCES Character,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE Inventory(
      CharacterName VARCHAR(20),
      Label VARCHAR(10) NOT NULL,
      size INTEGER default(63) NOT NULL,
      FOREIGN KEY (CharacterName) REFERENCES Character ON DELETE CASCADE,
      PRIMARY KEY (CharacterName, Label),
      CHECK(size > 0)
);

CREATE TABLE Item(
      Num INT NOT NULL,
      Name VARCHAR(20) NOT NULL,
      PRIMARY KEY(Num, Name)
);

CREATE TABLE IF NOT EXISTS Armor(
      ItemNum INT,
      ItemName VARCHAR(20),
      Resilience INT,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE IF NOT EXISTS ArmorPerks(
      ANum INT,
      AName VARCHAR(20),
      ArmorPerk VARCHAR(20),
      FOREIGN KEY (ANum, AName) REFERENCES Item (Num, Name) ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS Helmet(
      ItemNum INT,
      ItemName VARCHAR(20),
      VisionModifier INT,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
```

```
);

CREATE TABLE IF NOT EXISTS Arm(
      ItemNum INT,
      ItemName VARCHAR(20),
      AttackModifier INT,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE IF NOT EXISTS Chest(
      ItemNum INT,
      ItemName VARCHAR(20),
      HealthModifier INT,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE IF NOT EXISTS Leg(
      ItemNum INT,
      ItemName VARCHAR(20),
      SpeedModifier INT,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE IF NOT EXISTS Weapon(
      ItemNum INT,
      ItemName VARCHAR(20),
      Damage INT,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE IF NOT EXISTS WeaponPerks(
      WNum INT,
      WName VARCHAR(20),
      WeaponPerk VARCHAR(20),
      FOREIGN KEY (WNum, WName) REFERENCES Item (Num, Name) ON DELETE CASCADE
);


CREATE TABLE IF NOT EXISTS PrimaryWeapon(
      ItemNum INT,
      ItemName VARCHAR(20),
      PrimaryDmgMultiplier NUMERIC(3,2),
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE IF NOT EXISTS Secondary(
      ItemNum INT,
      ItemName VARCHAR(20),
      SecondaryDmgMultiplier INT,
```

```
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE IF NOT EXISTS Heavy(
      ItemNum INT,
      ItemName VARCHAR(20),
      HeavyDmgMultiplier INT,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE AutoRifle(
      ItemNum INT,
      ItemName VARCHAR(20),
      MagSizeLimit INTEGER NOT NULL,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE,
      CHECK (MagSizeLimit > 0)
);

CREATE TABLE Shotgun(
      ItemNum INT,
      ItemName VARCHAR(20),
      PalletSpread INT,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);

CREATE TABLE RocketLauncher(
      ItemNum INT,
      ItemName VARCHAR(20),
      ExplosionRadius INT,
      FOREIGN KEY (ItemNum, ItemName) REFERENCES Item (Num, Name) ON DELETE
CASCADE
);
```

Generating Reports:

1. Equip a random primary weapon from inventory to 'SunnyBoy'.

```
INSERT INTO Equip (CharacterName, Num, Name)
SELECT I.CharacterName AS CharacterName, I.Num AS Num, I.Name AS Name
FROM Inventory I
WHERE EXISTS (
   SELECT *
   FROM Primary P
   WHERE I.Num = P.Num AND I.Name = P.Name
) AND I.CharacterName = 'SunnyBoy'
ORDER BY RAND()
LIMIT 1
```

2. Equip an item

```
INSERT INTO Equip (CharacterName, ItemNum, ItemName)
```

```
SELECT S.CharacterName As CharacterName, S.ItemNum AS Num, S.ItemName AS
ItemName
FROM Stores S WHERE EXISTS
(SELECT * FROM Item I
WHERE S.ItemNum = I.Num AND S.ItemName = I.Name AND S.Label = 'Inventory')
AND S.CharacterName = 'cname';
```

3. Add an extra ArmorPerk "Cerebral Uplink" to all the Helmet Named "Eye of Another World"

```
INSERT INTO ArmorPerks (Num, Name, PerkName)
SELECT Num, Name, PerkName AS 'Cerebral Uplink'
FROM Item
WHERE Name = "Eye of Another World"
```

4. Add an Item to character's inventory

```
INSERT INTO Stores (CharacterName, Label, Num, Name)
Values ('SunnyBoy','Inventory',1,'Sun Shot');
```

5. Add an Item to character's inventory

```
INSERT INTO Stores (CharacterName, Label, Num, Name)
Values ('SunnyBoy','Vault',2,'Sun Shot');
```

6. Add a new Account

```
INSERT INTO Account(AccountName, Password, Email)
Values('GreenBluener', 'BlueGreener', 'veryBlue@gmail.com');
```

7. Add a new Character

```
INSERT INTO Character(CharacterName)
SELECT 'Hazardus' WHERE EXISTS (
   SELECT *
   FROM Account
   WHERE AccountName = 'Jamie');

INSERT INTO Owns (AccountName, CharacterName)
SELECT 'HAccount', 'Hazardus' WHERE EXISTS (
SELECT *
FROM Account
WHERE AccountName = 'HAccount');

INSERT INTO Storage(CharacterName, Label)
VALUES ('Hazardus', 'Vault');

INSERT INTO Inventory(CharacterName, Label)
VALUES ('Hazardus', 'Inventory');

INSERT INTO Have(CharacterName, Label) VALUES ('Hazardus', 'Vault');
INSERT INTO Have(CharacterName, Label) VALUES ('Hazardus', 'Inventory');
```

8. Add a new AutoRifle

```
INSERT INTO Item(Num, Name)
Values(2,'Sweet Bussiness');
INSERT INTO Weapon(ItemNum, ItemName, Damage)
Values(2,'Sweet Bussiness', 15);
INSERT INTO PrimaryWeapon(ItemNum, ItemName, PrimaryDmgMultiplier)
Values(2,'Sweet Bussiness', 0.7);
INSERT INTO AutoRifle(ItemNum, ItemName, MagSizeLimit)
```

```
Values(2, 'Sweet Bussiness', 150);
```

9. Add a new ShotGun

```
INSERT INTO Item(Num, Name)
Values(1,'Lord of Wolves');
INSERT INTO Weapon(ItemNum, ItemName, Damage)
Values(1,'Lord of Wolves', 50);
INSERT INTO Secondary(ItemNum, ItemName, SecondaryDmgMultiplier)
Values(1,'Lord of Wolves', 1.2);
INSERT INTO ShotGun(ItemNum, ItemName, PalletSpread)
Values(1,'Lord of Wolves', 15);
```

10. Add a new RocketLauncher

```
INSERT INTO Item(Num, Name)
Values(1,'Two Tailed Fox');
INSERT INTO Weapon(ItemNum, ItemName, Damage)
Values(1,'Two Tailed Fox', 200);
INSERT INTO Heavy(ItemNum, ItemName, HeavyDmgMultiplier)
Values(1,'Two Tailed Fox', 1.5);
INSERT INTO RocketLauncher(ItemNum, ItemName, ExplosionRadius)
Values(1,'Two Tailed Fox', 10);
```

11. Add a new Helmet

```
INSERT INTO Item(Num, Name)
Values(1, 'Eye of Another World');
INSERT INTO Armor(ItemNum, ItemName, Resilience)
Values(1, 'Eye of Another World', 15);
INSERT INTO Helmet(ItemNum, ItemName, VisionModifier)
Values(1, 'Eye of Another World', 3);
```

12. Add a new Arm

```
INSERT INTO Item(Num, Name)
Values(1, 'testArm');
INSERT INTO Armor(ItemNum, ItemName, Resilience)
Values(1, 'testArm', 15);
INSERT INTO Helmet(ItemNum, ItemName, AttactModifier)
Values(1, 'testArm', 3);
```

13. Add a new Chest

```
INSERT INTO Item(Num, Name)
Values(1, 'testChest');
INSERT INTO Armor(ItemNum, ItemName, Resilience)
Values(1, 'testChest', 15);
INSERT INTO Helmet(ItemNum, ItemName, HealthModifier)
Values(1, 'testChest', 3);
```

14. Add a new Leg

```
INSERT INTO Item(Num, Name)
Values(1, 'testLeg');
INSERT INTO Armor(ItemNum, ItemName, Resilience)
Values(1, 'testLeg', 15);
INSERT INTO Helmet(ItemNum, ItemName, SpeedModifier)
Values(1, 'testLeg', 3);
```

15. Delete a Character

```
DELETE FROM Character WHERE CharacterName = 'SilentSirius';
```

16. Delete an Item
```
DELETE FROM Item WHERE Num = 2 AND Name = 'Old Fasioned';
```

17. Update Password for an Account
```
UPDATE Account
SET Password = 'PvPGOD!!!!'
WHERE AccountName = 'SunnyBoy'
```

18. Update AttackModifier of all Arm
```
UPDATE Arm
SET AttackModifier = 2;
```

19. Update the HealthModifier of all Chest Named "Raiden Flux"
```
UPDATE Chest
SET HealthModifier = 3;
WHERE Name = 'Raiden Flux';
```

20. Update the Speed modifier of Leg that has a resilence more than 7
```
UPDATE Leg
SET SpeedModifier = 2
WHERE EXISTS (
   SELECT *
   FROM Armor A
   WHERE Leg.ItemNum = A.ItemNum
         AND Leg.ItemName = A.ItemName
         AND A.Resilience > 7
);
```

21. Increase all autoRifle damge by 10
```
UPDATE Weapon
SET Damage = Damage+10
WHERE EXISTS (
   SELECT *
   FROM AutoRifle A
   WHERE Weapon.ItemNum = A.ItemNum AND Weapon.ItemName = A.ItemName
);
```

22. Swap item between vault and inventory in stores
```
UPDATE Stores
SET Label = CASE
    WHEN Label = 'Vault'
          THEN 'Inventory'
    ELSE 'Vault'
END
WHERE ItemNum = 0
AND ItemName = 'S'
```

23. Show all account name
```
SELECT AccountName From Account;
```

24. List all the primary weapon IDs that havd a damage value higher than 20 in a character's storage, inventory and equipped slots.
```
SELECT P.ItemNum, P.ItemName
FROM PrimaryWeapon P
```

```
INNER JOIN Stores S ON P.ItemNum = S.ItemNum AND P.ItemName = S.ItemName
INNER JOIN Weapon W ON P.ItemNum = W.ItemNum AND P.ItemName = W.ItemName
WHERE W.Damage > 20 AND S.CharacterName = "Someone";
```

25. List all the armor that is stored in all storage for a certain character.

```
SELECT A.ItemNum, A.ItemName
FROM Armor A
WHERE EXISTS (
   SELECT *
   FROM Stores S
   WHERE S.ItemNum = A.ItemNum
         AND S.ItemName = A.ItemName
         AND S.CharacterName = 'SomeBody'
);
```

26. Calculate the total resilence of each players based on their eqquiped armor.

```
SELECT E.CharacterName, SUM(A.Resilience)
FROM Equip E, Armor A
WHERE E.ItemNum = A.ItemNum AND E.ItemName = A.ItemName
GROUP BY CharacterName
```

27. List all the CharacterName that have one or more Armor

```
SELECT CharacterName, COUNT(CharacterName)
FROM Stores S
WHERE EXISTS (
   SELECT *
   FROM Armor R
   WHERE S.ItemName = R.ItemName
)
GROUP BY CharacterName
HAVING COUNT (CharacterName) > 0;
```