

بخش اول:

User

کلید اصلی: UserID

:ویژگی‌ها

- Username (نام کاربری)
- Level (سطح کاربر)
- Gold (مقدار طلا)
- Elixir (مقدار الکسیری که کاربر دارد)
- DarkElixir (مقدار دارک الکسیری)
- Trophy (تعداد جامها)
- TownHallLevel

کاربران، موجودیت مرکزی سیستم هستند. هر کاربر می‌تواند ساختمان و نیرو داشته باشد، و هم حمله‌کننده باشد و هم مدافع.

Building

کلید اصلی: BuildingID

:ویژگی‌ها

- Name (نام ساختمان: مثل Gold Mine, Barracks, ...)
- UpgradeCost_Gold, UpgradeCost_Elixir, UpgradeCost_DarkElixir (هزینه ارتقاء با تفکیک نوع)
- (منبع)
- Width, Height (ابعاد ساختمان روی نقشه)
- Level (سطح ساختمان)
- Type (نوع ساختمان: Defensive, Resource, Army)

این جدول فقط اطلاعات پایه‌ی انواع ساختمان‌ها را نگه می‌دارد، بدون اشاره به کاربر خاص. کاربردش برای تعاریف عمومی و محاسبه هزینه و فضای لازم برای هر ساختمان است.

UserBuildin(واسطه)

کلید (UserBuilding_ID)

:ویژگی‌ها

- UserID (کلید خارجی به User)
- BuildingID (کلید خارجی به Building)

- XCoordinate, YCoordinate (موقعیت ساختمان روی نقشه کاربر)

این جدول مشخص می‌کنه که هر کاربر، کدام ساختمان‌ها رو روی نقشه‌ی خودش دارد، با چه موقعیتی. می‌توانه چند نمونه از یک نوع ساختمان داشته باشد.

Troop

کلید اصلی: TroopID

:ویژگی‌ها

- Name (نام نیرو: مثل Giant Archer)
- HitPoint (میزان حیات)
- Damage (میزان آسیب)
- DamageType (نوع آسیب: GroundToGround, GroundToAir, AirToBoth و ...)
- Capacity (فضای اشغالی در کمپ)
- Resource

تعریف کلی از انواع نیروها، بدون وابستگی به کاربر خاص.

UserTroop(واسطه)

کلید ترکیبی: (UserID, TroopID)

:ویژگی‌ها

- UserID (کلید خارجی به User)
- TroopID (کلید خارجی به Troop)
- Level

این جدول مشخص می‌کنه که هر کاربر، کدام نیروها رو دارد و در چه سطحی.

Attacks

کلید اصلی: AttackID

:ویژگی‌ها

- UserID_Offender (کلید خارجی به User)
- UserID_Defender (کلید خارجی به User)
- Castle_Destroyed (درصد تخریب)

- Stars_achieved (تعداد ستاره گرفته شده در حمله)
- Attack_Time (تاریخ و زمان حمله)

این جدول تعاملات جنگی بین کاربران رو نگه می داره. برای هر حمله، کاربر مهاجم و مدافع، همراه با نتایج ذخیره می شن.

User → UserBuilding

- 1 به N نوع رابطه
 - هر کاربر می تونه چند ساختمان در نقشه خودش داشته باشد.
 - از طرف User الزامی، چون هر ساختمان باید به یک کاربر تعلق داشته باشد.

UserBuilding → Building

- 1 به N نوع رابطه
 - هر ساختمان قرار داده شده، به یک نوع مشخص از ساختمان اشاره می کنه.
 - UserBuilding باید به یک نوع ساختمان اشاره کنه. الزامی

User → UserTroop

- 1 به N نوع رابطه
 - هر کاربر می تونه چند نوع نیرو رو آزاد کرده باشد.
 - بدون User معنا نداره. الزامی

UserTroop → Troop

- N به 1 نوع رابطه
 - هر UserTroop به یک نوع نیروی کلی اشاره دارد.
 - UserTroop باید به یک نیروی پایه متصل باشد. الزامی

Attacks → User (دو بار)

- برای مهاجم و مدافع 1 به N نوع رابطه
 - هر حمله دقیقاً یک مهاجم و یک مدافع دارد.
 - حمله بدون مهاجم و مدافع ممکن نیست

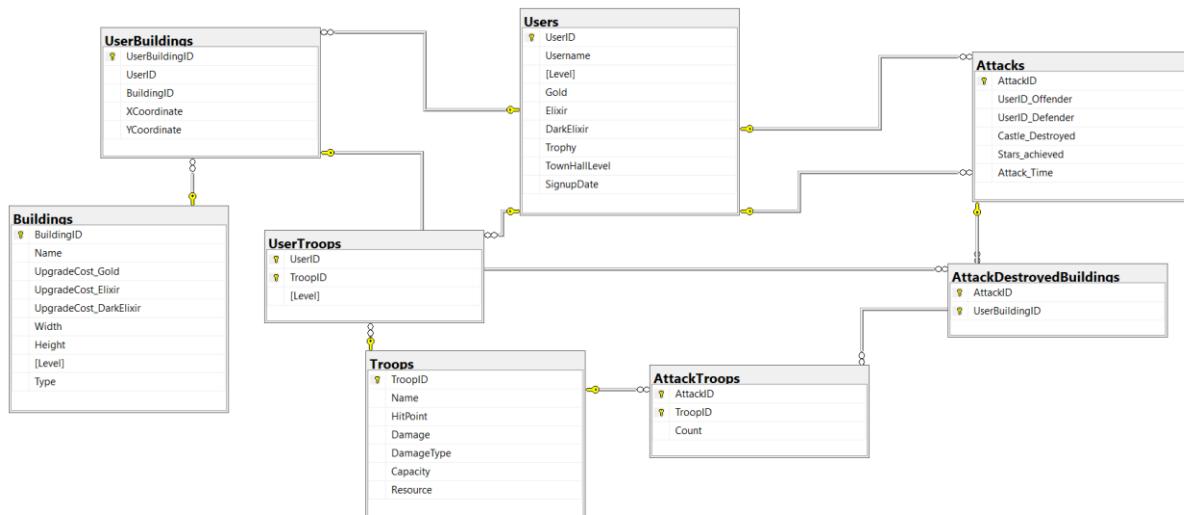
AttackTroops → Attacks و Troops

- نوع N به 1 در هر سمت
 - نشان می دهد در هر حمله چه نوع نیرویی و به چه تعدادی استفاده شده.

AttackDestroyedBuildings → Attacks و UserBuildings

نوع N به ۱

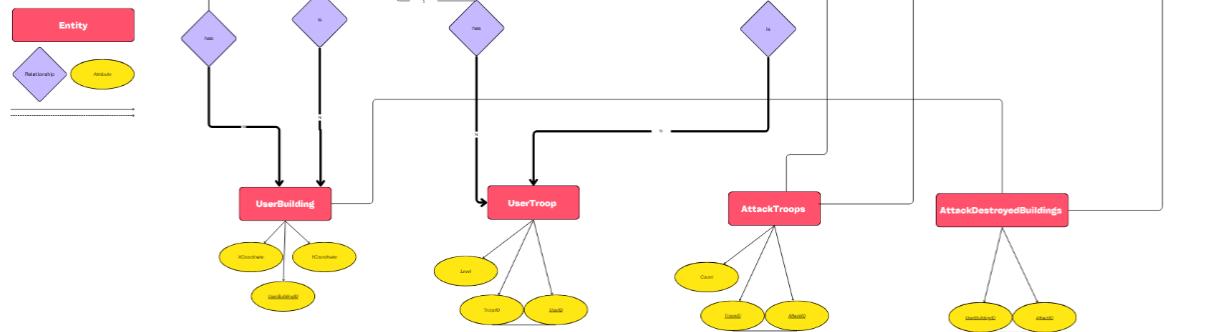
ساختمان‌هایی که در هر حمله تخریب شدن.



Link: [COC ER Diagram](#)

COC ER Diagram

Entity-relationship diagrams show the connections between multiple people, objects, or ideas under a single system. Get started with these simple shapes and lines.



بخش دوم:

```
CREATE TABLE Users (
    UserID INT PRIMARY KEY,
    Username VARCHAR(50) NOT NULL,
    Level INT DEFAULT 1,
    Gold INT DEFAULT 0,
    Elixir INT DEFAULT 0,
    DarkElixir INT DEFAULT 0,
    Trophy INT DEFAULT 0,
    TownHallLevel INT NOT NULL,
    SignupDate DATETIME NOT NULL DEFAULT GETDATE()
);

CREATE TABLE Buildings (
    BuildingID INT PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    UpgradeCost_Gold INT DEFAULT 0,
    UpgradeCost_Elixir INT DEFAULT 0,
    UpgradeCost_DarkElixir INT DEFAULT 0,
    Width INT NOT NULL,
    Height INT NOT NULL,
    Level INT DEFAULT 1,
    Type VARCHAR(50) NOT NULL CHECK (Type IN ('Defensive', 'Resource', 'Army'))
);

CREATE TABLE UserBuildings (
    UserBuildingID INT PRIMARY KEY IDENTITY(1,1),
    UserID INT NOT NULL,
    BuildingID INT NOT NULL,
    XCoordinate INT NOT NULL,
    YCoordinate INT NOT NULL,
    FOREIGN KEY (UserID) REFERENCES Users(UserID),
    FOREIGN KEY (BuildingID) REFERENCES Buildings(BuildingID)
);
```

```
CREATE TABLE Troops (
    TroopID INT PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    HitPoint INT NOT NULL,
    Damage INT NOT NULL,
    DamageType VARCHAR(50) NOT NULL CHECK (DamageType IN ('GroundToGround',
    'GroundToAir', 'GroundToBoth', 'AirToBoth')),
    Capacity INT NOT NULL,
    Resource VARCHAR(50),
);
CREATE TABLE UserTroops (
    UserID INT NOT NULL,
    TroopID INT NOT NULL,
    Level INT NOT NULL,
    PRIMARY KEY (UserID, TroopID),
    FOREIGN KEY (UserID) REFERENCES Users(UserID),
    FOREIGN KEY (TroopID) REFERENCES Troops(TroopID),
);
CREATE TABLE Attacks (
    AttackID INT PRIMARY KEY IDENTITY(1,1),
    UserID_Offender INT NOT NULL,
    UserID_Defender INT NOT NULL,
    Castle_Destroyed DECIMAL(0,2) DEFAULT 0.0,
    Stars_achieved INT CHECK (Stars_achieved BETWEEN 0 AND 5),
    Attack_Time INT NOT NULL,
    FOREIGN KEY (UserID_Offender) REFERENCES Users(UserID),
    FOREIGN KEY (UserID_Defender) REFERENCES Users(UserID)
);
```

۲. نمایش تمام نیروهایی که توسط کاربر خاصی آزاد (Unlocked) شده‌اند.

```

SELECT
    t.TroopID,
    t.Name,
    t.HitPoint,
    t.Damage,
    t.DamageType,
    t.Capacity,
    t.Resource,
    ut.Level AS UserTroopLevel
FROM
    UserTroops ut
JOIN
    Troops t ON ut.TroopID = t.TroopID
WHERE
    ut.UserID = 1;
  
```

| | TroopID | Name | HitPoint | Damage | DamageType | Capacity | Resource | UserTroopLevel |
|---|---------|-----------|----------|--------|----------------|----------|------------|----------------|
| 1 | 1 | Barbarian | 300 | 50 | GroundToGround | 1 | Elixir | 9 |
| 2 | 4 | Minion | 180 | 60 | AirToBoth | 2 | DarkElixir | 4 |
| 3 | 5 | Hog Rider | 800 | 200 | GroundToGround | 5 | DarkElixir | 7 |
| 4 | 10 | Golem | 4500 | 120 | GroundToGround | 30 | DarkElixir | 9 |

۳. نمایش تمام ساختمان‌های کاربر خاص با سطح بالاتر از ۵.

```

SELECT
    ub.UserBuildingID,
    b.BuildingID,
    b.Name,
    b.Level AS BuildingLevel,
    b.Type,
    ub.XCoordinate,
    ub.YCoordinate
FROM
    UserBuildings ub
JOIN
    Buildings b ON ub.BuildingID = b.BuildingID
WHERE
    ub.UserID = 0
    AND b.Level > 0;
  
```

| | UserBuildingID | BuildingID | Name | BuildingLevel | Type | XCoordinate | YCoordinate |
|---|----------------|------------|------------------|---------------|-----------|-------------|-------------|
| 1 | 765 | 69 | Archer Tower | 15 | Defensive | 49 | 3 |
| 2 | 766 | 25 | Elixir Collector | 10 | Resource | 9 | 13 |
| 3 | 767 | 90 | Wall | 7 | Defensive | 33 | 40 |
| 4 | 770 | 27 | Elixir Collector | 12 | Resource | 42 | 23 |
| 5 | 772 | 54 | Cannon | 15 | Defensive | 25 | 26 |
| 6 | 774 | 79 | Wizard Tower | 10 | Defensive | 40 | 25 |
| 7 | 776 | 27 | Elixir Collector | 12 | Resource | 38 | 6 |
| 8 | 777 | 115 | Barracks | 16 | Army | 4 | 11 |
| 9 | 778 | 53 | Cannon | 14 | Defensive | 13 | 36 |

۴. نمایش لیست تمام کاربران به همراه منابع آن‌ها (طلا، اکسیر، دارک اکسیر).

```

SELECT *
FROM
    Users;
  
```

| | UserID | Username | Level | Gold | Elixir | DarkElixir | Trophy | TownHallLevel | SignupDate |
|---|--------|---------------|-------|----------|----------|------------|--------|---------------|-----------------------------|
| 1 | 1 | margaret18 | 166 | 10450697 | 10223061 | 378636 | 4435 | 9 | 2025-06-20 21:40:26.3566667 |
| 2 | 2 | angel59 | 132 | 1530050 | 4700933 | 162364 | 1916 | 14 | 2025-06-20 21:40:26.3566667 |
| 3 | 3 | meganchavez | 40 | 9045718 | 1554299 | 427639 | 4292 | 4 | 2025-06-20 21:40:26.3566667 |
| 4 | 4 | frank45 | 4 | 5926411 | 5207835 | 230060 | 5832 | 10 | 2025-06-20 21:40:26.3600000 |
| 5 | 5 | gregoryfowler | 19 | 1460049 | 1932361 | 41471 | 5989 | 5 | 2025-06-20 21:40:26.3600000 |
| 6 | 6 | tracytujillo | 133 | 5443456 | 6836069 | 369097 | 399 | 3 | 2025-06-20 21:40:26.3600000 |
| 7 | 7 | bryan18 | 141 | 17190348 | 16222356 | 364832 | 4928 | 12 | 2025-06-20 21:40:26.3600000 |

۵. نمایش موقعیت (Y و X) تمام ساختمان‌های "Archer Tower" از نوع.

```
SELECT
    ub.XCoordinate,
    ub.YCoordinate,
    ub.UserID
FROM
    UserBuildings ub
JOIN
    Buildings b ON ub.BuildingID = b.BuildingID
WHERE
    b.Name = 'Archer Tower';
```

| | XCoordinate | YCoordinate | UserID |
|---|-------------|-------------|--------|
| 1 | 20 | 36 | 2 |
| 2 | 49 | 25 | 2 |
| 3 | 13 | 13 | 2 |
| 4 | 39 | 19 | 2 |
| 5 | 34 | 17 | 4 |
| 6 | 32 | 34 | 5 |
| 7 | 30 | 42 | 6 |
| 8 | 42 | 43 | 6 |
| 9 | 20 | 46 | 6 |

۶. نمایش نام تمام نیروهایی که نوع آسیب آن‌ها "AirToBoth" است.

```
SELECT
    Name
FROM
    Troops
WHERE
    DamageType = 'AirToBoth';
```

| | Name |
|---|--------|
| 1 | Minion |

۷. نمایش تمام پادگان‌های مربوط به کاربر 'Username = 'Warlord' با

```
SELECT
    u.Username,
    ub.UserBuildingID,
    ub.XCoordinate,
    ub.YCoordinate,
    b.Level AS BuildingLevel
FROM
    UserBuildings ub
JOIN
    Buildings b ON ub.BuildingID = b.BuildingID
JOIN
    Users u ON ub.UserID = u.UserID
WHERE
    u.Username = 'Warlord'
    AND b.Name = 'Barracks';
```

| | UserBuildingID | XCoordinate | YCoordinate | BuildingLevel |
|---|----------------|-------------|-------------|---------------|
| 1 | 1961 | 11 | 44 | 1 |
| 2 | 1970 | 23 | 6 | 9 |
| 3 | 1971 | 6 | 16 | 15 |

۸. نمایش تمام ساختمان‌هایی که عرض

(Height) یا ارتفاع (Width) آن‌ها بیش از ۴

است.

```
SELECT *
FROM
Buildings
WHERE
Width > ? OR Height > ?;
```

| BuildingID | Name | UpgradeCost_Gold | UpgradeCost_Elixir | UpgradeCost_DarkElixir | Width | Height | Level | Type |
|------------|------|------------------|--------------------|------------------------|-------|--------|-------|------|
| 1 | 110 | Laboratory | 4 | 4 | 1 | Army | | |
| 2 | 111 | Laboratory | 4 | 4 | 2 | Army | | |
| 3 | 112 | Laboratory | 4 | 4 | 3 | Army | | |
| 4 | 113 | Laboratory | 4 | 4 | 4 | Army | | |
| 5 | 114 | Laboratory | 4 | 4 | 5 | Army | | |
| 6 | 115 | Laboratory | 4 | 4 | 6 | Army | | |
| 7 | 116 | Laboratory | 4 | 4 | 7 | Army | | |
| 8 | 117 | Laboratory | 4 | 4 | 8 | Army | | |
| 9 | 118 | Laboratory | 4 | 4 | 9 | Army | | |
| 10 | 119 | Laboratory | 4 | 4 | 10 | Army | | |
| 11 | 120 | Laboratory | 4 | 4 | 11 | Army | | |
| 12 | 121 | Laboratory | 4 | 4 | 12 | Army | | |

| BuildingID | Name | Width | Height | Level | Type |
|------------|------|------------|--------|-------|------|
| 1 | 110 | Laboratory | 4 | 4 | 1 |
| 2 | 111 | Laboratory | 4 | 4 | 2 |
| 3 | 112 | Laboratory | 4 | 4 | 3 |
| 4 | 113 | Laboratory | 4 | 4 | 4 |
| 5 | 114 | Laboratory | 4 | 4 | 5 |
| 6 | 115 | Laboratory | 4 | 4 | 6 |
| 7 | 116 | Laboratory | 4 | 4 | 7 |
| 8 | 117 | Laboratory | 4 | 4 | 8 |
| 9 | 118 | Laboratory | 4 | 4 | 9 |
| 10 | 119 | Laboratory | 4 | 4 | 10 |
| 11 | 120 | Laboratory | 4 | 4 | 11 |
| 12 | 121 | Laboratory | 4 | 4 | 12 |

بخش سوم:

```
DECLARE @TargetUserID INT = ۱۳۳۴;

WITH TargetUserLevel AS (
    SELECT Level FROM Users WHERE UserID = @TargetUserID
),
SuccessfulAttackers AS (
    SELECT DISTINCT a.UserID_Offender AS UserID
    FROM Attacks a
    JOIN Users u ON a.UserID_Offender = u.UserID
    JOIN TargetUserLevel tu ON ABS(u.Level - tu.Level) <= ۱
    WHERE a.Stars_achieved >= ۱ OR a.Castle_Destroyed >= ۰,,
),
TroopUsageByPeers AS (
    SELECT ut.TroopID, COUNT(*) AS UsageCount
    FROM UserTroops ut
    JOIN SuccessfulAttackers sa ON ut.UserID = sa.UserID
    GROUP BY ut.TroopID
),
TargetUserTroops AS (
    SELECT TroopID, Level AS UserTroopLevel
    FROM UserTroops
    WHERE UserID = @TargetUserID
),
RecommendedTroops AS (
    SELECT
        tb.TroopID,
        tr.Name,
        tu.UserTroopLevel,
        tb.UsageCount
    FROM TroopUsageByPeers tb
    JOIN Troops tr ON tb.TroopID = tr.TroopID
    LEFT JOIN TargetUserTroops tu ON tr.TroopID = tu.TroopID
    WHERE tu.TroopID IS NULL OR tu.UserTroopLevel < (
        SELECT AVG(Level)
        FROM UserTroops ut
        WHERE ut.TroopID = tb.TroopID
    )
    SELECT TOP ۳
        TroopID,
        Name AS RecommendedTroop,
        ISNULL(UserTroopLevel, ۰) AS CurrentLevel,
        UsageCount AS PopularityAmongPeers
    FROM RecommendedTroops
    ORDER BY UsageCount DESC;
```

| TroopID | RecommendedTroop | CurrentLevel | PopularityAmongPeers |
|---------|------------------|--------------|----------------------|
| 1 | 9 | Wizard | 226 |
| 2 | 5 | Hog Rider | 221 |
| 3 | 1 | Barbarian | 207 |

۲. برای هر کاربر دلخواه، بسته به کاربران موفق در حمله (معیار موفقیت بسته به شما است) که در لول‌های نزدیکی هستند سه نیرو برای ارتقا پیشنهاد دهد.

۱. کاربران موفق در حمله را پیدا کن.

۲. از بین آن‌ها، آن‌هایی که سطح‌شان نزدیک به کاربر هدف است جدا کن.

۳. از بین این کاربران، محبوب‌ترین نیروهای استفاده‌شده را استخراج کن.

۴. آن نیروها را با نیروهای کاربر هدف مقایسه کن و آن‌هایی که سطح پایین‌تری دارد یا ندارد پیشنهاد بده.

۵. فقط ۳ نیروی برتر را برگردان.

۳. برای هر کاربر دلخواه، بسته به بینش خودتان تعیین کنید که آیا میزان جام (Trophy) های آن ها بسته به لول آن ها زیاد یا کم است.

```

DECLARE @TargetUserID INT = ۱۲۳۴;
DECLARE @TargetLevel INT;
DECLARE @TargetTrophy INT;
DECLARE @AvgTrophy FLOAT;
DECLARE @Result VARCHAR(۲۰);

SELECT
    @TargetLevel = Level,
    @TargetTrophy = Trophy
FROM
    Users
WHERE
    UserID = @TargetUserID;

SELECT
    @AvgTrophy = AVG(CAST(Trophy AS FLOAT))
FROM
    Users
WHERE
    Level = @TargetLevel;

IF @TargetTrophy > @AvgTrophy + ۳..
    SET @Result = 'High';
ELSE IF @TargetTrophy < @AvgTrophy - ۳..
    SET @Result = 'Low';
ELSE
    SET @Result = 'Normal';

SELECT
    @TargetUserID AS UserID,
    @TargetLevel AS Level,
    @TargetTrophy AS Trophy,
    @AvgTrophy AS AvgTrophyForLevel,
    @Result AS TrophyStatus;

```

۱. سطح (Level) کاربر هدف را پیدا کن.
۲. کاربران دیگر با همان سطح را پیدا کن.
۳. میانگین تعداد Trophy برای آن سطح را محاسبه کن.
۴. Trophy کاربر را با میانگین مقایسه کن.
۵. اگر بالاتر از میانگین + آستانه مشخصی بود → زیاد
در غیر این صورت → نرمال

| | UserID | Level | Trophy | AvgTrophyForLevel | TrophyStatus |
|---|--------|-------|--------|-------------------|--------------|
| 1 | 1234 | 184 | 5689 | 3122.35211267606 | High |

| | UserID | Level | Trophy | AvgTrophyForLevel | TrophyStatus |
|---|--------|-------|--------|-------------------|--------------|
| 1 | 4321 | 37 | 2513 | 3020 | Low |

```

WITH BuildingProgress AS (
  SELECT
    ub.UserID,
    SUM(b.Level) AS TotalBuildingLevel,
    SUM(b.UpgradeCost_Gold + b.UpgradeCost_Elixir + b.UpgradeCost_DarkElixir) AS TotalBuildingCost
  FROM UserBuildings ub
  JOIN Buildings b ON ub.BuildingID = b.BuildingID
  GROUP BY ub.UserID
),
TroopProgress AS (
  SELECT
    ut.UserID,
    SUM(ut.Level) AS TotalTroopLevel,
    COUNT(*) AS TotalTroopsUnlocked,
    COUNT(CASE WHEN Level >= 1 THEN 1 END) AS HighLevelTroops
  FROM UserTroops ut
  GROUP BY ut.UserID
),
DominantBuildingType AS (
  SELECT
    UserID, Type, CountPerType
  FROM (
    SELECT
      ub.UserID,
      b.Type,
      COUNT(*) AS CountPerType,
      ROW_NUMBER() OVER (PARTITION BY ub.UserID ORDER BY COUNT(*) DESC) AS rn
    FROM UserBuildings ub
    JOIN Buildings b ON ub.BuildingID = b.BuildingID
    GROUP BY ub.UserID, b.Type
  ) RankedTypes
  WHERE rn = 1
),
UserStats AS (
  SELECT
    u.UserID,
    u.Username,
    u.SignupDate,
    u.TownHallLevel,
    ISNULL(tp.TotalBuildingLevel, -1) AS BuildingLevel,
    ISNULL(tp.TotalTroopLevel, -1) AS TroopLevel,
    ISNULL(tp.TotalBuildingCost, -1) AS TotalCost,
    ISNULL(tp.TotalTroopsUnlocked, -1) AS TotalTroopsUnlocked,
    ISNULL(tp.HighLevelTroops, -1) AS HighLevelTroops,
    ISNULL(bt.Type, 'Unknown') AS DominantBuildingType,
    DATEDIFF(SECOND, u.SignupDate, GETDATE()) AS ActiveSeconds
  FROM Users u
  LEFT JOIN BuildingProgress bp ON u.UserID = bp.UserID
  LEFT JOIN TroopProgress tp ON u.UserID = tp.UserID
  LEFT JOIN DominantBuildingType bt ON u.UserID = bt.UserID
),
Final AS (
  SELECT
    *,
    (BuildingLevel + TroopLevel + TownHallLevel) AS TotalProgress,
    CAST((BuildingLevel + TroopLevel + TownHallLevel) AS FLOAT) / NULLIF(TotalCost + ActiveSeconds, -1) AS EfficiencyScore,
    CASE
      WHEN TroopLevel > BuildingLevel THEN 'Troop Focused'
      ELSE 'Building Focused'
    END AS FocusType,
    CASE
      WHEN CAST(HighLevelTroops AS FLOAT) / NULLIF(TotalTroopsUnlocked, -1) >= 0.5 THEN 'Specialist'
      ELSE 'Generalist'
    END AS TroopUpgradeStyle
  FROM UserStats
)
SELECT
  UserID, Username, TownHallLevel, EfficiencyScore, FocusType, DominantBuildingType, TroopUpgradeStyle, TotalProgress, TotalCost, ActiveDays
FROM Final
ORDER BY EfficiencyScore DESC;

```

۴. تحلیل کنید کدام کاربران در طول

زمان پیشرفت سریعتری دارند (نسبت به زمان و منابع مصرف شده در ارتقای نیرو و ساختمانها) و چه الگوهایی در رفتار آنها وجود دارد.

| | |
|-----------------------------|--|
| EfficiencyScore | قدرت خوب کاربر منابع و زمان رو به پیشرفت تبديل کرده |
| FocusType | بیشتر نیرو ارتقا داده یا ساختمان؟ |
| DominantBuildingType | کدوم نوع ساختمان بیشتر ساخته؟ |
| TroopUpgradeStyle | آیا فقط روی چند نیروی خاص تمرکز کرده؟ (Specialist) (Generalist) |
| TotalProgress | مجموع سطح نیرو + ساختمان |
| TotalCost | جمع منابع صرف شده برای ساختمانها |
| ActiveDays | از زمان ثبت نام تا الان چند روز گذشته |

| | UserID | Username | TownHallLevel | EfficiencyScore | FocusType | DominantBuildingType | TroopUpgradeStyle | TotalProgress | TotalCost | ActiveDays |
|----|--------|------------------|---------------|-----------------|------------------|----------------------|-------------------|---------------|-----------|------------|
| 1 | 10357 | jamesjohnson | 3 | 4.6E-05 | Troop Focused | Defensive | Generalist | 66 | 300000 | 13.0600000 |
| 2 | 8964 | todd95 | 14 | 3.7E-05 | Troop Focused | Resource | Generalist | 64 | 1445000 | 3.0600000 |
| 3 | 13024 | fhughes | 8 | 2.6E-05 | Troop Focused | Army | Generalist | 135 | 3510000 | 19.0600000 |
| 4 | 11179 | andrewskathleen | 11 | 2.4E-05 | Troop Focused | Defensive | Generalist | 136 | 4494000 | 14.0600000 |
| 5 | 1163 | paul36 | 6 | 2.4E-05 | Troop Focused | Army | Generalist | 102 | 542400 | 43.0600000 |
| 6 | 12505 | savannahherandez | 16 | 2.1E-05 | Troop Focused | Army | Generalist | 54 | 1050000 | 18.0600000 |
| 7 | 5682 | jennifer98 | 1 | 2E-05 | Troop Focused | Defensive | Generalist | 45 | 1230000 | 12.0600000 |
| 8 | 2021 | margaretbuchanan | 2 | 1.8E-05 | Troop Focused | Defensive | Generalist | 73 | 4054000 | 0.0600000 |
| 9 | 2767 | leahgreen | 9 | 1.7E-05 | Troop Focused | Army | Generalist | 93 | 870000 | 53.0600000 |
| 10 | 9056 | iconner | 5 | 1.7E-05 | Troop Focused | Resource | Specialist | 52 | 130300 | 34.0600000 |
| 11 | 9191 | gtran | 5 | 1.7E-05 | Building Focused | Defensive | Generalist | 33 | 126000 | 21.0600000 |

۵. برای هر لول موجود در پایگاه داده، تحلیل کنید که چند درصد حمله‌ها زیر یک دقیقه به اتمام می‌رسند.

```
WITH AttackData AS (
    SELECT
        u.Level,
        a.Attack_Time,
        CASE WHEN a.Attack_Time < 1 THEN 1 ELSE 0 END AS IsUnderOneMinute
    FROM Attacks a
    JOIN Users u ON a.UserID_Offender = u.UserID
),
LevelStats AS (
    SELECT
        Level,
        COUNT(*) AS TotalAttacks,
        SUM(IsUnderOneMinute) AS FastAttacks
    FROM AttackData
    GROUP BY Level
)
SELECT
    Level,
    TotalAttacks,
    FastAttacks,
    ROUND(1 * FastAttacks / NULLIF(TotalAttacks, 0), 2) AS FastAttackPercentage
FROM LevelStats
ORDER BY Level;
```

| | |
|--------------------|--|
| AttackData | به ازای هر حمله، تعیین می‌کنه که آیا زیر ۶۰ ثانیه بوده یا نه، همراه با مهاجم Level |
| LevelStats | گروهبندی بر اساس Level و محاسبه‌ی مجموع و حمله‌های سریع |
| خروجی نهایی | درصد محاسبه می‌کنه و مرتب می‌کنه بر اساس Level |

| Level | Results | | |
|-------|--------------|-------------|----------------------|
| | TotalAttacks | FastAttacks | FastAttackPercentage |
| 1 | 1181 | 75 | 6.350000000000000 |
| 2 | 1261 | 69 | 5.470000000000000 |
| 3 | 1165 | 62 | 5.320000000000000 |
| 4 | 937 | 65 | 6.940000000000000 |
| 5 | 1058 | 76 | 7.180000000000000 |
| 6 | 1425 | 98 | 6.880000000000000 |
| 7 | 1441 | 83 | 5.760000000000000 |
| 8 | 1246 | 74 | 5.940000000000000 |
| 9 | 1218 | 78 | 6.400000000000000 |
| 10 | 1374 | 84 | 6.110000000000000 |
| 11 | 1188 | 70 | 5.890000000000000 |
| 12 | 1316 | 77 | 5.850000000000000 |

۶. بسته به بازیکن‌هایی که در دفاع موفق بوده اند (معیار موفقیت را خودتان تعیین کنید)، نشان دهید اختلاف میانگین لول دیوارها و میانگین لول تالار اصلی چقدر است.

```
WITH SuccessfulDefenders AS (
    SELECT DISTINCT UserID_Defender AS UserID
    FROM Attacks
    WHERE Castle_Destroyed < 3 AND Stars_achieved <= 1
),
WallLevels AS (
    SELECT
        ub.UserID,
        b.Level AS WallLevel
    FROM UserBuildings ub
    JOIN Buildings b ON ub.BuildingID = b.BuildingID
    WHERE b.Name LIKE '%Wall%'
),
WallAvg AS (
    SELECT AVG(CAST(WallLevel AS FLOAT)) AS AvgWallLevel
    FROM WallLevels
    WHERE UserID IN (SELECT UserID FROM SuccessfulDefenders)
),
TownHallAvg AS (
    SELECT AVG(CAST(TownHallLevel AS FLOAT)) AS AvgTownHallLevel
    FROM Users
    WHERE UserID IN (SELECT UserID FROM SuccessfulDefenders)
)
SELECT
    w.AvgWallLevel,
    t.AvgTownHallLevel,
    ROUND(w.AvgWallLevel - t.AvgTownHallLevel, 2) AS LevelDifference
FROM WallAvg w, TownHallAvg t;
```

| | |
|----------------------------|---|
| SuccessfulDefenders | شناسایی مدافعانی که کمتر از ۳۰٪ شکست خوردهند |
| WallLevels | استخراج سطح دیوار برای این افراد |
| WallAvg | میانگین سطح دیوارها |
| TownHallAvg | میانگین سطح Town Hall |
| خروجی | اختلاف میانگین دیوار و تالار اصلی فقط برای مدافعان موفق |

| Results | Messages | | |
|---------|------------------|------------------|-----------------|
| | AvgWallLevel | AvgTownHallLevel | LevelDifference |
| 1 | 8.47320172673037 | 8.43852113216231 | 0.03 |

(امتیازی)

```

CREATE TABLE AttackTroops (
    AttackID INT NOT NULL,
    TroopID INT NOT NULL,
    Count INT NOT NULL,
    PRIMARY KEY (AttackID, TroopID),
    FOREIGN KEY (AttackID) REFERENCES Attacks(AttackID),
    FOREIGN KEY (TroopID) REFERENCES Troops(TroopID)
);
CREATE TABLE AttackDestroyedBuildings (
    AttackID INT NOT NULL,
    UserBuildingID INT NOT NULL,
    PRIMARY KEY (AttackID, UserBuildingID),
    FOREIGN KEY (AttackID) REFERENCES Attacks(AttackID),
    FOREIGN KEY (UserBuildingID) REFERENCES
    UserBuildings(UserBuildingID)
);

```

```

WITH SuccessfulAttacks AS (
    SELECT
        AttackID,
        UserID_Offender AS UserID
    FROM Attacks
    WHERE Castle_Destroyed >= 5 OR Stars_achieved >= 2
),
TroopUsage AS (
    SELECT
        sa.UserID,
        at.TroopID,
        SUM(at.Count) AS TotalUsage
    FROM SuccessfulAttacks sa
    JOIN AttackTroops at ON sa.AttackID = at.AttackID
    GROUP BY sa.UserID, at.TroopID
),
RankedTroops AS (
    SELECT
        tu.UserID,
        tu.TroopID,
        tu.TotalUsage,
        RANK() OVER (PARTITION BY tu.UserID ORDER BY tu.TotalUsage DESC)
    AS rk
    FROM TroopUsage tu
)
SELECT
    rt.UserID,
    u.Username,
    rt.TroopID,
    t.Name AS TroopName,
    rt.TotalUsage
FROM RankedTroops rt
JOIN Users u ON rt.UserID = u.UserID
JOIN Troops t ON rt.TroopID = t.TroopID
WHERE rk = 1
ORDER BY rt.UserID;

```

۷. با پیاده سازی های جداول برای نگهداری سرباز های استفاده شده در حمله، و ساختمان های نابود شده برای هر حمله، تعیین کنید که برای هر کاربر کدام سربازها بیشترین تاثیر را در تهاجم های هر بازیکن دارد.

تأثیر = مجموع تعداد استفاده شده در حملات موفق

| | |
|--------------------------|---|
| SuccessfulAttacks | فیلتر حملات موفق ($\leq 50\%$ تخریب یا ≤ 2 ستاره) |
| TroopUsage | جمع کل استفاده از هر نوع سرباز برای هر کاربر |
| RankedTroops | پیدا کردن پر کاربرد ترین سرباز (با $RANK()$ برای هر کاربر) |
| خروجی نهایی | نام کاربر + نام مؤثر ترین سرباز + تعداد استفاده در حملات موفق |

بخش چهارم:

(1NF)

✓ هر سلول فقط یک مقدار اتمی (Atomic) داشته باشد.

✓ هیچ مجموعه تکرارشونده نداشته باشیم.

✓ همه‌ی ردیف‌ها ساختار یکسان داشته باشند.

✓ جداول باید (Primary Key) مشخصی داشته باشند.

بررسی تک‌تک جداول تا 1NF:

Users

UserID (PK) •

همه ویژگی‌ها اتمی هستند. •

Buildings

BuildingID (PK) •

تمام ستون‌ها اتمی هستند. •

UserBuildings

UserBuildingID (PK) •

مختصات اتمی هستند، جداگانه ذخیره شدن. •

Troops

TroopID (PK) •

ستون‌های Resource و DamageType مقادیر منفرد دارند. •

UserTroops

(UserID, TroopID) (PK) •

همه ویژگی‌ها اتمی هستند. •

Attacks

AttackID (PK) •

هیچ ستون لیستی یا تکراری ندارد. •

AttackTroops

(AttackID, TroopID) (PK) •

یک مقدار عددی اتمی است. •

AttackDestroyedBuildings

(AttackID, UserBuildingID) (PK) •

هر سطر نماینده‌ی یک ساختمان تخریب شده است → اتمی •

(2NF)

✓ در **1NF** باشد

✓ هیچ ویژگی غیرکلیدی به بخشی از کلید ترکیبی وابسته نباشد. یعنی وابستگی جزئی نداشته باشیم

✓ برای جدول‌هایی که کلیدشان (تکستونه) است، بررسی **2NF** اتوماتیک برقرار است.

بررسی تک‌تک جداول تا **2NF**

Users

• ساده : UserID

Buildings

• ساده : BuildingID

UserBuildings

• ساده : UserBuildingID

• ویژگی‌های جدول (XCoordinate, YCoordinate) به کلید کامل وابسته هستند.

Troops

• ساده : TroopID

UserTroops

• کلید ترکیبی (UserID, TroopID)

• ویژگی Level : وابسته به ترکیب UserID + TroopID است. نه فقط یکی.

Attacks

• ساده : AttackID

• تمام ویژگی‌ها مستقیماً به کلید وابسته‌اند.

AttackTroops

• کلید ترکیبی : (AttackID, TroopID)

• ویژگی Count : وابسته به ترکیب Attack + Troop

AttackDestroyedBuildings

• کلید ترکیبی : (AttackID, UserBuildingID)

(3NF)

✓ در **2NF** باشد

✓ هیچ وابستگی گذری (Transitive Dependency) بین صفات غیرکلیدی نباشد

✓ هر ستون غیرکلیدی باید مستقیماً و فقط به کلید اصلی وابسته باشد، نه به ستون غیرکلیدی دیگر.

بررسی تک تک جداول تا ***3NF***:

Users

- همه صفات مستقیماً به UserID وابسته‌اند
- هیچ صفت غیرکلیدی، دیگری را تعیین نمی‌کند

Buildings

- فرض بر این است که:
 - در طراحی پایگاه داده مطمئن هستیم که هر اسم ساختمان فقط و فقط یک نوع دارد
 - تعیین *Name* → *Type* برقرار است
 - باید *Type* را از جدول جدا کرد و به جدول جدیدی *BuildingTypes* منتقل کرد

- `CREATE TABLE BuildingTypes (`
 `Name VARCHAR(٥٠) PRIMARY KEY,`
 `Type VARCHAR(٢٠) CHECK (Type IN ('Defensive', 'Resource', 'Army'))`
`);`

پر کردن Buildings از داده‌های موجود در BuildingTypes

- `INSERT INTO BuildingTypes (Name, Type)`
`SELECT DISTINCT Name, Type`
`FROM Buildings;`

- `ALTER TABLE Buildings`
`DROP COLUMN Type;`

حذف ستون Type از جدول Buildings و ایجاد Foreign Key بین Buildings و BuildingTypes

- `ALTER TABLE Buildings`
`ADD CONSTRAINT FK_Buildings_BuildingTypes`
`FOREIGN KEY (Name) REFERENCES BuildingTypes(Name);`

UserBuildings

- همه چیز به کلید اصلی وابسته است
- هیچ وابستگی گذری بین صفات نیست

Troops

- همه صفات مستقیماً به TroopID وابسته هستند
- هیچ وابستگی گذری وجود ندارد

UserTroops

- فقط به ترکیب UserID + TroopID وابسته است
- هیچ صفت غیرکلیدی دیگه‌ای هم نداریم

Attacks

هیچ صفتی به صفت غیرکلیدی وابسته نیست •

AttackTroops

فقط Count داریم، که به کلید ترکیبی وابسته است •

AttackDestroyedBuildings

هیچ ویژگی دیگهای وجود نداره •

| جدول | Primary Key | Candidate Keys |
|--------------------------|----------------------------|----------------|
| Users | UserID | UserID |
| Buildings | BuildingID | BuildingID |
| BuildingTypes | Name | Name |
| UserBuildings | UserBuildingID | UserBuildingID |
| Troops | TroopID | TroopID |
| UserTroops | (UserID, TroopID) | همین ترکیب |
| Attacks | AttackID | AttackID |
| AttackTroops | (AttackID, TroopID) | همین ترکیب |
| AttackDestroyedBuildings | (AttackID, UserBuildingID) | همین ترکیب |