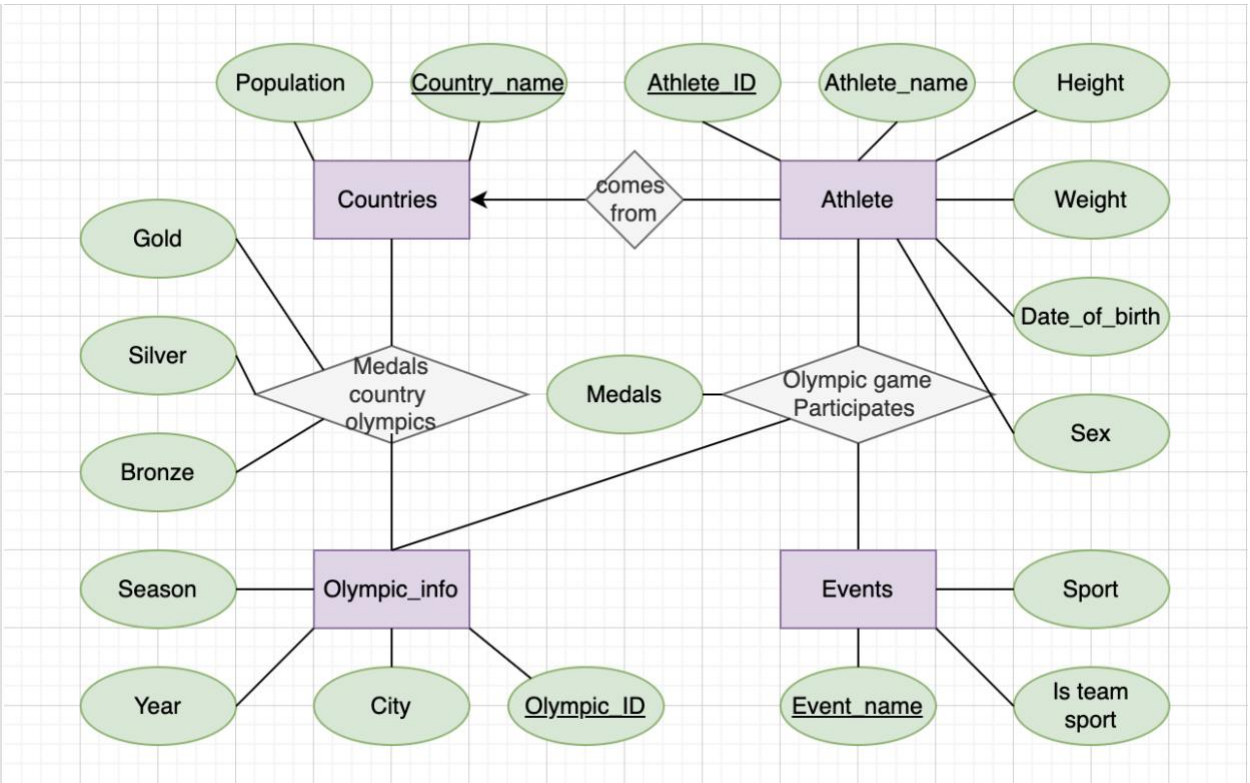
להלן הסכמה של בסיס הנתונים:



הסבר על הטבלאות:

ראשית, בכל הטבלאות הPrimary Keys מזהות באופן חד חד ערכי כל Entry, ובנוסף תומכים ברוב השאילתות שמתבססות בעיקר על הPK.

עבור שאילתות שבהן נדרש סידור או איחוד על בסיס עמודה אחרת, הוספנו INDEX.

1. Countries (Country_name, population) - Country_name is PK and a FK to **Athlete** table.
   טבלה זו מכילה את שם המדינה ואת גודל האוכלוסייה שלה.
2. Athlete (Athlete_ID, Athlete_name, Height, Weight, Age, Sex, Country_name) - Athlete_ID is PK and a FK to **Olympic_game_participates** table, Country_name is FK to **Countries** table.
   טבלה זו מכילה את כל האינפורמציה על המשתתפים באולימפיאדות השונות – מזהה חד-חד ערכי של משתתף, שם המשתתף, גובה, משקל, תאריך לידה ומין.

כמו כן, מאחר ובשאילתות שלנו מבוצעים JOIN על בסיס המדינה ממנה באים האתלטים, **יצרנו INDEX** על בסיס Country כדי לזרז את זמני החיפוש.

3. Events (Event_name, Sport, Is_team_sport) - Event_name is PK and FK to **Olympic_game_participates** table.

רשימת מקצים אולימפיים.- הטבלה מכילה רשימה של שמות מקצים ובהתאמה מה הוא הספורט אשר המקצה שייך לו והאם זה ספורט קבוצתי או יחידני.

כמו כן, מאחר ובאחת השאילתות אנחנו מעוניינים לעשות SORT על בסיס הספורט לפיו יש הכי פחות משתתפים, **יצרנו INDEX** על בסיס Sport כדי לזרז אז זמני החיפוש.

4. Olympic_info (Olympic_ID, Season, Year, City) - Olympic_game is the PK and FK to **Olympic_game_participates** table.

   טבלה זו מכילה את כל האולימפיאדות שהיו ומידע על כל אולימפיאדה – מזהה חד-חד ערכי לאולימפיאדה, האם היא הייתה אולימפיאדת חורף או קיץ, באיזו עיר נערכה האולימפיאדה ובאיזו שנה.

5. Olympic_game_participates (Olympic_ID, Event_name, Athlete_ID, Medals) - Olympic_ID, Event_name, Athlete_ID are all minimal Key, Olympic_ID is a FK to **Olympic_info** table, Event_name is a FK to **Events** table and Athlete_ID is a FK to **Athlete** table.

   טבלה זו מפרטת מה הם המקצים שהיו בכל אולימפיאדה ומיהם המשתתפים בכל אחד מהמקצים.
   כמו כן, הטבלה מפרטת עבור כל משתתף האם הוא זכה במדליה כלשהי ואם כן באיזו מדליה.

השאילתות בהן השתמשנו:

שאילתת הFULL TEXT- כמה משתתפים שלחה מדינה ספציפית:

```
SELECT athlete.country, COUNT(DISTINCT olympic_game_participants.athlete_id) AS total_participants,
FROM olympic_game_participants
JOIN athlete ON olympic_game_participants.athlete_id = athlete.athlete_id AND
athlete.country LIKE '{country_name}'
GROUP BY athlete.country
```

כאשר country_name זה הפרמטר אותו המשתמשת בוחרת.

שאילתות קומפלקסיות:

1. 10 המדינות ששלחו הכי הרבה משתתפים:

```
SELECT athlete.country, COUNT(DISTINCT olympic_game_participants.athlete_id) AS total_participants
FROM olympic_game_participants JOIN athlete
ON olympic_game_participants.athlete_id = athlete.athlete_id
GROUP BY athlete.country
ORDER BY total_participants DESC
LIMIT 10
```

2. 10 המדינות שיחס המשתתפים לאוכלוסייה שלהן הוא הכי גבוה, באחוזים:

```
Select A.Country , COUNT(olympic_game_participants.athlete_id) / population.population
From Countries as C, Olympic_game_participates as OGP, Athlete as A
Where A.Country_name  = C.Country_name
AND OGP.Athlete_ID = A.Athlete_ID
Group by A.Country_name
Order by A.country C.population DESC
Limit 10
```

3. 10 ענפי הספורט עם הכי הרבה משתתפים ולאחר מכן עם הכי פחות משתתפים:

```
Select E.sport
From Olympic_game_participates as OGP, Events as E
Where E.Event_name  = OGP. Event_name
Group by E.sport
Order by SUM(OGP.Athlete_ID) DESC
Limit 10

Select E.sport
From Olympic_game_participates as OGP, Events as E
Where E.Event_name  = OGP. Event_name
Group by E.sport
Order by SUM(OGP.Athlete_ID) ASC
Limit 10
```

4. 10 המדינות עם יחס האוכלוסייה וכמות מדליות הכי גבוה:

```sql
WITH medals_count AS (
    SELECT
        country,
        SUM(gold + silver + bronze) AS total_medals,
        population
    FROM
        medals
        JOIN population ON medals.country = population.country
    GROUP BY
        country, population
)

SELECT
    country,
    (total_medals / population) AS medals_population_ratio
FROM
    medals_count
ORDER BY
    medals_population_ratio DESC
LIMIT
    10
```

5. 10 המדינות עם יחס המשתתפים וכמות מדליות הכי גבוה:

```sql
WITH medals_count AS (
    SELECT
        country,
        SUM(gold + silver + bronze) AS total_medals
    FROM
        medals
    GROUP BY
        country
), total_participants_count AS (
    SELECT
      country,
      COUNT(DISTINCT athlete_id) as total_participants
    FROM
      olympic_game_participants
    JOIN athlete ON olympic_game_participants.athlete_id = athlete.athlete_id
    GROUP BY
      country
)

SELECT
    medals_count.country,
    (medals_count.total_medals / total_participants_count.total_participants) AS medals_participants_ratio
FROM
    medals_count
JOIN total_participants_count ON medals_count.country = total_participants_count.country
ORDER BY
    medals_participants_ratio DESC
LIMIT
    10
```

מקורות מידע

מקור הAPI:

את הAPI לקחנו Countries-Cities מהאתר Rapid-API. הבאנו ממנו מידע עדכני על מדינות והאוכלוסייה שלהן, אותו שמרנו בקובץ CSV שלאחר מכן נדחף לטבלה population.

מאחר ויש הגבלה בכמות הגישות שאפשר לעשות לAPI וניתן לגשת כל כמה שניות, יצרנו את קובץ הCSV מראש לפני ההגשה.

הקוד שמייצר את הCSV ומבצע גישות POST לAPI נמצא בקובץ get_api_data.py בקבצי ההגשה.

מקורות מידע על האולימפיאדה:

5 הטבלאות הנוספות נוצרות באמצעות טבלאות CSV מצורפות מהאתר Kaggle.com, מ Olympic Historical Dataset From Olympedia.org.

כל קבצי הCSV מצורפים.