מסמך תיעוד- פרויקט סיום קורס בסיסי נתונים

אפליקציית "המעשר"

מבנה ה-DB:

מטרת האפליקציה שלנו היא לבנות למשתמש פלייליסטים חדשים על פי נתונים שונים ולאפשר לו להיחשף לשירים/אמנים/אלבומים חדשים. לכן, הישויות המרכזיות שקבענו בהתחלה היו שיר, אומן ואלבום שכן לכל אחת מהן פרמטרים היכולים לעניין את המשתמש בעת הרכבת פלייסליסט. אספנו את הנתונים שקיימים ב-API לגבי כל אחת מן הישויות הללו והתחלנו בבניית דיאגרמה במטרה לגזור ממנה את הטבלאות הנדרשות (כפי שלמדנו בכיתה).

היחסים בין שלושת הישויות הללו נקבעו לפי ההכרות שלנו עם התעשיית המוסיקה ולאחר בחינת הנתונים:

1. אמן-אלבום (Made-by): כל אמן יכול ליצור מספר לאבומים. אך אפשרי גם כי בכל אלבום יהיו מספר אמנים שותפים (כמו אלבום זוגי, או אלבום של להקה). לכן מדובר בקשר של רבים לרבים.
2. שיר-אלבום (In): בכל אלבום יש מספר שירים, וכל שיר מופיע באלבום אחד. לכן מדובר בקשר של רבים (שירים) ליחיד (אלבום). בנוסף, הבחנו כי לקשר הזה יש מאפיין יחודי, והוא המספר של השיר בתוך האלבום.
3. שיר-אמן: כל אמן מבצע מספר שירים וכל שיר יכולים לבצע מספר אמנים. עם זאת, כאן החלטנו כי הקשר בין אמן לשיר בעצם מתבטא בקשר שבין שיר לאלבום ובקשר בין אלבום לאמן.

לאחר מכן החלטנו כי אחד הנתונים שחסר באפליקציה ויהיה שימושי למשתמש זה לדעת מה פופולאריות השירים. מצאנו אתר המציג לגבי 64 מדינות שונות, מה היו השירים הכי מושמעים בהן בשבוע האחרון וכמה פעמים הם הושמעו בהן. לכן הוספנו לדיאגרמה ישות מדינה והחלטנו לקשר אותה לשיר. אפשרי היה לחבר אותה לאומן שכן האתר בו השתמשנו מראה גם את כמות ההשמעות שיש לאומן לפי מדינה, אך החלטנו שמכיוון שהשמעה במהותה היא של שיר מסויים, יהיה יותר נכון להציג את כמות ההשמעות של שיר במדינה. ומכך אם נרצה נוכל גם לחשב מה כמות ההשמעות שיש לאמן במדינה- על ידי סכימת כמות ההשמעות של כל השירים שלו.

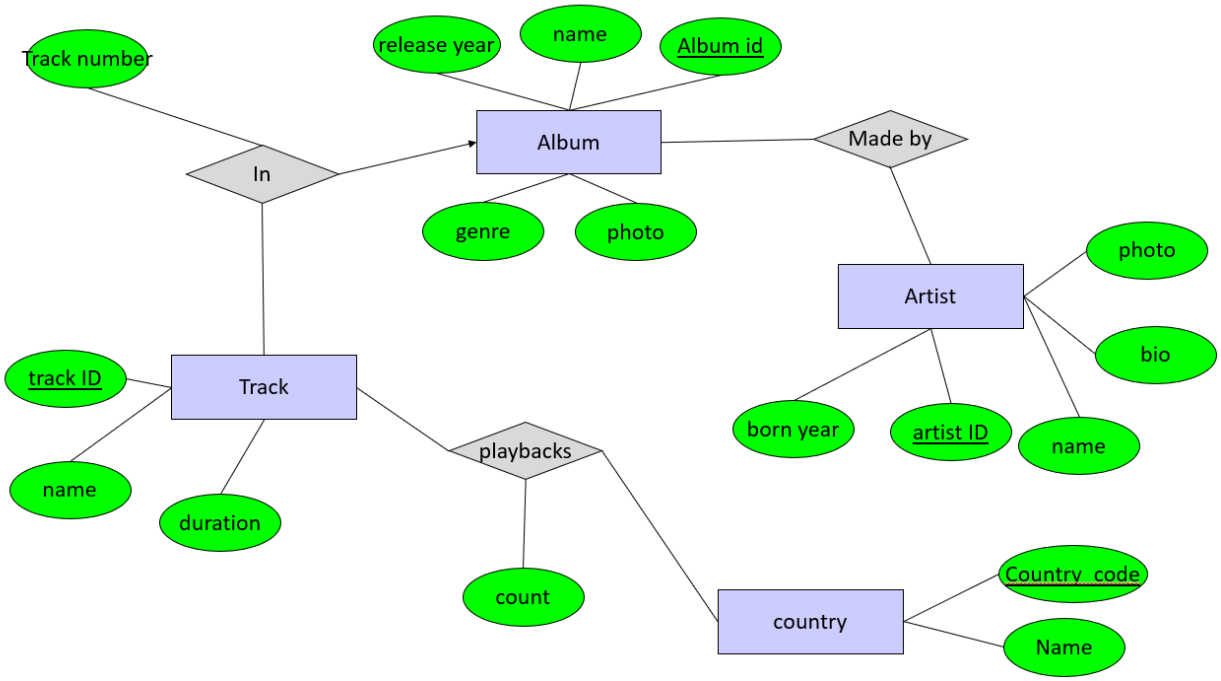
4. שיר-השמעות (Playbacks): שיר יכול להיות מושמע במדינה אחת או בכמה מדינות, ובכל מדינה מושמעים הרבה שירים. לכן מדובר בקשר של רבים לרבים. בנוסף לכל שיר יש כמות השמעות מסויימת בכל מדינה וזהו מאפיין יחודי של הקשר שבין שיר למדינה.

מפתחות (keys): עבור הטבלאות של האמנים, שירים ואלבומים קבענו כי יהיה מזהה id שהינו מספר יחודי לכל שורה. זאת מכיוון שהשמות שלהם אינם חד חד ערכיים ולא נמצא נתון אחר שהינו יחודי. עבור טבלת המדינות קבענו כי שם הקוד של כל מדינה יהיה המפתח שלה.

סוגי השדות (types): עבור כל שדה קבענו את סוגו בהתאם לנתון אותו הוא מייצג והגבלנו את אורכו על מנת לייעל את ה-DB.

אינדקסים (indexes): יפורט בהמשך, לגבי כל שאילתא.

דיאגרמה-



מבנה הטבלאות-

כפי שלמדנו בכיתה, מתוך הדיאגרמה שבנינו נגזרו הטבלאות הנחוצות. לכל ישות יש טבלה משלה (Artist, Album, Track, Country) המכילה שדה לכל נתון שיש לגבי הישות הזאת שמופיע בדיאגרמה. נוסף על כך, לכל קשר של רבים-לרבים (אמן-אלבום, מדינה-שיר) ישנה טבלה המכילה את המפתחות הזרים של טבלאות הישויות המשתתפות בקשר. לקשר שבין שיר לאלבום אין טבלה מכיוון שזהו קשר של רבים (שירים) ליחיד (אלבום) ולכן בטבלת השירים לכל שיר יש שדה של אלבום בו מופיע מפתח זר- המזהה היחודי משל האלבום.

1. **Track** (track_id, name, duration, album_id, track_number)
   - Abum_id is a foreign key from the Album table.
2. **Album** (album_id, name, release_year, genre, photo)
3. **Album_Artist** (album_id, artist_id)
   - Album_id is a foreign key from the Album table.
   - Artist_id is a foreign key from the Artist table.
4. **Artist** (artist_id, name, birth_year, gender, bio, photo)
5. **Country** (country_code, name)
6. **Playbacks** (track_id, country_code, count)
   - Track_id is a foreign key from the Track table.
   - Country_code is a foreign key from the Country table.

הסבר על כל טבלה-

1. **Track-** טבלת שירים. כל שורה מכילה נתונים לגבי שיר אחד. המפתח של הטבלה הינו מספר מזהה הניתן לכל שיר והינו יחודי.
האינדקס שנבחר עבור טבלה זו הוא- duration וכן אינדוקס full-text על עמודת name.

שדה	הסבר	foreign key?	סוג
track_id	המפתח. מזהה יחודי לכל שיר	לא	INT
name	שם השיר	לא	VARCHAR(255)
duration	אורך השיר (במילישניות)	לא	INT
album_id	המזהה של האלבום שהשיר מופיע בו	כן	INT
track_number	המספר של השיר הזה בתוך האלבום	לא	SMALLINT

2. **Album-** טבלת אלבומים. כל שורה מכילה נתונים לגבי אלבום אחד. המפתח של הטבלה הינו מספר מזהה הניתן לכל אלבום והינו יחודי.
האינדקסים שנבחרו עבור טבלה זו הם- release_year, genre

שדה	הסבר	foreign key?	סוג
album_id	המפתח. מזהה יחודי לכל אלבום	לא	INT
name	שם האלבום	לא	VARCHAR(255)
release_year	השנה בה יצא האלבום	לא	YEAR
genre	הז'אנר המוסיקלי של האלבום	לא	VARCHAR(30)
photo	תמונה של עטיפת האלבום	לא	VARCHAR(255)

3. **Album_artist-** טבלת זוגות של אמן ואלבום שהוא היוצר שלו. שני השדות בו הם foreign key של הטבלות המתאימות- Artist ו-Album.

שדה	הסבר	foreign key?	סוג
album_id	מזהה יחודי של אלבום	כן	INT
artist_id	מזהה יחודי של אומן	כן	INT

4. **Artist-** טבלת האמנים. כל שורה מכילה נתונים על אומן אחד (זמר, להקה, די-ג'יי, יוצר). המפתח של הטבלה הינו מספר מזהה הניתן לכל אומן והינו יחודי.
האינדקסים שנבחרו עבור טבלה זו הם-birth_year ו- FULL TEXT INDEX על ה-bio וגם על עמודת name.

שדה	הסבר	foreign key?	סוג
artist_id	המפתח. מזהה יחודי לכל אלבום	לא	INT
name	שם האמן	לא	VARCHAR(255)
birth_year	השנה בה האמן נולד	לא	YEAR

bio	ביוגרפיה של האומן	לא	TEXT
photo	תמונה של האומן	לא	VARCHAR (255)

5. **Country** – טבלת מדינות. כל שורה מכילה שם של מדינה וקיצור שם המדינה, אשר הינו המפתח.

שדה	הסבר	foreign key?	סוג
country_code	קיצור שם המדינה, המפתח.	לא	VARCHAR(6)
country	שם המדינה	לא.	VARCHAR(255)

6. **Playbacks-** טבלה המרכזת את כמות ההשעמות לשיר, לפי מדינה. השדות track_id ו-country_code הם foreign key של הטבלות המתאימות- Track ו-Country.
האינדקס שנבחר עבור טבלה זו הוא- count.

שדה	הסבר	foreign key?	סוג
track_id	המזהה יחודי לכל שיר	כן	INT
country_code	שמקוד של המדינה	כן	VARCHAR(6)
count	כמות ההשמעות של השיר במדינה	לא	BIGINT

השאילתות:

באפליקציה ישנם שלושה סוגים של שאילתות, כאשר משני הסוגים הראשונים ישנן שאילתות פשוטות וגם מורכבות:

i. שאילתות מוכנות מראש – שאילתות אלה מופיעות בעמוד הראשי של האפליקציה, בליווי תמונה מתאימה. כדי להריצן ולקבל את הפלט שלהן, על המשתמש ללחוץ על הכפתור הרלוונטי של השאילתה.
ii. שאילתות בהרכבה אישית – שאילתות אשר מקבלות קלט מהמשתמש אשר משפיע על אופן הרצת השאילתה. על מנת להגיע לשאילתות אלה, יש ללחות על הכפתור make your own top 10 שנמצא בעמוד הראשי.
iii. שאילתות עזר – אלה שאילתות פשוטות שניתן להריץ על גבי התוצאות של השאילתות האחרות (למשל עבור אומן מסוים שהתקבל בשאילתה, לקבל את אלבומיו).

שאילתות מוכנות מראש

המורכבות:

1. **האלבומים הכי מושמעים סה"כ**

- פרמטרים לבחירת המשתמש: אין
- התוצאה: רשימת עשרת האלבומים אשר כמות ההשמעות הכוללת לכל שיריהם יחד (גלובלית) הכי גבוהה.
- המידע המוצג: שם האלבום, שם האומן, כמות ההשמעות סה"כ בכל שירי האלבום (גלובלית), השיר הכי מושמע מתוך האלבום – שמו וכמות ההשמעות שלו, קישור לתמונת האלבום.
- אופטימיזציה ומבנה הDB: על מנת ליעל שליפה זו, בחרנו לאנדקס את הטבלה playbacks לפי שדה הcount (כמות ההשמעות), שכן אנו מבצעים לפיו JOIN בשליפה (בין כמות ההשמעות המקסימלית לשיר בכל אלבום, שמתקבלת בשליפה הפנימית, לבין playbacks בשליפה החיצונית – זאת על מנת לקבל את השם של השיר הנשמע ביותר באלבום). למעט שדה זה, כל יתר השדות שבשימוש בשליפה הם כבר מפתחות/מפתחות זרים, ועל כן אין עוד איך ליעל.
- השליפה:

```
SELECT album_name, artist_name, total_global_playbacks, tl.max_song AS
max_song_name,
tl.play_count AS max_song_plays, photo_link, album_id
FROM (SELECT album.album_id AS album_id, album.name AS album_name,
artist.name AS artist_name,
      SUM(playbacks.count) AS total_global_playbacks, album.photo AS
      photo_link, MAX(playbacks.count) AS max_plays
      FROM track, album, artist, album_artist, playbacks
      WHERE track.track_id = playbacks.track_id
      AND track.album_id = album.album_id
      AND track.album_id = album_artist.album_id
      AND artist.artist_id = album_artist.artist_id
      AND playbacks.country_code = "global"
      GROUP BY track.album_id, album.name, artist.name
      ORDER BY SUM(playbacks.count) DESC
      LIMIT 10) AS top_albums
JOIN (SELECT t.album_id AS aid, l.count play_count, t.name max_song
      from track t, playbacks l
      WHERE t.track_id = l.track_id AND l.country_code = "global") AS tl
ON top_albums.album_id = tl.aid AND top_albums.max_plays = tl.play_count
```

2. **האומנים המובילים לפי כמות השמעות ממוצעות**

- פרמטרים לבחירת המשתמש: אין
- התוצאה: רשימת עשרת האומנים המושמעים ביותר גלובלית, בממוצע לפי כל שיריהם.
- המידע המוצג: שם האומן, ממוצע השמעות לשיר של האומן, סך כמות ההשמעות של האומן בכל שיריו (גלובלית), השיר הכי מושמע של האומן – שם וכמות השמעות, לינק לתמונה של האומן.
- אופטימיזציה ומבנה הDB: בדומה לשליפה הקודמת, גם שליפה זו מתייעלת ע"י כך שהגדרנו את שדה count בplaybacks להיות אינדקס, שכן גם כן נעשה לפיו JOIN מסיבה דומה. כל יתר השדות שבשימוש הם מפתחות רגילים / זרים.
- השליפה:

```
SELECT artist_playbacks.artist_name AS artist_name,
artist_playbacks.average_artist_playback AS average_artist_playback,
artist_playbacks.sum_artist_playbacks AS sum_artist_playbacks,
artist_songs.max_song AS most_played_song, artist_songs.play_count AS
most_played_sound_count,
artist_playbacks.photo, artist_playbacks.artist_id
      FROM( SELECT ar.artist_id AS artist_id, ar.name AS artist_name,
      ar.photo AS photo, ROUND(AVG(p.count)) AS average_artist_playback,
      SUM(p.count) sum_artist_playbacks, MAX(p.count) AS max_artist_playbacks
      FROM artist ar, track t, album_artist alar, playbacks p
      WHERE t.track_id = p.track_id
      AND t.album_id = alar.album_id
      AND ar.artist_id = alar.artist_id
      AND p.country_code = "global"
      GROUP BY ar.artist_id, ar.name, ar.photo
      HAVING COUNT(t.track_id) > 3
      ORDER BY AVG(p.count) DESC
      LIMIT 10) AS artist_playbacks
JOIN (SELECT aa.artist_id AS art_id, t.name AS max_song, l.count AS
play_count
FROM track t, playbacks l, album_artist aa
WHERE t.track_id = l.track_id AND l.country_code = "global"
AND t.album_id = aa.album_id) artist_songs
ON artist_playbacks.artist_id = artist_songs.art_id
AND artist_playbacks.max_artist_playbacks = artist_songs.play_count
ORDER BY artist_playbacks.average_artist_playback DESC
```

3. **האלבומים הארוכים ביותר**

- פרמטרים לבחירת המשתמש: אין
- התוצאה: עשרת האלבומים הארוכים ביותר, מבחינת אורך כל השירים שלהם יחד.
- המידע המוצג: שם האלבום, שם האומן, אורך כולל של האלבום, השיר הארוך ביותר – שמו ואורכו, לינק לתמונת האלבום.
- אופטימיזציה ומבנה הDB: על מנת ליעל שליפה זו, אינדקסנו את הטבלה track לפי שדה duration, שכן בעזרתו אנו מבצעים JOIN שספיצי לאורכי עשרת האלבומים הכי ארוכים. גם כאן, יתר השדות שבשימוש הם כבר מפתחות, ועל כן אין עוד איך ליעל.
- השליפה:

```
SELECT album_name, artist_name, album_length, t.name AS longest_song,
longest_song, album_photo, t.album_id AS album_id
FROM
```

```
      (SELECT album.album_id AS album_id, album.name AS album_name,
      artist.name AS artist_name,
      SUM(track.duration) AS album_length, MAX(track.duration) AS
      longest_song, album.photo AS album_photo
      FROM track, album, artist, album_artist
      WHERE track.album_id = album.album_id
      AND track.album_id = album_artist.album_id
      AND artist.artist_id = album_artist.artist_id
      GROUP BY track.album_id, album.name, artist.name
      ORDER BY SUM(track.duration) DESC
      LIMIT 10
      ) AS longest_albums
JOIN track t
ON longest_albums.album_id = t.album_id AND longest_albums.longest_song =
t.duration
ORDER BY album_length DESC
```

4. **שירי אהבה ותיקים Old Love Songs**

- <u>פרמטרים לבחירת המשתמש:</u> אין
- <u>התוצאה:</u> עשרה שירים אשר בשמם מופיעה המילה love, ויצאו מלפני שנת 1980.
- <u>המידע המוצג:</u> שם השיר, שם האומן, אורך השיר, שנת יציאת השיר, שם האלבום.
- <u>אופטימיזציה ומבנה הDB:</u> מאחר ואנו מבצעים בשאילתה חיפוש בתוך המחרוזות של עמודת track.name, החלטנו לבצע בטבלת track אינדוקס full-text לעמודת name (שם השיר) על מנת ליעל אותה ואת השאילתה הבאה, במקום לבצע בהן שימוש לא יעיל בLIKE. בנוסף, היות העמודה realese_year בטבלה album אינדקס בטבלה, עוזר לייעל את השליפה.
- <u>השליפה:</u>

```
SELECT track.name AS track_name, artist.name AS artist_name, track.duration
AS duration,
album.release_year AS release_year, album.name AS album_name,track.track_id
AS track_id
FROM
      (SELECT max(t.track_id) AS track_id, a.artist_id AS artist_id
      FROM track AS t, album AS al, album_artist AS ala, artist AS a
      WHERE t.album_id = al.album_id
      AND al.album_id = ala.album_id
      AND ala.artist_id = a.artist_id
      AND MATCH(t.name) AGAINST("love")
      AND al.release_year < 1980
      GROUP BY a.artist_id, a.name
      LIMIT 10) AS christmas_songs,
track, album, artist
WHERE christmas_songs.track_id = track.track_id
AND christmas_songs.artist_id = artist.artist_id
AND album.album_id = track.album_id
```

5. **שירי חג המולד Christmas Songs**

- <u>פרמטרים לבחירת המשתמש:</u> אין
- <u>התוצאה:</u> עשרה שירי חג המולד – שירים שבשמם המילה Christmas, ולא יותר משיר אחד מכל אומן.
- <u>המידע המוצג:</u> שם השיר, שם האומן, אורך השיר, שנת יציאת השיר, שם האלבום.
- <u>אופטימיזציה ומבנה הDB:</u> בדומה לשאילתה 4, אינדוקס עמודת track.name בfull_text מייעל את החיפוש שנעשה בשליפה לשירים שמכילים בשמם את המילה Christmas.
- <u>השליפה:</u>

```
SELECT track.name AS track_name, artist.name AS artist_name, track.duration
AS duration,
album.release_year AS release_year, album.name AS album_name,track.track_id
AS track_id
FROM
      (SELECT max(t.track_id) AS track_id, a.artist_id AS artist_id
      FROM track AS t, album AS al, album_artist AS ala, artist AS a
      WHERE t.album_id = al.album_id
      AND al.album_id = ala.album_id
      AND ala.artist_id = a.artist_id
      AND MATCH (t.name) AGAINST("christmas")
      GROUP BY a.artist_id, a.name
      LIMIT 10) AS christmas_songs,
track, album, artist
WHERE christmas_songs.track_id = track.track_id
AND christmas_songs.artist_id = artist.artist_id
AND album.album_id = track.album_id
```

שאילתות פשוטות:

1. **מוסיקה צרפתית French music**
   - התוצאה: רשימת עשרת השירים המושמעים ביותר בצרפת.
   - השליפה:

```
SELECT DISTINCT t.name AS Track, a.name AS Artist, t.duration AS Duration,
p.count AS Streams, t.track_id
FROM track AS t, playbacks AS p, album AS al, album_artist AS ala,artist AS a
WHERE t.track_id = p.track_id AND
t.album_id = al.album_id AND
al.album_id = ala.album_id AND
ala.artist_id = a.artist_id AND
p.country_code = "fr"
ORDER BY p.count DESC
LIMIT 10
```

2. **מוסיקת פופ Pop music**
   - התוצאה: רשימת עשרת השירים המושמעים ביותר גלובלית מז'אנר הפופ.
   - השליפה:

```
SELECT DISTINCT t.track_id, t.name AS Track, a.name AS Artist, t.duration AS
Duration, p.count AS Streams, t.track_id
FROM track AS t, playbacks AS p, album AS al, album_artist AS ala,artist AS a
WHERE t.track_id = p.track_id
AND t.album_id = al.album_id
AND al.album_id = ala.album_id
AND ala.artist_id = a.artist_id
AND p.country_code = "global"
AND al.genre = "pop"
ORDER BY p.count DESC
LIMIT 10
```

3. **מוסיקת קאנטרי Country Music**
   - התוצאה: עשרה שירים מז'אנר הקאנטרי, לא יותר מאחד מאלבום.
   - השליפה:

```
SELECT track.name AS track_name, track.duration, artist.name AS artist_name,
album.name AS album_name, track.track_id
```

```
FROM album JOIN track ON album.album_id = track.album_id
JOIN album_artist ON album.album_id = album_artist.album_id
JOIN artist ON artist.artist_id = album_artist.artist_id
WHERE album.genre = "Country"
ORDER BY track.track_number
LIMIT 10
```

4. **טופ 10 ישראלי לשנת 2019 -Israel Top 10 2019**
   - התוצאה: עשרה שירים המושמעים ביותר בישראל, שיצאו בשנת 2019.
   - השליפה:

```
SELECT t.name AS track_name, p.count AS streams, ar.name AS artist_name,
al.name AS album_name, t.track_id AS track_id
FROM track t, playbacks p, album al, album_artist ala, artist ar
WHERE p.country_code = "il"
AND t.track_id = p.track_id
AND al.album_id = t.album_id
AND al.album_id = ala.album_id
AND ala.artist_id = ar.artist_id
AND al.release_year = 2019
ORDER BY p.count desc
LIMIT 10
```

5. **שירי פרנק סינטרה Frank Sinatra songs**
   - התוצאה: עשרה שירים של הזמר פרנק סינטרה.
   - השליפה:

```
SELECT track.name AS track_name, track.duration AS duration, al.name AS
album_name, al.release_year AS release_year,
track.track_id AS track_id
FROM (
      SELECT MIN(t.track_id) AS track_id, al.album_id AS album_id
      FROM track t, album_artist ala, album al
      WHERE t.album_id = ala.album_id
      AND al.album_id = t.album_id
      AND ala.artist_id = 755
      GROUP BY al.album_id
      LIMIT 10) AS sinatra_songs,
track, album al
WHERE track.track_id = sinatra_songs.track_id
AND al.album_id = sinatra_songs.album_id
```

שאילתות בהרכבה אישית

שאילתות מורכבות:

1. **שאילתת Full-Text – חיפוש אמנים לפי מילים בביוגרפיה ושנת לידה**
   - פרמטרים לבחירת המשתמש: בשליפה זו המשתמש יכול להכניס מספר מילות מפתח לחיפוש בביוגראפיה של אומן. הוא יכול לבחור מילים מתוך רשימה מוכנה מראש אשר אנו יודעים שתניב תוצאות (כדוגמת rapper, actress וכו'), או להקליד באופן חופשי. כמו כן, הוא יכול לבחור להזין טווח שנים בו ירצה להגביל את החיפוש על האומנים, מבחינת שנת הלידה שלהם.
   - התוצאה: רשימה של עשרה אומנים אשר נולדו בטווח השנים המבוקש, ושבביוגראפיה שלהם קיימות כל המילים אותם חיפש המשתמש.

- המידע המוצג: שם האומן, שנת לידתו, לינק לתמונה של האומן.
- אופטימיזציה ומבנה הDB: שליפה זו זוכה לייעול ע"י כל שעמודת הbio של טבלת artist, מאונדקסת בתצורת full text, ובכך מאפשרת שליפות יעילות על מילים מתוכה. כמו כן, גם עמודת birth_year מהווה אינדקס, ולכן גם ביצוע הגבלות לפיה יהיה מהיר.
- השליפה: (בתצורתה הכללית בפייתון)

```
SELECT NAME AS artist_name, birth_year, photo, artist_id
FROM artist
WHERE MATCH (bio) AGAINST("{word1}")
#AND MATCH (bio) AGAINST("{word2}") -etc. for as many words as the user wants

AND artist.birth_year >= {YEAR1}
AND artist.birth_year <= {YEAR2}
```

2. **השירים המושמעים ביותר לפי מדינה או מדינות**

- פרמטרים לבחירת המשתמש: שמות של מדינות (כמה שהמשתמש רוצה).
- התוצאה: רשימת עשרת השירים הכי נשמעים בכל המדינות שהמשתמש בחר (סה"כ).
- המידע המוצג: שם השיר, כמות השמעות של השיר בכל המדינות המבוקשות סה"כ, שם האומן המבצע, שם האלבום, אורך השיר.
- אופטימיזציה ומבנה הDB: כל השדות שמסוג id (וכן השדה country_code) הם מפתחות או מפתחות זרים בטבלאותיהם, ועל כן כבר משמשים כאינדקסים ומזרזים את כל הjoins שנעשים.
שקלנו לאנדקס בטבלה country לפי העמודה country.name, שבעזרתה נעשית השליפה, אך מאחר ובטבלה זו רק 65 רשומות, הגענו למסקנה כי אין לכך הצדקה ממשית, שכן כל שליפה תחזור מהר מאוד בכל אופן. על כן כדי ליעל את השליפה, לא נותר עוד מה להוסיף.
נציין רק כי אנו מבצעים בשליפה זו INNER JOIN בין tracks לבין playbacks שכן לשיר חייבים להיות כמות השמעות כלשהי על מנת להיכלל בתוצאות, אבל עם כל יתר הטבלאות אנו עושים LEFT JOIN, שכן אין חובה שלשיר בהכרח יופיעו יתר הנתונים.
- השליפה:

```
SELECT SUM(playbacks.count) AS num_plays, track.name AS track_name,
artist.name AS artist_name,
track.duration AS duration, album.name AS album_name, track.track_id
FROM (track INNER JOIN playbacks ON track.track_id = playbacks.track_id)
LEFT JOIN country ON playbacks.country_code = country.country_code
LEFT JOIN album ON track.album_id = album.album_id
LEFT JOIN album_artist ON track.album_id = album_artist.album_id
LEFT JOIN artist ON artist.artist_id = album_artist.artist_id
WHERE
(
country.name = "{country_name1}"
#OR playbacks.country_code = "{country_name2}"
#...more countries can be added...
)
GROUP BY track.track_id, track.name, track.duration, artist.name ,album.name
ORDER BY SUM(playbacks.count) DESC
LIMIT 10
```

3. **השירים המושמעים ביותר בין שנים נתונות**

- פרמטרים לבחירת המשתמש: טווח השנים בו ירצה לקבל את השירים הכי מושמעים.
- התוצאה: רשימת עשרת השירים המושמעים ביותר (גלובלית) שיצאו בין השנים שהמשתמש הזין.
- המידע המוצג: שם השיר, שם האומן, שם האלבום, מספרו של השיר באלבום, סך כל ההשמעות של השיר (גלובלית, ובכל הזמנים), שנת יציאת השיר.

- אופטימיזציה ומבנה הDB: כדי ליעל שאילתה זו, נעזרנו באינדקס בטבלה tracks לפי אורך השיר, וכן הוספנו אינדקס בטבלה album לפי שנת יציאת האלבום, שכן לפי שדה זה מתבצעת השאילתה. יתר השדות שבשימוש הם כבר מפתחות.
- השליפה:

```
SELECT track.name as track_name, artist.name AS artist_name, album.name AS
album_name,
track.track_number AS track_number, playbacks.count AS total_plays,
album.release_year AS release_year, track.track_id AS track_id
FROM (track INNER JOIN playbacks ON track.track_id = playbacks.track_id)
LEFT JOIN
((album INNER JOIN album_artist ON album.album_id = album_artist.album_id)
INNER JOIN artist ON album_artist.artist_id = artist.artist_id)
ON track.album_id = album.album_id
WHERE playbacks.country_code = "global"
AND album.release_year >= {year1}
AND album.release_year <= {year2}
ORDER BY playbacks.count DESC
LIMIT 10
```

4. **שיר הכי מושמע, מכל אומן הכי מושמע בז'אנר מסוים**

- פרמטרים לבחירת המשתמש: ז'אנר מבוקש.
- התוצאה: רשימה של עשרה שירים, אחד מכל אומן מבין עשרת האומנים שצברו הכי הרבה השמעות סה"כ (גלובלית) בז'אנר אותו ביקש המשתמש.
- המידע המוצג: שם השיר, מספר ההשמעות של השיר (גלובלית), שם האומן, שם האלבום, סך ההשמעות של האומן בז'אנר.
- אופטימיזציה ומבנה הDB: על מנת ליעל את השליפה, אינדקסנו את הטבלה album לפי genre. כמו כן השליפה נעזרת בכך שהטבלה playbacks מאונדקסת לפי כמות ההשמעות count, ומכך שכל יתר השדות שבשימוש הם מפתחות.
- השליפה:

```
SELECT top_track_name, top_track_plays, artist_name, top_track_album,
total_artist_plays_in_genre, track_id
FROM
(#top 10 artists in genre globaly:
SELECT ar.artist_id as artist_id, ar.name AS artist_name,
SUM(p.count) AS total_artist_plays_in_genre, MAX(p.count) max_song_plays,
al.genre
FROM artist ar, track t, playbacks p, album_artist alar, album al
WHERE t.track_id = p.track_id
AND t.album_id = alar.album_id
AND alar.artist_id = ar.artist_id
AND t.album_id = al.album_id
AND p.country_code = "global"
AND al.genre = "{genre}"  #genre chosen by user
GROUP BY ar.artist_id, ar.name
ORDER BY SUM(p.count) DESC
LIMIT 10) AS top_singers
JOIN (SELECT t2.name AS top_track_name, t2.track_id AS track_id,
ar2.artist_id AS ar_id, al2.name AS top_track_album,
p2.count AS top_track_plays
FROM track t2, artist ar2, album al2, album_artist alar2, playbacks p2
WHERE t2.album_id = alar2.album_id
```

```
AND ar2.artist_id = alar2.artist_id
AND t2.album_id = al2.album_id
AND t2.track_id = p2.track_id) AS tracks_plays
ON top_singers.artist_id = tracks_plays.ar_id AND top_singers.max_song_plays
= tracks_plays.top_track_plays
ORDER BY top_singers.total_artist_plays_in_genre desc
```

5. **אלבומים לפי שם אומן (שאילתת full-text) Albums by Artist Name**
   - פרמטרים לבחירת המשתמש: שם של אומן (כולל שם חלקי).
   - התוצאה: רשימה של עד עשרה אלבומים של האומן אותו חיפש המשתמש בשמו (אם קיים).
   - המידע המוצג: שם האומן, שם אלבום, שנת יציאת האלבום.
   - אופטימיזציה ומבנה הDB: שליפה שלו מיועלת בעזרת האינדוקס full-text של עמודת name בטבלה artist.
   - השליפה:

```
SELECT ar.name AS artist_name, al.name AS album_name, al.release_year AS
release_year, al.album_id AS album_id
FROM album al, album_artist ala, artist ar
WHERE MATCH(ar.name) AGAINST("{first_name}")
#AND MATCH(ar.name) AGAINST("{last_name}")
AND ar.artist_id = ala.artist_id
AND ala.album_id = al.album_id
ORDER BY ar.artist_id, al.release_year
LIMIT 10
```

6. **חיפוש שירים לפי שם Songs by name**
   - פרמטרים לבחירת המשתמש: שם או שם חלקי שם שיר.
   - התוצאה: רשימה של עשרה שירים המכילים בשמם את השם שחיפש המשתמש.
   - המידע המוצג: שם השיר, שם האומן, שם האלבום.
   - אופטימיזציה ומבנה הDB: שליפה שלו מיועלת בעזרת האינדוקס full-text של עמודת name בטבלה track.
   - השליפה:

```
SELECT track.name AS track_name, artist.name AS artist_name, album.name AS
album_name,
album.release_year AS release_year, album.genre AS genre, track.track_id AS
track_id
FROM track, album, album_artist, artist
WHERE MATCH(track.name) AGAINST ("{first_name}")
#AND MATCH(track.name) AGAINST ("{second_name}")
AND track.album_id = album.album_id
AND track.album_id = album_artist.album_id
AND album_artist.artist_id = artist.artist_id
LIMIT 10
```

שאילתות עזר

כאמור, ניתנות להרצה על גבי תוצאות של השאילתות האחרות.

1. החזרת מידע על שיר ספציפי (לפי track_id) – שם האומן, אורך השיר, שם האלבום ושנת יציאה.
2. החזרת מידע על אומן ספציפי (לפי artist_id).
   - מוחזר כלל המידע על האומן, כולל הביוגראפיה שלו ושלושת שיריו המושמעים ביותר.
3. החזרת אלבומים של אומן (לפי artist_id).

4. החזרת מידע על אלבום (לפי album_id).
5. החזרת כל השירים באלבום (לפי album_id).

שימוש ב-API ומקורות מידע נוספים:

1. https://theaudiodb.com/
מדובר במאגר מידע המכיל נתונים העוסקים במוסיקה ומאפשר את השליפת באמצעות API בפורמט JSON. המאגר מכיל נתונים שונים אודות אמנים, שירים ואלבומים ומהווה את מקור המידע המרכזי עבור האפליקציה שלנו. השליפות ממנו נעשו באמצעות שליחת בקשות GET וקבלת מענה בפורמט JSON.
עבור כל אמן מרשימת אמנים התחלתית (המונה 2,000 אמנים), שלפנו את המידע הקיים לגביו, את המידע לגבי כל אלבומיו וכן את המידע על כל שיר מתוך אלבומים אלו.
2. https://kworb.net/
מאגר מידע נוסף אשר שימש אותנו, ממנו הבאנו את הנתונים על כמות ההשמעות שיש לכל שיר בחלוקה לפי מדינות. באתר יש טבלת השמעות שירים ללכ-64 מדינות שונות בעולם, ועל מנת להשתמש במידע ביצענו web scraping ומכל דף של מדינה שלפנו את הנתונים הללו.
מאתר זה גם לקחנו את רשימת האמנים ההתחלתית עליה התבססנו וכן את רשימת המדינות (יצרנו מהן קובצי csv).

מבנה הקוד:

1. שרת + קליינט:

1. py.run – קובץ ההרצה הראשי של התוכנית המכיל את קונפיגורציית כתובת ה-ip והפורט.
2. templates/__init__.py. – קובץ האתחול של קומפוננטות המערכת.
3. py.views/main/templates – קובץ קונפיגורציית הנתיבים של השרת.
4. py.query/main/templates – קובץ ניהול התקשורת מול שרת ה-MySql.
5. /public/templates – מכילה את פלט הקומפילציה של קומפוננטות ה-JavaScript.
6. /static/templates – מכילה את קובץ ה-html הראשי, ואת קומפוננטות ה-JavaScript אשר בשימוש המערכת.

2. API + DB:

1. Api_data_retriever.py – ביצוע בקשות ל-API של https://theaudiodb.com/ והכנסת הנתונים המתקבלים ל-DB.
2. Charts_retriever.py – הכנסת תוכן טבלאות ההשמעות שבאתר https://kworb.net/ ל-DB
3. Countries_retriever.py – הוספת רשימת המדינות מה-CSV ל-DB.
4. Tables.py – קובץ python המכיל את הפקודות SQL ליצירת הטבלאות של ה-DB.
5. Db_connector – פונקציות להתממשקות עם ה-DB.

ספריות חיצוניות:

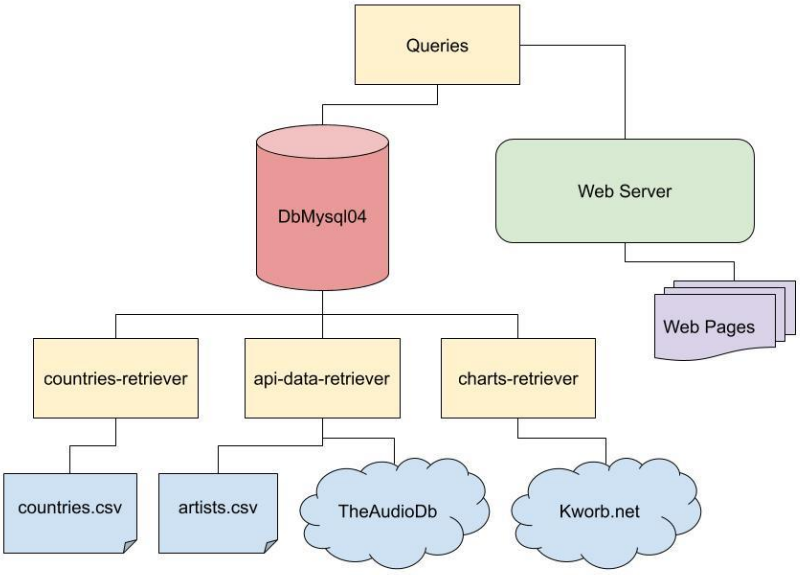
Python Back End:

1. csv- לקריאת קבצי הcsv של האומנים והמדינות
2. requests- לביצוע בקשות GET לאתרים מהם לקחנו את המידע
3. beautifulSoup – ספריה לביצוע parsing של דפי html.
4. mysql.connector – לביצוע הגישה ל-DB
5. flask- ניהול השרת

Front End:

6. React- ספריית JavaScript נבחרת
7. Material UI- יצירת קומפוננטות אינטואיטיביות ואיכותיות.

Flow כללי של האפליקציה:



על ידי שימוש בקליינט, דף אינטרנט המבוסס HTML ו-React, המשתמש יכול לבחור אחת מבין עשר השאילתות המוכנות מראש, ללא קלט של המשתמש, או לבחור באחת מבין שש השאילתות אשר דורשות הכנסת ארגומנטים לבחירתו, ולקבל את המידע בהתאם. כמו כן, על גבי התוצאות המתקבלות, ניתן בלחיצת כפתור על פריט (למשל על שיר שחזר) לקבל מידע מעמיק יותר על אותו פריט.
לכל שאילתא מספר שונה של ארגומנטים לבחירת המשתמש (או שאין כלל), אשר יופיעו לקבלת

קלט, בעת בחירת השאילתא.
כאשר המשתמש ילחץ על כפתור השליחה, פונקציית JavaScript תאסוף את המידע הרלוונטי, תשלח אותו לשרת הflask- אשר יעבד את המידע, ויתקשר מול שרת ה-MySql לשם קבלת המידע.
השרת יעבד את המידע לכדי JSON ויעביר אותו בחזרה לקליינט, אשר יעבד אותו לתוך טבלת המידע הייעודית באתר.