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.

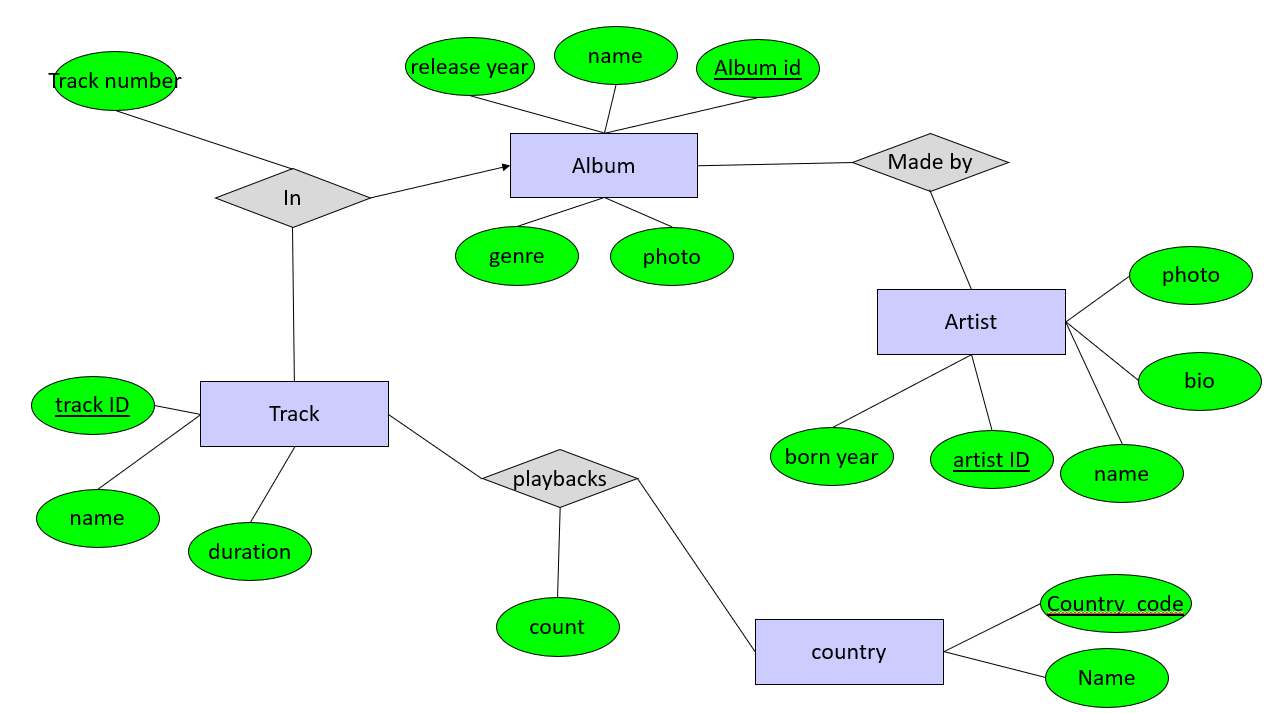
לאחר מכן החלטנו כי אחד הנתונים שחסר באפליקציה ויהיה שימושי למשתמש זה לדעת מה פופולאריות השירים. מצאנו אתר המציג לגבי 64 מדינות שונות, מה היו השירים הכי מושמעים בהן בשבוע האחרון וכמה פעמים הם הושמעו בהן. לכן הוספנו לדיאגרמה ישות מדינה והחלטנו לקשר אותה לשיר. אפשרי היה לחבר אותה לאומן שכן האתר בו השתמשנו 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.

1. 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.

1. **Album** (album\_id, name, release\_year, genre, photo)
2. **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.

1. **Artist** (artist\_id, name, birth\_year, gender, bio, photo)
2. **Country** (country\_code, name)
3. **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 |

1. **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) |

1. **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 |

1. **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) |

1. **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) |

1. **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 | שםקוד 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:

1. 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.
2. 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.
3. 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

1. **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**

1. **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**

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

* פרמטרים לבחירת המשתמש: אין
* התוצאה: עשרה שירים אשר בשמם מופיעה המילה love, ויצאו מלפני שנת 1980.
* המידע המוצג: שם השיר, שם האומן, אורך השיר, שנת יציאת השיר, שם האלבום.
* אופטימיזציה ומבנה הDB: 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.
* השליפה:

**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

1. **שירי חג המולד Christmas Songs**

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

**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

1. **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

1. **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

1. **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**

1. **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}

1. **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 זוINNER 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

1. **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

1. **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**

1. **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

1. **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.

1. <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. .py.\_\_init\_\_/templates – 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:

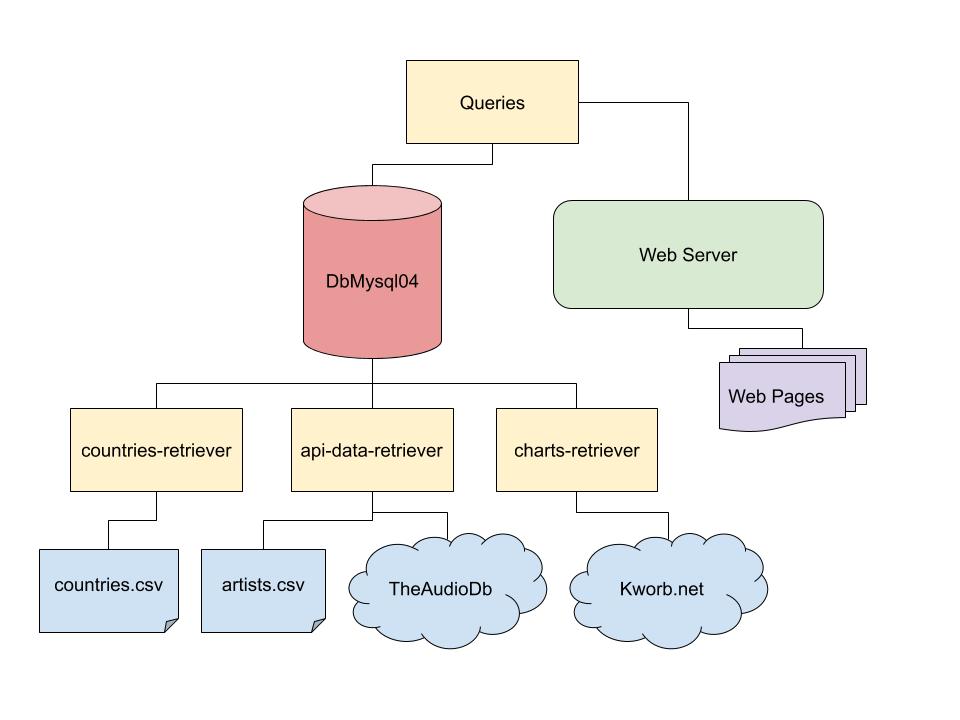
**Python Back End:**

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:**

1. React- ספריית JavaScript נבחרת
2. 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 ה-flask 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.