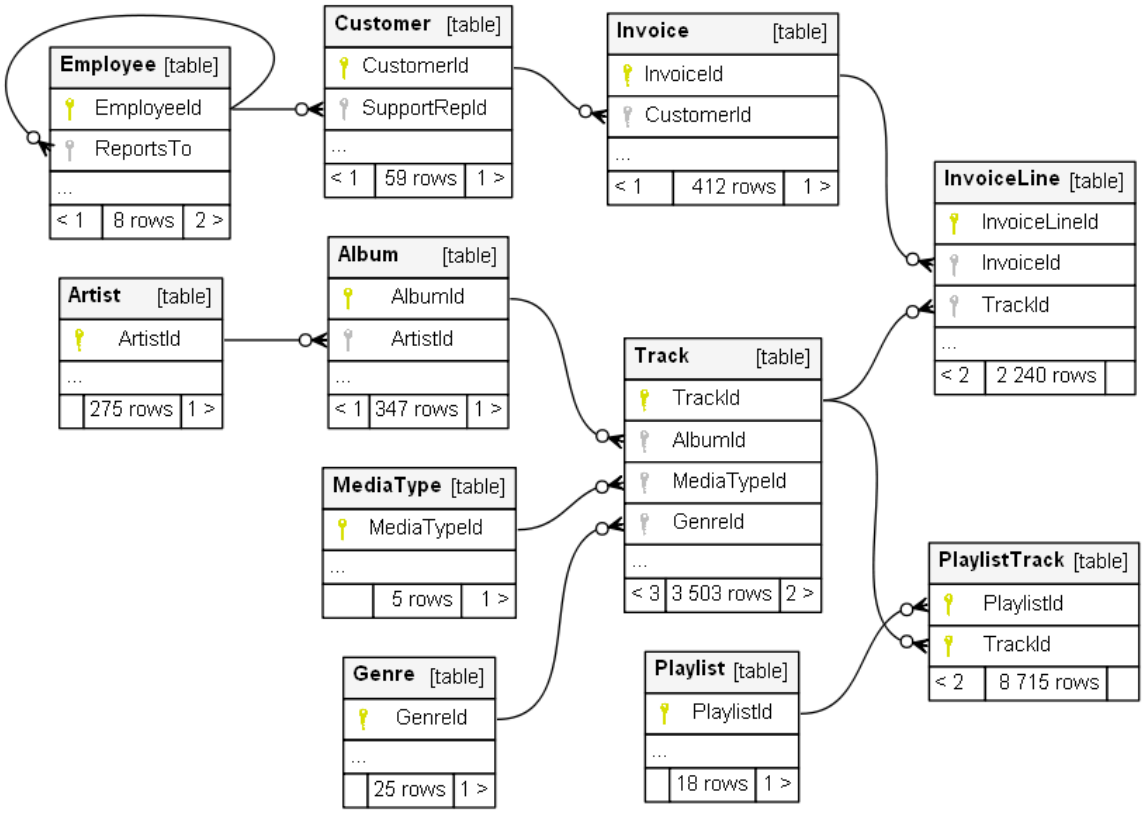
# פרויקט סיום

**לפניכם CHINOOK DATABASE. ה-DB מכיל DATA של שירים מ-ITUNES, הכולל:**

- פלייליסטים
- ז'אנרים
- אמנים ואלבומים
- סוגי קבצים
- רכישות של שירים
- לקוחות
- עובדים שעזרו ללקוחות בכל רכישה
- היררכיה ארגונית של העובדים

להלן הסכמה של ה-DB:



## טעינת בסיס הנתונים

טענו את ה-DB באמצעות קובץ ה-Dump המצורף או באמצעות קובץ ה-sql המצורף.

פקודות cmd בהנחה שהקבצים נמצאים ב-C (שימו לב לגרסה של ה-postgres, עשוי לשנות את הניתוב):

עברו להיות תחת התיקייה הרלוונטית של postgres:

```
cd C:\Program Files\PostgreSQL\15\bin
```

הריצו את קובץ ה-dump או את קובץ פקודות ה-sql:

import with dump file:

```
psql -h 127.0.0.1 -U postgres -f C:\chinook_dump.sql
```

import with sql code:

```
psql -h 127.0.0.1 -U postgres -f C:\chinook_db.sql
```

## פייתון- PANDAS (10 נק')

לפניכם 3 קבצי txt המכילים בתוכם 2 טבלאות המתקשרות לטבלת Employee שב-Database.

לכל Employee יש מחלקה מקושרת – Department. בטבלת employee קיים רק הקישור באמצעות FK שהינו department_id.

הטבלאות המצורפות בשלושת הקבצים מכילות את המחלקה והשם שלה. בנוסף לכל מחלקה יש תתי מחלקות מקושרות ולכל תת מחלקה קיים תקציב.

**המטרה**: להוסיף לסכמת stg טבלה שתכיל את שמות המחלקות ואת התקציב של כל מחלקה (סכום התקציבים של תתי המחלקות). אין צורך להציג את תתי המחלקות ושמן.

הטבלה תראה כך:

department_budget:

| department_id | department_name | budget |
|---|---|---|
| | | |

3 הקבצים המצורפים:

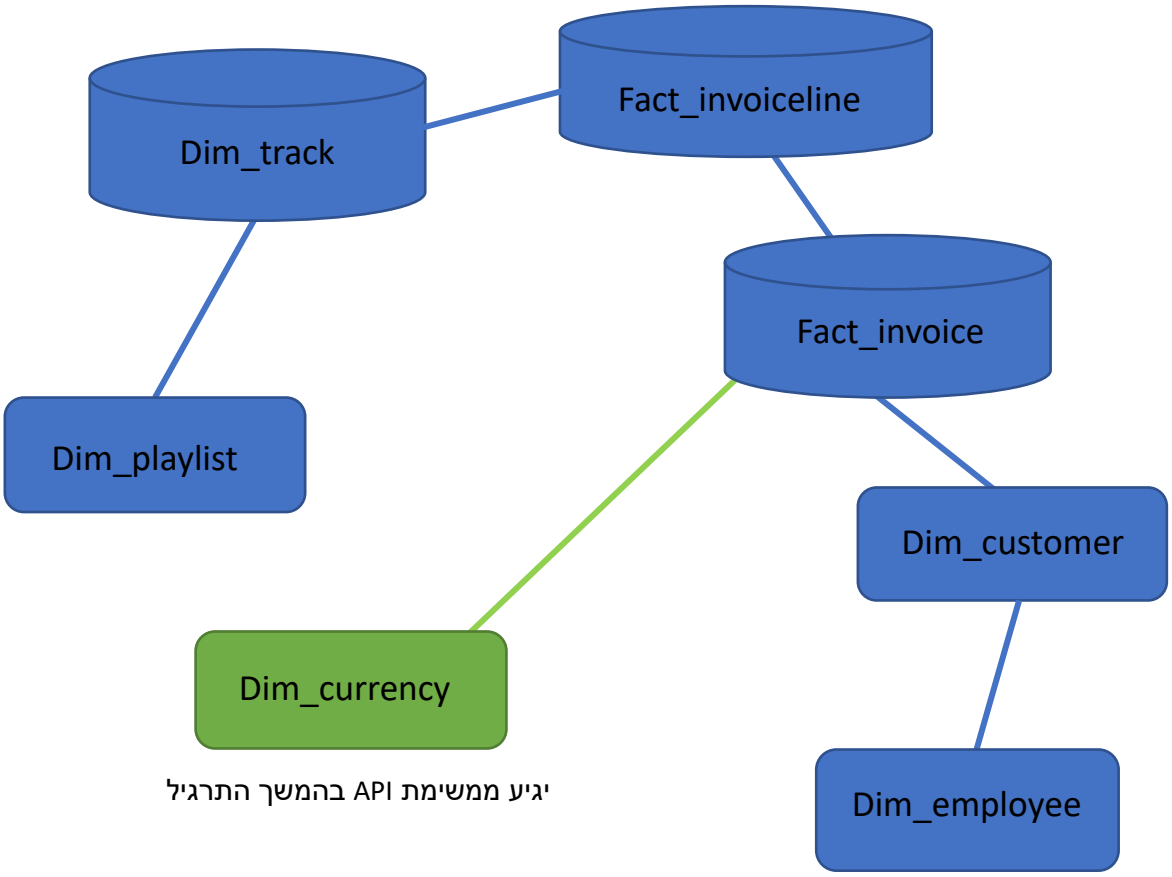
1. קובץ המכיל טבלה של מזהה מחלקה ושם מחלקה. Format הקובץ: ישנה הפרדה של delimiter.
2. קובץ המכיל חלק מטבלת תתי מחלקות ותקציבן. Format הקובץ: json.
3. קובץ המכיל את החלק השני מטבלת תתי מחלקות ותקציבן. Format הקובץ: json הבנוי בפורמט שונה מה-json הראשון.

**המשימה** – השתמשו ב-python בלבד, היעזרו בספריות (pandas, sqlalchemy (create_engine:

1. עליכם לקרוא את הקבצים באמצעות python pandas.
2. ליצור 2 dataframeים אחד עבור כל טבלה כך ש:
   a. dataframe אחד יורכב מ-union של שני קבצי ה-json (כיוון ששניהם מהווים חלק מאותה הטבלה).
   b. ה-dataframe השני יורכב מהקובץ בפורמט ה-delimiter.
3. לבצע join בין שתי הטבלאות ולסכום את התקציב עבור כל מחלקה.
4. להתחבר ל-postgres ל-chinook ולהזרים אליו את הטבלה לסכמת stg.

## DBT (30 נק')

להלן הסכמה הרצויה של ה-data warehouse:




יגיע ממשימת API בהמשך התרגיל

**עבור כל הטבלאות שתיצרו:**

- יש להוסיף שדה שיכיל את זמן הריצה של ה-dbt
- השתמשו ב-source ב-from

Dim_playlist

- הביאו לטבלת dimension את כלל השדות מ-playlisttrack
- חברו את טבלת playlist והביאו ממנה את כלל השדות
- נסו לחשוב, האם יש משהו שדורש התייחסות עקב כך שמדובר בחיבור של שתי טבלאות שונות עם שני תאריכי עדכון שונים? (מבחינת אופי הבאת המידע והתעדכנותו, Materialization וכו')

Dim_customer

- הביאו לטבלת dimension את כלל השדות מ-customer
- הלקוח מעוניין שהשדות של שם פרטי ושם משפחה יהיו בפורמט של אות ראשונה גדולה והשאר קטנות (בחלק מהרשומות במקור כל האותיות גדולות)
- הוסיפו שדה שבו יהיה הדומיין מכתובת המייל

Dim_employee

- הביאו לטבלת dimension את כלל השדות מ-employee
- חברו את טבלת department_budget והוסיפו ממנה את שם המחלקה ואת תקציב המחלקה
- הוסיפו שדה ובו מספר השנים בהם העובד כבר מועסק
- הוסיפו שדה שיכיל את הדומיין מכתובת המייל
- **בונוס** - הוסיפו שדה אשר יצביע האם העובד הוא מנהל is_manager (ערכים 0 או 1), הסבר:
  - השדה reportsto מכיל עבור כל עובד מי המנהל הארגוני שלו (ה-customerid של המנהל)
  - על כן, אם ה-customerid של עובד מסויים קיים בשדה reportsto של עובד כלשהו זה אומר שהוא מנהל
  - השתמשו ב-case שבודק אם ה-customerid של כל עובד קיים ברשימת ה-customerid של המנהלים (השדה reportsto, כאמור)

Dim_track

- הביאו  לטבלת dimension את כלל השדות מ-track
- חברו את טבלאות: album, artist, mediatype, genre והביאו מהן את כלל השדות
- אילו שדות אין צורך להביא מטבלת track?
- הפכו את השדה של משך השיר במילישניות למשך השיר בשניות
  - **בונוס** הוסיפו שדה שיציג את משך השיר כ- MI:SS (למשל, במקום 301 שניות שיהיה רשום 05:01)

Fact_invoice

- הכינו טבלת fact ל-invoice והביאו ממנה את כלל השדות
- האם צריך להביא מ-invoice את השדות של הכתובת? הסבר
- הפכו את הטבלה כך שתהיה Materialized - Incremental

Fact_invoiceline

- הכינו טבלת fact לטבלה invoiceline והביאו ממנה את כלל השדות
- הפכו את הטבלה כך שתהיה Materialized - Incremental

## API CURRENCIES (10 נק')

הסכומים הרשומים ב-Database הם בדולרים. כחלק מניתוח המידע, נרצה להציג חלק מהתוצאות בשקלים. לשם כך, נשתמש ב-API כדי לקבל נתוני המרות מטבע עבור כל הימים הרלוונטיים.

יש מספר שירותים המאפשרים גישה ל-API אשר יכול להחזיר נתוני המרה ושערים בין סוגי מטבעות רבים.

הוציאו טבלה שתכיל עבור כל תאריך רלוונטי את ערך ההמרה מדולר לשקל. תוכלו לבצע את המשימה בדרך שתרצו, להלן דוגמא לדרך אחת:

שלבי העבודה:

1. מצאו את התאריכים הרלוונטיים שיש להביא עבורם מידע.
2. הכינו סקריפט פייתון שיתשאל את ה-API ויביא את יחס ההמרה מדולר לשקל עבור כל יום בתקופת הזמן הרלוונטית.
3. הזרימו את המידע הנ"ל לתוך טבלה ייעודית ב-Data Warehouse.

בכל שירות API תוכלו למצוא דוקומנטציה והסברים לדרך שיש לתשאל אותו, איזה פרמטרים מקבל, מה בדיוק מחזיר וכו'.

## POWER BI (50 נק')

בנו מספר ויזואליזציות וניתוחים על ה-data warehouse. פתרו את הבאים והוסיפו לאחר מכן 3 משלכם.

1. הציגו את ה-top 5 של האמנים עם הכי הרבה אלבומים.
2. הציגו את ה-top 5 של האמנים עם הכי הרבה שירים (tracks).
3. הציגו את ה-top 5 של הז'אנרים עם הכי הרבה שירים.
4. הציגו את הפלייליסט עם הכי הרבה שירים, הפלייליסט עם הכי מעט שירים, ואת ממוצע השירים בפלייליסט.
5. מיהם 5 הלקוחות שרכשו בסכומים הגבוהים ביותר? מה הסכום (בדולרים וגם בשקלים)?
6. הציגו גרף של סכום מכירות עבור כל חודש.
7. האם קיימת קורלציה בין אורך השיר לבין הרווחים שהוא מניב במכירות?
8. הציגו רווחים לפי מדינה עבור 5 המדינות עם הכי הרבה רווחים ו-5 המדינות עם הכי מעט רווחים.
9. בונוס: בתוך כל מדינה משאלה 8, מהו אחוז הרווחים של כל ז'אנר מתוך כלל הרווחים במדינה?

שימו את הגרפים בשני דאשבורדים שונים, והוסיפו פילטרים שיחתכו את שניהם.