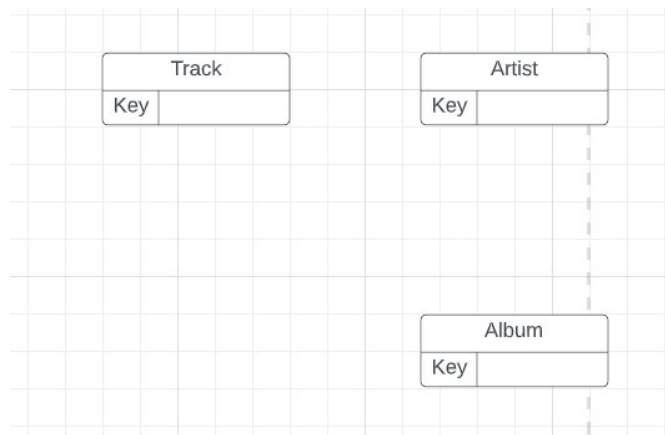
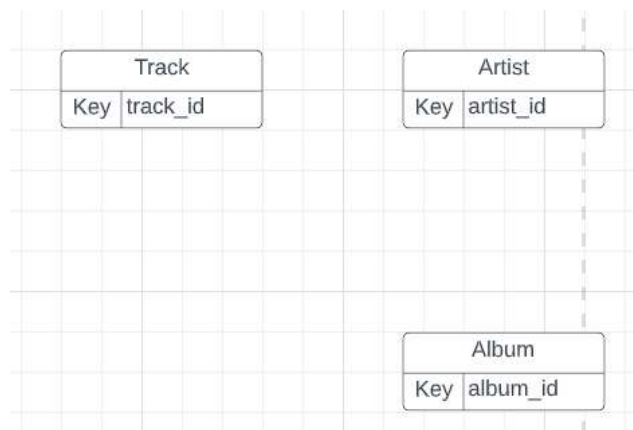


Step1: identify entities:



Step2: identify key attribute/s for each entity.



Step 3: draw the relationships.

According to the missing value check, a track will always be assigned to an album

```
combined_df['artist_id'].isnull().any()
✓ 0.0s
False

combined_df['album_id'].isnull().any()
✓ 0.0s
False

combined_df['track_id'].isnull().any()
✓ 0.0s
False
```

And this part of code shows that a track belongs to one album only, an album belongs to one artist only.

```

# Group by 'TrackID' and count the unique 'AlbumID' occurrences
unique_album_counts = combined_df.groupby('track_id')['album_id'].nunique()

# Check if any 'TrackID' has more than one unique 'AlbumID'
multiple_albums = unique_album_counts[unique_album_counts > 1]

print(multiple_albums)
✓ 0.0s
Series([], Name: album_id, dtype: int64)

# Group by 'track_id' and count the unique 'artist_id' occurrences
unique_album_counts = combined_df.groupby('track_id')['artist_id'].nunique()

# Check if any 'TrackID' has more than one unique 'AlbumID'
multiple_albums = unique_album_counts[unique_album_counts > 1]

print(multiple_albums)
✓ 0.0s
Series([], Name: artist_id, dtype: int64)

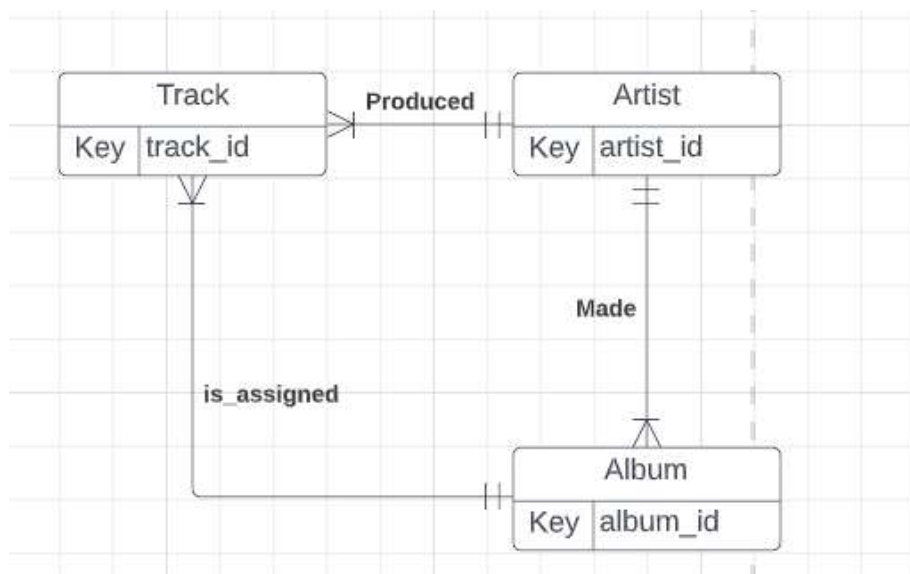
# Group by 'album_id' and count the unique 'artist_id' occurrences
unique_album_counts = combined_df.groupby('album_id')['artist_id'].nunique()

# Check if any 'TrackID' has more than one unique 'AlbumID'
multiple_albums = unique_album_counts[unique_album_counts > 1]

print(multiple_albums)
✓ 0.0s
Series([], Name: artist_id, dtype: int64)

```

So the relationships:

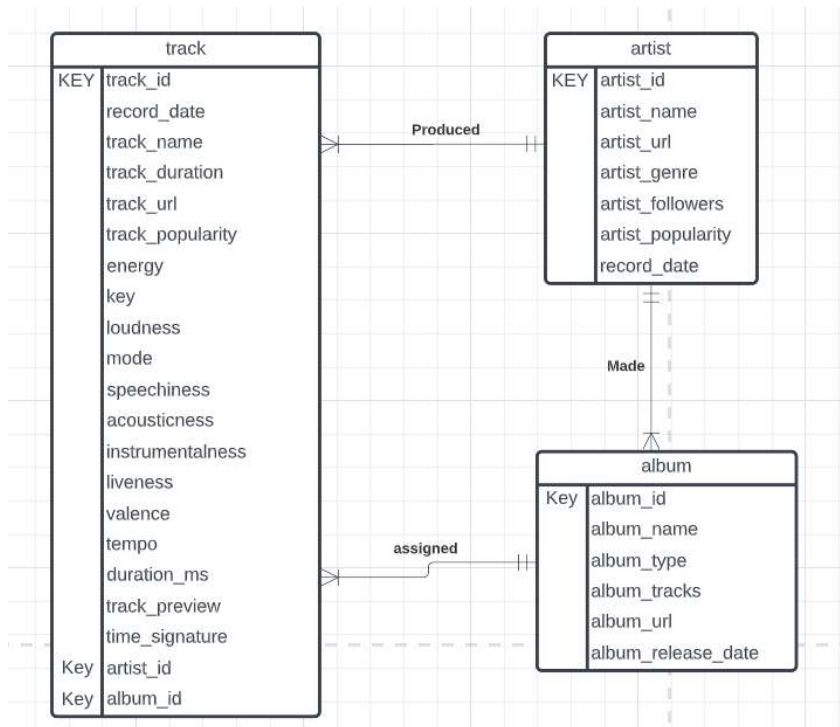


Step 4: add non-key attributes:

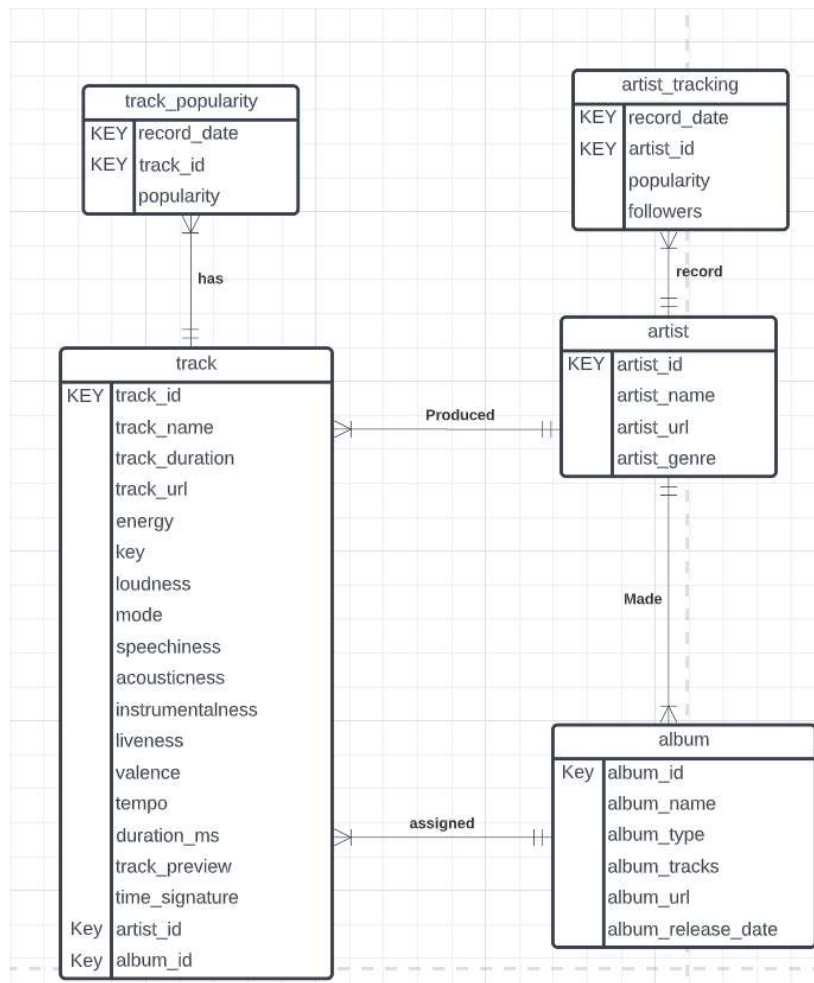
Record_date -> multivalued attribute

Popularity -> multivalued attribute

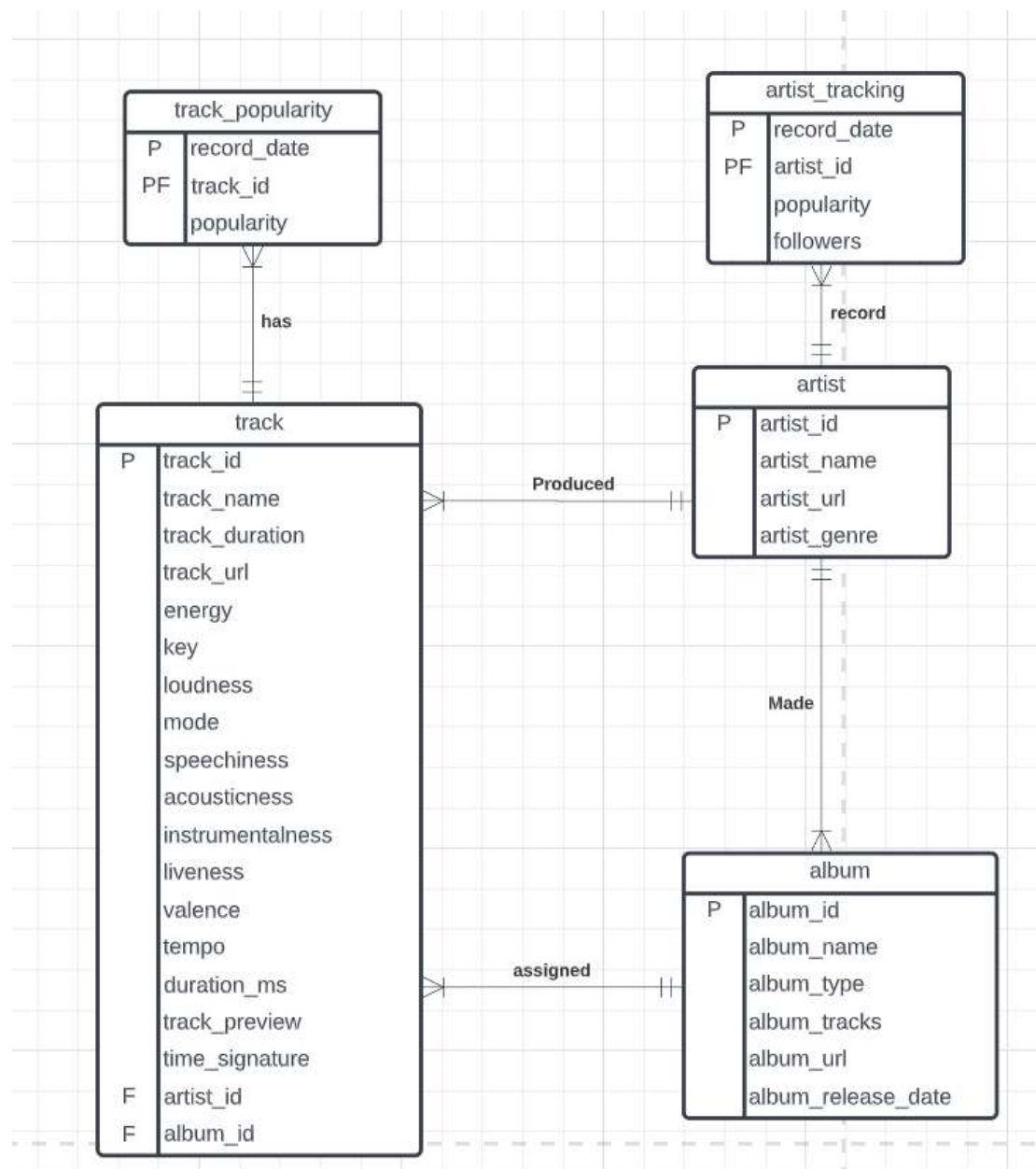
Artist popularity -> multivalued attribute



Step 5: remove multivalued attribute and create a new entity



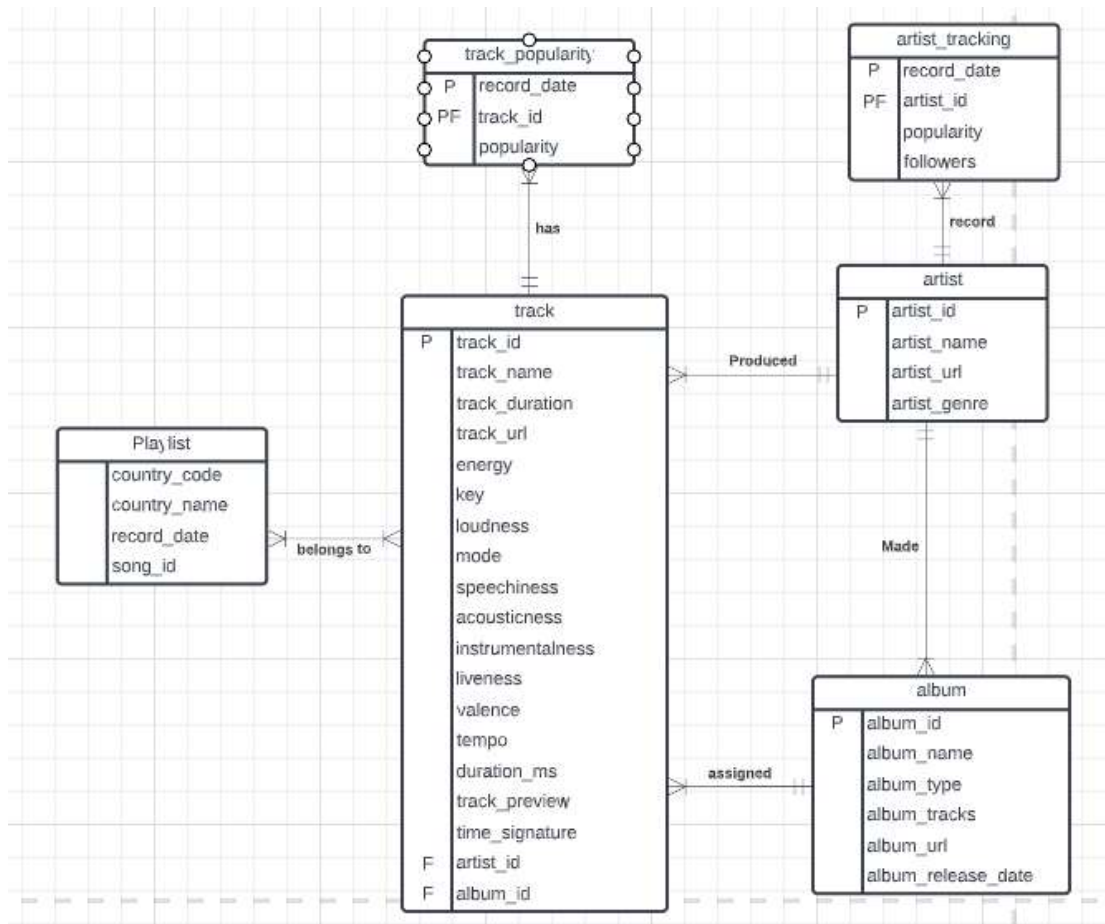
Step 6: Drawing Logical Model, Identify Primary Key and Foreign Key



Step7: Set constraint and add new entity based on case study

case: spotify has top songs playlist for country, all of these tracks are found from those playlists

Action: add playlist entity



A playlist contains one or more tracks, and a track can belong to one or more playlist, now it is a composite entity. Therefore, I will create a bridge entity: `Playlist_track_id`

