Master in AI & DS – A.A. 2024/2025

Progetto per il modulo Databases

Modellazione di una Piattaforma di Streaming Musicale

A cura di: Francesca Aceto, Rosa Maria Bruno, Francesca Ricci

1. Introduzione

Nel contesto attuale, in cui la fruizione dei contenuti multimediali avviene principalmente tramite piattaforme digitali, i servizi di streaming musicale rappresentano una delle principali modalità attraverso cui gli utenti accedono alla musica. Una piattaforma di streaming musicale è un servizio che offre agli utenti l'accesso on-demand a un vasto catalogo di brani musicali, album, podcast, di solito tramite una sottoscrizione a pagamento o fruibile gratuitamente con pubblicità. L'obiettivo principale è quello di fornire all'utente finale un'esperienza di ascolto personalizzata e accessibile su vari dispositivi (smartphone, computer, tablet).

Questo progetto si pone l'obiettivo di modellare un sistema informativo che descriva le principali funzionalità di una piattaforma di streaming musicale, ispirandosi ai moderni servizi di musica on-demand, come Spotify, Apple Music e Amazon Music, con funzionalità che spaziano dalla fruizione di un catalogo musica, all'ascolto di brani fino alla gestione personalizzata della musica tramite creazione di playlist. Per semplicità, vengono considerati solo i brani musicali e non i podcast.

1.2 Motivazione della scelta del contesto

La scelta di una piattaforma di streaming musicale come dominio applicativo per la progettazione nasce da molteplici motivazioni. Innanzitutto, si tratta di un contesto reale, attuale e molto diffuso, con milioni di utenti attivi ogni giorno in tutto il mondo. La complessità intrinseca nella gestione dei contenuti, nelle interazioni utente e nelle dipendenze tra le entità rende questo caso d'uso particolarmente adatto ad un'analisi strutturata tramite strumenti di modellazione concettuale e logica. Dal punto di vista didattico e progettuale, un sistema di questo tipo si presta bene per:

- sperimentare la modellazione di dati complessi e relazioni N:M, con gestione di chiavi primarie ed esterne;

- approfondire le tecniche di progettazione concettuale e logica, tramite diagramma E/R e successiva traduzione in schema relazionale;

- applicare interrogazioni SQL di varia natura (selezioni, join, aggregazioni, query innestate) su uno scenario realistico;

- gettare le basi per eventuali sviluppi futuri, come l'integrazione di meccanismi di analisi predittiva, raccomandazione, clustering e personalizzazione tramite tecniche di machine learning.

Infine, il contesto musicale rappresenta un ambito coinvolgente anche dal punto di vista creativo ed emozionale, e si presta bene ad attività di simulazione e progettazione senza perdere il legame con applicazioni reali e potenzialmente implementabili.

1.3 Obiettivi

Lo scopo principale del progetto è quello di fornire una base di dati per supportare le principali funzionalità di una piattaforma di streaming musicale. Nello specifico, il sistema permetterà di:

- gestire un catalogo di canzoni, album, artisti e generi musicali;

- tenere traccia degli utenti che usufruiscono della piattaforma;

- monitorare le attività degli utenti, tra cui la creazione di playlist personali;

- tracciare ed organizzare le relazioni tra gli utenti e gli artisti seguiti;

- collegare ogni brano ad uno specifico album e ad un genere musicale, mantenendo anche il legame con l'artista o gli artisti che lo interpretano;

- monitorare il numero di riproduzione di brani come possibile metrica di popolarità;

- supportare ricerche e query che permettano, ad esempio, di filtrare contenuti per genere, artista o numero di ascolti.

Questa struttura informativa potrà successivamente essere estesa ed integrata con tool più avanzati per offrire funzionalità quali la raccomandazione personalizzata di brani, l'analisi dei gusti musicali o la generazione di automatica di playlist basate su preferenze esplicite o implicite, aprendo la strada all'integrazione di moduli di intelligenza artificiale.

2. Descrizione del Contesto: Piattaforma di Streaming Musicale

Alla base di una piattaforma di streaming c'è un vasto catalogo di *contenuti musicali*, interpretati da *artisti* e gestiti tipicamente da *case discografiche*. Gli *utenti* sono i principali fruitori e possono cercare brani specifici, navigare per *genere*, *artista* o *album*, creare *playlist* personalizzate e *riprodurre* i brani in streaming. La piattaforma tiene traccia delle abitudini di ascolto degli utenti per offrire suggerimenti personalizzati e statistiche.

Si vuole rappresentare una base dati per la gestione dei contenuti e delle interazioni all'interno di una piattaforma di streaming musicale, tenendo conto delle seguenti informazioni:

- **Contenuti Musicali**: ogni brano musicale, o **canzone**, è caratterizzato da un titolo, una durata, un codice identificativo univoco, una data di uscita con cui viene distribuito al pubblico, un genere musicale rispetto al quale è classificato. I brani musicali sono l'unità fondamentale, sono interpretati da uno o più artisti e possono essere organizzati in album o meno (singoli).

- **Album**: ciascun album ha un codice identificativo, un titolo, una data di pubblicazione con cui viene distribuito al pubblico, e può essere di diverse tipologie (es. album studio, album live, EP, ecc.). Gli album sono attribuiti ad uno o più specifici artisti.

- **Artisti**: un artista è identificato da un codice univoco, un nome d'arte, un nome reale (opzionale), dalla sua nazionalità.

- **Generi Musicali**: i brani sono categorizzati in uno o più generi musicali (es. Pop, Rock, Elettronica, ecc.), ciascuno identificato da un nome univoco.

- **Case Discografiche**: gli album ed i singoli musicali possono essere prodotti e distribuiti da case discografiche, identificate da un nome univoco e da un recapito espresso da indirizzo e telefono.

- **Utente**: è colui che usufruisce del servizio di streaming. Gli utenti della piattaforma sono identificati da una username univoca fornita in fase di registrazione, password (criptata), indirizzo e-mail, data di nascita, Paese da cui ascolta, data di registrazione alla piattaforma, tipo di abbonamento (es. Free, Premium) e data di ultimo accesso. Gli utenti interagiscono con i contenuti della piattaforma in diversi modi: possono riprodurre musica, seguire artisti e creare playlist.

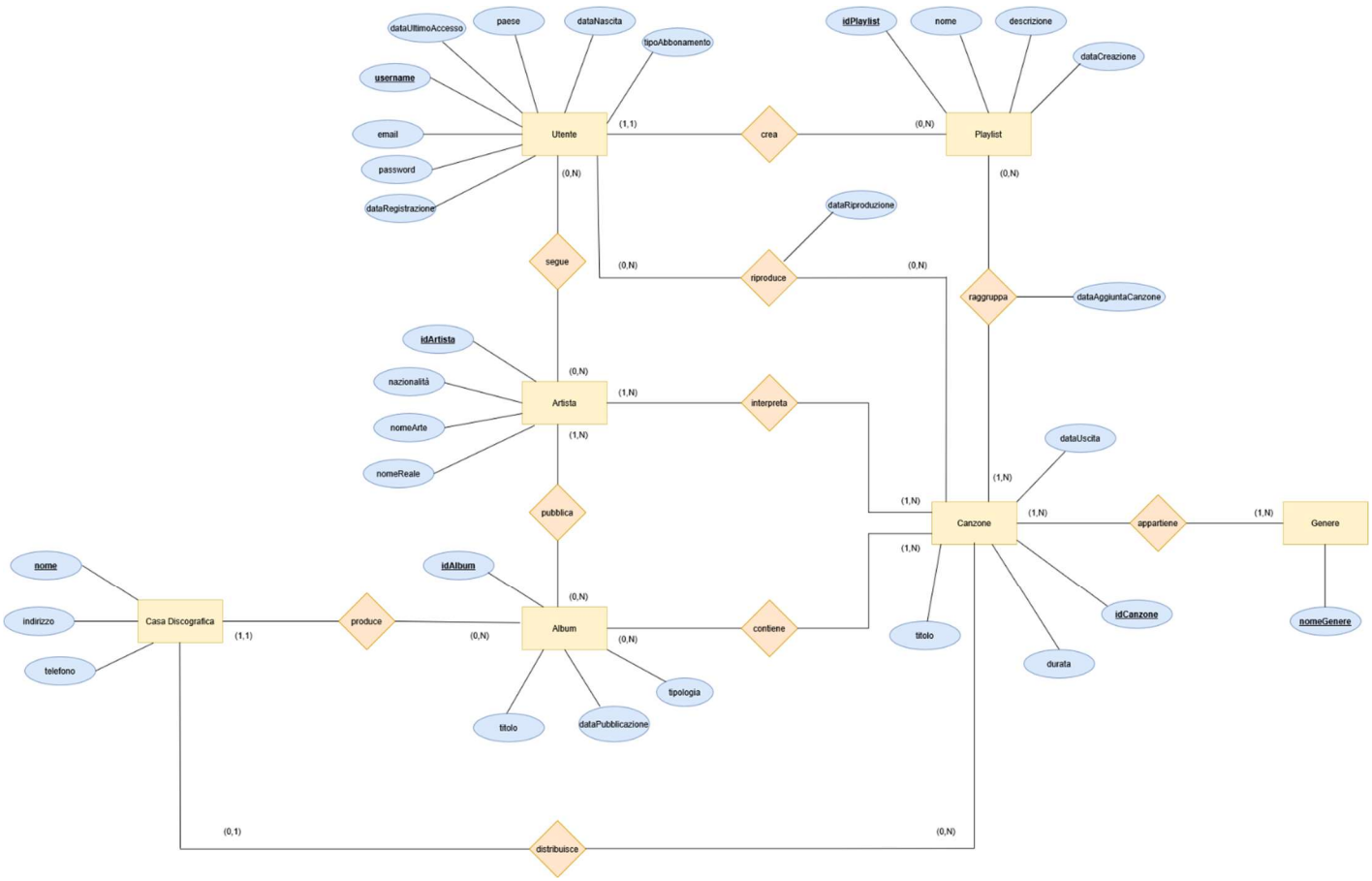
- **Playlist**: gli utenti possono creare liste personalizzate di brani, ovvero playlist personali. Ciascuna playlist è caratterizzata da un id univoco, un nome, una data di creazione, una eventuale descrizione e l'id dell'utente che l'ha creata. Una playlist può contenere uno o più brani musicali.

- **Interazioni Utente**: la piattaforma può memorizzare le sessioni di riproduzione dei brani da parte degli utenti, registrando quale brano è stato ascoltato da quale utente e in quale data/ora, ricavando il numero di ascolti totali degli utenti. Inoltre, può tenere traccia degli artisti che un utente segue e dei brani che un utente ha aggiunto in playlist.

- **Classifiche e Statistiche**: la piattaforma potrebbe aver bisogno di dati aggregati per generare report periodici, ad esempio sui brani più ascoltati, sugli artisti che pubblicano con una data casa discografica, sugli artisti che non hanno pubblicato album, ecc.

3. Progettazione dello schema E/R

Il modello concettuale della base di dati per la modellazione di una piattaforma di streaming musicale è rappresentato dal seguente diagramma E/R:



da cui emergono le entità, le relazioni tra di esse, gli attributi significativi di entità e relazioni, nonché le chiavi identificative.

Entità e attributi

- UTENTE: username, email, password, dataRegistrazione, paese, dataNascita, tipoAbbonamento, dataUltimoAccesso

  Fornisce informazioni sugli utenti registrati, identificati dalla loro username fornita in fase di registrazione.

- ARTISTA: idArtista, nomeArte, nomeReale, nazionalità

  Esprime i dati degli artisti, identificandoli con un id univoco e includendo informazioni circa il nome d'arte, il nome reale e la nazionalità.

- ALBUM: idAlbum, titolo, dataPubblicazione, tipologia

  Permette di modellare il concetto di Album musicale, fornendo informazioni significative quali il titolo, la data di pubblicazione, la tipologia.

- CASA DISCOGRAFICA: nome, indirizzo, telefono

  Rappresenta le etichette discografiche, che possono pubblicare album/singoli brani.

- CANZONE: idCanzone, durata, titolo, dataUscita

  Rappresenta la singola traccia musicale.

- GENERE: nomeGenere

  Fornisce un modo per classificare le canzoni in base al loro stile musicale.

- PLAYLIST: idPlaylist, nome, descrizione, dataCreazione

  Modella il concetto di playlist, ovvero una collezione personalizzata di canzoni creata da un utente.

Relazioni e cardinalità

Le associazioni tra le entità sono espresse tramite relazioni e cardinalità. Dal momento che esistono diversi standard sulla direzione di lettura delle cardinalità delle relazioni, per chiarezza le cardinalità vengono testualmente riportate di seguito:

- CREA, con cardinalità 1:N tra UTENTE e PLAYLIST: un utente può creare (0,N) playlist, una specifica playlist è creata da uno ed un solo utente. La cardinalità minima 0 è un caso valido, dato che un utente potrebbe registrarsi

e non creare necessariamente playlist personalizzate, ma usare la piattaforma solo per ascoltare playlist predefinite o solo canzoni;

- SEGUE, con cardinalità N:N tra UTENTE e ARTISTA: permette di capire quali artisti un utente eventualmente segue. Un utente può infatti seguire (0,N) artisti, un artista può essere seguito da (0,N) utenti;

- RIPRODUCE, con cardinalità N:N tra UTENTE e CANZONE ed attributo dataRiproduzione: permette di tenere traccia delle riproduzioni di canzoni da parte degli utenti, registrandone la data di riproduzione. In particolare, un utente può riprodurre (0,N) canzoni, una canzone può essere riprodotta da (0,N) utenti. La cardinalità minima 0 è un caso valido e comune poiché un utente appena registrato, o un utente che semplicemente non ha ancora ascoltato nulla, avrà infatti riprodotto zero canzoni.

- RAGGRUPPA, con cardinalità N:N tra PLAYLIST e CANZONE ed attributo dataAggiuntaCanzone: una playlist contiene (1,N) canzoni, una canzone può essere contenuta in (0,N) playlist;

- PUBBLICA, con cardinalità N:N tra ARTISTA e ALBUM: un artista può pubblicare (0,N) album, un dato album può essere pubblicato da (1,N) artisti. Esistono infatti artisti che hanno pubblicato solo singoli, ma non album;

- CONTIENE, con cardinalità N:N tra ALBUM e CANZONE: permette di capire le canzoni che appartengono ad un album. In particolare, un album contiene (1:N) canzoni, una canzone può essere in più album (ad esempio, un brano originale su un album studio e poi su un "greatest hits"), ma può anche non essere incluso in alcun album ed esistere come singolo autonomo;

- INTERPRETA, con cardinalità N:N tra ARTISTA e CANZONE: associa le canzoni agli artisti che le interpretano. Un artista interpreta (1,N) canzoni, una canzone è interpretata da (1,N) artisti (un esempio sono i *featuring*, ovvero le collaborazioni tra artisti diversi);

- PRODUCE, con cardinalità 1:N tra CASA DISCOGRAFICA e ALBUM: una casa discografica può produrre (0,N) album, un album è prodotto da una casa discografica;

- DISTRIBUISCE, con cardinalità 1:N tra CASA DISCOGRAFICA e CANZONE: una canzone può essere o non essere distribuita (0,1) da una casa discografica; analogamente, una casa discografica può distribuire (0,N) canzoni. Esistono infatti sempre più artisti indipendenti che pubblicano i loro brani direttamente sulle piattaforme di streaming. Per questo, un brano non è necessariamente associato ad una casa discografica;

- APPARTIENE, con cardinalità N:N tra CANZONE e GENERE: classifica le canzoni in base al loro genere musicale. Una canzone appartiene ad uno o più (1,N) generi musicali, un genere musicale classifica una o più (1,N) canzoni.

4. Realizzazione dello schema logico

Lo schema E/R sviluppato può essere convertito in uno schema logico relazionale. Le entità e le relazioni individuate devono essere mappate opportunamente in tabelle del database, tenendo conto degli attributi, delle chiavi primarie ed esterne e delle tabelle associative derivate dalle relazioni molti-a-molti.

Tabelle principali

1) **UTENTE** (**username**, email, password, dataRegistrazione, paese, dataNascita, tipoAbbonamento, dataUltimoAccesso)

   PK: username

2) **ARTISTA** (**idArtista**, nomeArte, nomeReale, nazionalita)

   PK: idArtista

3) **CASA_DISCOGRAFICA** (**nome**, indirizzo, telefono)

   PK: nome

4) **ALBUM** (**idAlbum**, titolo, dataPubblicazione, tipologia, **casaDiscografica**)

   PK: idAlbum

   FK: CASA_DISCOGRAFICA.nome

5) **CANZONE** (**idCanzone**, durata, titolo, dataUscita, **casaDiscografica**)

   PK: idCanzone

   FK: CASA_DISCOGRAFICA.nome

6) **GENERE**(**nomeGenere**)

   PK: nomeGenere

7) **PLAYLIST**(**idPlaylist**, nome, descrizione, dataCreazione, **utente)**

   PK: idPlaylist

   FK: UTENTE.username

Nel modello concettuale sono presenti relazioni uno-a-molti; per permettere di mantenere il collegamento tra le entità e garantire l'integrità referenziale, nel modello logico la chiave primaria dell'entità dal lato "uno" viene inserita come chiave esterna nell'entità dal lato "molti". È quello che succede con le tabelle **ALBUM** e **CANZONE** rispetto a CASA_DISCOGRAFICA, e con la tabella **PLAYLIST** rispetto ad UTENTE.

Relazioni e Tabelle associative

8) Relazione SEGUE (UTENTE --- ARTISTA)

   SEGUE(**utente**, **idArtista**)

   PK: utente, idArtista

   FK: UTENTE.username, ARTISTA.idArtista

9) Relazione PUBBLICA (ARTISTA --- ALBUM)

   PUBBLICAZIONE_ALBUM(**idArtista**, **idAlbum**)

   PK: idArtista, idAlbum

   FK: ARTISTA.idArtista, ALBUM.idAlbum

10) Relazione CONTIENE (ALBUM --- CANZONE)

    ALBUM_CANZONE(**idAlbum**, **idCanzone**)

    PK: idAlbum, idCanzone

    FK: ALBUM.idAlbum, CANZONE.idCanzone

11) Relazione INTERPRETA (ARTISTA --- CANZONE)

    INTERPRETAZIONE_CANZONE(**idArtista**, **idCanzone**)

    PK: idArtista, idCanzone

FK: ARTISTA.idArtista, CANZONE.idCanzone

12) Relazione APPARTIENE (CANZONE --- GENERE)

GENERE_CANZONE(**nomeGenere**, **idCanzone**)

PK: nomeGenere, idCanzone

FK: GENERE.nomeGenere, CANZONE.idCanzone

13) Relazione RAGGRUPPA (PLAYLIST --- CANZONE)

PLAYLIST_CANZONE(**idPlaylist**, **idCanzone,** dataAggiuntaCanzone)

PK: idPlaylist, idCanzone

FK: PLAYLIST.idPlaylist, CANZONE.idCanzone

14) Relazione RIPRODUCE (UTENTE --- CANZONE)

RIPRODUZIONE_CANZONE(**utente**, **idCanzone**, dataRiproduzione)

PK: utente, idCanzone

FK: UTENTE.username, CANZONE.idCanzone

Come si può vedere, le relazioni molti-a-molti presenti nel modello concettuale vengono trasformate in tabelle associative nel modello logico, in conformità alle regole di progettazione relazionale. Ogni relazione molti-a-molti viene rappresentata tramite una nuova tabella che include come chiavi esterne le chiavi primarie delle entità coinvolte, costituendo così una chiave primaria composta.

4.1 Creazione Schema e Tabelle nel DBMS "MySQL"

Il DBMS di riferimento con cui modellare lo schema del database per la rappresentazione della piattaforma di streaming musicale è MySQL. A supporto della creazione delle tabelle e dell'inserimento di valori si rivela molto utile il tool visuale MySQL Workbench.

In questa sezione sono riportate le istruzioni per la creazione del database "*piattaforma_streaming_musicale*" e delle tabelle descritte nella sezione precedente. Ogni tabella include gli attributi individuati per descrivere al meglio il contesto di interesse, le chiavi primarie (PK) e le chiavi esterne (FK) necessarie per garantire l'integrità referenziale.

Tramite i seguenti script è possibile definire la struttura completa del database relazionale, in modo da supportare le funzionalità descritte in precedenza per la creazione e gestione di artisti, utenti, canzoni e generi musicali, case discografiche, playlist e le relative interazioni.

- Creazione DB

```
mysql>
mysql> create database piattaforma_streaming_musicale;
```

```
mysql> use piattaforma_streaming_musicale;
Database changed
mysql>
```

- Tabella GENERE

```
mysql> CREATE TABLE genere (nomeGenere CHAR(25), PRIMARY KEY(nomeGenere));
Query OK, 0 rows affected (0.633 sec)
```

- Tabella CASA_DISCOGRAFICA

```
mysql> CREATE TABLE casa_discografica (nome VARCHAR(50) PRIMARY KEY, indirizzo VARCHAR(50),
 telefono VARCHAR(15));
Query OK, 0 rows affected (0.156 sec)
```

- Tabella ALBUM

```
mysql> CREATE TABLE album (idAlbum NUMERIC PRIMARY KEY, titolo VARCHAR(50) NOT NULL,
tipologia VARCHAR(30), dataPubblicazione DATE NOT NULL, casaDiscografica VARCHAR(50)
NOT NULL, FOREIGN KEY (casaDiscografica) REFERENCES casa_discografica(nome));
Query OK, 0 rows affected (0.222 sec)
```

- Tabella ARTISTA

```
mysql> CREATE TABLE artista (idArtista NUMERIC PRIMARY KEY, nomeArte VARCHAR(100) NOT
 NULL, nomeReale VARCHAR(100), nazionalita VARCHAR(50));
Query OK, 0 rows affected (0.210 sec)
```

Di seguito si riporta screen di creazione delle restanti tabelle, realizzate tramite interfaccia di MySQL Workbench.

- Tabella UTENTE

```
CREATE TABLE `utente` (
  `username` varchar(25) NOT NULL,
  `email` varchar(50) NOT NULL,
  `password` varchar(15) NOT NULL,
  `dataNascita` date NOT NULL,
  `paese` varchar(15) DEFAULT NULL,
  `dataRegistrazione` datetime NOT NULL,
  `tipoAbbonamento` varchar(15) NOT NULL DEFAULT 'Free',
  `dataUltimoAccesso` datetime NOT NULL,
  PRIMARY KEY (`username`),
  UNIQUE KEY `email_UNIQUE` (`email`),
  UNIQUE KEY `username_UNIQUE` (`username`)
)
```

- Tabella CANZONE

```
CREATE TABLE `canzone` (
  `idCanzone` int NOT NULL AUTO_INCREMENT,
  `titolo` varchar(100) NOT NULL,
  `durata` time DEFAULT NULL,
  `dataUscita` date DEFAULT NULL,
  `casaDiscografica` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`idCanzone`),
  KEY `casaDiscografica_idx` (`casaDiscografica`),
  CONSTRAINT `casaDiscografica` FOREIGN KEY (`casaDiscografica`) REFERENCES `casa_discografica` (`nome`)
)
```

- Tabella PLAYLIST

```
CREATE TABLE `playlist` (
  `idPlaylist` int NOT NULL AUTO_INCREMENT,
  `nome` varchar(50) NOT NULL,
  `descrizione` text,
  `dataCreazione` date NOT NULL,
  `utente` varchar(50) NOT NULL,
  PRIMARY KEY (`idPlaylist`),
  KEY `playlist_ibfk_1` (`utente`),
  CONSTRAINT `playlist_ibfk_1` FOREIGN KEY (`utente`) REFERENCES `utente` (`username`)
)
```

- Tabella SEGUE

```
CREATE TABLE `segue` (
  `utente` varchar(25) NOT NULL,
  `idArtista` int NOT NULL,
  PRIMARY KEY (`utente`,`idArtista`),
  KEY `segue_ibfk_2` (`idArtista`),
  CONSTRAINT `segue_ibfk_1` FOREIGN KEY (`utente`) REFERENCES `utente` (`username`),
  CONSTRAINT `segue_ibfk_2` FOREIGN KEY (`idArtista`) REFERENCES `artista` (`idArtista`)
)
```

- Tabella PUBBLICAZIONE_ALBUM

```
CREATE TABLE `pubblicazione_album` (
  `idArtista` int NOT NULL,
  `idAlbum` int NOT NULL,
  PRIMARY KEY (`idArtista`,`idAlbum`),
  KEY `idAlbum` (`idAlbum`),
  CONSTRAINT `pubblicazione_album_ibfk_1` FOREIGN KEY (`idArtista`) REFERENCES `artista` (`idArtista`),
  CONSTRAINT `pubblicazione_album_ibfk_2` FOREIGN KEY (`idAlbum`) REFERENCES `album` (`idAlbum`)
)
```

- Tabella ALBUM_CANZONE

```
CREATE TABLE `album_canzone` (
  `idAlbum` int NOT NULL,
  `idCanzone` int NOT NULL,
  PRIMARY KEY (`idAlbum`,`idCanzone`),
  KEY `idCanzone` (`idCanzone`),
  CONSTRAINT `album_canzone_ibfk_1` FOREIGN KEY (`idAlbum`) REFERENCES `album` (`idAlbum`),
  CONSTRAINT `album_canzone_ibfk_2` FOREIGN KEY (`idCanzone`) REFERENCES `canzone` (`idCanzone`)
)
```

- Tabella INTERPRETAZIONE_CANZONE

```
CREATE TABLE `interpretazione_canzone` (
  `idArtista` int NOT NULL,
  `idCanzone` int NOT NULL,
  PRIMARY KEY (`idArtista`,`idCanzone`),
  KEY `idCanzone` (`idCanzone`),
  CONSTRAINT `interpretazione_canzone_ibfk_1` FOREIGN KEY (`idArtista`) REFERENCES `artista` (`idArtista`),
  CONSTRAINT `interpretazione_canzone_ibfk_2` FOREIGN KEY (`idCanzone`) REFERENCES `canzone` (`idCanzone`)
)
```

- Tabella GENERE_CANZONE

```
CREATE TABLE `genere_canzone` (
  `idCanzone` int NOT NULL,
  `nomeGenere` varchar(50) NOT NULL,
  PRIMARY KEY (`idCanzone`,`nomeGenere`),
  KEY `nomeGenere` (`nomeGenere`),
  CONSTRAINT `genere_canzone_ibfk_1` FOREIGN KEY (`idCanzone`) REFERENCES `canzone` (`idCanzone`),
  CONSTRAINT `genere_canzone_ibfk_2` FOREIGN KEY (`nomeGenere`) REFERENCES `genere` (`nomeGenere`)
)
```

- Tabella PLAYLIST_CANZONE

```
CREATE TABLE `playlist_canzone` (
  `idPlaylist` int NOT NULL,
  `idCanzone` int NOT NULL,
  `dataAggiuntaCanzone` datetime NOT NULL,
  PRIMARY KEY (`idPlaylist`,`idCanzone`),
  KEY `idCanzone` (`idCanzone`),
  CONSTRAINT `playlist_canzone_ibfk_1` FOREIGN KEY (`idPlaylist`) REFERENCES `playlist` (`idPlaylist`),
  CONSTRAINT `playlist_canzone_ibfk_2` FOREIGN KEY (`idCanzone`) REFERENCES `canzone` (`idCanzone`)
)
```

- Tabella RIPRODUZIONE_CANZONE

```
CREATE TABLE `riproduzione_canzone` (
  `utente` varchar(25) NOT NULL,
  `idCanzone` int NOT NULL,
  `dataRiproduzione` datetime NOT NULL,
  PRIMARY KEY (`utente`,`idCanzone`),
  KEY `idCanzone` (`idCanzone`),
  CONSTRAINT `riproduzione_canzone_ibfk_1` FOREIGN KEY (`utente`) REFERENCES `utente` (`username`),
  CONSTRAINT `riproduzione_canzone_ibfk_2` FOREIGN KEY (`idCanzone`) REFERENCES `canzone` (`idCanzone`)
)
```

Ecco come appare infine il DB completo di tutte le tabelle:

4.2 Popolamento delle tabelle con dati di esempio

- Popolamento tabella UTENTE

```
INSERT INTO utente VALUES
('luca_rossi', 'luca.rossi@mail.com', '5f4c28410a3a60ba38d1588c5a8f78a5971f9962', '1990-05-15', 'Italia',
'2024-01-01 23:00:56', 'Premium', '2025-03-12 02:54:10'),
('marioB', 'mario.b@example.com', '64cd299ebc50489dee363c5f606677340031b177', '1988-11-30', 'Italia',
'2024-02-15 13:03:00', 'Base', '2025-02-20 20:02:03'),
('vincent', 'vincent.aronne95@example.com', '9d321a4881ee91e45a2a5625b045d39f5b98dbbd', '1995-07-22',
'Francia', '2023-11-10 20:07:50', 'Premium', '2024-06-15 10:20:05'),
('johnSmith92', 'john.smith@example.com', '58f831106d1d1c79fa5f9f83590d206246ca85de', '1992-03-18',
'USA', '2024-01-20 10:10:02', 'Premium', '2025-07-01 08:05:03'),
('emma_jones', 'emma.j@example.com', '6f1c15458c34d33c76ab2db672cc166f1a0c5314', '1998-09-05', 'UK',
'2023-12-05 05:02:10', 'Base', '2025-05-25 01:03:24');
```

```
mysql> INSERT INTO utente VALUES('luca_rossi', 'luca.rossi@mail.com', '5f4c28410a3a60
ba38d1588c5a8f78a5971f9962', '1990-05-15', 'Italia', '2024-01-01 23:00:56', 'Premium'
, '2025-03-12 02:54:10'),('marioB', 'mario.b@example.com', '64cd299ebc50489dee363c5f6
06677340031b177', '1988-11-30', 'Italia', '2024-02-15 13:03:00', 'Base', '2025-02-20
20:02:03'),('vincent', 'vincent.aronne95@example.com', '9d321a4881ee91e45a2a5625b045d
39f5b98dbbd', '1995-07-22', 'Francia', '2023-11-10 20:07:50', 'Premium', '2024-06-15
10:20:05'),('johnSmith92', 'john.smith@example.com', '58f831106d1d1c79fa5f9f83590d206
246ca85de', '1992-03-18', 'USA', '2024-01-20 10:10:02', 'Premium', '2025-07-01 08:05:
03'),('emma_jones', 'emma.j@example.com', '6f1c15458c34d33c76ab2db672cc166f1a0c5314',
 '1998-09-05', 'UK', '2023-12-05 05:02:10', 'Base', '2025-05-25 01:03:24');
```

- Popolamento tabella CASA_DISCOGRAFICA

```
INSERT INTO casa_discografica VALUES ('Interscope Records', 'Santa Monica, California', '+1-310-865-1000');
INSERT INTO casa_discografica VALUES ('Columbia Records', 'New York City, New York', '+1-212-833-4000');
INSERT INTO casa_discografica VALUES ('Roc-A-Fella Records', 'New York City, New York', '+1-212-841-8000');
INSERT INTO casa_discografica VALUES ('XL Recordings', 'London, England', '+44-20-7636-7777');
INSERT INTO casa_discografica VALUES ('Carosello Records', 'Milano, Italia', '+39-02-3659-1240');
```

- Popolamento tabella ARTISTA

```
INSERT INTO artista VALUES (1, 'Marshall Mathers', 'Eminem', 'USA');
INSERT INTO artista VALUES (2, 'Adele Laurie Blue Adkins', 'Adele', 'UK');
INSERT INTO artista VALUES (3, 'Shawn Corey Carter', 'Jay-Z', 'USA');
INSERT INTO artista VALUES (4, 'Billie Eilish Pirate Baird O''Connell', 'Billie Eilish', 'USA');
INSERT INTO artista VALUES (5, 'Francesco Tarducci', 'Nesli', 'Italia');
INSERT INTO artista VALUES (6, 'Stefano Tognini', 'Marracash', 'Italia');
INSERT INTO artista VALUES (7, 'Robyn Fenty', 'Rihanna', 'Barbados');
```

- Popolamento tabella GENERE

```
INSERT INTO genere VALUES ('Pop');
INSERT INTO genere VALUES ('Rock');
INSERT INTO genere VALUES ('Hip-Hop');
INSERT INTO genere VALUES ('Alternative');
INSERT INTO genere VALUES ('R&B');
INSERT INTO genere VALUES ('Soul');
```

- Popolamento tabelle ALBUM, CANZONE e tabella associative coinvolte

```
-- Album 1: Eminem - The Marshall Mathers LP
INSERT INTO album VALUES (1, 'The Marshall Mathers LP', 'Studio', '2000-05-23', 'Interscope Records');
INSERT INTO canzone VALUES (1, 'The Real Slim Shady', '00:04:44', '2000-05-23', 'Interscope Records');
INSERT INTO album_canzone VALUES (1, 1);
INSERT INTO genere_canzone VALUES (1, 'Hip-Hop');
INSERT INTO interpretazione_canzone VALUES (1, 1);

INSERT INTO canzone VALUES (2, 'Stan', '00:06:44', '2000-11-21', 'Interscope Records');
INSERT INTO album_canzone VALUES (1, 2);
INSERT INTO genere_canzone VALUES (2, 'Hip-Hop');
INSERT INTO interpretazione_canzone VALUES (1, 2);


-- Album 2: Adele - 21
INSERT INTO album VALUES (2, '21', 'Studio', '2011-01-24', 'Columbia Records');
INSERT INTO canzone VALUES (3, 'Rolling in the Deep', '00:03:48', '2010-11-29', 'Columbia Records');
INSERT INTO album_canzone VALUES (2, 3);
INSERT INTO genere_canzone VALUES (3, 'Soul');
INSERT INTO interpretazione_canzone VALUES (2, 3);

INSERT INTO canzone VALUES (4, 'Someone Like You', '00:04:45', '2011-01-24', 'Columbia Records');
INSERT INTO album_canzone VALUES (2, 4);
INSERT INTO genere_canzone VALUES (4, 'Soul');
INSERT INTO interpretazione_canzone VALUES (2, 4);


-- Album 3: Jay-Z - The Blueprint
INSERT INTO album VALUES (3, 'The Blueprint', 'Studio', '2001-09-11', 'Roc-A-Fella Records');
INSERT INTO canzone VALUES (5, 'Izzo (H.O.V.A.)', '00:04:00', '2001-08-20', 'Roc-A-Fella Records');
INSERT INTO album_canzone VALUES (3, 5);
INSERT INTO genere_canzone VALUES (5, 'Hip-Hop');
INSERT INTO interpretazione_canzone VALUES (3, 5);

INSERT INTO canzone VALUES (6, 'Takeover', '00:05:13', '2001-09-11', 'Roc-A-Fella Records');
INSERT INTO album_canzone VALUES (3, 6);
INSERT INTO genere_canzone VALUES (6, 'Hip-Hop');
INSERT INTO interpretazione_canzone VALUES (3, 6);


-- Album 4: Billie Eilish - When We All Fall Asleep
INSERT INTO album VALUES (4, 'When We All Fall Asleep, Where Do We Go?', 'Studio', '2019-03-29', 'Interscope Records');
INSERT INTO canzone VALUES (7, 'bad guy', '00:03:14', '2019-03-29', 'Interscope Records');
INSERT INTO album_canzone VALUES (4, 7);
INSERT INTO genere_canzone VALUES (7, 'Pop');
INSERT INTO interpretazione_canzone VALUES (4, 7);

INSERT INTO canzone VALUES (8, 'when the party''s over', '00:03:16', '2018-10-17', 'Interscope Records');
INSERT INTO album_canzone VALUES (4, 8);
INSERT INTO genere_canzone VALUES (8, 'Alternative');
INSERT INTO interpretazione_canzone VALUES (4, 8);


-- Album 5: Nesli - 1989
INSERT INTO album VALUES (5, '1989', 'Studio', '2012-10-16', 'Carosello Records');
INSERT INTO canzone VALUES (9, 'Buongiorno', '00:03:45', '2012-09-10', 'Carosello Records');
INSERT INTO album_canzone VALUES (5, 9);
INSERT INTO genere_canzone VALUES (9, 'Hip-Hop');
INSERT INTO interpretazione_canzone VALUES (5, 9);

-- Collaborazione: Eminem feat. Rihanna - Love The Way You Lie
INSERT INTO canzone VALUES (10, 'Love The Way You Lie', '00:04:23', '2010-06-22', 'Interscope Records');
INSERT INTO album_canzone VALUES (1, 10);  -- Appartiene all'album di Eminem
INSERT INTO genere_canzone VALUES (10, 'Hip-Hop');
INSERT INTO interpretazione_canzone VALUES (1, 10);
INSERT INTO interpretazione_canzone VALUES (7, 10);  -- Rihanna partecipa
```

- Popolamento tabella PUBBLICAZIONE_ALBUM

```
INSERT INTO pubblicazione_album VALUES (1, 1);  -- Eminem pubblica The Marshall Mathers LP
INSERT INTO pubblicazione_album VALUES (2, 2);  -- Adele pubblica 21
INSERT INTO pubblicazione_album VALUES (3, 3);  -- Jay-Z pubblica The Blueprint
INSERT INTO pubblicazione_album VALUES (4, 4);  -- Billie Eilish pubblica When We All Fall Asleep
INSERT INTO pubblicazione_album VALUES (5, 5);  -- Nesli pubblica 1989
```

- Popolamento tabelle PLAYLIST e PLAYLIST_CANZONE

```
INSERT INTO playlist VALUES (1, 'Hip-Hop Classics', 'I migliori classici hip-hop', '2024-03-01', 'luca_rossi');
INSERT INTO playlist_canzone VALUES (1, 1, '2025-06-02 01:07:17');  -- The Real Slim Shady
INSERT INTO playlist_canzone VALUES (1, 5, '2025-04-12 10:01:10');  -- Izzo (H.O.V.A.)
INSERT INTO playlist_canzone VALUES (1, 6, '2025-05-22 00:01:12');  -- Takeover

INSERT INTO playlist VALUES (2, 'Soul Vibes', 'Canzoni soul per rilassarsi', '2024-03-05', 'vincent');
INSERT INTO playlist_canzone VALUES (2, 3, '2025-06-02 01:12:44');  -- Rolling in the Deep
INSERT INTO playlist_canzone VALUES (2, 4, '2025-05-10 21:15:40');  -- Someone Like You

INSERT INTO playlist VALUES (3, 'Modern Mix', 'Pop e alternative contemporanei', '2024-02-20', 'johnSmith92');
INSERT INTO playlist_canzone VALUES (3, 7, '2025-03-02 11:10:24');  -- bad guy
INSERT INTO playlist_canzone VALUES (3, 8, '2025-05-02 03:25:35');  -- when the party's over
INSERT INTO playlist_canzone VALUES (3, 10, '2025-05-17 09:27:23'); -- Love The Way You Lie
```

- Popolamento tabella RIPRODUZIONE_CANZONE

```
INSERT INTO riproduzione_canzone VALUES ('luca_rossi', 1, '2025-05-01 08:30:00');
INSERT INTO riproduzione_canzone VALUES ('luca_rossi', 5, '2025-05-01 09:15:00');
INSERT INTO riproduzione_canzone VALUES ('marioB', 3, '2025-05-02 14:20:00');
INSERT INTO riproduzione_canzone VALUES ('marioB', 7, '2025-05-02 15:05:00');
INSERT INTO riproduzione_canzone VALUES ('vincent', 4, '2025-05-03 10:45:00');
INSERT INTO riproduzione_canzone VALUES ('vincent', 8, '2025-05-03 11:30:00');
INSERT INTO riproduzione_canzone VALUES ('johnSmith92', 10, '2025-05-04 18:20:00');
INSERT INTO riproduzione_canzone VALUES ('johnSmith92', 6, '2025-05-04 19:05:00');
INSERT INTO riproduzione_canzone VALUES ('emma_jones', 2, '2025-05-05 12:15:00');
INSERT INTO riproduzione_canzone VALUES ('emma_jones', 9, '2025-05-05 13:00:00');
```

- Popolamento tabella SEGUE

```
INSERT INTO segue VALUES ('luca_rossi', 1);  -- Eminem
INSERT INTO segue VALUES ('luca_rossi', 3);  -- Jay-Z
INSERT INTO segue VALUES ('marioB', 2);  -- Adele
INSERT INTO segue VALUES ('marioB', 4);  -- Billie Eilish
INSERT INTO segue VALUES ('vincent', 2);  -- Adele
INSERT INTO segue VALUES ('vincent', 5);  -- Nesli
INSERT INTO segue VALUES ('johnSmith92', 3);  -- Jay-Z
INSERT INTO segue VALUES ('johnSmith92', 7);  -- Rihanna
INSERT INTO segue VALUES ('emma_jones', 4);  -- Billie Eilish
INSERT INTO segue VALUES ('emma_jones', 1);  -- Eminem
```

5. QUERY SQL

Avendo a disposizione la struttura finale del database, comprensiva di valori esemplificativi, è possibile interrogare le tabelle per rispondere a diverse tipologie di query, dalle più semplici alle più complesse che presuppongono aggregazioni e query innestate.

1) Selezionare tutti gli artisti italiani e americani

   SELECT nomeArte, nazionalita

   FROM artista

   WHERE nazionalita = 'USA' OR nazionalita = 'Italia';

nomeArte	nazionalita
Eminem	USA
Jay-Z	USA
Billie Eilish	USA
Nesli	Italia
Marracash	Italia

2) Selezionare le canzoni con durata superiore ai 4 minuti

   SELECT titolo, durata

   FROM canzone

   WHERE TIME_TO_SEC(durata) > 240;

titolo	durata
The Real Slim Shady	00:04:44
Stan	00:06:44
Someone Like You	00:04:45
Takeover	00:05:13
Love The Way You Lie	00:04:23

3) Selezionare il catalogo completo: album, artisti e case discografiche

```
SELECT A.titolo AS album, AR.nomeArte AS artista, CD.nome AS casa_discografica
FROM album A
JOIN pubblicazione_album PA ON A.idAlbum = PA.idAlbum
JOIN artista AR ON PA.idArtista = AR.idArtista
JOIN casa_discografica CD ON A.casaDiscografica = CD.nome;
```

album	artista	casa_discografica
The Marshall Mathers LP	Eminem	Interscope Records
21	Adele	Columbia Records
The Blueprint	Jay-Z	Roc-A-Fella Records
When We All Fall Asleep, Where Do ...	Billie Eilish	Interscope Records
1989	Nesli	Carosello Records

4) Trovare tutti i brani dell'artista "Eminem", restituiti in ordine di durata decrescente (dal brano più lungo al più breve)

```
SELECT C.titolo AS brano, A.titolo AS album, C.durata
FROM canzone C
JOIN album_canzone AC ON C.idCanzone = AC.idCanzone
JOIN album A ON AC.idAlbum = A.idAlbum
JOIN interpretazione_canzone I ON C.idCanzone = I.idCanzone
JOIN artista AR ON I.idArtista = AR.idArtista
WHERE AR.nomeArte = 'Eminem'
ORDER BY C.durata DESC
```

brano	album	durata
Love The Way You Lie	The Marshall Mathers LP	00:04:23
Stan	The Marshall Mathers LP	00:06:44
The Real Slim Shady	The Marshall Mathers LP	00:04:44

5) Elencare tutte le playlist presenti, indicando per ciascuna il nome della playlist, il numero di canzoni in essa contenute, l'utente che l'ha creata ed il tipo di abbonamento sottoscritto dall'utente.

```
SELECT P.nome AS playlist, COUNT(PC.idCanzone) AS num_canzoni, U.username AS utente, U.tipoAbbonamento

FROM playlist P JOIN utente U ON P.utente = U.username JOIN playlist_canzone PC ON P.idPlaylist = PC.idPlaylist

GROUP BY P.idPlaylist;
```

playlist	num_canzoni	utente	tipoAbbonamento
Hip-Hop Classics	3	luca_rossi	Premium
Soul Vibes	2	vincent	Premium
Modern Mix	3	johnSmith92	Premium

6) Contare il numero di canzoni per genere e mostrare i risultati in ordine di conteggio decrescente

```
SELECT G.nomeGenere, COUNT(GC.idCanzone) AS num_canzoni

FROM genere_canzone GC RIGHT JOIN genere G ON GC.nomeGenere = G.nomeGenere

GROUP BY G.nomeGenere

ORDER BY num_canzoni DESC;
```

nomeGenere	num_canzoni
Hip-Hop	6
Soul	2
Alternative	1
Pop	1
R&B	0
Rock	0

6.1) Contare il numero di canzoni per genere, escludendo i generi senza canzoni. Mostrare i risultati in ordine decrescente

```
SELECT G.nomeGenere, COUNT(GC.idCanzone) AS num_canzoni

FROM genere_canzone GC, genere G

WHERE GC.nomeGenere = G.nomeGenere

GROUP BY G.nomeGenere

ORDER BY num_canzoni DESC;
```

nomeGenere	num_canzoni
Hip-Hop	6
Soul	2
Alternative	1
Pop	1

7) Per ogni artista, calcolare la durata media delle canzoni presenti nei suoi album, il numero totale di canzoni ed il titolo dell'album.

```
SELECT ar.nomeArte AS artista,
SEC_TO_TIME(TRUNCATE(AVG(TIME_TO_SEC(c.durata)), 0))  AS
durata_media_canzoni, COUNT(DISTINCT c.idCanzone) AS num_canzoni, a.titolo
AS album

FROM artista ar

JOIN pubblicazione_album pa ON ar.idArtista = pa.idArtista JOIN album a ON
pa.idAlbum = a.idAlbum JOIN album_canzone ac ON a.idAlbum = ac.idAlbum JOIN
canzone c ON ac.idCanzone = c.idCanzone

GROUP BY ar.idArtista, a.titolo;
```

artista	durata_media_canzoni	num_canzoni	album
Eminem	00:05:17	3	The Marshall Mathers LP
Adele	00:04:16	2	21
Jay-Z	00:04:36	2	The Blueprint
Billie Eilish	00:03:15	2	When We All Fall Asleep, Where Do ...
Nesli	00:03:45	1	1989

8) Trovare tutte le canzoni che non sono state aggiunte ad una playlist

```
SELECT c.titolo AS canzone, ar.nomeArte AS artista
FROM canzone c, interpretazione_canzone i, artista ar
WHERE c.idCanzone = i.idCanzone AND
        i.idArtista = ar.idArtista AND
   NOT EXISTS (
   SELECT *
   FROM playlist_canzone pc
   WHERE pc.idCanzone = c.idCanzone
);
```

	canzone	artista
▶	Stan	Eminem
	Buongiorno	Nesli

9) Selezionare gli utenti con abbonamento ‘Premium’ che non hanno mai riprodotto né inserito in una loro playlist alcuna canzone di genere ‘Pop’

```
SELECT u.username, u.email
FROM utente u
WHERE u.tipoAbbonamento = 'Premium'
AND NOT EXISTS (
 -- utenti con riproduzione di canzoni Pop
   SELECT *
   FROM riproduzione_canzone rc
   JOIN genere_canzone gc ON rc.idCanzone = gc.idCanzone
   WHERE rc.utente = u.username AND gc.nomeGenere = 'Pop'
```

```
)

AND u.username NOT IN (

    -- utenti con playlist contenenti canzoni Pop

    SELECT DISTINCT p.utente

    FROM playlist p

    JOIN playlist_canzone pc ON p.idPlaylist = pc.idPlaylist JOIN genere_canzone gc
ON pc.idCanzone = gc.idCanzone

    WHERE gc.nomeGenere = 'Pop'

);
```

	username	email
▶	luca_rossi	luca.rossi@example.com
	vincent	vincent.aronne95@example.com

10) Elencare i brani più ascoltati nell'intervallo di tempo compreso tra '2025-05-02' e '2025-05-04', mostrando per ciascuno il titolo, l'artista che lo interpreta ed il numero totale di volte in cui è stato riprodotto. Ordinare i risultati dal brano più ascoltato al meno ascoltato.

```
SELECT c.titolo, ar.nomeArte AS artista, COUNT(rc.idCanzone) AS ascolti

FROM riproduzione_canzone rc, canzone c, interpretazione_canzone ic, artista ar

WHERE rc.idCanzone = c.idCanzone AND c.idCanzone = ic.idCanzone AND
ic.idArtista = ar.idArtista AND rc.dataRiproduzione BETWEEN '2025-05-02' AND
'2025-05-04'

GROUP BY c.idCanzone, ar.nomeArte, c.titolo

ORDER BY ascolti DESC;
```

	titolo	artista	ascolti
▶	Rolling in the Deep	Adele	1
	bad guy	Billie Eilish	1
	Someone Like You	Adele	1
	when the party's over	Billie Eilish	1

11) Elencare tutti gli artisti che pubblicano con la casa discografica “Columbia Records”

```
SELECT DISTINCT ar.nomeArte as artista, a.tipologia as tipoAlbum

FROM artista ar

JOIN pubblicazione_album pa ON ar.idArtista = pa.idArtista JOIN album a ON pa.idAlbum = a.idAlbum

WHERE a.casaDiscografica = 'Columbia Records';
```

	artista	tipoAlbum
▶	Adele	Studio

12) Trovare i 5 brani ascoltati dall’utente ‘luca_rossi’, con una durata superiore ai 3 minuti

```
SELECT c.titolo, ar.nomeArte AS artista, COUNT(*) AS ascolti, c.durata

FROM riproduzione_canzone rc, canzone c, interpretazione_canzone ic, artista ar

WHERE ic.idArtista = ar.idArtista AND rc.idCanzone = c.idCanzone AND
        c.idCanzone = ic.idCanzone AND rc.utente = 'luca_rossi'

GROUP BY c.idCanzone, ar.nomeArte

HAVING TIME_TO_SEC(c.durata) > 180

ORDER BY ascolti DESC

LIMIT 5;
```

	titolo	artista	ascolti	durata
▶	Izzo (H.O.V.A.)	Jay-Z	1	00:04:00
	The Real Slim Shady	Eminem	1	00:04:44

6. CONCLUSIONI

La modellazione di un contesto reale e attuale come quello di una piattaforma di streaming musicale ha rappresentato un buon pretesto per applicare i principi di base della progettazione e implementazione delle basi di dati relazionali. L'analisi del contesto ha permesso di identificare gli attori e gli oggetti chiave (utenti, artisti, album, canzoni, generi, case discografiche) e di delineare, seppur in maniera semplificata, le interazioni che caratterizzano un sistema di streaming musicale. Attraverso le fasi di progettazione concettuale ed E/R, è stato possibile tradurre i requisiti testuali in un modello formale, catturando le entità, i loro attributi, le cardinalità e la natura delle relazioni. La successiva conversione in schema logico utilizzando MySQL, ha permesso di mappare il modello concettuale in una struttura relazionale concreta, con la corretta definizione di tabelle, chiavi primarie e chiavi esterne per garantire l'integrità referenziale dei dati. L'implementazione delle query SQL ha infine permesso di validare la struttura del database, dimostrando di poter estrarre informazioni significative e rispondere a quesiti complessi sul catalogo musicale e sul comportamento degli utenti. Dalle semplici selezioni e proiezioni alle query più articolate con join multipli, aggregazioni e query innestate, il database si è rivelato uno strumento valido per la gestione e l'analisi dei dati musicali.

Il modello è stato progettato per garantire una rappresentazione fedele delle funzionalità principali di una moderna piattaforma musicale, ma con alcune semplificazioni. Non tiene infatti conto della fruizione di podcast ed audiolibri da parte degli utenti. Inoltre, il dominio delle piattaforme di streaming è vasto ed in continua evoluzione. Pur costituendo una buona base per la rappresentazione di una piattaforma di streaming musicale, presenta ampi margini di miglioramento.

Possibili sviluppi futuri potrebbero includere:

- la gestione dei podcast, integrando le entità per podcast, episodi, categorie di podcast, con relazioni simili a quelle di album/canzoni;
- la gestione degli audiolibri, richiedendo la modellazione di entità per libri, narratori, editori e capitoli;
- l’estensione con funzionalità di tracciamenti delle interazioni utente (ascolti, preferenze, ecc.) per offrire suggerimenti personalizzati e per futuri moduli di analisi;
- strumenti di analisi avanzata e machine learning applicando algoritmi di raccomandazione più sofisticata.

Questi sviluppi non solo aumenterebbero la complessità del database, ma aprirebbero anche le porte a funzionalità più ricche e ad un'analisi dei dati più profonda, essenziali per il successo di una piattaforma di streaming nel mercato attuale.