**Дахненко Михаил**

**Part 1:** Выбор Сценария

Для данной работы выбран сценарий: Сервис потоковой передачи музыки. Эта система будет управлять артистами, альбомами, песнями, пользователями и пользовательскими плейлистами.

**Part 2:** Проектирование Базы Данных и Документация

**Идентификация Сущностей и Атрибутов:**

1. Артисты (Artists):

2. Альбомы (Albums):

3. Песни (Songs):

4. Пользователи (Users):

5. Плейлисты (Playlists):

5. Пользовательские плейлисты (PlaylistSongs):

**Проектирование Таблиц:**

1. Table Name: Artists

○ Description: Хранит информацию об артистах.

○ Attributes:

■ ArtistID: INTEGER, PK, NOT NULL, UNIQUE

■ Name: VARCHAR(255), NOT NULL, UNIQUE

■ Country: VARCHAR(100)

○ Constraints:

■ PK\_Artists: PRIMARY KEY (ArtistID)

■ UQ\_AuthorFullName: UNIQUE (FirstName, LastName)

2. Table Name: Albums

○ Description: Содержит информацию об альбомах.

○ Attributes:

■ AlbumID: INTEGER, PK, NOT NULL, UNIQUE

■ Title: VARCHAR(255), NOT NULL

■ ReleaseYear: INTEGER

■ ArtistID: INTEGER, FK (REFERENCES Artists), NOT NULL

○ Constraints:

■ PK\_Albums: PRIMARY KEY (AlbumID)

■ CHK\_ReleaseYear: CHECK (ReleaseYear >= 1900 AND

ReleaseYear <= 2025)

■ FK\_Albums\_Artists: FOREIGN KEY (AuthorID) REFERENCES

Artists(ArtistID)

3. Table Name: Songs

○ Description: Хранит данные о песнях.

○ Attributes:

■ SongID: INTEGER, PK, NOT NULL, UNIQUE

■ Title: VARCHAR(255), NOT NULL

■ Duration: INTEGER, NOT NULL, CHECK (Duration > 0)

■ AlbumID: INTEGER, FK (REFERENCES Albums), NOT NULL

○ Constraints:

■ PK\_SONGS: PRIMARY KEY (SongID)

■ FK\_Songs\_Albums: FOREIGN KEY (AlbumID) REFERENCES Albums (AlbumID)

4. Table Name: Users

○ Description: Данные о пользователях сервиса.

○ Attributes:

■ UserID: INTEGER, PK, NOT NULL, UNIQUE

■ FirstName: VARCHAR(100), NOT NULL

■ LastName: VARCHAR(100), NOT NULL

■ Email: VARCHAR(255), UNIQUE, NOT NULL

○ Constraints:

■ PK\_Users: PRIMARY KEY (UserID)

■ UQ\_Email: UNIQUE (Email)

5. Table Name: Playlists

○ Description: Плейлисты, созданные пользователями.

○ Attributes:

■ PlaylistID: INTEGER, PK, NOT NULL, UNIQUE

■ Title: VARCHAR(255), NOT NULL

■ UserID: INTEGER, FK (REFERENCES Users), NOT NULL

■ CreationDate: DATE NOT NULL DEFAULT CURRENT\_DATE

○ Constraints:

■ PK\_Playlists: PRIMARY KEY (PlaylistID)

■ FK\_Playlists\_Users: FOREIGN KEY (UserID) REFERENCES Users(UserID)

6. Table Name: PlaylistSongs

○ Description: Связь многие-ко-многим между плейлистами и песнями.

○ Attributes:

■ PlaylistSongID: INTEGER, PK, NOT NULL, UNIQUE

■ PlaylistID: INTEGER, FK (REFERENCES Playlists), NOT NULL

■ SongID: INTEGER, FK (REFERENCES Songs), NOT NULL

■ AddedDate: DATE NOT NULL DEFAULT CURRENT\_DATE

○ Constraints:

■ PK\_PlaylistSongs: PRIMARY KEY (PlaylistSongID)

■ FK\_PlaylistSongs\_Playlists: FOREIGN KEY (PlaylistID) REFERENCES Playlists(PlaylistID)

■ FK\_PlaylistSongs\_Songs: FOREIGN KEY (SongID) REFERENCES Songs(SongID)

■ UQ\_Playlist\_Song: UNIQUE (PlaylistID, SongID) -- чтобы песня не повторялась в одном плейлисте

**Взаимосвязи:**

● Artists и Albums (Один-ко-Многим): Однин артист может выпустить несколько альбомов.

● Albums и Songs (Один-ко-Многим): Один альбом содержит несколько песен.

● Users и Playlists (Один-ко-Многим): Один пользователь может создать несколько плейлистом.

● Playlists и Songs (Многие-ко-Многим): Один плейлист может содержать несколько песен, а одна песня может входить в разные плейлисты.

**Part 3: ER-Диаграмма**

