| | |
|---|---|
| Przedmiot: Bazy danych<br>Temat: Diagram ER, dokładny opis tabel | Data: 30.11.2023r. |
| Grupa: PS.02<br>Zespół:<br>- Dominik Gąsowski (lider)<br>- Wojciech Domański | Prowadzący: mgr inż. Marek Kopczewski<br><br>Ocena: |

## 1. Projekt bazy danych

Nasz projekt obejmuje bazę danych aplikacji Spotify. Spotify to aplikacja muzyczna umożliwiająca tworzenie playlist, słuchanie muzyki itp.

## 2. Diagram ER

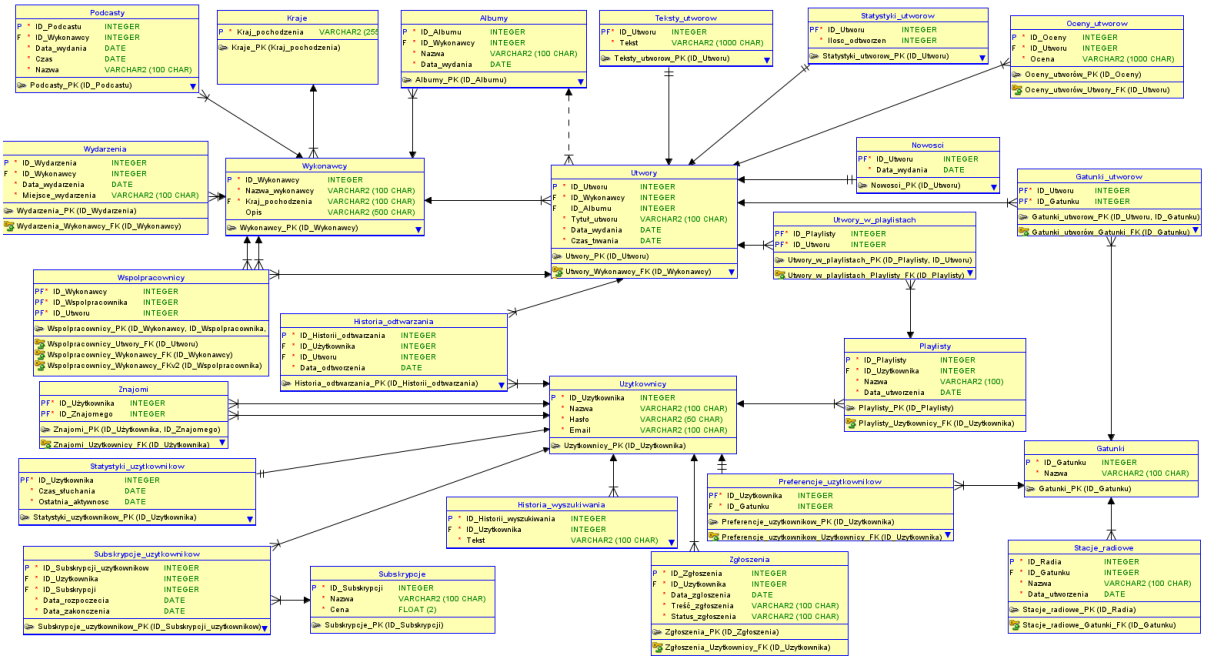


Diagram ER

## 3. SQL

### 3.1. Tworzenie tabel

Opis tabel:

**Utwory –** Tabela zawierająca informacje na temat utworów. Posiada ona klucz podstawowy „ID_Utworu” i dwa klucze obce: „ID_Wykonawcy” oraz „ID_Albumu”. Utwory są połączone z wykonawcami relacją wiele do jednego. Każdy wykonawca może mieć wiele utworów.

**Wykonawcy –** Tabela o wykonawcach. Ma klucz podstawowy ID_Wykonawcy. Jest połączona z utworami relacją wspomnianą punkt wyżej oraz z albumami.

**Albumy** – Tabela na temat albumów muzycznych. Posiada klucz podstawowy „ID_Albumu” oraz klucz obcy nawiązujący do tabeli Utwory. Jest połączona z utworami relacją jeden do wielu. Dopuszczalne są nulle (nie każdy utwór musi należeć do jakiegoś albumu).

**Teksty_utworow** – Tabela połączona z tabelą Utwory relacją jeden do jeden. Każdy utwór ma swój tekst (nie bierzemy pod uwagę melodii bez słów).

**Statystyki_utworow** – Tabela połączona w ten sam sposób jak tabela wyżej (Teksty_utworow). Zawiera pole ilość_odtworzen (każda piosenka została ileś razy odsłuchana)

**Oceny_utworów** – Zawiera informacje na temat ocen utworów. Ma klucz podstawowy „ID_Oceny” oraz klucz obcy „ID_Utworu”, dzięki któremu jest połączona relacją z utworami wiele do jednego.

**Podcasty** – Tabela na temat podcastów. Zawiera informacje na temat podcastów, klucz podstawowy ID_Podcastu oraz klucz obcy do tabeli wykonawcy. Jest połączona z wykonawcami relacją wiele do jednego (jeden wykonawca może mieć wiele podcastów)

**Wydarzenia** – Tabela zawierająca informacje na temat koncertów, meet-upów. Ma klucz podstawowy ID_Wydarzenia oraz klucz obcy ID_Wykonawcy. Jest połączona z wykonawcami relacją wiele do jednego (każdy wykonawca z biegiem czasu organizuje co raz więcej wydarzeń).

**Nowosci** – Tabela zawierające najnowsze utwory i ich daty wydania. Zawiera klucz obcy ID_Utworu. Jest połączona z utworami relacją jeden do jeden (nowy utwór mieści się również w tabeli Utwory).

**Utwory_w_playlistach** – Tabela łącznikowa między tabelą „Utwory” oraz „Playlisty”. Dzięki niej przyporządkowujemy utwory do poszczególnych playlist. Zawiera tylko klucze obce do obu tabel.

**Gatunki_utworow** – Tabela łącznikowa między tabelą „Utwory” i „Gatunki”. Mówi nam o tym, do jakiego gatunku należy dany utwór. Zawiera tylko 2 pola: klucze obce do tabel wymienionych wyżej.

**Gatunki** – Zawiera klucz podstawowy „ID_Gatunku” oraz jego nazwę. Tabela ta mówi nam o możliwych gatunkach, do jakich mogą zaliczać się utwory.

**Playlisty** – Tabela zawierająca klucz podstawowy „ID_Playlisty” oraz klucz obcy „ID_Uzytkownika”. Tabela jest połączona z tabelą „Uzytkownicy” relacją wiele do jednego. (Każdy użytkownik może posiadać wiele playlist).

**Uzytkownicy** – Tabela o użytkownikach korzystających ze Spotify. Zawiera informacje na temat użytkowników (ich login,email,haslo). Zawiera klucz podstawowy „ID_Uzytkownika”.

**Historia_odtwarzania** – Tabela łącznikowa między tabelą „Utwory” i „Uzytkownicy”. Mówi nam ona jakich utworów słuchali użytkownicy. Dodatkowo posiada ona datę odsłuchania z godziną.

Znajomi – tabela łącznikowa między użytkownikami. Posiada ona dwa klucze obce, które mówią nam kto kogo zna z użytkowników.

**Historia_wyszukiwania** – Tabela zawierająca informacje na temat tego, czego szukał użytkownik na Spotify. Zapisujemy w niej na bieżąco to co wpisuje użytkownik szukając piosenki.

**Preferencje_uzytkownikow** – Tabela ma dwa klucze obce: „ID_Gatunku” i „ID_Uzytkownika”. Dzięki niej wiemy jakie gatunki lubi dany użytkownik i możemy mu proponować trafniejsze utwory.

**Stacje_radiowe** – Posiada klucz podstawowy „ID_Radia” oraz klucz obcy łączący tabelę z Gatunkami w relacji wiele do jednego (każdy gatunek może być wyborem wielu stacji).

**Zgłoszenia** – Tabela zawierająca informacje na temat zgłoszeń użytkowników. Ma klucz podstawowy ID_Zgloszenia oraz klucz obcy łączący tę tabelę z „Uzytkownicy”. Dodaliśmy również datę zgłoszenia, dzięki której będziemy mogli rozpatrywać na bieżąco zgłoszenia.

**Statystyki_uzytkownikow** – zawiera klucz obcy do tabeli użytkowników oraz informacje takie jak czas słuchania utworów oraz ostatnia aktywność w Spotify. Na jej podstawie możemy m.in. tworzyć podsumowania na koniec roku.

**Subskrypcje** – Tabela zawierająca informacje na temat dostępnych subskrypcji (m.in. darmowa).

**Subskrypcje_uzytkownikow** – Tabela łącznikowa między użytkownikami i subskrypcjami. Dzięki niej możemy tworzyć zestawienia na temat tego, jakie subskrypcje posiadają użytkownicy.

**Współpracownicy** – Tabela łącznikowa. Zawiera informacje na temat utworów, które zostały stworzone przez więcej niż jednego wykonawcę. Mamy 3 klucze obce: ID_Utworu (z tabeli „Utwory”), ID_Wspolpracownika (z tabeli „Uzytkownicy”) oraz ID_Uzytkownika (również do tabeli „Uzytkownicy”).

Przykładowe komendy tworzące tebele:

```
CREATE TABLE Kraje(Kraj_pochodzenia varchar(250) NOT NULL, PRIMARY KEY(Kraj_pochodzenia));
```

```
CREATE TABLE Wykonawcy(ID_Wykonawcy INT(3) NOT NULL AUTO_INCREMENT, Nazwa_wykonawcy varchar(100) NOT NULL, Kraj_pochodzenia varchar(250) NOT NULL, Opis varchar(500), PRIMARY KEY(ID_Wykonawcy), FOREIGN KEY(Kraj_pochodzenia) REFERENCES Kraje(Kraj_pochodzenia));
```

## 3.2 Modyfikacja tabel

Po zakończeniu tworzenia tabel nastąpiła potrzeba modyfikacji niektórych z nich. Do tego użyliśmy dyrektywy ‘ALTER TABLE’.

Przykładowe komendy modyfikujące tabele:

```
ALTER TABLE wspolpracownicy ADD PRIMARY KEY(ID_Wykonawcy,ID_Wspolpracownika,ID_Utworu);
```

```
ALTER TABLE Subskrypcje_uzytkownikow MODIFY ID_Subskrypcji_uzytkownikow INTEGER(3) NOT NULL AUTO_INCREMENT FIRST;
```

### 3.3 Wprowadzanie danych

Po stworzeniu oraz zmodyfikowaniu tabel wypełniliśmy naszą bazę przykładowymi danymi, do tego użyliśmy komendy ‘INSER INTO’.

Przykładowe komendy wprowadzające dane:

INSERT INTO Podcasty(ID_Wykonawcy,Data_wydania,Czas,Nazwa) VALUES(1,"1652-01-13",400,"Trailer GTA VI"),(1,"1651-07-16",500,"Podcast3"),(2,"2012-04-12",900,"Podcast4"),(6,"2019-08-07",12000,"Podcast5"),(6,"2016-12-16",5000,"Podcast6"),(1,"1921-01-30",1921,"Podcast7"),(10,"1892-12-13",2100,"Podcast8"),(4,"2018-11-12",900,"gasnijpokadet"),(3,"2012-12-20",1200,"aedbtnhgOIDflu");

INSERT INTO Statystyki_utworow(ID_Utworu, Ilosc_odtworzen) VALUES (1, 150), (2, 230), (3, 180), (4, 300), (5, 270), (6, 210), (7, 320), (8, 250), (9, 190), (10, 280);

### 3.4 Modyfikacja oraz usuwanie danych

Przy użyiu bazy danych przećwiczyliśmy modyfikowanie oraz usuwanie przechowywanych danych używając instrukcji ‘UPDATE’ oraz ‘DELETE’.

Przykładowe komendy modyfikujące dane:

UPDATE albumy SET nazwa="nowa nazwa" WHERE ID_Albumu = 1;

UPDATE historia_odtwarzania SET Data_odtworzenia = '2025-12-12 08:30:00' WHERE ID_Uzytkownika=1 AND ID_Utworu=1;

Przykładowe komendy usuwające dane:

DELETE FROM Podcasty WHERE YEAR(Data_wydania) = 2017;

DELETE FROM Playlisty WHERE Nazwa LIKE 'A%';

### 3.5 Tworzenie widoków

Stworzyliśmy dwa widoki: pierwszy zawierające dane z jednej tabeli, drugi łączący dwie tabele.

Do stworzenia pierwszego z nich użyliśmy polecenia:
CREATE VIEW widok1 AS SELECT nazwa, email FROM użytkownicy WHERE id_uzytkownika > 5;

Drugi widok:

CREATE VIEW widok2 AS SELECT subskrypcje_uzytkownikow.Data_rozpoczecia, subskrypcje_uzytkownikow.Data_zakonczenia FROM subskrypcje, subskrypcje_uzytkownikow WHERE subskrypcje.id_subskrypcji=subskrypcje_uzytkownikow.id_subskrypcji_uzytkownikow AND subskrypcje.Nazwa="Basic";

Po ich stworzeniu były widocze w liście tabel oraz mogliśmy z nich korzystać jak z normalnych tabel:

```
MariaDB [spotify]> select * from widok2;
+------------------+------------------+
| Data_rozpoczecia | Data_zakonczenia |
+------------------+------------------+
| 2023-01-01       | 2023-12-31       |
| 2023-06-08       | 2023-12-08       |
+------------------+------------------+
2 rows in set (0.001 sec)
```

### 3.6 Tworzenie indeksów

Podobnie jak w przypadku widoków, stworzyliśmy dwa indeksy, pierwszy zawierający jedną kolumnę, drugi łączący trzy kolumny.

Do stworzenia pierwszego z nich użyliśmy polecenia:
CREATE INDEX indeks1 ON utwory(ID_Utworu);

Drugi indeks:

CREATE INDEX indeks1 ON Albumy(ID_Albumu, Nazwa, Data_wydania);

### 3.7 Uprawnienia dla użytkowników

Stworzyliśmy dwóch użytkowników: uzytkownik1 oraz uzytkownik2:

create user 'uzytkownik1'@'localhost' IDENTIFIED BY 'uzytkownik1';

create user 'uzytkownik2'@'localhost' IDENTIFIED BY 'uzytkownik2';

Nowi użytkownicy zostali pomyślnie utworzeni:

```
MariaDB [spotify]> select user, host from mysql.user;
+-------------+-----------+
| User        | Host      |
+-------------+-----------+
| root        | 127.0.0.1 |
| root        | ::1       |
| pma         | localhost |
| root        | localhost |
| uzytkownik1 | localhost |
| uzytkownik2 | localhost |
+-------------+-----------+
6 rows in set (0.001 sec)
```

Następnie nadaliśmy im prawa do oczytu, modyfikacji oraz usuwania odpowiednich tabel, widoków.

GRANT SELECT, INSERT, UPDATE, DELETE ON widok1 TO 'uzytkownik1'@'localhost';

GRANT SELECT, INSERT, UPDATE, DELETE ON widok2 TO 'uzytkownik2'@'localhost';

```
MariaDB [spotify]> SHOW GRANTS FOR 'uzytkownik1'@'localhost';
+-----------------------------------------------------------------------------------------------------------------------------+
| Grants for uzytkownik1@localhost                                                                                            |
+-----------------------------------------------------------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `uzytkownik1`@`localhost` IDENTIFIED BY PASSWORD '*D408ABFC58FAFF8D420CC419E347E2A60B43130D' |
| GRANT SELECT, INSERT, UPDATE, DELETE ON `spotify`.`widok1` TO `uzytkownik1`@`localhost`                                     |
+-----------------------------------------------------------------------------------------------------------------------------+
2 rows in set (0.000 sec)
```

Uprawnienia użytkownika pierwszego

```
MariaDB [spotify]> SHOW GRANTS FOR 'uzytkownik2'@'localhost';
+-----------------------------------------------------------------------------------------------------------------------------+
| Grants for uzytkownik2@localhost                                                                                            |
+-----------------------------------------------------------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `uzytkownik2`@`localhost` IDENTIFIED BY PASSWORD '*957095D3CFAEC5C0317DC60D871560507E1E2874' |
| GRANT SELECT, INSERT, UPDATE, DELETE ON `spotify`.`widok2` TO `uzytkownik2`@`localhost`                                     |
+-----------------------------------------------------------------------------------------------------------------------------+
2 rows in set (0.000 sec)
```

Uprawnienia użytkownika drugiego

Sprawdziliśmy działanie uprawnień poprzez wyświetlanie widoków:

```
MariaDB [spotify]> select * from widok2;
ERROR 1142 (42000): SELECT command denied to user 'uzytkownik1'@'localhost' for table `spotify`.`widok2`
MariaDB [spotify]> select * from widok1;
+----------------+-------------------------+
| nazwa          | email                   |
+----------------+-------------------------+
| MelodicDreamer | dreamer@example.com     |
| BeatFinder     | beatfinder@example.com  |
| TuneChaser     | chaser@example.com      |
| MusicVoyager   | voyager@example.com     |
| SongDreamer    | songdreamer@example.com |
+----------------+-------------------------+
5 rows in set (0.000 sec)
```

```
MariaDB [spotify]> select * from widok1;
ERROR 1142 (42000): SELECT command denied to user 'uzytkownik2'@'localhost' for table `spotify`.`widok1`
MariaDB [spotify]> select * from widok2;
+------------------+------------------+
| Data_rozpoczecia | Data_zakonczenia |
+------------------+------------------+
| 2023-01-01       | 2023-12-31       |
| 2023-06-08       | 2023-12-08       |
+------------------+------------------+
2 rows in set (0.000 sec)
```

**Wnioski:**

Zaprojektowana przez nas baza danych jest w trzeciej postaci normalnej, do tego wymagane było stworzenie paru dodatkowych tabel. Widoki w bazach danych są przydatne przy skomplikowanych zapytaniach, zamiast zagnieżdżonych ‘selectów’ możemy używać widoków. Są one również stosowane w celach bezpieczeństwa baz – pracownicy używający ich nie pracują bezpośrednio na tabelach. Wykorzystaliśmy to przy tworzeniu użytkowników – pierwszy miał tylko dostęp do widoku ‘widok1’, drugi natomiast to widoku ‘widok2’. Indeksy w bazach danych są strukturami, które mają zwiększyć efektywność wyszukiwania, sortowanie itp. Przyspieszają one dostęp do rekordów w tabeli.

Pozostałe instrukcje SQL znajdują się w pliku sql.txt.