МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГУ»)

Факультет прикладной математики, информатики и механики

Кафедра математического обеспечения ЭВМ

ОТЧЕТ

по дисциплине РСУБД

Тема: разработка базы данных «Музыкальная коллекция»

Направление 02.03.02 - Фундаментальная информатика и информационные технологии

Профиль – «Инженерия программного обеспечения»

Обучающийся _____ студ. 61 гр. 4 курса В. А. Брянцев

Руководитель _____ доц. физ.-мат. наук, проф. И.Ф. Астахова

Воронеж 2024

Постановка задачи

База данных “Музыкальная коллекция”. В БД необходимо хранить информацию о характеристиках альбома: название, дата издания, стиль музыкальных произведений, число дорожек (tracks), общее время звучания в минутах, тип носителя (компакт-диск, магнитофонная лента, грампластинка).

Интерес представляет информация и о песнях, представленных в альбоме (название песни, дата написания, стиль музыки, длительность песни в минутах).

Информация об исполнителях песен должна включать следующие данные: фамилия, имя, дата рождения, место рождения, дата смерти, основной стиль музыки. Кроме того, потребуются данные о песнях исполнителя (является ли исполнитель певцом или певцом и автором песни, дата записи или исполнения).

Надо учесть, что исполнитель может входить в состав одной или нескольких музыкальных групп. В этом случае в БД должна храниться информация о дате, когда исполнитель присоединился к данной группе и о дате, когда исполнитель покинул группу.

Информация о группах должна содержать следующее: название группы, дата основания, дата распада, основной стиль музыки.

Наконец, БД должна хранить информацию о стилях музыки.

Написать пакет, состоящий из процедур и функций, которые позволили выполнить следующие действия с базой данных:

1) Вывести текущих участников заданной группы;
2) Посчитать сумму длительности всех альбомов;
3) Вывести информацию (имя, фамилия, псевдоним, группа, год присоединения к группе, год ухода из группы) об исполнителях, которые используют псевдоним;
4) Посчитать количество выпущенных альбомов на каждый вид дистрибуции;
5) Вывести все песни, автором которых является заданный исполнитель;
6) Посчитать количество песен, для которых заданный исполнитель является автором.

Предусмотреть разработку триггеров, обеспечивающие каскадные изменения в связанных таблицах.

Описание структуры таблиц базы данных

Опишем структуру таблиц, связи между ними и произведём их нормализацию.

В базе данных представлены две справочные таблицы:

Таблица MusicStyles содержит названия музыкальных стилей (поле style_name) и их порядковые номера (поле style_id). Поле style_id является первичным ключом.

style_id	style_name
1	Alternative rock
2	Nu-metal
3	Horror trap
…	…

Таблица MediaTypes содержит названия носителей, на которых может быть выпущен альбом (поле type_name) и их порядковые номера (поле type_id). Поле type_id является первичным ключом.

type_id	type_name
1	Digital
2	Vinyl
3	CD

Таблица Albums содержит информацию о музыкальных альбомах:

album_id	album_name	release_date	style_id	total_tracks	total_duration	media_type_id
1	Скафандр и бабочка	30-NOV-17	1	9	31	3
2	Вкус Крови	27-SEP-19	1	9	32	1
…	…	…	…	…	…	…

- album_id – идентификатор альбома, который также является первичным ключом таблицы.
- album_name – название альбома.
- release_date – дата выпуска альбома.
- style_id – идентификатор музыкального стиля (внешний ключ, который ссылается на поле style_id таблицы MusicStyles).
- total_tracks – количество песен в альбоме.
- total_duration – длительность альбома в минутах.
- media_type_id – идентификатор носителя (внешний ключ, который ссылается на поле type_id таблицы MediaTypes).

Таблица Artists содержит информацию об исполнителях:

artist_id	first_name	last_name	nickname	main_style_id
1	Андрей	Грязнов	NULL	1
2	Кирилл	Бабиев	NULL	1
…	…	…	…	…

- artist_id – идентификатор исполнителя, который также является первичным ключом.
- first_name – имя исполнителя.
- last_name – фамилия исполнителя.
- nickname – псевдоним исполнителя.
- main_style_id – идентификатор основного музыкального стиля исполнителя (внешний ключ, который ссылается на поле style_id таблицы MusicStyles).

Таблица Bands содержит информацию о группах:

band_id	band_name	formation_year	breakup_year	main_style_id
1	ТАйМСКВЕР	2009	NULL	1
2	ZOMBIEZ	2017	NULL	3
…	…	…	…	…

- band_id – идентификатор группы, который также является первичным ключом.
- band_name – название группы.
- formation_year – год основания группы.
- breakup_year – год распада группы.
- main_style_id – идентификатор основного музыкального стиля группы (внешний ключ, который ссылается на поле style_id таблицы MusicStyles).

Таблица Songs содержит информацию о песнях:

song_id	song_name	style_id	duration	album_id	band_id
1	Крошечное солнце	1	3	1	1
2	Тени	1	3	1	1
…	…	…	…	…	…

- song_id – идентификатор песни, который также является первичным ключом.
- song_name – название песни.
- style_id – идентификатор музыкального стиля песни (внешний ключ, который ссылается на поле style_id таблицы MusicStyles).
- duration – длительность песни в минутах.
- album_id – идентификатор альбома, к которому относится песня (внешний ключ, который ссылается на поле album_id таблицы Albums).
- band_id – идентификатор группы, к которой относится песня (внешний ключ, который ссылается на поле band_id таблицы Bands).

В базе данных также представлены таблицы, реализующие связи многие-ко-многим:

Таблица Authorship обеспечивает хранение информации об авторстве исполнителей над песнями. Она имеет два поля: artist_id – идентификатор исполнителя и song_id – идентификатор песни. Оба поля являются внешними ключами, которые ссылаются на соответствующие таблицы.

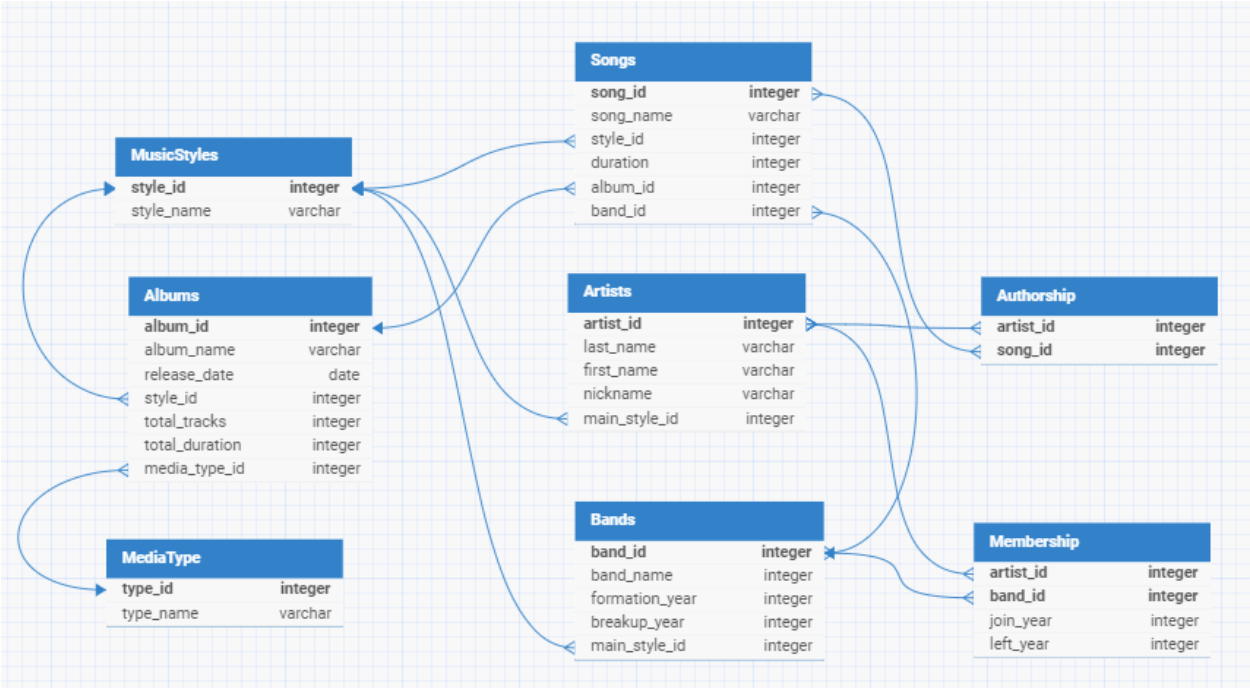
artist_id	song_id
1	1
1	2
…	…

Таблица Membership обеспечивает хранение информации о принадлежности исполнителей к группам. Она имеет четыре поля: artist_id – идентификатор исполнителя, band_id – идентификатор группы, join_year – год присоединения исполнителя к группе, left_year – год, в который исполнитель группу покинул. Поля artist_id и band_id являются внешними ключами, которые ссылаются на соответствующие таблицы.

artist_id	band_id	join_year	left_year
1	1	2009	NULL
2	1	2013	NULL
…	…	…	…

Во всех таблицах изменения родительского ключа разрешаются, но при этом осуществляется коррекция всех значений внешних ключей, ссылающихся на модифицируемое значение родительского ключа. Таким образом обеспечивается ссылочная целостность базы данных. База данных находится в нормализованном состоянии.

ER-диаграмма базы данных:



Создание и заполнение таблиц

Таблица MusicStyles:

```
CREATE TABLE MusicStyles (
    style_id INTEGER NOT NULL PRIMARY KEY,
    style_name VARCHAR(100) NOT NULL
);
INSERT INTO MusicStyles(style_id, style_name) VALUES(1, 'Alternative rock');
INSERT INTO MusicStyles(style_id, style_name) VALUES(2, 'Nu-metal');
INSERT INTO MusicStyles(style_id, style_name) VALUES(3, 'Horror trap');
INSERT INTO MusicStyles(style_id, style_name) VALUES(4, 'Metalcore');
INSERT INTO MusicStyles(style_id, style_name) VALUES(5, 'Rapcore');
INSERT INTO MusicStyles(style_id, style_name) VALUES(6, 'Russian rap');
INSERT INTO MusicStyles(style_id, style_name) VALUES(7, 'Rock');
INSERT INTO MusicStyles(style_id, style_name) VALUES(8, 'Trap');
INSERT INTO MusicStyles(style_id, style_name) VALUES(9, 'Alternative');
```

Таблица MediaTypes:

```
CREATE TABLE MediaTypes (
    type_id INTEGER NOT NULL PRIMARY KEY,
    type_name VARCHAR(100) NOT NULL
);
INSERT INTO MediaTypes(type_id, type_name) VALUES (1, 'Digital');
```

```
INSERT INTO MediaTypes(type_id, type_name) VALUES (2, 'Vinyl');
INSERT INTO MediaTypes(type_id, type_name) VALUES (3, 'CD');
```

Таблица Albums:

```
CREATE TABLE Albums (
    album_id INTEGER NOT NULL PRIMARY KEY,
    album_name VARCHAR(100) NOT NULL,
    release_date DATE NOT NULL,
    style_id NUMBER REFERENCES MusicStyles(style_id),
    total_tracks NUMBER NOT NULL,
    total_duration NUMBER NOT NULL,
    media_type_id NUMBER REFERENCES MediaTypes(type_id)
);
INSERT INTO Albums(album_id, album_name, release_date, style_id, total_tracks,
total_duration, media_type_id)
VALUES(1, 'Скафандр и бабочка', TO_DATE('30-11-2017', 'DD-MM-YYYY'), 1, 9, 31,
3);
INSERT INTO Albums(album_id, album_name, release_date, style_id, total_tracks,
total_duration, media_type_id)
VALUES(2, 'Вкус Крови', TO_DATE('27-09-2019', 'DD-MM-YYYY'), 1, 9, 32, 1);
INSERT INTO Albums(album_id, album_name, release_date, style_id, total_tracks,
total_duration, media_type_id)
VALUES(3, 'KATHARZIZ', TO_DATE('23-02-2023', 'DD-MM-YYYY'), 3, 16, 56, 1);
INSERT INTO Albums(album_id, album_name, release_date, style_id, total_tracks,
total_duration, media_type_id)
VALUES(4, 'Ужас', TO_DATE('19-10-2023', 'DD-MM-YYYY'), 9, 8, 27, 1);
INSERT INTO Albums(album_id, album_name, release_date, style_id, total_tracks,
total_duration, media_type_id)
VALUES(5, 'Дизмораль', TO_DATE('20-12-2021', 'DD-MM-YYYY'), 1, 8, 27, 1);
INSERT INTO Albums(album_id, album_name, release_date, style_id, total_tracks,
total_duration, media_type_id)
VALUES(6, 'Мрак', TO_DATE('2-02-2020', 'DD-MM-YYYY'), 9, 5, 17, 3);
```

Таблица Artists:

```
CREATE TABLE Artists (
    artist_id INTEGER NOT NULL PRIMARY KEY,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    nickname VARCHAR(100),
    main_style_id NUMBER REFERENCES MusicStyles(style_id)
);

INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(1, 'Андрей', 'Грязнов', NULL, 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(2, 'Кирилл', 'Бабиев', NULL, 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(3, 'Дмитрий', 'Теплов', NULL, 1);
```

```
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(4, 'Николай', 'Карпенко', NULL, 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(5, 'Степан', 'Четвериков', NULL, 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(6, 'Евгений', 'Тарцус', NULL, 7);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(7, 'Егор', 'Батов', NULL, 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(8, 'Александр', 'Юдин', NULL, 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(9, 'Никита', 'Карасев', 'Кит', 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(10, 'Никита', 'Лукьянов', NULL, 1);

INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(11, NULL, NULL, 'Zombie Whytte', 3);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(12, NULL, NULL, 'Zombie Red', 3);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(13, NULL, NULL, 'Zombie Black', 8);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(14, NULL, NULL, 'Purple Z', 8);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(15, NULL, NULL, 'Zombie Gr€¥', 3);

INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(16, 'Игорь', 'Власьев', 'Ваганыч', 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(17, 'Игорь', 'Петров', NULL, 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(18, 'Виктор', 'Бурко', NULL, 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(19, 'Владимир', 'Евдокимов', NULL, 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(20, 'Алексей', 'Луцай', NULL, 1);
INSERT INTO Artists(artist_id, first_name, last_name, nickname, main_style_id)
VALUES(21, 'Виктор', 'Евстратов', NULL, 1);
```

Таблица Bands:

```
CREATE TABLE Bands (
    band_id INTEGER NOT NULL PRIMARY KEY,
    band_name VARCHAR(100) NOT NULL,
    formation_year INTEGER NOT NULL,
    breakup_year INTEGER,
    main_style_id NUMBER REFERENCES MusicStyles(style_id)
);
INSERT INTO Bands(band_id, band_name, formation_year, breakup_year,
main_style_id)
```

```
VALUES(1, 'ТАЙМСКВЕР', 2009, NULL, 1);
INSERT INTO Bands(band_id, band_name, formation_year, breakup_year,
main_style_id)
VALUES(2, 'ZOMBIEZ', 2017, NULL, 3);
INSERT INTO Bands(band_id, band_name, formation_year, breakup_year,
main_style_id)
VALUES(3, 'этажность', 2013, NULL, 9);
INSERT INTO Bands(band_id, band_name, formation_year, breakup_year,
main_style_id)
VALUES(4, 'ЖЩ', 2011, NULL, 1);
```

Таблица Songs:

```
CREATE TABLE Songs (
    song_id INTEGER NOT NULL PRIMARY KEY,
    song_name VARCHAR(100) NOT NULL,
    style_id NUMBER REFERENCES MusicStyles(style_id),
    duration NUMBER NOT NULL,
    album_id NUMBER REFERENCES Albums(album_id),
    band_id NUMBER REFERENCES Bands(band_id)
);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(1, 'Крошечное солнце', 1, 3, 1, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(2, 'Тени', 1, 3, 1, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(3, 'Реквием', 1, 3, 1, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(4, 'Из вечносолнечного льда', 1, 4, 1, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(5, 'Мой серый город', 1, 3, 1, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(6, 'Декабрь', 1, 4, 1, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(7, 'Космос молчит', 1, 3, 1, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(8, 'Я иду за тобой', 1, 4, 1, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(9, 'Декабрь (акустика)', 1, 3, 1, 1);

INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(10, 'Интро', 1, 0, 2, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(11, 'Вкус крови', 1, 4, 2, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(12, 'Я.Л.Н.Л.Л', 1, 4, 2, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(13, 'Соль и пепел', 1, 4, 2, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(14, 'Самурай', 1, 3, 2, 1);
```

```
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(15, 'Копия копии', 1, 4, 2, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(16, 'Пульс', 1, 4, 2, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(17, 'Облако', 1, 4, 2, 1);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(18, 'Я.Л.Н.Л.Л. (Master Version)', 1, 4, 2, 1);

INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(20, 'MEATEATERZ', 3, 3, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(21, 'RATTENLOCH', 3, 4, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(22, 'GEHENNA', 5, 3, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(23, 'TAG 8', 3, 3, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(24, 'HERZSTILLSTAND', 5, 4, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(25, 'ZOMBIEMASK', 3, 4, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(26, 'URNE', 3, 4, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(27, 'WALL OF Z', 5, 3, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(28, 'UNSTERBLICH', 8, 3, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(29, 'ONYX', 8, 3, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(30, 'ZETTEL AM ZEH', 3, 3, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(31, 'MENSCH UND ICH', 8, 4, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(32, 'ANTICHRIST', 8, 5, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(33, 'KILL SHIT', 3, 4, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(34, 'ICH VS ICH 2', 8, 5, 3, 2);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(35, 'KATHARZIZ', 8, 3, 3, 2);

INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(40, 'Ужас без конца', 1, 3, 4, 3);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(41, 'Ладожский вокзал', 9, 3, 4, 3);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(42, 'Защити нас', 1, 3, 4, 3);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(43, 'Пиво', 9, 3, 4, 3);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
```

```
VALUES(44, 'Страна глухих', 9, 4, 4, 3);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(45, 'Сон', 9, 4, 4, 3);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(46, 'Ничего не хочется', 1, 3, 4, 3);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(47, 'Жизнь', 9, 4, 4, 3);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(48, 'Смелость', 1, 3, 5, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(49, 'Там же,где и я', 1, 3, 5, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(50, 'Врать', 1, 2, 5, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(51, 'Дизмораль', 9, 4, 5, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(52, 'Король', 1, 4, 5, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(53, 'Тоска', 9, 4, 5, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(54, 'Крепким словом', 9, 2, 5, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(55, 'Разберутся без нас', 1, 4, 5, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(56, 'Мне Не Нужен Адрес', 1, 4, 6, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(57, 'Мрак', 9, 2, 6, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(58, 'Пусть Люди Запомнят Меня Таким', 9, 5, 6, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(59, 'Когда Я Умру', 9, 3, 6, 4);
INSERT INTO Songs(song_id, song_name, style_id, duration, album_id, band_id)
VALUES(60, 'Жить Так Долго', 9, 3, 6, 4);
```

Таблица Authorship:

```
CREATE TABLE Authorship (
    artist_id INTEGER NOT NULL,
    song_id INTEGER NOT NULL,
    PRIMARY KEY (artist_id, song_id),
    FOREIGN KEY (artist_id) REFERENCES Artists(artist_id),
    FOREIGN KEY (song_id) REFERENCES Songs(song_id)
);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 1);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 2);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 2);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 3);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 4);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 5);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 6);
```

```
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 6);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 7);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 7);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 8);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 9);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 9);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 10);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 10);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 11);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 11);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 12);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 12);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 13);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 13);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 14);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 14);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 15);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 15);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 16);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 16);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 17);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 17);
INSERT INTO Authorship(artist_id, song_id) VALUES(1, 18);
INSERT INTO Authorship(artist_id, song_id) VALUES(2, 18);

INSERT INTO Authorship(artist_id, song_id) VALUES(16, 40);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 41);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 42);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 43);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 44);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 45);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 46);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 47);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 48);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 49);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 50);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 51);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 52);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 53);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 54);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 55);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 56);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 57);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 58);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 59);
INSERT INTO Authorship(artist_id, song_id) VALUES(16, 60);
```

Таблица Membership:

```
CREATE TABLE Membership (
```

```
    artist_id INTEGER NOT NULL,
    band_id INTEGER NOT NULL,
    join_year INTEGER NOT NULL,
    left_year INTEGER,
    PRIMARY KEY (artist_id, band_id),
    FOREIGN KEY (artist_id) REFERENCES Artists(artist_id),
    FOREIGN KEY (band_id) REFERENCES Bands(band_id)
);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(1, 1,
2009, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(2, 1,
2013, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(3, 1,
2015, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(4, 1,
2022, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(5, 1,
2024, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(6, 1,
2015, 2021);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(7, 1,
2015, 2022);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(8, 1,
2020, 2021);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(9, 1,
2021, 2021);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(10, 1,
2021, 2024);

INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(11, 2,
2017, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(12, 2,
2017, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(13, 2,
2017, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(14, 2,
2017, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(15, 2,
2017, NULL);

INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(16, 3,
2013, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(16, 4,
2011, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(17, 4,
2013, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(18, 4,
2011, NULL);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(19, 4,
2020, NULL);
```

```
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(20, 4, 2011, 2011);
INSERT INTO Membership(artist_id, band_id, join_year, left_year) VALUES(21, 4, 2013, 2020);
```

Описание программного продукта

Для базы данных был создан пакет MusicPack со следующими функциями и процедурами:

1) GetCurrentMembers(bandName VARCHAR2) – процедура для вывода текущих участников заданной группы (название группы передается в качестве параметра bandName).
2) GetTotalDuration – функция для подсчета суммы длительности всех альбомов в базе данных. Возвращает значение NUMBER – сумму длительности.
3) GetArtistsWithNickname – процедура, которая выводит имя, фамилию, псевдоним, группу, дату вступления в группу и дату выхода из группы исполнителей, которые используют псевдоним.
4) CountAlbumsByMediaType – процедура, которая считает количество альбомов, выпущенных на каждом виде носителя.
5) GetSongsByArtist(artistName VARCHAR2) – процедура, которая выводит все песни, для которых заданный исполнитель является автором (имя и фамилия или псевдоним исполнителя передается в качестве параметра artistName).
6) CountSongsByArtist(artistName VARCHAR2) – функция, которая считает количество песен, для которых заданный исполнитель является автором (имя и фамилия или псевдоним исполнителя передается в качестве параметра artistName). Функция возвращает значение NUMBER – количество песен.

```
CREATE OR REPLACE PACKAGE MusicPack AS
    -- 1. Процедура для вывода текущих участников заданной группы
    PROCEDURE GetCurrentMembers(bandName VARCHAR2);

    -- 2. Функция для подсчета суммы длительности всех альбомов
    FUNCTION GetTotalDuration RETURN NUMBER;

    -- 3. Процедура для вывода всех исполнителей с псевдонимом и групп, в которых они состоят или состояли
    PROCEDURE GetArtistsWithNickname;

    -- 4. Процедура для подсчета количества выпущенных альбомов на каждый вид дистрибуции
    PROCEDURE CountAlbumsByMediaType;

    -- 5. Процедура для вывода всех песен, автором которых является заданный исполнитель
    PROCEDURE GetSongsByArtist(artistName VARCHAR2);

    -- 6. Функция для подсчета количества песен, для которых заданный исполнитель является автором
    FUNCTION CountSongsByArtist(artistName VARCHAR2) RETURN NUMBER;
END MusicPack;

CREATE OR REPLACE PACKAGE BODY MusicPack AS
```

```
    -- 1. Процедура для вывода текущих участников заданной группы
    PROCEDURE GetCurrentMembers(bandName VARCHAR2) IS
    BEGIN
        FOR rec IN (
            SELECT a.first_name, a.last_name, a.nickname, m.join_year,
m.left_year
            FROM Bands b
            JOIN Membership m ON b.band_id = m.band_id
            JOIN Artists a ON m.artist_id = a.artist_id
            WHERE b.band_name = bandName AND m.left_year IS NULL
        ) LOOP
            DBMS_OUTPUT.PUT_LINE('Artist: ' || rec.first_name || ' ' ||
rec.last_name || ' ' || rec.nickname || ', Joined: ' || rec.join_year);
        END LOOP;
    END GetCurrentMembers;

    -- 2. Функция для подсчета суммы длительности всех альбомов
    FUNCTION GetTotalDuration RETURN NUMBER IS
        total_duration NUMBER;
    BEGIN
        SELECT SUM(total_duration)
        INTO total_duration
        FROM Albums;
        RETURN total_duration;
    END GetTotalDuration;

    -- 3. Процедура для вывода всех исполнителей с псевдонимом и групп, в которых
они состоят или состояли
    PROCEDURE GetArtistsWithNickname IS
    BEGIN
        FOR rec IN (
            SELECT a.nickname, a.first_name, a.last_name, b.band_name,
m.join_year, m.left_year
            FROM Artists a
            LEFT JOIN Membership m ON a.artist_id = m.artist_id
            LEFT JOIN Bands b ON m.band_id = b.band_id
            WHERE a.nickname IS NOT NULL
        ) LOOP
            DBMS_OUTPUT.PUT_LINE('Artist: ' || rec.first_name || ' ' ||
rec.last_name || ' (Nickname: ' || rec.nickname || '), Band: ' || rec.band_name
|| ', Joined: ' || rec.join_year || ', Left: ' || rec.left_year);
        END LOOP;
    END GetArtistsWithNickname;

    -- 4. Процедура для подсчета количества выпущенных альбомов на каждый вид
дистрибуции
    PROCEDURE CountAlbumsByMediaType IS
    BEGIN
        FOR rec IN (
            SELECT mt.type_name, COUNT(a.album_id) AS album_count
            FROM MediaTypes mt
```

```
            JOIN Albums a ON mt.type_id = a.media_type_id
            GROUP BY mt.type_name
        ) LOOP
            DBMS_OUTPUT.PUT_LINE('Media Type: ' || rec.type_name || ', Albums: '
|| rec.album_count);
        END LOOP;
    END CountAlbumsByMediaType;

    -- 5. Процедура для вывода всех песен, автором которых является заданный
исполнитель
    PROCEDURE GetSongsByArtist(artistName VARCHAR2) IS
    BEGIN
        FOR rec IN (
            SELECT s.song_name
            FROM Songs s
            JOIN Authorship au ON s.song_id = au.song_id
            JOIN Artists ar ON au.artist_id = ar.artist_id
            WHERE ar.first_name || ' ' || ar.last_name = artistName OR
ar.nickname = artistName
        ) LOOP
            DBMS_OUTPUT.PUT_LINE('Song: ' || rec.song_name);
        END LOOP;
    END GetSongsByArtist;

    -- 6. Функция для подсчета количества песен, для которых заданный исполнитель
является автором
    FUNCTION CountSongsByArtist(artistName VARCHAR2) RETURN NUMBER IS
        song_count NUMBER;
    BEGIN
        SELECT COUNT(s.song_id)
        INTO song_count
        FROM Songs s
        JOIN Authorship au ON s.song_id = au.song_id
        JOIN Artists ar ON au.artist_id = ar.artist_id
        WHERE ar.first_name || ' ' || ar.last_name = artistName OR ar.nickname =
artistName;

        RETURN song_count;
    END CountSongsByArtist;
END MusicPack;
```

Примеры вызовов функций и процедур:

Процедура GetCurrentMembers:

Вызов:

```
BEGIN
    MusicPack.GetCurrentMembers('ТАйМСКВЕР');
END;
```

Вывод:

Artist: Андрей Грязнов , Joined: 2009

Artist: Кирилл Бабиев , Joined: 2013

Artist: Дмитрий Теплов , Joined: 2015

Artist: Николай Карпенко , Joined: 2022

Artist: Степан Четвериков , Joined: 2024

Функция GetTotalDuration:

Вызов:

```
DECLARE
    DurationSum NUMBER;
BEGIN
    DurationSum := MusicPack.GetTotalDuration();
    DBMS_OUTPUT.put_line('Total duration of all albums (in min): ' ||
DurationSum);
END;
```

Вывод: Total duration of all albums (in min): 190

Процедура GetArtistsWithNickname:

Вызов:

```
BEGIN
    MusicPack.GetArtistsWithNickname();
END;
```

Вывод:

Artist: Никита Карасев (Nickname: Кит), Band: ТАйМСКВЕР, Joined: 2021, Left: 2021

Artist: (Nickname: Zombie Whytte), Band: ZOMBIEZ, Joined: 2017, Left:

Artist: (Nickname: Zombie Red), Band: ZOMBIEZ, Joined: 2017, Left:

Artist: (Nickname: Zombie Black), Band: ZOMBIEZ, Joined: 2017, Left:

Artist: (Nickname: Purple Z), Band: ZOMBIEZ, Joined: 2017, Left:

Artist: (Nickname: Zombie Gr€¥), Band: ZOMBIEZ, Joined: 2017, Left:

Artist: Игорь Власьев (Nickname: Ваганыч), Band: этажность, Joined: 2013, Left:

Artist: Игорь Власьев (Nickname: Ваганыч), Band: ЖЩ, Joined: 2011, Left:

Процедура CountAlbumsByMediaType:

Вызов:

```
BEGIN
    MusicPack.CountAlbumsByMediaType();
END;
```

Вывод:

Media Type: Digital, Albums: 4

Media Type: CD, Albums: 2

Процедура GetSongsByArtist:

Вызов:

```
BEGIN
    MusicPack.GetSongsByArtist('Ваганыч');
END;
```

Вывод:

Song: Ужас без конца

Song: Ладожский вокзал

Song: Защити нас

Song: Пиво

Song: Страна глухих

Song: Сон

Song: Ничего не хочется

Song: Жизнь

Song: Смелость

Song: Там же,где и я

Song: Врать

Song: Дизмораль

Song: Король

Song: Тоска

Song: Крепким словом

Song: Разберутся без нас

Song: Мне Не Нужен Адрес

Song: Мрак

Song: Пусть Люди Запомнят Меня Таким

Song: Когда Я Умру

Song: Жить Так Долго

Функция CountSongsByArtist:

Вызов:

```
DECLARE
    SongsCount NUMBER;
    ArtistName VARCHAR2(100) := 'Андрей Грязнов';
BEGIN
    SongsCount := MusicPack.CountSongsByArtist(ArtistName);
```

```
    DBMS_OUTPUT.put_line('Number of songs authored by ' || ArtistName || ': ' ||
SongsCount);
END;
```

Вывод: Number of songs authored by Андрей Грязнов: 18

Для обеспечения каскадных изменений в таблицах реализованы следующие триггеры:

Триггеры CascadeStyleUpdate, CascadeMediaUpdate, CascadeAlbumUpdate, CascadeSongUpdate, CascadeArtistUpdate, CascadeBandUpdate обеспечивают обновление полей идентификаторов в связанных таблицах.

Триггеры CascadeAlbumDelete, CascadeSongDelete, CascadeArtistDelete, CascadeBandDelete обеспечивают удаление несуществующих записей из связанных таблиц после удаления данных из таблицы, к которой привязан триггер.

```
CREATE OR REPLACE TRIGGER CascadeStyleUpdate
AFTER UPDATE OF style_id
ON MusicStyles
FOR EACH ROW
BEGIN
    UPDATE Albums
    SET style_id = :NEW.style_id
    WHERE style_id = :OLD.style_id;

    UPDATE Bands
    SET main_style_id = :NEW.style_id
    WHERE main_style_id = :OLD.style_id;

    UPDATE Songs
    SET style_id = :NEW.style_id
    WHERE style_id = :OLD.style_id;

    UPDATE Artists
    SET main_style_id = :NEW.style_id
    WHERE main_style_id = :OLD.style_id;
END;

CREATE OR REPLACE TRIGGER CascadeMediaUpdate
AFTER UPDATE OF type_id
ON MediaTypes
FOR EACH ROW
BEGIN
    UPDATE Albums
    SET media_type_id = :NEW.type_id
    WHERE media_type_id = :OLD.type_id;
END;

CREATE OR REPLACE TRIGGER CascadeAlbumUpdate
AFTER UPDATE OF album_id
ON Albums
FOR EACH ROW
```

```
BEGIN
    UPDATE Songs
    SET album_id = :NEW.album_id
    WHERE album_id = :OLD.album_id;
END;

CREATE OR REPLACE TRIGGER CascadeSongUpdate
AFTER UPDATE OF song_id
ON Songs
FOR EACH ROW
BEGIN
    UPDATE Authorship
    SET song_id = :NEW.song_id
    WHERE song_id = :OLD.song_id;
END;

CREATE OR REPLACE TRIGGER CascadeArtistUpdate
AFTER UPDATE OF artist_id
ON Artists
FOR EACH ROW
BEGIN
    UPDATE Authorship
    SET artist_id = :NEW.artist_id
    WHERE artist_id = :OLD.artist_id;

    UPDATE Membership
    SET artist_id = :NEW.artist_id
    WHERE artist_id = :OLD.artist_id;
END;

CREATE OR REPLACE TRIGGER CascadeBandUpdate
AFTER UPDATE OF band_id
ON Bands
FOR EACH ROW
BEGIN
    UPDATE Songs
    SET band_id = :NEW.band_id
    WHERE band_id = :OLD.band_id;

    UPDATE Membership
    SET band_id = :NEW.band_id
    WHERE band_id = :OLD.band_id;
END;

CREATE OR REPLACE TRIGGER CascadeAlbumDelete
AFTER DELETE ON Albums
FOR EACH ROW
BEGIN
    DELETE FROM Songs
    WHERE album_id = :OLD.album_id;
END;
```

```
CREATE OR REPLACE TRIGGER CascadeSongDelete
AFTER DELETE ON Songs
FOR EACH ROW
BEGIN
    DELETE FROM Authorship
    WHERE song_id = :OLD.song_id;
END;

CREATE OR REPLACE TRIGGER CascadeArtistDelete
AFTER DELETE ON Artists
FOR EACH ROW
BEGIN
    DELETE FROM Authorship
    WHERE artist_id = :OLD.artist_id;

    DELETE FROM Membership
    WHERE artist_id = :OLD.artist_id;
END;

CREATE OR REPLACE TRIGGER CascadeBandDelete
AFTER DELETE ON Bands
FOR EACH ROW
BEGIN
    DELETE FROM Membership
    WHERE band_id = :OLD.band_id;

    DELETE FROM Songs
    WHERE band_id = :OLD.band_id;
END;
```