МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационные системы и технологии

Специальность 1–40 01 01 Программное обеспечение информационных технологий

Специализация 1–40 01 01 Программное обеспечение информационных технологий (программирование интернет – изданий)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

«Музыкальная площадка»

Выполнил студент Эльзаров Урмат Эльзарович

(Ф.И.О.)

Руководитель проекта асс., Н.И. Уласевич

(учен. степень, звание, должность, Ф.И.О., подпись)

Заведующий кафедрой к.т.н., доц. Смелов В.В .

(учен. степень, звание, должность, Ф.И.О., подпись)

Консультант:  асс., Н.И. Уласевич.

(учен. степень, звание, должность, Ф.И.О., подпись)

Нормоконтролер: асс., Н.И. Уласевич

(учен. степень, звание, должность, Ф.И.О., подпись)

Курсовой проект защищен с оценкой

**Содержание**

[Введение 3](#_Toc130217198)

[1. Постановка задачи 4](#_Toc130217199)

[1.1. Обзор аналогичных решений 4](#_Toc130217200)

[1.2. Техническое задание 6](#_Toc130217201)

[1.3. Выбор средств реализации программного продукта 7](#_Toc130217202)

[1.4. Вывод 7](#_Toc130217203)

[2. Проектирование базы данных 8](#_Toc130217204)

Введение

Музыка является древнейшим искусством. В таком искусстве есть только одно ограничение – фантазия композитора. С помощью музыки можно развеять тоску, задуматься над бытием или отношением к жизни, загрустить или просто отдохнуть. Именно поэтому музыку считают неотъемлемой частью нашей жизни.

Современную жизнь довольно тяжело представить без музыки. Она буквально повсюду, треки люди слушают в общественном транспорте или в торговых центрах, они ходят на концерты или создают свои собственные музыкальные коллекции. Такая популяризация музыки в жизни общества легко объяснима, ведь она дарит всем слушателям уникальную возможность снять стресс, избавиться от излишнего эмоционального напряжения. Некоторые психологи даже используют музыкальную терапию в своей практике, она помогает их клиентам бороться с депрессиями или апатиями.  
Сегодня каждый пользователь может [слушать музыку](https://muzonov.net/) прямо в режиме онлайн со специализированных музыкальных сайтов. Подобных ресурсов в сети интернет достаточно много, каждый из них предлагает свои уникальные условия для непосредственного взаимодействия с пользователями. Так, например, для работы с возможностями одних платформ обязательно нужна предварительная регистрация, а на других сайтах можно свободно работать и без нее. У каждого музыкального web ресурса, безусловно, имеются свои определенные достоинства и нюансы.  
Целью данной работы являлась разработка реляционной базы данных для музыкальной площадки. Эта база данных должна составлялась для обеспечения клиента доступом к музыкальным композициям, имеющимся на ней. Так же было необходимо разработать соответствующее приложение, для демонстрации её работы.

База данных — это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. Реляционная база данных – это набор данных с предопределенными связями между ними. Эти данные организованны в виде набора таблиц, состоящих из столбцов и строк. В качестве СУБД для базы данных была выбрана Oracle 12c, в связи с ее высокой производительностью и надежностью.

В основной части будут затронуты все аспекты разработки проекта и обоснованы некоторые технические приёмы, к которым приходилось прибегнуть, с целью реализации работы веб-сервера с базой данных.

Для обеспечения безопасности пользователей приложения в моем курсовом проекте используется технология шифрования паролей от аккаунта перед записью их в базу данных. А также для обеспечения работы приложения использовались мультимедийные типы данных при хранении изображений и аудио файлов.

В пояснительной записке вы сможете найти краткую информацию о похожих продуктах, архитектуре, реализации проекта, руководстве пользователя.

1. Постановка задачи
   1. **Обзор аналогичных решений**

Музыка играет огромную роль в жизни человека. Одной из главных ее функций является объединять людей, поскольку язык музыки понятен без перевода. Значимость музыки заключается в ее способности обогащать жизнь человека и общества в целом.

Изобретение радио и кинематографа позволило перейти на следующий уровень инженерных разработок – телевидения, которое стало играть важную роль в продвижении музыки и популяризации артистов. В начале 1980-х гг. музыкальное телевидение затмило важность радио. А сегодня стриминговые сервисы стали еще популярнее и удобнее телевидения.

Одним из основных трендов музыкальных приложений остается курируемая музыка. Это так называемые «радиостанции», создающиеся с минимальных использованием компьютерных алгоритмов. В составлении плейлистов участвуют люди – музыкальные эксперты, знаменитости и т.д.

При разработке своего приложения я обратила внимания на самые успешные примеры популярнейших музыкальных площадок. Рассмотрим их ниже.

Сегодня Spotify – крупнейший в мире музыкальный стриминговый сервис. Он располагает библиотекой c 35 миллионами песен и имеет 160 миллионов активных пользователей. Интерфейс этого сервиса можно увидеть на рисунке 1.1.

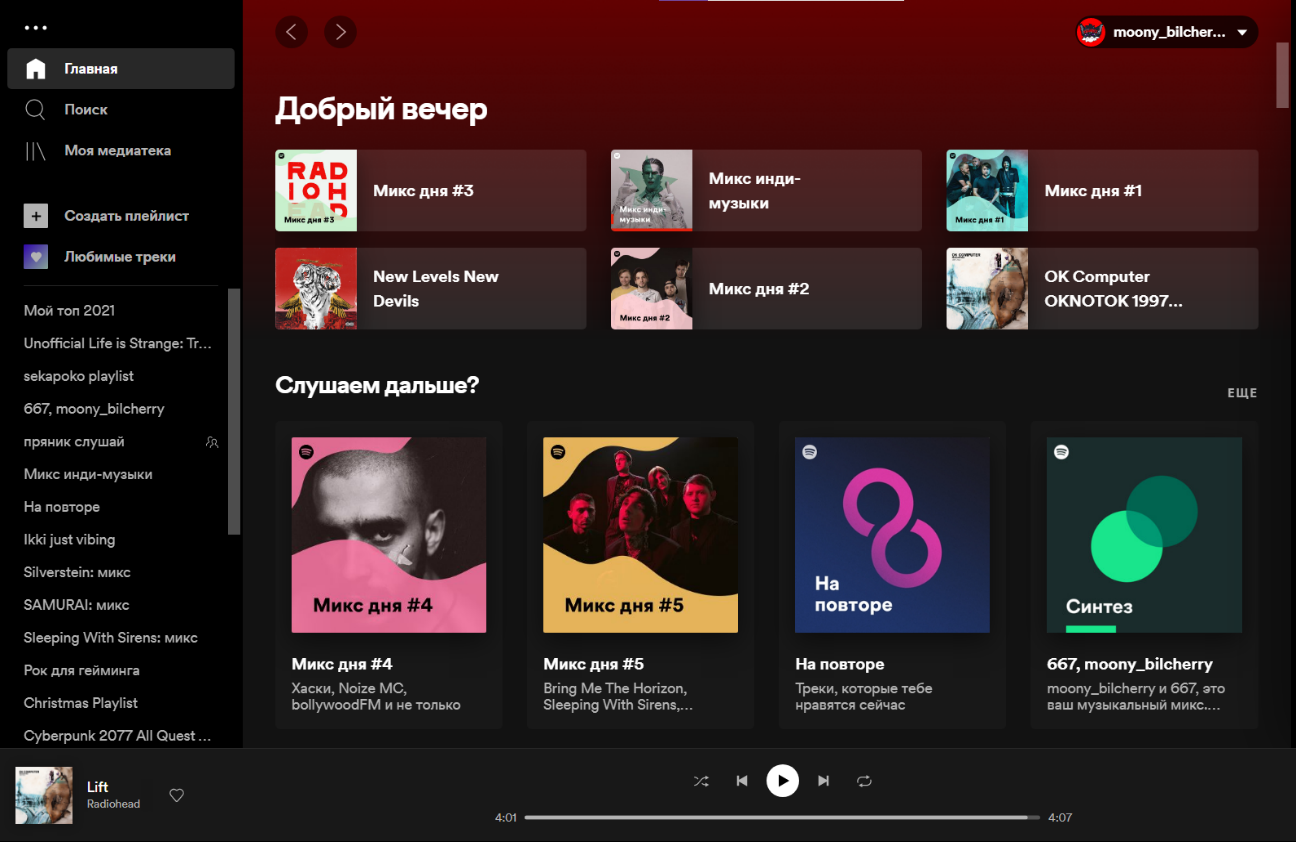


Рисунок 1.1 – Домашняя страница Spotify

С домашней страницы в Spotify пользователь может воспользоваться поиском, попасть в свою библиотеку, обратиться к недавно прослушанным плейлистам и открыть настойки своего аккаунта. Также Spotify по умолчанию имеет темный интерфейс

По популярности Apple Music идёт следом за Spotify, с 56 миллионами пользователей и большей по размеру библиотекой в 45 миллионов песен. Рассмотрим эту площадку на рисунке 1.2.

Apple Music на домашней странице имеет доступ к расширенному поиску. Пользователь получает рекомендации на основании своих прослушиваний и так же первым делом может обратить свое внимание на новые релизы от знакомых исполнителей. В боковом меню можно перейти к списку известных исполнителей, альбомов и песен.

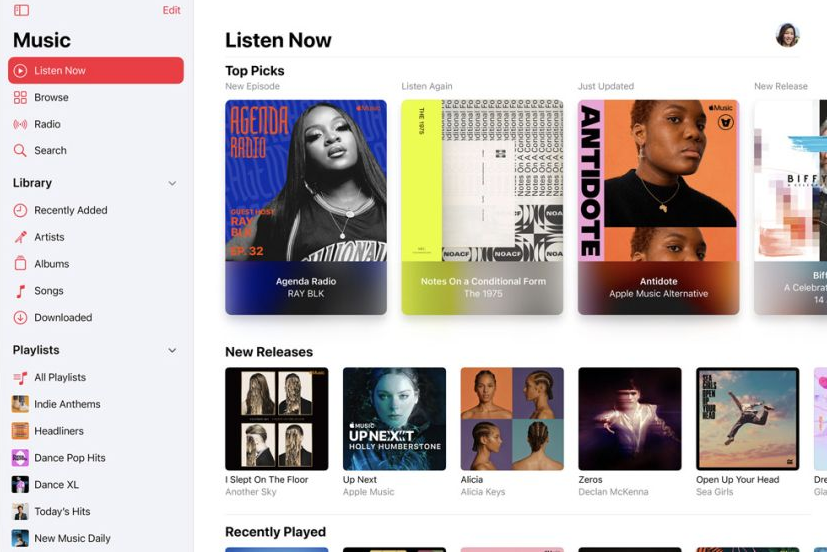


Рисунок 1.2 – Домашняя страница Apple Music

Разобрав два приведенных выше сервиса для стриминга музыки, можно сформировать общий функционал необходимый для работы приложения.

Прежде всего, естественно, база данных должна корректно работать с изображениями и аудио файлами. Обязательно должен быть поиск песен по исполнителю, альбому и названию песни, также пользователь должен быть способен сохранять в личный плейлист понравившиеся ему песню.

Также стоит ввести простейшее меню настроек, чтобы пользователь мог изменить свой пароль или логин по желанию.

* 1. **Техническое задание**

Обзор вышеперечисленных известных аналогов позволяет проанализировать все преимущества и недостатки альтернативных возможностей и позволяет сформулировать список требований, предъявляемых к программному средству, разрабатываемому в данном курсовом проекте. Программное средство должно обеспечивать возможность выполнения перечисленных ниже функций:

* управление администратором базой данных;
* возможность пользователю зарегистрироваться или войти в существующую учетную запись;
* возможность авторизированным пользователям просмотра информации о композициях, альбомах и плейлиста;
* Поиск аудиозаписей по альбому и исполнителю.
* Загрузка аудиозаписей на платформу администратором.
* Взаимодействие с базой данных при помощи хранимых процедур.
  1. **Выбор средств реализации программного продукта**

При разработке приложения были использованы:

* интегрированная среда разработки Microsoft Visual Studio 2022;
* программная платформа .NET Framework 6.0;
* язык программирования C#;
* расширяемый язык разметки XAML;
* технология WPF;
* пакет Oracle Managed Data Access для .NET
* Oracle 12c.
  1. **Вывод**

В данном разделе мы просмотреть и проанализировать аналогичные решения благодаря чему мы смогли узнать о том как дола быть реализована наше приложение и какие данные нужны для создания базы данных .

1. Проектирование базы данных
   1. Таблицы базы данных

Для реализации базы данных музыкальной площадки было разработано 6 таблиц. Диаграмма базы данных со структурой связей представлена в приложении А.

При разработке приложение получило рабочее название «MoonyFM», соответственно схема базы данных называется DBMoonyFM. В ее структуру входят следующие таблицы: role\_table, user\_table, artist\_table, album\_table, song\_table и saved\_table. Ниже мы разберем каждую из них более подробно.

Таблица role\_table представляет собой перечень ролей, которые могут иметь пользователи. Имеющиеся столбцы: role\_id (идентификатор роли, типа number(10), первичный ключ) и role\_name (текстовое название роли, тип varchar2(30)). Имеющиеся по умолчанию роли: USER (рядовой пользователь) и OWNER (пользователь с правами администратора).

Таблица user\_table содержит информацию о пользователях, а именно: user\_id (идентификатор пользователя, тип number(10), первичный ключ), user\_login (логин пользователя, типа varchar2(30)), user\_password (пароль пользователя, тип varchar2(30), шифруется в процессе регистрации и расшифровывается при проверке во время авторизации) и user\_role (идентификатор роли, тип number(10), внешний ключ).

Таблица artist\_table содержит информацию о музыкальных исполнителях: artist\_id (идентификатор исполнителя, тип number(10), первичный ключ), artist\_name (имя исполнителя, тип varchar2(10)).

Таблица album\_table представляет собой альбомы, выпущенные исполнителями, и потому содержит следующие столбцы: album\_id (идентификатор альбома, тип number(10), первичный ключ), artist\_id (идентификатор исполнителя, тип number(10), внешний ключ), album\_name (текстовое название альбома, тип varchar2(30)), album\_released (год выпуска альбома, тип number(10)) и album\_blob (столбец для хранения обложки альбома, тип данных blob).

Таблица song\_table отвечает за хранение информации о музыкальных композициях, имеющихся в базе данных. А именно: song\_id (идентификатор песни, тип number(10), первичный ключ), song\_artist (идентификатор исполнителя, тип number(10), внешний ключ), song\_album (идентификатор альбома, тип number(10), внешний ключ), song\_name (текстовое название песни, тип varchar2(30)) и song\_blob (столбец для хранения аудиофайла песни, тип данных blob).

Последняя таблица, saved\_table, используется для сопоставления пользователя с понравившейся ему композицией. Благодаря ей пользователи сохраняют песни в свой плейлист. Информация, хранимая в данной таблице: saved\_id (идентификатор экземпляра сохраненной песни, number(10), первичный ключ), saved\_user (идентификатор пользователя, сохранившего песню; тип number(10), внешний ключ) и saved\_song (идентификатор сохраненной песни, тип number(10), внешний ключ).

* 1. Процедуры и функции для решения поставленных задач

Для управления данными через приложение администратор использует хранимые процедуры. Хранимая процедура – объект базы данных, представляющий собой набор SQL-инструкций, который компилируется один раз и хранится на сервере.

Написанные мной в ходе разработки курсового проекта процедуры и функции можно разбить на несколько категорий. Их мы рассмотрим ниже.

* + 1. Идентификация пользователей

Во-первых, был написан блок процедур и функций, отвечающих за идентификацию пользователей.

Прежде всего, рассмотрим функцию encryption\_password. Ее реализацию можно увидеть на рисунке 2.1. Эта функция позволяет зашифровать пароль пользователя. Она принимает один входной параметр: пароль пользователя, который следует зашифровать. В моем курсовом проекте для шифрования используется алгоритм AES с 128-разрядным ключом, метод сцепления CBC и механизм заполнения PKCS#5.

Для шифрования необходимо выбрать ключ. В приложении в качестве ключа используется строка «0710196810121972». Это значение относится к типу данных VARCHAR2. Чтобы использовать его в функции ENCRYPT, необходимо сначала преобразовать его к типу RAW. Для этого мы воспользуемся функцией STRING\_TO\_RAW встроенного пакета UTL\_I18N. Также вторым параметром указываем кодировку. Переменная l\_in\_val типа VARCHAR2 также должна быть преобразована к типу RAW. Переменную, которая будет передаваться в качестве возвращаемого значения функции, требуется преобразовать в шестнадцатеричный формат. Для этого используем функцию RAWTOHEX.



Рисунок 2.1 – Процедура encryption\_password

В обратных целях была разработана функция decryption\_password. Она в качестве параметра принимает зашифрованный пароль и должна в результате своей работе вернуть его в расшифрованном виде. Используя тот же самый ключ, передаем его в качестве одного из параметров в функцию DECRYPT, перед этим преобразовав его в RAW при помощи функции STRING\_TO\_RAW. Так же не забываем передать в функцию параметр typ и зашифрованный пароль типа RAW.

Результат функции преобразуем к строке при помощи функции RAW\_TO\_CHAR пакета UTL\_I18N. Реализацию этой функции можно увидеть на рисунке 2.2.

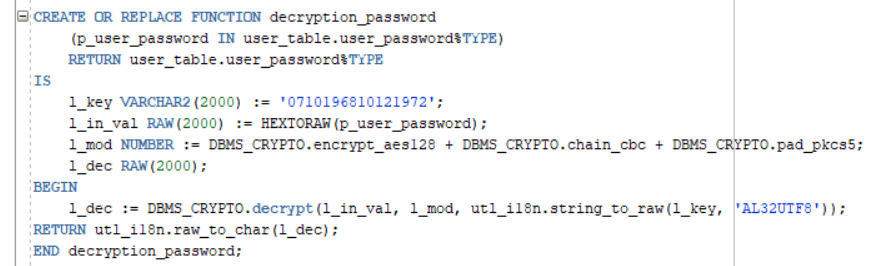


Рисунок 2.2 – Процедура decryption\_password

Далее была разработана хранимая процедура register\_user, с помощью которой происходит регистрация пользователей в приложении. Эта процедура принимает два параметра: p\_user\_login (логин пользователя, которого нужно зарегистрировать) и p\_user\_password (пароль от аккаунта этого пользователя). В случае, если указанный логин уже есть в базе данных, поднимается исключение. Если исключения не произошло, то строка успешно вставляется в таблицу user\_table, зашифровав пароль в процессе.

Процедура check\_role позволяет найти роль пользователя и выгружает её для дальнейших манипуляций. Принимает один входной параметр, а именно: логин пользователя (p\_user\_login), а также один выходной параметр: название роли пользователя (o\_user\_role).

Для авторизации пользователя в приложении разработана процедура log\_in\_user. У нее есть два входных параметра: p\_user\_login (логин пользователя) и p\_user\_password (пароль пользователя). Если указанная комбинация логина и пароля отсутствует в таблице user\_table, то поднимается исключение. Если исключения не произошло, то в приложение передается три выходных параметра, которые пригодятся далее в разработке логики приложения и будут занесены в поля статического класса DataWorker: o\_user\_id (идентификатор пользователя, прошедшего авторизацию), o\_user\_login (логин пользователя), o\_user\_role (роль пользователя).

* + 1. Процедуры для манипуляций в меню «Настройки»

Процедура search\_user позволяет найти информацию о пользователе и выгружает данные о нем из таблицу user\_table. Она используется для отображения данных в меню «Настройки». Процедура принимает один входной параметр, а именно: логин пользователя (p\_user\_login), а также два выходных параметра, а именно: логин пользователя (p\_user\_login), расшифрованный пароль пользователя (p\_user\_password).

Процедура update\_user\_login позволяет изменить логин пользователя. Данная процедура принимает два входных параметра: старый логин пользователя (p\_user\_login) и новый логин пользователя (p\_new\_user\_login), которым следует заменить старый.

Процедура update\_user\_password позволяет изменить пароль пользователя. Эта принимает два входных параметра, а именно: старый пароль пользователя (p\_user\_password) и расшифрованный новый пароль пользователя (p\_new\_user\_password), который заменит в таблице старый и будет зашифрован в процессе.

* + 1. Процедуры для добавления данных в таблицы

Процедура create\_artist служит для добавления нового артиста в базу данных. Данная процедура принимает один параметр: p\_artist\_name (имя артиста, который будет добавлен в базу данных). Поскольку все процедуры, описанные в данном пункте, аналогичны по своему алгоритму, имеет смысл показать в качестве примера только одну их них. Таким образом, на рисунке 2.3 можно увидеть реализацию функции create\_artist.

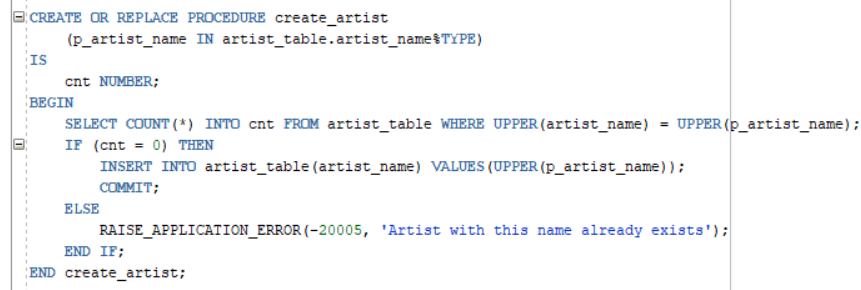


Рисунок 2.3 – Процедура create\_artist

Далее была разработана процедура для добавления нового альбома в базу данных. Эта процедура принимает три параметра: p\_artist\_name (имя исполнителя, которому принадлежит создаваемый альбом), p\_album\_name (название создаваемого альбома) и p\_album\_released (год выпуска альбома). Так как в хранимых процедурах нельзя передавать данные объемом более 32 Кб, изображение, служащее обложкой альбома, передается отдельно через операцию INSERT.

Процедура create\_song служит для добавления новой песни в базу данных. Эта процедура принимает три параметра: p\_artist\_name (имя исполнителя, которому принадлежит создаваемая песня), p\_album\_name (название альбома, к которому будет относится песня) и p\_song\_name (название добавляемой песни). Аудиофайл песни передается отдельно через операцию INSERT.

Процедура save\_song служит для добавления пользователем песни в свой плейлист. Эта процедура принимает два параметра: p\_user\_id (идентификатор пользователя), и p\_song\_id (идентификатор песни).

* + 1. Процедуры для изменения данных в таблицах

В этом блоке будут описаны процедуры, с помощью которых администратором изменяются данные в таблицах.

Процедура update\_artist отвечает за замену старого имени исполнителя на новое. Она соответственно принимает эти имена как параметры: p\_old\_artist и p\_new\_artist. Реализацию этой процедуры будем использовать как пример функций изменения. Она изображена на рисунке 2.4.

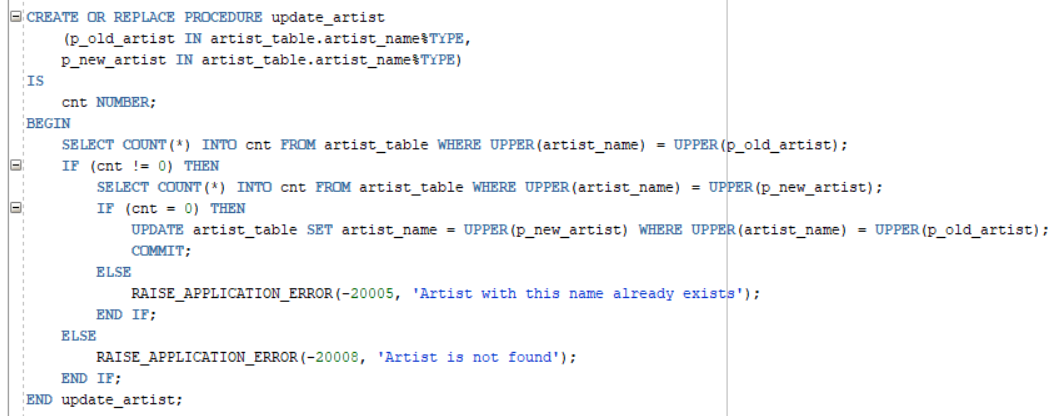


Рисунок 2.4 – Процедура update\_artist

Процедура update\_album\_name позволяет обновить название альбома. Она принимает два параметра: p\_ablum\_id (идентификатор альбома) и p\_new\_name (имя альбома, которым будет заменено старое).

Процедура update\_album\_year позволяет обновить год выпуска альбома. Она принимает два параметра: p\_ablum\_id (идентификатор альбома) и p\_new\_year (год выпуска альбома, которым будет заменен старый).

Процедура update\_song\_name позволяет обновить название уже существующей песни. Она принимает два параметра: p\_song\_id (идентификатор песни) и p\_new\_name (новое название для песни).

* + 1. Процедуры для удаления данных из таблиц

В этом блоке будут описаны процедуры, с помощью которых администратором удаляются данных из таблиц. Поскольку таблицы создавались с параметром ON DELETE CASCADE, удаление значений значительно упрощается. Когда данные в родительском ключе удаляются, все строки в дочерней таблице, которые зависят от значений удаленного родительского ключа, также удаляются.

Процедура delete\_artist отвечает за удаление исполнителя из базы данных. Она принимает один параметр: p\_id (идентификатор удаляемого исполнителя). Реализацию этой процедуры будем использовать как пример функций удаления. Она изображена на рисунке 2.5. При удалении исполнителя удалятся все связанные с ним альбомы, песни и экземпляры песен, сохраненных в плейлисты пользователей.

Процедура delete\_album отвечает за удаление альбома из базы данных. Она принимает один параметр: p\_id (идентификатор удаляемого альбома). При удалении альбома удалятся все связанные с ним песни и экземпляры песен, сохраненных в плейлисты пользователей.

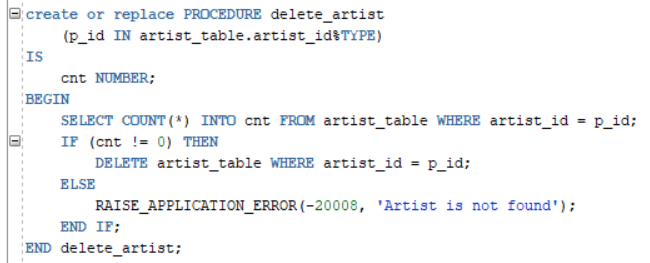


Рисунок 2.5 – Процедура delete\_artist

Процедура delete\_song отвечает за удаление песни из базы данных. Она принимает один параметр: p\_id (идентификатор удаляемой песни). При удалении песни удалятся все связанные с ней экземпляры песен, сохраненных в плейлисты пользователей.

Процедура remove\_song отвечает за удаление песни из плейлиста конкретного пользователя. Она принимает два параметра: p\_user\_id (идентификатор пользователя) и p\_song\_id (идентификатор удаляемой из плейлиста песни).

С помощью процедуры delete\_user пользователь может удалить свой аккаунт. Процедура принимает один параметр: p\_login (логин пользователя). При этом удалятся все данные о составе его плейлиста.

* + 1. Процедура для добавления 100 000 строк

С помощью процедуры insert\_100k\_users было зарегистрировано 100 000 пользователей. Она изображена на рисунке 2.6.

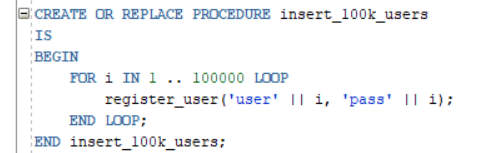


Рисунок 2.6 – Процедура insert\_100k\_users

Поскольку процесс получения администратором информации о всех пользователях занимал около двух минут, было решено отображать их в окне администратора по 20.

* + 1. Процедура для экспорта и импорта таблиц в XML

Для реализации обоих процедур потребовалось создать директорий с желаемым расположением при подключении под пользователем SYS к инстансу и выдать пользователю, владеющему базой данных, гран на чтение и запись из этого директория. Это изображено на рисунке 2.7.

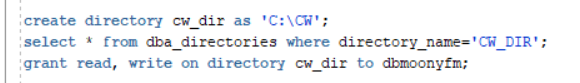


Рисунок 2.7 – Создание директория CW\_DIR

В рамках задания нужно было разработать процедуру, экспортирующую данные из таблицы, содержащей 100 000 строк, в XML документ. С этой целью разработана процедура users\_export. Она изображена на рисунке 2.8.

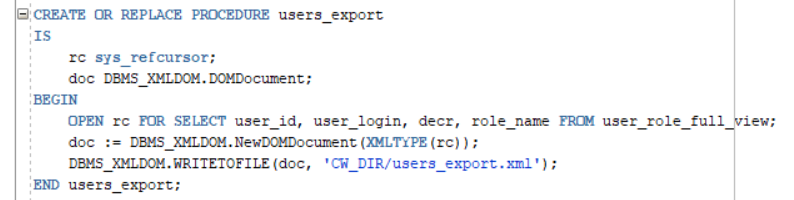


Рисунок 2.8 – Процедура users\_export

Также нужно было написать процедуру, которая бы, наоборот, импортировала существующие данные из XML документа в таблицу базы данных. В этом случае данные вставляются в таблицу artist\_table. Процедура artist\_import изображена на рисунке 2.9.

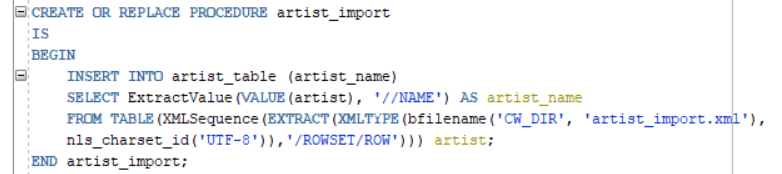


Рисунок 2.9 – Процедура artist\_import

* 1. Представления базы данных

Для удобства и повышения производительности при обращении с базой данных были разработаны следующие представления.

Представление user\_role\_full\_view соединяет между собой таблицы user\_table и user и возвращает следующие столбцы: user\_table.user\_id, user\_table.user\_login, decryption\_password(user\_table.user\_password) as decr, role\_table.role\_name. Оно используется при выводе информации о пользователях в окне администратора, а также при авторизации пользователя для проверки роли.

Представление artist\_album\_view объединяет таблицы artist\_table и album\_table и возвращает следующие столбцы: album\_table.album\_id, artist\_table.artist\_name, album\_table.album\_name, album\_table.album\_released, album\_table.album\_blob. Оно используется для отображения плиток с альбомами.

Представление artist\_album\_song\_view объединяет таблицы artist\_table, album\_table и song\_table и возвращает следующие столбцы: song\_table.song\_id, artist\_table.artist\_name, album\_table.album\_name, song\_table.song\_name, album\_table.album\_released, album\_table.album\_blob, song\_table.song\_blob. Оно содержит наиболее полную информацию о каждой из композиций.

Последнее представление, artist\_album\_song\_user\_view, дополняет предыдущее представление, дополнительно объединяя его с таблицей user\_table. Это используется для корректного отображения плиток с песнями во вкладке «Плейлист».

* 1. Пользователь базы данных

В разработанной мной базе данных присутствует один пользователь, который является хозяином схемы – DBMoonyFM. Он имеет грант на подключение к базе данных, грант на выборку, добавление, изменение и удаление данных из всех таблиц, описанных в пункте 2.1, и из всех представлений, описанных в пункте 2.3. Он может создавать таблицы, процедуры, представления и сессии. Также ему выдан грант на выполнение процедур и функций стандартного пакета dbms\_crypto.

Заключение

Результатом выполнения курсового проекта стало .NET приложение с использованием технологии WPF для операционной системы Windows. В ходе курсовой работы была разработана база данных в Oracle 12c на тему «Музыкальная площадка». Приложение является средством взаимодействия клиента с базой данных.

Были реализованы основные требования, указанные в листе задания вместе со следующими пунктами:

* регистрация и авторизация пользователей музыкальной площадки с учётом выданных им ролей;
* пароли при регистрации шифруются;
* мультимедийные типы данных, корректно записывающиеся в базу данных и затем воспроизводящиеся;
* в процессе разработки производительность была проверена средством диагностики SolarWinds DPA;
* поиск песен по исполнителю, альбому и названию самой песни;
* группировка песен по исполнителям, альбомам;
* возможность каждому пользователю сохранять понравившиеся песни в свой плейлист;
* добавление исполнителей администратором;
* добавление альбомов администратором;
* добавление песен администратором;
* администратор имеет доступ к информации о пользователях (включая их пароли);
* все пользователи могут изменять свой логин и пароль;
* все пользователи кроме администратора могут удалить свой аккаунт по желанию;
* импорт и экспорт данных таблиц в XML формат;
* заполнение таблицы на 100 000 записей (таблица пользователей).

Проект не может окончательно претендовать на полноценный коммерческий продукт, поскольку для демонстрации технологии и работы с базой данных были опущены некоторые атрибуты современных стриминговых площадок, реализация которых не относится к основной теме курсового проекта (например, программирование искусственного интеллекта, собирающего информацию о предпочтениях пользователя и формирующее на их основе подборки).

Протестировав все компоненты приложения, можно прийти к заключению, что все основные требования выполнены и приложение работает исправно.

Список литературы

1. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: https://metanit.com – Дата доступа: 15.10.2021.

2. Oracle-dba.ru [Электронный ресурс] / Режим доступа: https://oracle-dba.ru – Дата доступа: 27.10.2021.

3. Шифрование в PL/SQL: уроки программирования [Электронный ресурс] / Режим доступа: <https://oracle-patches.com/db/sql/3939-shifrovanie-v-pl-sql-uroki-programmirovaniya> – Дата доступа: 16.10.2021.

4. DBMS\_XMLDOM [Электронный ресурс] / Режим доступа: <https://docs.oracle.com/cd/B1930601/appdev.102/b14258/dxmldom.htm#i1076719> – Дата доступа: 20.10.2021.

5. Developing and Using Stored Procedures [Электронный ресурс] / Режим доступа: <https://docs.oracle.com/cd/B2835901/appdev.111/b28843/tdddgprocedures.htm> – Дата доступа: 26.10.2021.

6. Stackoverflow.com [Электронный ресурс] / Режим доступа: https://stackoverflow.com – Дата доступа: 08.12.2021

Приложение А

