**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»**

Інститут прикладного системного аналізу

Кафедра математичних методів системного аналізу

Курсова робота

**з дисципліни «Системи баз даних» на тему: «****Магазин музичних видань»**

Виконав (-ла):

студент (-ка) I курсу, групи KA-08

Рентюк Віталій Валерійович

Керівник:

Афанасьєва Ірина Валентинівна

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент (-ка)

Київ – 2022 рік

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ННК „ІПСА” НТУУ „КПІ”** | | | | | | |
| (назва вищого закладу освіти) | | | | | | |
| Кафедра | | ***Математичних методів системного аналізу*** | | | | |
| Дисципліна | | ***Системи баз даних*** | | | | |
| Галузь знань | | ***122 Комп’ютерні науки*** | | | | |
| Курс | ***другий*** | | Група | ***КА—08*** | Семестр | ***другий*** |

**ЗАВДАННЯ**

**на курсовий проект(роботу) студента**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Бондаренка Тимофія Андрійовича | | | | |
| (прізвище, ім’я, по батькові) | | | | |
| 1. Тема проекту(роботи) | Аукціон | | | |
| 2. Строк здачі студентом закінченого проекту(роботи) | | | ***XX.06.2022 р.*** | |
| 3. Вихідні дані до проекту(роботи) | |  | | |
|  | | | | |
|  | | | | |
|  | | | | |
| 4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці) | | | |  |
| ***1. Опис предметної сфери та постановка задачі.*** | | | | |
| ***2. Концептуальна модель бази даних. Діаграми потоків даних.*** | | | | |
| ***3. Інфологічна модель бази даних. Діаграми «сутність– зв’язок».*** | | | | |
| ***4. Даталогічна модель бази даних.*** | | | | |
| ***5. Фізична модель бази даних. Засоби забезпечення цілісності.*** | | | | |
| ***6. Опис реалізації проекту бази даних.*** | | | | |
| ***7. Висновки.*** | | | | |
| ***8. Джерела інформації.*** | | | | |
| ***9. Додаток А. Скрипт створення схеми БД*** | | | | |
| 5. Дата видачі завдання |  | | | |

**ЗМІСТ**

[ОПИС ПРЕДМЕТНОЇ СФЕРИ ТА ПОСТАНОВКА ЗАДАЧІ 3](#_bookmark0)

[КОНЦЕПТУАЛЬНА МОДЕЛЬ ІНФОРМАЦІЙНОЇ СИСТЕМИ.ДІАГРАМИ ПОТОКІВ](#_bookmark1) [ДАНИХ 4](#_bookmark1)

[ІНФОЛОГІЧНА МОДЕЛЬ БАЗИ ДАНИХ. ДІАГРАМИ “СУТНІСТЬ –ЗВʼЯЗОК” 6](#_bookmark2)

[ДАТАЛОГІЧНА МОДЕЛЬ БАЗИ ДАНИХ 9](#_bookmark3)

[ФІЗИЧНА МОДЕЛЬ БАЗИ ДАНИХ. ЗАСОБИ ЗАБЕЗПЕЧЕННЯЦІЛІСНОСТІ 12](#_bookmark4)

[ОПИС РЕАЛІЗАЦІЇ ПРОЕКТУ БАЗИ ДАНИХ 13](#_bookmark5)

[ВИСНОВКИ 15](#_bookmark6)

[ДЖЕРЕЛА ІНФОРМАЦІЇ 16](#_bookmark7)

[ДОДАТОК А 17](#_bookmark8)

# ОПИС ПРЕДМЕТНОЇ СФЕРИ ТА ПОСТАНОВКА ЗАДАЧІ

Торгівельна площадка – це приклад програмного продукту, що використовує бази даних та їхні можливості, а саме форми, запити та історію певних дій. Даний продукт може мати практичне використання, адже є одним із засобів торгівлі, зокрема онлайн.

Мета даної роботи - сформувати вимоги до проекту “ Магазин музичних видань ”, продумати ролі та їх можливості, створити схеми DFD та ERD. Створити необхідні таблиці з продуманими полями, прописати для них тригери, процедури та запити з використанням мови MySQL[1]. Розробити консольний додаток мовою Python[2].

Вимоги до програмного продукту - створити продукт, який дозволяє користувачам створювати запити та пропозиції на різні видання, укладати угоди, приймаючи пропозиції. Створити роль адміна, який може адмініструвати торгівельну площадку шляхом видалення недоброчесних пропозицій та бану користувачів.

Доступ до системи забезпечується за допомогою консольного додатку[3].

# КОНЦЕПТУАЛЬНА МОДЕЛЬ ІНФОРМАЦІЙНОЇ СИСТЕМИ. ДІАГРАМИ ПОТОКІВ ДАНИХ

Система має містити 3 типи користувачів (покупець, продавець, гість та адміністратор).

Тип “Продавець” складається з логіна, пароля та рейтинга. Він може переглядати пропозиції інших користувачів, створювати свої пропозиції, оновлювати їх та приймати заклики Покупців.

Тип “Покупець” є альтернативною версією продавця. Тобто він також має можливість переглядати пропозиції, але створює не пропозиції, а запити на видання, на які в свою чергу можуть відгукнутись продавці і приймає пропозиції продавця.

Тип “Адміністратор” має можливість адміністрування. Він може видалити заклик чи пропозицію користувача та зняти частину його рейтингових балів. Також він має можливість заблокувати користувача, якщо він зробив багато порушень.

Тип “Гість ” не має особливих полів, але має можливість зайти у систему як один із вище перелічених типів або зареєструватися. Також він може переглядати категорії і товари як і покупець чи покупець, проте не може ні приймати ні створювати свої пропозиції чи заклики.

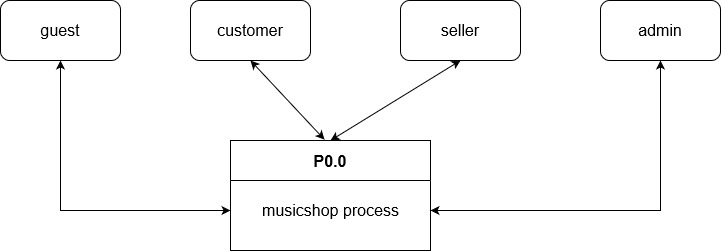
Взаємодію даних сутностей можна показати у більш компактному вигляді, наприклад DFD діаграмі:

Рис. 1. Діаграма потоків даних P0

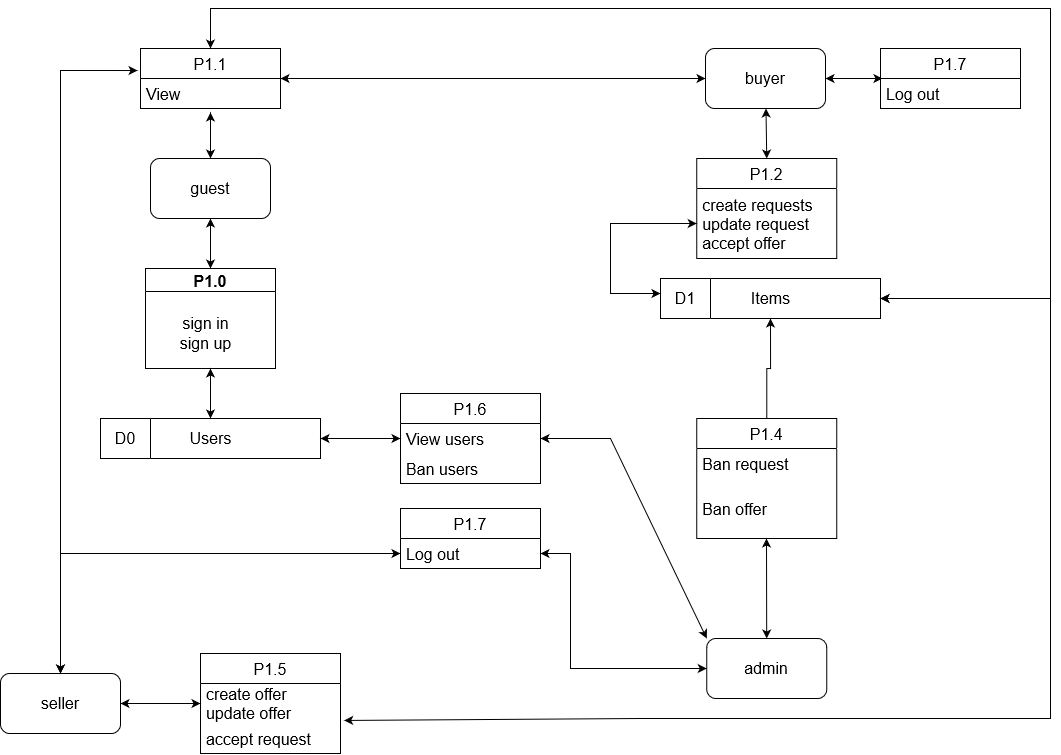


Рис. 2. Діаграма потоків даних P1

## ІНФОЛОГІЧНА МОДЕЛЬ БАЗИ ДАНИХ. ДІАГРАМИ “СУТНІСТЬ – ЗВʼЯЗОК”

Зв’язок таблиць у програмному продукті можна показати за допомогою наступної ERD схеми:

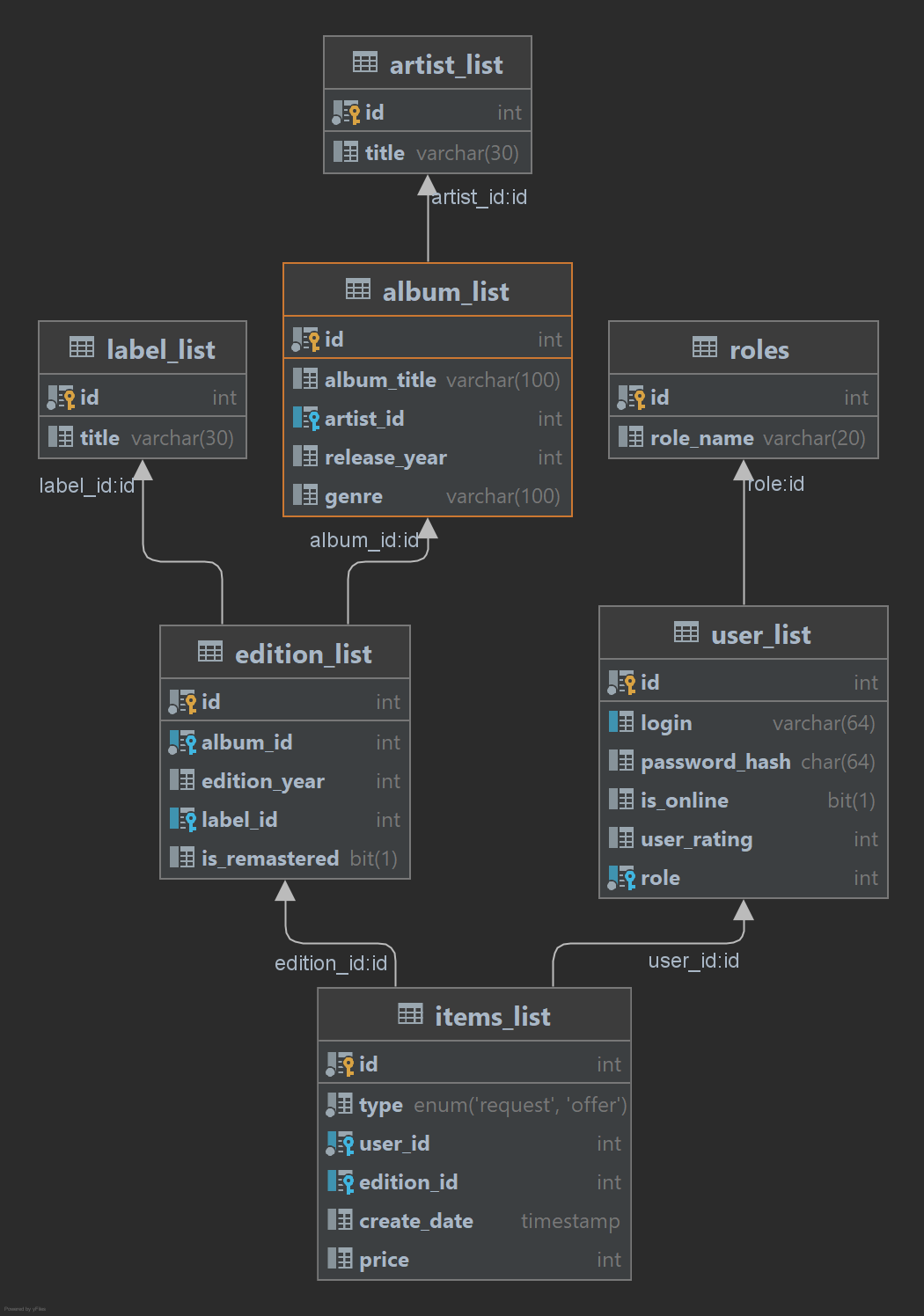


Рис. 3. Діаграма “Сутність - Зв’язок”

Таблиця 1. Звʼязки між сутностями схеми “shop of musical editions”

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Інша  сутність | Атрибут іншої  сутності | Пояснення |
| Таблиця user\_list | | | |
| role | roles | id | Користувач має мати одну роль |
| id | Items\_list | user\_id | Користувач може створити декілька пропозицій |
| Таблиця role | | | |
| id | user\_list | role | Роль може мати  декілька користувачів |
| Таблиця items\_list | | | |
| user\_id | User\_list | id | У пропозиції повинен бути користувач, що створив її |
| edition\_id | Edition\_list | id | Пропозиція завжди являє собою певне музичне видання |
| Таблиця edition\_list | | | |
| id | Items\_list | Edition\_id | Декілька пропозицій можуть стосуватись одного видання |
| Label\_id | Label\_list | id | У музичного видання є видавництво, що є виробником |
| Album\_id | Album\_list | id | Музичне видання це завжди якийсь музикальний альбом |
| Таблиця album\_list | | | |
| id | Edition\_id | album\_id | Один альбом може мати багато різних видань |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Artist\_id | Artist\_list | | id | | У альбома є виконувач | |
| Artist\_list | | | | | | |
| id | | Album\_list | | Artist\_id | | У одного виконувача може бути багато альбомів |

# ДАТАЛОГІЧНА МОДЕЛЬ БАЗИ ДАНИХ

Необхідно розглянути, визначити та описати атрибути кожної з таблиць, що наведені на Рисунку 3.

Таблиця 2. Опис атрибутів

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Сутність | Назва атрибуту | Ключ | Обов’язкове значення | Тип даних | Обмеження |
| Таблиця user\_list | | | | | |
| id | Номер  користувача | PK | Так | Ціле число | Більше за 0 |
| User\_rating | Прізвище  користувача |  | Так | Ціле число | - |
| login | Логін  користувача | SK | Так | Текст | Менше за 64 символа |
| password\_ hash | Захешований пароль  користувача |  | Так | Текст | 64 символа |
| role | Номер ролі | FK | Так | Ціле число | Наявність  серед role.id |
| is\_online | Змінна, що визначає чи активний даний профіль  користувача |  | Так | Бульовий тип | {0,1} |
| Таблиця role | | | | | |
| id | Номер ролі | PK | Так | Ціле число | Більше за 0 |
| Role\_name | Назва ролі |  | Так | Текст | Менше за 20 символів |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Таблиця items\_list | | | | | |
| id | Номер  категорії | PK | Так | Ціле число | Більше за 0 |
| type | Тип пропозиції |  | Так | Текст | {request, offer} |
| User\_id | Номер користувача | FK | Так | Ціле число | Більше 0 |
| Edition\_id | Номер видання | FK | Так | Ціле число | Більше 0 |
| Create\_date | Дата та час створення |  | Так | Дата | Новіша за 01.01.1970 |
| price | Ціна |  | Ні | Ціле число | Більше 0 |
| Таблиця edition\_list | | | | | |
| id | Номер товару | PK | Так | Ціле число | Більше за 0 |
| Album\_id | Номер Альбому | FK | Так | Ціле число | Наявність серед album\_list.id |
| Edition\_year | Рік видання |  | Ні | Ціле число | Більше 0 |
| label\_id | Номер  видавництва | FK | Так | Ціле число | Наявність серед label\_list.id |
| Is\_remastered | Видання є ремастером, чи ні? |  | Ні | Бульовий тип | 0,1 |
| Таблиця album\_list | | | | | |
| id | Номер альбому | PK | Так | Ціле число | Більше за 0 |
| Album\_title | Назва альбому |  | Так | Текст | Довжина менша за 100 |
| artist\_id | Номер  виконавця | FK | Так | Ціле число | Наявність  серед artist\_list.id |
| Release\_year | Рік випуску альбома |  | Ні | Дійсне число | Більше за 0 |
| Genre | Жанр альбому |  | Ні | Текст | Довжина менша за 100 |
| Artist\_list | | | | | |
| id | Номер виконавця | PK | Так | Ціле число | Більше за 0 |
| title | Назва виконавця |  | Ні | Текст | Довжина менша за 30 |
| Label\_list | | | | | |
| id | Номер видавництва | PK | Так | Ціле число | Більше за 0 |
| title | Назва видавництва |  | Ні | Текст | Довжина менша за 30 |

# ФІЗИЧНА МОДЕЛЬ БАЗИ ДАНИХ. ЗАСОБИ ЗАБЕЗПЕЧЕННЯ ЦІЛІСНОСТІ

Виходячи з вищевказаних інфологічних та даталогічних моделей за допомогою системи керування базами даних MySQL було створено та реалізовано систему, що задовільняє всім потребам. Зокрема для забезпечення цілісності бази даних було використано наступні правила:

Для контролю за правильністю звʼязків між реляціями, а саме задля перевірки чи є зовнішній ключ в необхідній таблиці було використано правило FOREIGN KEY. Дане правило було вжито для звʼязків, що зазначені в Таблиці 1, а конкретно атрибутів з Таблиці 2 помічених FK.

Для контролю за коректною ідентифікацією записів у таблиці було використано правило PRIMARY KEY, що одночасно гарантує не нульовість запису та його унікальність. Дане правило було вжито для атрибутів помічених в Таблиці 2 як PK.

Для контролю за обовʼязковою наявністю даних було використано правило NOT NULL та DEFAULT, що гарантує не нулевість атрибуту в 1 випадку та надає значення за замовчуванням у другому. Дане правило було вжиту для атрибутів, що відмічені в Таблиці 2 як обовʼязкові.

Для контролю за унікальністю значень було використано правило UNIQUE, яке не допускає повторень конкретних атрибутів у таблиці. Дане правило було застосоване для атрибутів помічених SK в Таблиці 2.

# ОПИС РЕАЛІЗАЦІЇ ПРОЕКТУ БАЗИ ДАНИХ

Основна частина проекту була реалізована на мові Python(в. 3.10.5). Особливість даної мови полягає у великій гнучкості використання завдяки відсутності обов’язкового використання типів даних, наявності великої кількості opensource-пакетів.

Для реалізації даного проекту було використано наступні пакети:

* mysql-connector-python 8.0.29 Для підключення та керування базами даних[4]
* hashlib – Для шифрування тексту[5]
* typing – для спрощеної анотації типів даних[6]
* tabulate – для виведення таблиць у консоль[7]

Сам проект складається з 4 файлів. Дані файли: database.py, main.py, model.py. Використання подібної структури проекту дозволило збільшити зрозумілість коду, детальніше про кожен з них:

Таблиця 3. файли та їх опис

|  |  |
| --- | --- |
| Назва сервісу | Опис сервісу |
| database.py | Призначений для зв’язку застосунку з базою даних. Складається з функцій для виклику SQL-запитів та функції підключення до бази даних. Підключення здійснюється за допомогою облікових даних (користувач, пароль, хост і порт). |
| main.py | Відповідає за консольний інтерфейс та взаємодії додатку з користувачем. |

|  |  |
| --- | --- |
| model.py | Призначений для зберігання основних структур даних, що використовуються у проєкті. |

Для виконання операцій, що потребують змін одночасно в декількох таблицях було використано конструкцію try except, що збільшує безпечність коду, адже зміни в таблицях будуть зроблені лише у випадку відсутності помилок при всіх етапах виконання заклику.

У додатку реалізована опрацювання помилок і тому навіть при некоректному вводу програма буде викликати введення даних до тих пір, поки користувач не введе їх правильно.

# ВИСНОВКИ

У результаті даної роботи було розроблено проектне рішення, яке відноситься до концептуальної моделі інформаційної системи. Було виконане інфологічне моделювання, яку в подальшому перетворено на даталогічну (реляційну) модель даних та була розроблена ER-схема. По даним моделям та схемам було розроблене проектне рішення, що відносилось до фізичної моделі бази даних, а саме створені таблиці та визначені типи даних для атрибутів та їх властивостей. В подальшому між таблицями було встановлено зв’язки та задано правила цілісності даних. Було розроблено та створено запити до бази даних у середовищі СКБД MySQL. За допомогою мови програмування Python було розроблено програмне забезпечення, що задовольняє концептуальним умовам проекту та дозволяє обмінюватись даними з вище створеною базою даних. Було реалізовано консольний інтерфейс.

Також були отримані практичні навички роботи з СКБД MySQL, а саме створення базових запитів INSERT, UPDATE, DELETE, SELECT. Та формування більш складних запитів з використанням об’єднань, фільтрів, сортувань та вибірок.

Програмний код проекту було завантажено до github репозиторія[8].

# ДЖЕРЕЛА ІНФОРМАЦІЇ

1. Документація СКБД MySQL 5.6. URL: <https://dev.mysql.com/doc/refman/5.6/en/>(дата звернення 19.06.2022 ).
2. Документація Мови програмування Python. URL:  [https://docs.python.org/3/](https://go.dev/doc/) (дата звернення: 19.06.2022)
3. Інформація про поняття консольний додаток. URL:  [http://nikolay.in.ua/navchaemos/visual basic/stvorennya-konsolnikh-dodatkiv](%20http://nikolay.in.ua/navchaemos/visual%20basic/stvorennya-konsolnikh-dodatkiv%20) (дата звернення 19.06.2022)
4. Документація Python MySQL Connector. URL:  [https://dev.mysql.com/doc/connector-python/en/](https://datatracker.ietf.org/doc/html/rfc7519) (дата звернення: 19.06.2022)
5. Документація hashlib. URL:  [https://docs.python.org/3/library/hashlib.html](https://pkg.go.dev/github.com/go-sql-driver/mysql%40v1.6.0) (дата звернення 19.06.2022)
6. Документація typing . URL: https://docs.python.org/3/library/typing.html (дата звернення: 19.06.2022)
7. Документація tabulate. URL:

[https://pypi.org/project/tabulate/](%20https://pypi.org/project/tabulate/%20) (дата звернення 19.06.2022)

1. Програмний код проекту. URL: https://github.com/Vitalii-Rent/Music-editions-store (дата звернення 19.06.2022)

# ДОДАТОК А

CREATE DATABASE music\_shop;

CREATE TABLE music\_shop.user\_list

(

id INT PRIMARY KEY AUTO\_INCREMENT,

login VARCHAR(64) UNIQUE,

password\_hash CHAR(64),

is\_online BIT DEFAULT 0,

user\_rating INT DEFAULT 0,

role INT NOT NULL,

FOREIGN KEY (role) REFERENCES music\_shop.roles (id) ON DELETE CASCADE

);

CREATE TABLE music\_shop.roles

(

id INT PRIMARY KEY AUTO\_INCREMENT,

role\_name VARCHAR(20)

);

/\*CREATE TABLE music\_shop.users\_roles

(

user\_id INT NOT NULL,

role\_id INT NOT NULL,

FOREIGN KEY (user\_id) REFERENCES music\_shop.user\_list (id) ON DELETE CASCADE,

FOREIGN KEY (role\_id) REFERENCES music\_shop.user\_role (id) ON DELETE CASCADE

);\*/

CREATE TABLE music\_shop.artist\_list

(

id INT PRIMARY KEY AUTO\_INCREMENT,

title NVARCHAR(30)

);

CREATE TABLE music\_shop.album\_list

(

id INT PRIMARY KEY AUTO\_INCREMENT,

album\_title NVARCHAR(100),

artist\_id INT,

#genre\_title NVARCHAR(30),

release\_year INT,

genre NVARCHAR(100),

FOREIGN KEY (artist\_id) REFERENCES music\_shop.artist\_list (id) ON DELETE CASCADE);

CREATE TABLE music\_shop.label\_list

(

id INT PRIMARY KEY AUTO\_INCREMENT,

title NVARCHAR(30)

);

CREATE TABLE music\_shop.edition\_list

(

id INT PRIMARY KEY AUTO\_INCREMENT,

album\_id INT NOT NULL,

edition\_year INT,

label\_id INT,

is\_remastered BIT,

FOREIGN KEY (album\_id) REFERENCES music\_shop.album\_list (id) ON DELETE CASCADE,

FOREIGN KEY (label\_id) REFERENCES music\_shop.label\_list (id) ON DELETE CASCADE

);

CREATE TABLE music\_shop.items\_list

(

id INT PRIMARY KEY AUTO\_INCREMENT,

type ENUM ('request', 'offer') NOT NULL,

user\_id INT NOT NULL,

#user\_rating int NOT NULL,

edition\_id INT,

create\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

price INT,

FOREIGN KEY (user\_id) REFERENCES music\_shop.user\_list (id) ON DELETE CASCADE,

#FOREIGN KEY (user\_rating) REFERENCES music\_shop.user\_list (user\_rating) ON DELETE CASCADE,

FOREIGN KEY (edition\_id) REFERENCES music\_shop.edition\_list (id) ON DELETE CASCADE

);