НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет Программной инженерии и компьютерной техники

Информатика

Лабораторная работа № 4 "Введение в базы данных."

Выполнил студентка

Слудная Виктория Евгеньевна

Группа № Р3122

Преподаватель: Болдырева Елена Александровна

г. Санкт-Петербург

2024

Оглавление

[Задание 3](#_Toc184044479)

[1. Схема базы данных. 3](#_Toc184044480)

[2. Подключение к базам данных. 3](#_Toc184044481)

[3. Создание таблиц 4](#_Toc184044482)

[4. Вставка записей 7](#_Toc184044483)

[5. Извлечение данных записей. 9](#_Toc184044484)

[6. Обновление записей таблицы. 14](#_Toc184044485)

[7. Удаление записей таблицы. 14](#_Toc184044486)

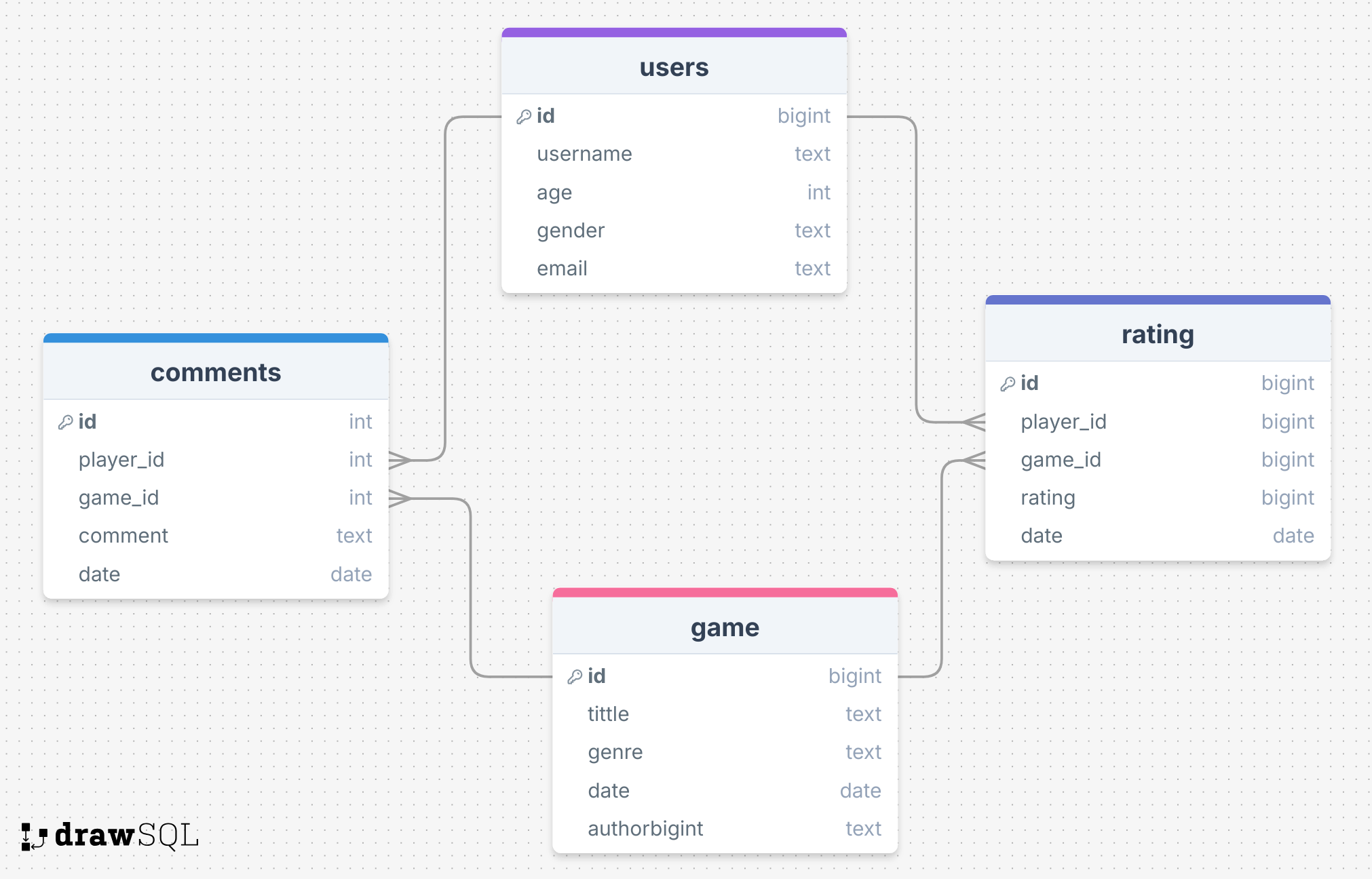
[Вывод. 15](#_Toc184044487)

# Задание

Необходимо создать и подключиться к базам данных SQLite и MySQL с помощью библиотек Python. Добавить туда записи. Выполнить различные запросы.

# 1. Схема базы данных.

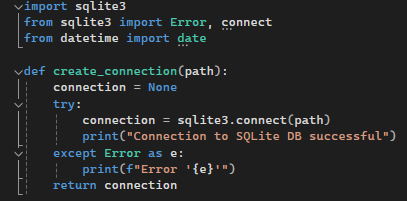
В качестве собственной базы данных я буду использовать следующую.



*Рис. 1. Собственная база данных.*

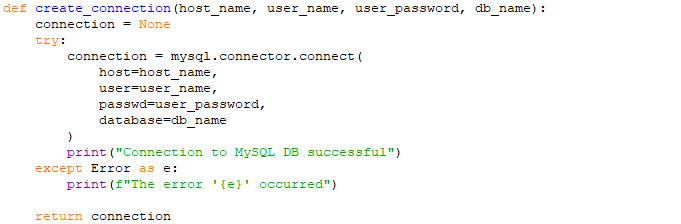
# 2. Подключение к базам данных.

Для подключения к базам данных SQLite используем библиотеку в Python sqlite3 и следующую функцию:



*Рис. 2. Функция для подключения к базе данных. SQLite*

Для подключения в MySQL необходимо было установить библиотеку mysql и локальный сервер. Используем похожую функцию:



*Рис. 3. Функция для подключения к базе данных. MySQL*

При успешном подключении получим сообщение:

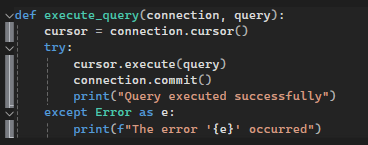


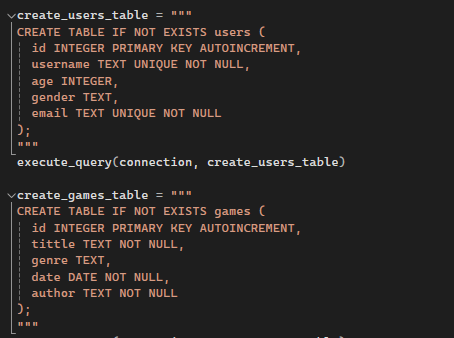
*Рис. 4. Вывод при успешном создании.*

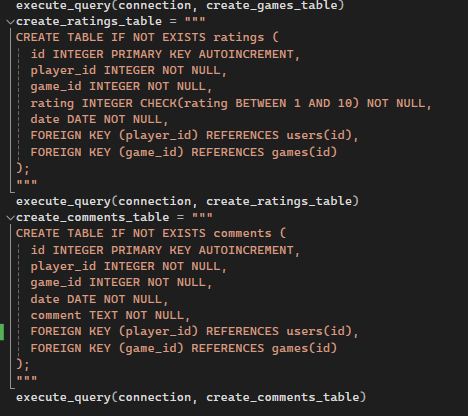
# 3. Создание таблиц

Для создания таблиц в SQLite и MySQL используем следующие запросы:

Для SQLite:

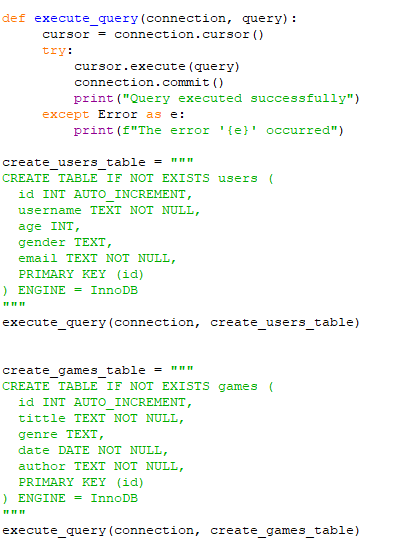


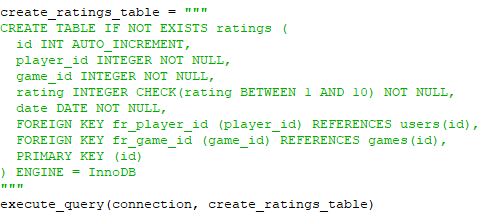


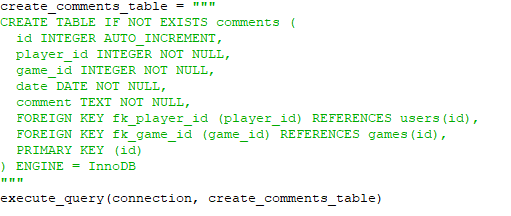


*Рис. 5-7. Создание таблиц. SQLite*

Для MySQL:





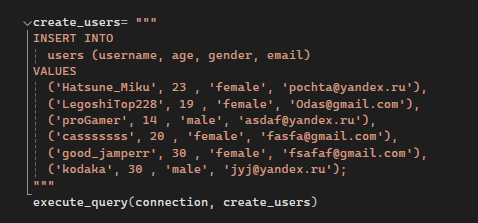


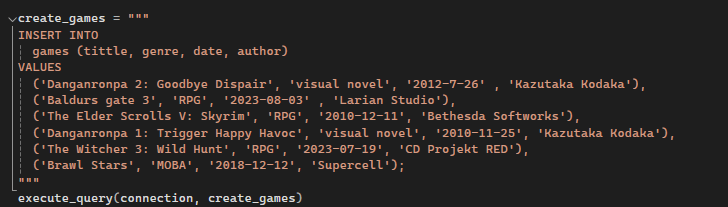
*Рис. 8-10. Создание таблиц. MySQL*

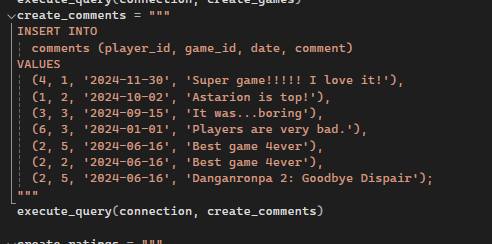
# 4. Вставка записей

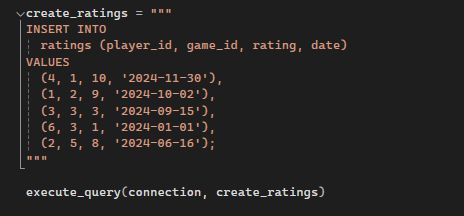
Для добавления записей в таблицу используем запросы с INSERT INTO:

Для SQLite:



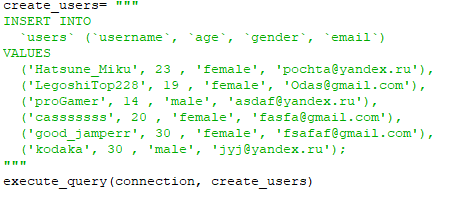


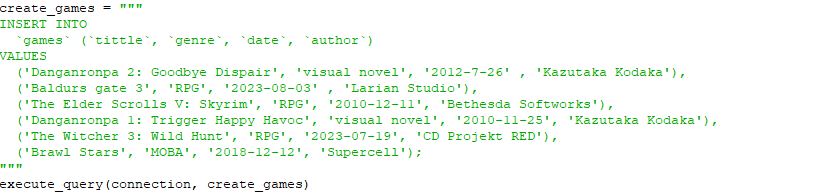
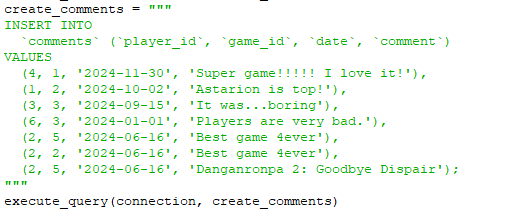


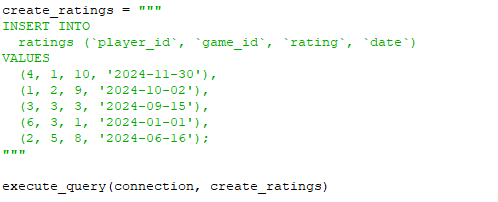


*Рис. 11-14. Добавление данных. SQLite*

Для MySQL:



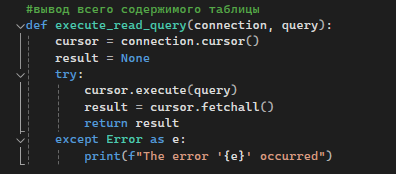
 



*Рис. 14-18. Создание таблиц. MySQL*

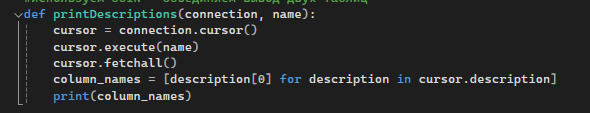
# 5. Извлечение данных записей.

Для извлечения данных из таблиц используются SELECT-запросы с использованием различных дополнительных операций. Синтаксис для запросов в SQLite и MySQL совпадает.



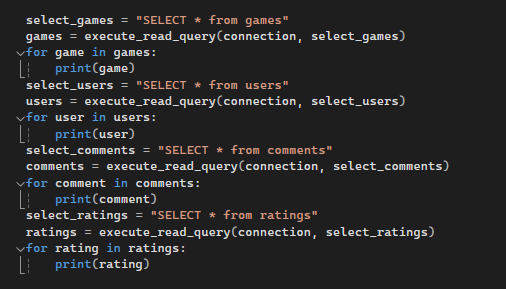
*Рис. 19. Функция для вывода запроса.*

Функция для вывода наименований столбцов:



*Рис. 20. Функция для вывода имен столбцов.*

Простые SELECT-запросы:



*Рис. 21. Пример создания SELECT-запросов.*

Пример вывода:

(1, 'Danganronpa 2: Goodbye Dispair', 'visual novel', '2012-7-26', 'Kazutaka Kodaka')

(2, 'Baldurs gate 3', 'RPG', '2023-08-03', 'Larian Studio')

(3, 'The Elder Scrolls V: Skyrim', 'RPG', '2010-12-11', 'Bethesda Softworks')

(4, 'Danganronpa 1: Trigger Happy Havoc', 'visual novel', '2010-11-25', 'Kazutaka Kodaka')

(5, 'The Witcher 3: Wild Hunt', 'RPG', '2023-07-19', 'CD Projekt RED')

(6, 'Brawl Stars', 'MOBA', '2018-12-12', 'Supercell')

(7, 'Sally Face', 'horror', '2016-01-21', 'Portable Moose')

(1, 'Hatsune\_Miku', 23, 'female', 'pochta@yandex.ru')

(2, 'LegoshiTop228', 19, 'female', 'Odas@gmail.com')

(3, 'proGamer', 14, 'male', 'asdaf@yandex.ru')

(4, 'casssssss', 20, 'female', 'fasfa@gmail.com')

(5, 'good\_jamperr', 30, 'female', 'fsafaf@gmail.com')

(6, 'kodaka', 30, 'male', 'jyj@yandex.ru')

(7, 'nicejumper', 14, 'female', 'error@yandex.ru')

(1, 4, 1, '2024-11-30', 'Super game!!!!! I love it!')

(2, 1, 2, '2024-10-02', 'Astarion is top!')

(3, 3, 3, '2024-09-15', 'It was...boring')

(4, 6, 3, '2024-01-01', 'Players are very bad.')

(5, 2, 5, '2024-06-16', 'Best game 4ever')

(6, 2, 2, '2024-06-16', 'Best game 4ever')

(7, 2, 5, '2024-06-16', 'Danganronpa 2: Goodbye Dispair')

(8, 7, 7, '2024-06-22', 'Super!!!')

(1, 4, 1, 10, '2024-11-30')

(2, 1, 2, 9, '2024-10-02')

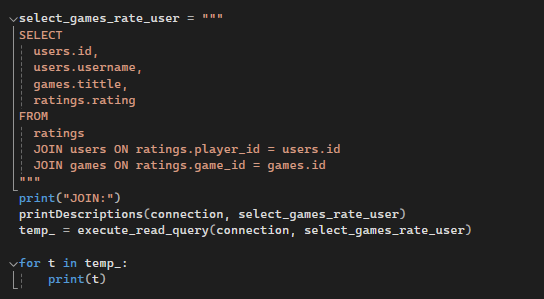
(3, 3, 3, 3, '2024-09-15')

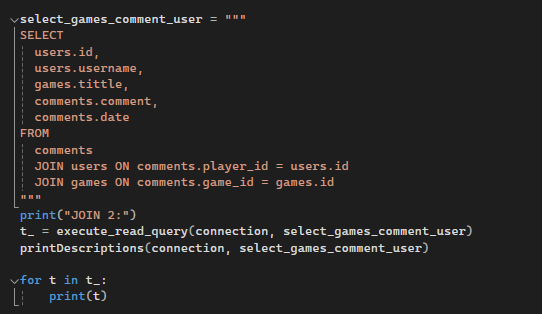
(4, 6, 3, 1, '2024-01-01')

(5, 2, 5, 8, '2024-06-16')

(6, 4, 7, 9, '2024-11-16')

INNER JOIN:





*Рис. 22. Пример создания INNER JOIN -запросов.*

JOIN:

['id', 'username', 'tittle', 'rating']

(4, 'casssssss', 'Danganronpa 2: Goodbye Dispair', 10)

(1, 'Hatsune\_Miku', 'Baldurs gate 3', 9)

(3, 'proGamer', 'The Elder Scrolls V: Skyrim', 3)

(6, 'kodaka', 'The Elder Scrolls V: Skyrim', 1)

(2, 'LegoshiTop228', 'The Witcher 3: Wild Hunt', 8)

(4, 'casssssss', 'Sally Face', 9)

JOIN 2:

['id', 'username', 'tittle', 'comment', 'date']

(4, 'casssssss', 'Danganronpa 2: Goodbye Dispair', 'Super game!!!!! I love it!', '2024-11-30')

(1, 'Hatsune\_Miku', 'Baldurs gate 3', 'Astarion is top!', '2024-10-02')

(3, 'proGamer', 'The Elder Scrolls V: Skyrim', 'It was...boring', '2024-09-15')

(6, 'kodaka', 'The Elder Scrolls V: Skyrim', 'Players are very bad.', '2024-01-01')

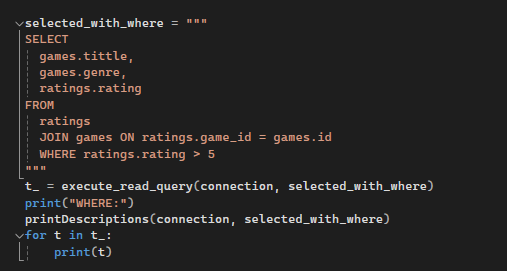
(2, 'LegoshiTop228', 'The Witcher 3: Wild Hunt', 'Best game 4ever', '2024-06-16')

(2, 'LegoshiTop228', 'Baldurs gate 3', 'Best game 4ever', '2024-06-16')

(2, 'LegoshiTop228', 'The Witcher 3: Wild Hunt', 'Danganronpa 2: Goodbye Dispair', '2024-06-16')

(7, 'nicejumper', 'Sally Face', 'Super!!!', '2024-06-22')

WHERE:



*Рис. 23. Пример создания WHERE-запросов.*

WHERE:

['tittle', 'genre', 'rating']

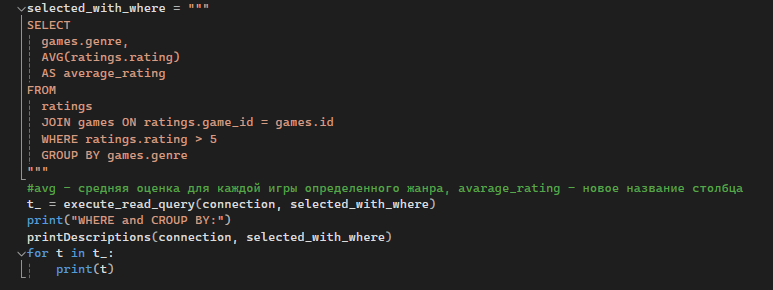
('Danganronpa 2: Goodbye Dispair', 'visual novel', 10)

('Baldurs gate 3', 'RPG', 9)

('The Witcher 3: Wild Hunt', 'RPG', 8)

('Sally Face', 'horror', 9)

GROUP BY:



*Рис. 24. Пример создания CROUP BY-запросов.*

WHERE and CROUP BY:

['genre', 'average\_rating']

('RPG', 8.5)

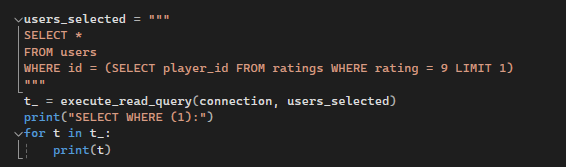
('horror', 9.0)

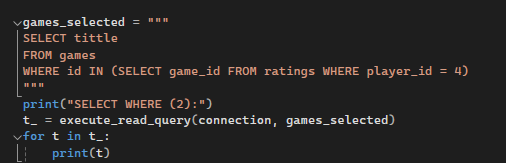
('visual novel', 10.0)

SELECT WHERE (1):

(1, 'Hatsune\_Miku', 23, 'female', 'pochta@yandex.ru')

Вложенный SELECT:





*Рис. 25. Пример создания вложенных SELECT-запросов.*

SELECT WHERE (1):

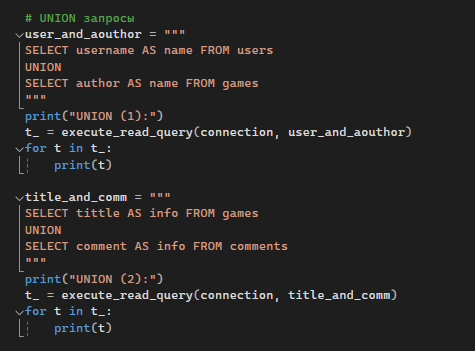
(1, 'Hatsune\_Miku', 23, 'female', 'pochta@yandex.ru')

SELECT WHERE (2):

('Danganronpa 2: Goodbye Dispair',)

('Sally Face',)

UNION:



*Рис. 26. Пример создания UNION-запросов.*

UNION (1):

('Bethesda Softworks',)

('CD Projekt RED',)

('Hatsune\_Miku',)

('Kazutaka Kodaka',)

('Larian Studio',)

('LegoshiTop228',)

('Portable Moose',)

('Supercell',)

('casssssss',)

('good\_jamperr',)

('kodaka',)

('nicejumper',)

('proGamer',)

UNION (2):

('Astarion is top!',)

('Baldurs gate 3',)

('Best game 4ever',)

('Brawl Stars',)

('Danganronpa 1: Trigger Happy Havoc',)

('Danganronpa 2: Goodbye Dispair',)

('It was...boring',)

('Players are very bad.',)

('Sally Face',)

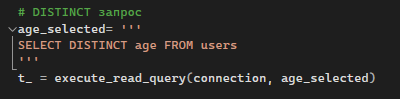
('Super game!!!!! I love it!',)

('Super!!!',)

('The Elder Scrolls V: Skyrim',)

('The Witcher 3: Wild Hunt',)

DISTINCT:



*Рис. 27. Пример создания DISTING-запросов.*

(23,)

(19,)

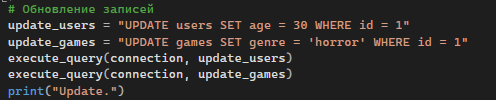
(14,)

(20,)

(30,)

# 6. Обновление записей таблицы.

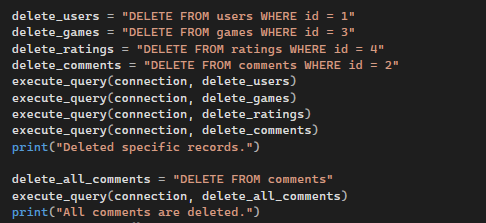
Обновление записей таблицы с помощью команды UPDATE происходит одинаково для SQLite и MySQL:



*Рис. 28. Обновление.*

# 7. Удаление записей таблицы.

Удаление так же происходит одинаково:



*Рис. 29. Удаление.*

# Вывод.

В ходе данной лабораторной работы я научилась подключаться к различным СУБД с помощью библиотек Python для работы с SQL базами данных, управлять базами данных SQLite и MySQL, выполнять запросы к базе данных внутри приложения Python.