НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Информатика Лабораторная работа №4

«Введение в Базы данных»

Выполнил студент:

Дмитришен Кирилл Русланович

Группа №P3124

Преподаватель:

Болдырева Елена Александровна

г. Санкт-Петербург

2023

Оглавление

[Диаграмма зависимостей 3](#_Toc152576720)

[Задания: 4](#_Toc152576721)

[SQLite 5](#_Toc152576722)

[ВЫВОД 14](#_Toc152576724)

[Выводы 17](#_Toc152576727)

# Порядок выполнения:

1. Изучить материалы для подготовки.

2. Выполнить задания-примеры для того, чтобы разобраться с принципом взаимодействия с

базами данных с помощью языка программирования Python.

3. Выполнить задания для самостоятельной работы: SQLLite (обязательное задание – 60% от

максимальной оценки) и MySQL (необязательное задание – 40% от максимальной оценки).

4. Составить отчет.

# SQLite

import sqlite3  
from sqlite3 import Error  
  
def create\_connection(path):  
 connection = None  
 try:  
 connection = sqlite3.connect(path)  
 print("Connection to SQLite DB successful")  
 except Error as e:  
 print(f"The error '{e}' occurred")  
 return connection  
  
connection = create\_connection("C://Users/dmitr/downloads/sm2\_app")  
  
def execute\_query(connection, query):  
 cursor = connection.cursor()  
 try:  
 cursor.execute(query)  
 connection.commit()  
 print("Query executed successfully")  
 except Error as e:  
 print(f"The error '{e}' occurred")  
  
  
create\_authors\_table = """  
CREATE TABLE IF NOT EXISTS authors (  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 name TEXT NOT NULL,  
 dol TEXT,  
 gender TEXT,  
 nationality TEXT  
);  
"""  
  
execute\_query(connection, create\_authors\_table)  
  
create\_books\_table = """  
CREATE TABLE IF NOT EXISTS books(  
 id INTEGER PRIMARY KEY AUTOINCREMENT,   
 title TEXT NOT NULL,   
 description TEXT NOT NULL,   
 author\_id INTEGER NOT NULL,   
 FOREIGN KEY (author\_id) REFERENCES authors (id)  
);  
"""  
  
execute\_query(connection, create\_books\_table)  
  
create\_readers\_table = """  
CREATE TABLE IF NOT EXISTS readers (  
 id INTEGER PRIMARY KEY AUTOINCREMENT,   
 comment\_id INTEGER NOT NULL,   
 book\_id integer NOT NULL,   
 FOREIGN KEY (comment\_id) REFERENCES comment (id), FOREIGN KEY (book\_id) REFERENCES books (id)  
);  
"""  
  
execute\_query(connection, create\_readers\_table)  
  
create\_comments\_table = """  
CREATE TABLE IF NOT EXISTS comments (  
 id INTEGER PRIMARY KEY AUTOINCREMENT,   
 text TEXT NOT NULL,   
 reader\_id INTEGER NOT NULL,   
 book\_id INTEGER NOT NULL,   
 FOREIGN KEY (reader\_id) REFERENCES readers (id) FOREIGN KEY (book\_id) REFERENCES books (id)  
);  
"""  
  
execute\_query(connection, create\_comments\_table)  
  
print("\n-----------------------------------------------------------------------\n")  
#############################################################3  
  
create\_authors = """  
INSERT INTO  
 authors (name, dol, gender, nationality)  
VALUES  
 ('Leo Tolstoy', '1828-1910', 'male', 'Russia'),  
 ('William Shakespeare', '1564-1616', 'male', 'England'),  
 ('J. K. Rowling', '1965-#', 'female', 'England'),  
 ('Agatha Christie', '1890-1976', 'female', 'England'),  
 ('R. L. Stine', '1943-#', 'male', 'U.S.');  
"""  
  
execute\_query(connection, create\_authors)  
  
create\_books = """  
INSERT INTO  
 books (title, description, author\_id)  
VALUES  
 ("War and Peace", "There's always a way back to life through love", 1),  
 ("Anna Karenina", "Impulsiveness leads to tragedies", 1),  
 ("Hamlet", "Indecision can drive you mad", 2),  
 ("Harry Potter and the Philosopher’s Stone", "Be good and don't be bad", 3),  
 ("Murder on the Orient Express", "Don't trust the gardener or something", 4),  
 ("Fear Street", "Boo", 5);  
"""  
  
execute\_query(connection, create\_books)  
  
create\_readers = """  
INSERT INTO  
 readers (comment\_id, book\_id)  
VALUES  
 (1, 1),  
 (2, 2),  
 (3, 3),  
 (4, 4),  
 (5, 5),  
 (6, 6);  
"""  
  
execute\_query(connection, create\_readers)  
  
create\_comments = """  
INSERT INTO  
 comments (text, reader\_id, book\_id)  
VALUES  
 ('Heartbreaking', 1, 1),  
 ('Meh', 2, 2),  
 ('Hard to read', 3, 3),  
 ('Fun for a 9-year old kid', 4, 4),  
 ('Did not read lol', 5, 5),  
 ('Scary or not, havent decided yet', 5, 6);  
"""  
  
execute\_query(connection, create\_comments)  
  
print("\n-----------------------------------------------------------------------\n")  
#####################################################################  
  
def execute\_read\_query(connection, query):  
 cursor = connection.cursor()  
 result = None  
 try:  
 cursor.execute(query)  
 result = cursor.fetchall()  
 return result  
 except Error as e:  
 print(f"The error '{e}' occurred")  
  
select\_authors = "SELECT \* from authors"  
authors = execute\_read\_query(connection, select\_authors)  
for author in authors:  
 print(author)  
  
print("\n-----------------------------------------------------------------------\n")  
  
select\_authors\_books = """  
SELECT  
 authors.id,  
 authors.name,  
 books.title  
FROM  
 books  
 INNER JOIN authors ON authors.id = books.author\_id  
"""  
authors\_books = execute\_read\_query(connection, select\_authors\_books)  
for authors\_book in authors\_books:  
 print(authors\_book)  
  
print("\n-----------------------------------------------------------------------\n")  
  
select\_books\_comments = """  
SELECT  
 books.id,  
 books.title,  
 comments.text  
FROM  
 comments  
 INNER JOIN books ON books.id = comments.book\_id  
"""  
books\_comments = execute\_read\_query(connection, select\_books\_comments)  
for books\_comments in books\_comments:  
 print(books\_comments)  
  
print('\n')  
  
select\_reader\_comments = """  
SELECT  
 text as comment,  
 COUNT(comments.id) as ids  
FROM  
 readers,  
 comments  
WHERE  
 readers.id = comments.reader\_id  
GROUP BY  
 readers.comment\_id  
"""  
reader\_comments = execute\_read\_query(connection, select\_reader\_comments)  
for reader\_comment in reader\_comments:  
 print(reader\_comment)  
  
print('\n')  
  
update\_book\_description = """  
UPDATE  
 books  
SET  
 description = "Changed my mind, its nice"  
WHERE  
 id = 2  
"""  
execute\_query(connection, update\_book\_description)  
  
select\_book\_description = "SELECT description FROM books WHERE id = 2"  
book\_description = execute\_read\_query(connection, select\_book\_description)  
for description in book\_description:  
 print(description)  
  
  
print('\n')  
  
delete\_comment = "DELETE FROM comments WHERE id > 2"  
execute\_query(connection, delete\_comment)  
  
select\_comments = "SELECT \* from comments"  
comments = execute\_read\_query(connection, select\_comments)  
for comment in comments:  
 print(comment)

# MySQL:

import mysql.connector  
from mysql.connector import Error  
  
def create\_connection(host\_name, user\_name, user\_password):  
 connection = None  
 try:  
 connection = mysql.connector.connect(  
 host=host\_name,  
 user=user\_name,  
 passwd=user\_password  
 )  
 print("Connection to MySQL DB successful")  
 except Error as e:  
 print(f"The error '{e}' occurred")  
 return connection  
connection = create\_connection("localhost", "user", "1234")  
  
  
def create\_database(connection, query):  
 cursor = connection.cursor()  
 try:  
 cursor.execute(query)  
 print("Database created successfully")  
 except Error as e:  
 print(f"The error '{e}' occurred")  
  
  
  
# create\_database\_query = "CREATE DATABASE sm\_app"  
# create\_database(connection, create\_database\_query)  
  
  
def create\_connection(host\_name, user\_name, user\_password, db\_name):  
 connection = None  
 try:  
 connection = mysql.connector.connect(  
 host=host\_name,  
 user=user\_name,  
 passwd=user\_password,  
 database=db\_name  
 )  
 print("Connection to MySQL DB successful")  
 except Error as e:  
 print(f"The error '{e}' occurred")  
 return connection  
  
connection = create\_connection("localhost", "user", "1234", "sm\_app")  
  
  
def execute\_query(connection, query):  
 cursor = connection.cursor()  
 try:  
 cursor.execute(query)  
 connection.commit()  
 print("Query executed successfully")  
 except Error as e:  
 print(f"The error '{e}' occurred")  
  
  
  
create\_authors\_table = """  
CREATE TABLE IF NOT EXISTS authors (  
 id INT AUTO\_INCREMENT,  
 name TEXT NOT NULL,  
 dol TEXT,  
 gender TEXT,  
 nationality TEXT,  
 PRIMARY KEY (id)  
) ENGINE = InnoDB  
"""  
  
execute\_query(connection, create\_authors\_table)  
  
create\_books\_table = """  
CREATE TABLE IF NOT EXISTS books(  
 id INT AUTO\_INCREMENT,   
 title TEXT NOT NULL,   
 description TEXT NOT NULL,   
 author\_id INT NOT NULL,   
 FOREIGN KEY (author\_id) REFERENCES authors (id),  
 PRIMARY KEY (id)  
);  
"""  
  
execute\_query(connection, create\_books\_table)  
  
  
  
  
create\_readers\_table = """  
CREATE TABLE IF NOT EXISTS readers (  
 id INT AUTO\_INCREMENT,   
 comment\_id INT NOT NULL,   
 book\_id INT NOT NULL,   
 FOREIGN KEY (book\_id) REFERENCES books (id),  
 PRIMARY KEY(id)  
);  
"""  
  
execute\_query(connection, create\_readers\_table)  
  
create\_comments\_table = """  
CREATE TABLE IF NOT EXISTS comments (  
 id INT AUTO\_INCREMENT,   
 text TEXT NOT NULL,   
 reader\_id INT NOT NULL,   
 book\_id INT NOT NULL,   
 FOREIGN KEY (reader\_id) REFERENCES readers (id),  
 FOREIGN KEY (book\_id) REFERENCES books (id),  
 PRIMARY KEY (id)  
);  
"""  
  
execute\_query(connection, create\_comments\_table)  
  
alter\_table\_query = """  
ALTER TABLE readers  
ADD CONSTRAINT `comment\_id` FOREIGN KEY (`comment\_id`) REFERENCES `comments` (`comment\_id`) ON DELETE CASCADE ON UPDATE CASCADE;  
"""  
execute\_query(connection, create\_readers\_table)  
  
print("\n1-----------------------------------------------------------------------\n")  
  
  
create\_authors = """  
INSERT INTO  
 authors (name, dol, gender, nationality)  
VALUES  
 ('Leo Tolstoy', '1828-1910', 'male', 'Russia'),  
 ('William Shakespeare', '1564-1616', 'male', 'England'),  
 ('J. K. Rowling', '1965-#', 'female', 'England'),  
 ('Agatha Christie', '1890-1976', 'female', 'England'),  
 ('R. L. Stine', '1943-#', 'male', 'U.S.');  
"""  
  
execute\_query(connection, create\_authors)  
  
create\_books = """  
INSERT INTO  
 books (title, description, author\_id)  
VALUES  
 ("War and Peace", "There's always a way back to life through love", 1),  
 ("Anna Karenina", "Impulsiveness leads to tragedies", 1),  
 ("Hamlet", "Indecision can drive you mad", 2),  
 ("Harry Potter and the Philosopher’s Stone", "Be good and don't be bad", 3),  
 ("Murder on the Orient Express", "Don't trust the gardener or something", 4),  
 ("Fear Street", "Boo", 5);  
"""  
  
execute\_query(connection, create\_books)  
  
create\_readers = """  
INSERT INTO  
 readers (comment\_id, book\_id)  
VALUES  
 (1, 1),  
 (2, 2),  
 (3, 3),  
 (4, 4),  
 (5, 5),  
 (6, 6);  
"""  
  
execute\_query(connection, create\_readers)  
  
create\_comments = """  
INSERT INTO  
 comments (text, reader\_id, book\_id)  
VALUES  
 ('Heartbreaking', 1, 1),  
 ('Meh', 2, 2),  
 ('Hard to read', 3, 3),  
 ('Fun for a 9-year old kid', 4, 4),  
 ('Did not read lol', 5, 5),  
 ('Scary or not, havent decided yet', 5, 6);  
"""  
  
execute\_query(connection, create\_comments)  
  
delete\_comment = "DELETE FROM authors WHERE id > 5"  
execute\_query(connection, delete\_comment)  
  
delete\_comment = "DELETE FROM books WHERE id > 6"  
execute\_query(connection, delete\_comment)  
  
delete\_comment = "DELETE FROM readers WHERE id > 6"  
execute\_query(connection, delete\_comment)  
  
delete\_comment = "DELETE FROM comments WHERE id > 6"  
execute\_query(connection, delete\_comment)  
  
print("\n2-----------------------------------------------------------------------\n")  
  
def execute\_read\_query(connection, query):  
 cursor = connection.cursor()  
 result = None  
 try:  
 cursor.execute(query)  
 result = cursor.fetchall()  
 return result  
 except Error as e:  
 print(f"The error '{e}' occurred")  
  
  
select\_authors = "SELECT \* from authors"  
authors = execute\_read\_query(connection, select\_authors)  
for author in authors:  
 print(author)  
  
  
print("\n3-----------------------------------------------------------------------\n")  
  
select\_authors\_books = """  
SELECT  
 authors.id,  
 authors.name,  
 books.title  
FROM  
 books  
 INNER JOIN authors ON authors.id = books.author\_id  
"""  
authors\_books = execute\_read\_query(connection, select\_authors\_books)  
for authors\_book in authors\_books:  
 print(authors\_book)  
  
print("\n4-----------------------------------------------------------------------\n")  
  
select\_books\_comments = """  
SELECT  
 books.id,  
 books.title,  
 comments.text  
FROM  
 comments  
 INNER JOIN books ON books.id = comments.book\_id  
"""  
books\_comments = execute\_read\_query(connection, select\_books\_comments)  
for books\_comments in books\_comments:  
 print(books\_comments)  
  
print("\n5-----------------------------------------------------------------------\n")  
  
select\_reader\_comments = """  
SELECT  
 readers.comment\_id,  
 comments.text as comment,  
 COUNT(comments.id) as ids  
FROM  
 readers  
 JOIN comments ON readers.id = comments.reader\_id  
GROUP BY  
 readers.comment\_id, comments.text;  
  
"""  
reader\_comments = execute\_read\_query(connection, select\_reader\_comments)  
  
for reader\_comment in reader\_comments:  
 print(reader\_comment)  
  
print('\n')  
  
update\_book\_description = """  
UPDATE  
 books  
SET  
 description = "Changed my mind, its nice"  
WHERE  
 id = 2  
"""  
execute\_query(connection, update\_book\_description)  
  
select\_book\_description = "SELECT description FROM books WHERE id = 2"  
book\_description = execute\_read\_query(connection, select\_book\_description)  
for description in book\_description:  
 print(description)  
  
  
print("\n6-----------------------------------------------------------------------\n")  
  
  
delete\_comment = "DELETE FROM comments WHERE id > 2"  
execute\_query(connection, delete\_comment)  
  
select\_comments = "SELECT \* from comments"  
comments = execute\_read\_query(connection, select\_comments)  
for comment in comments:  
 print(comment)

# ВЫВОД (очевидно, одинаковый в обоих случаях)

Connection to SQLite DB successful

Query executed successfully

Query executed successfully

Query executed successfully

Query executed successfully

-----------------------------------------------------------------------

Query executed successfully

Query executed successfully

Query executed successfully

Query executed successfully

-----------------------------------------------------------------------

(1, 'Leo Tolstoy', '1828-1910', 'male', 'Russia')

(2, 'William Shakespeare', '1564-1616', 'male', 'England')

(3, 'J. K. Rowling', '1965-#', 'female', 'England')

(4, 'Agatha Christie', '1890-1976', 'female', 'England')

(5, 'R. L. Stine', '1943-#', 'male', 'U.S.')

-----------------------------------------------------------------------

(1, 'Leo Tolstoy', 'War and Peace')

(1, 'Leo Tolstoy', 'Anna Karenina')

(2, 'William Shakespeare', 'Hamlet')

(3, 'J. K. Rowling', 'Harry Potter and the Philosopher’s Stone')

(4, 'Agatha Christie', 'Murder on the Orient Express')

(5, 'R. L. Stine', 'Fear Street')

-----------------------------------------------------------------------

(1, 'War and Peace', 'Heartbreaking')

(2, 'Anna Karenina', 'Meh')

(3, 'Hamlet', 'Hard to read')

(4, 'Harry Potter and the Philosopher’s Stone', 'Fun for a 9-year old kid')

(5, 'Murder on the Orient Express', 'Did not read lol')

(6, 'Fear Street', 'Scary or not, havent decided yet')

('Heartbreaking', 1)

('Meh', 1)

('Hard to read', 1)

('Fun for a 9-year old kid', 1)

('Did not read lol', 2)

Query executed successfully

('Changed my mind, its nice',)

Query executed successfully

(1, 'Heartbreaking', 1, 1)

(2, 'Meh', 2, 2)

Выводы:

Я узнал как устроены базы данных в SQLite и MySQL и научился работать с ними используя язык Python и программу PyCharm.