

# МЕТОДЫ ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ БАЗЫ ДАННЫХ POSTGRESQL

## КУРСОВАЯ РАБОТА

студентки 3 курса 351 группы

направления 09.03.04 — Программная инженерия

факультета КНиИТ

Быковой Марии Дмитриевны

Научный руководитель

зав.каф.техн.пр.,

к. ф.-м. н., доцент И. А. Батраева

**Цель курсовой работы:** разработка базы данных в PostgreSQL, настройка ее защищенности и реализация оконного приложения для взаимодействия с базой с помощью Python .

**Задачи:**

- изучение основ безопасности баз данных;
- анализ способов организации безопасности базы данных;
- создание оконного приложения на языке Python для взаимодействия с базой;
- рассмотрение различных видов SQL-инъекций и способов защиты от них.

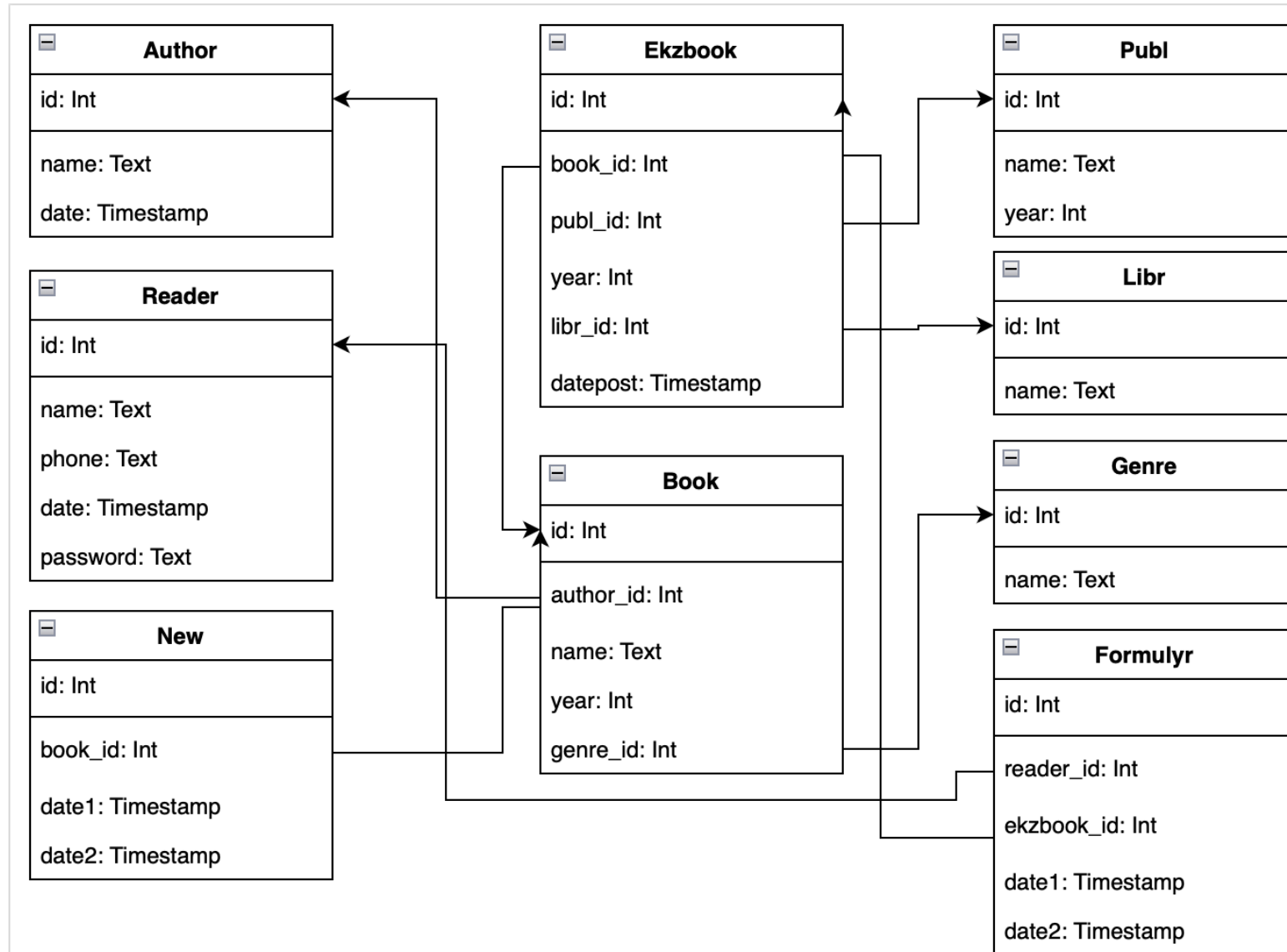
**Инструменты:** PostgreSQL, Tkinter Python.

# Понятие защищенной базы данных

Безопасность данных — это состояние защищенности, при котором обеспечиваются конфиденциальность, доступность и целостность данных.

Конфиденциальность отвечает за обеспечение доступа к данным, только санкционированным пользователями. Целостность исключает несанкционированное изменение структуры и содержания данных. Доступность позволяет обеспечить доступ к данным, санкционированным пользователями, по их первому требованию.

# Схема базы данных «Библиотечная система»



# Пример SQL-запроса создания сущности

```
CREATE TABLE author(  
  id INT PRIMARY KEY,  
  name text,  
  date timestamp)
```

```
CREATE TABLE reader(  
  id INT PRIMARY KEY,  
  name text,  
  phone text,  
  "date" timestamp,  
  password text)
```

```
CREATE TABLE book(  
  id INT PRIMARY KEY,  
  author_id integer,  
  name text,  
  year integer,  
  genre_id integer)
```

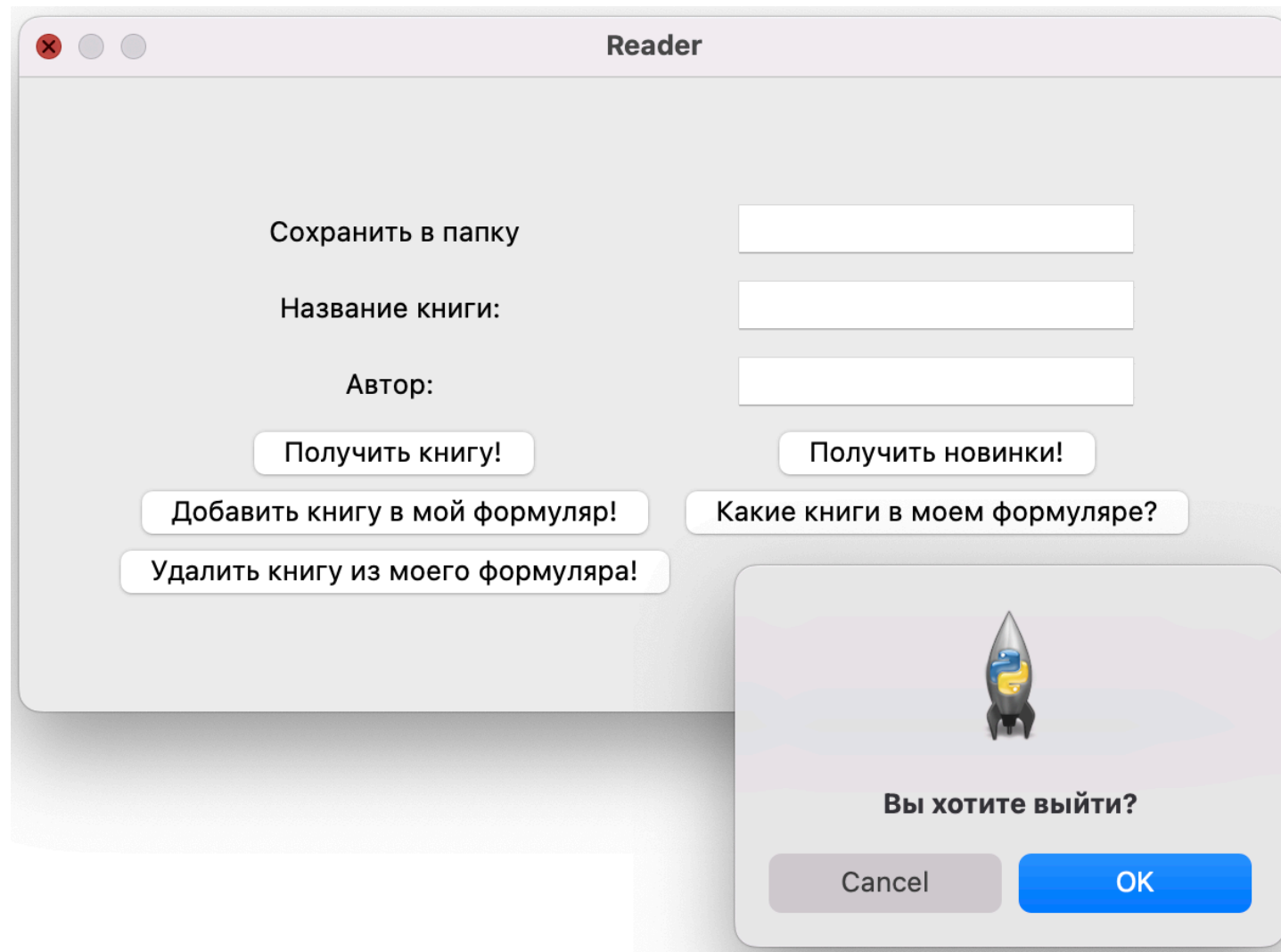
# Ролевая модель доступа

*Управление доступом на основе ролей* — развитие политики избирательного управления доступом, при этом права доступа (привилегии) субъектов системы на объекты группируются с учётом специфики их применения, образуя роли.

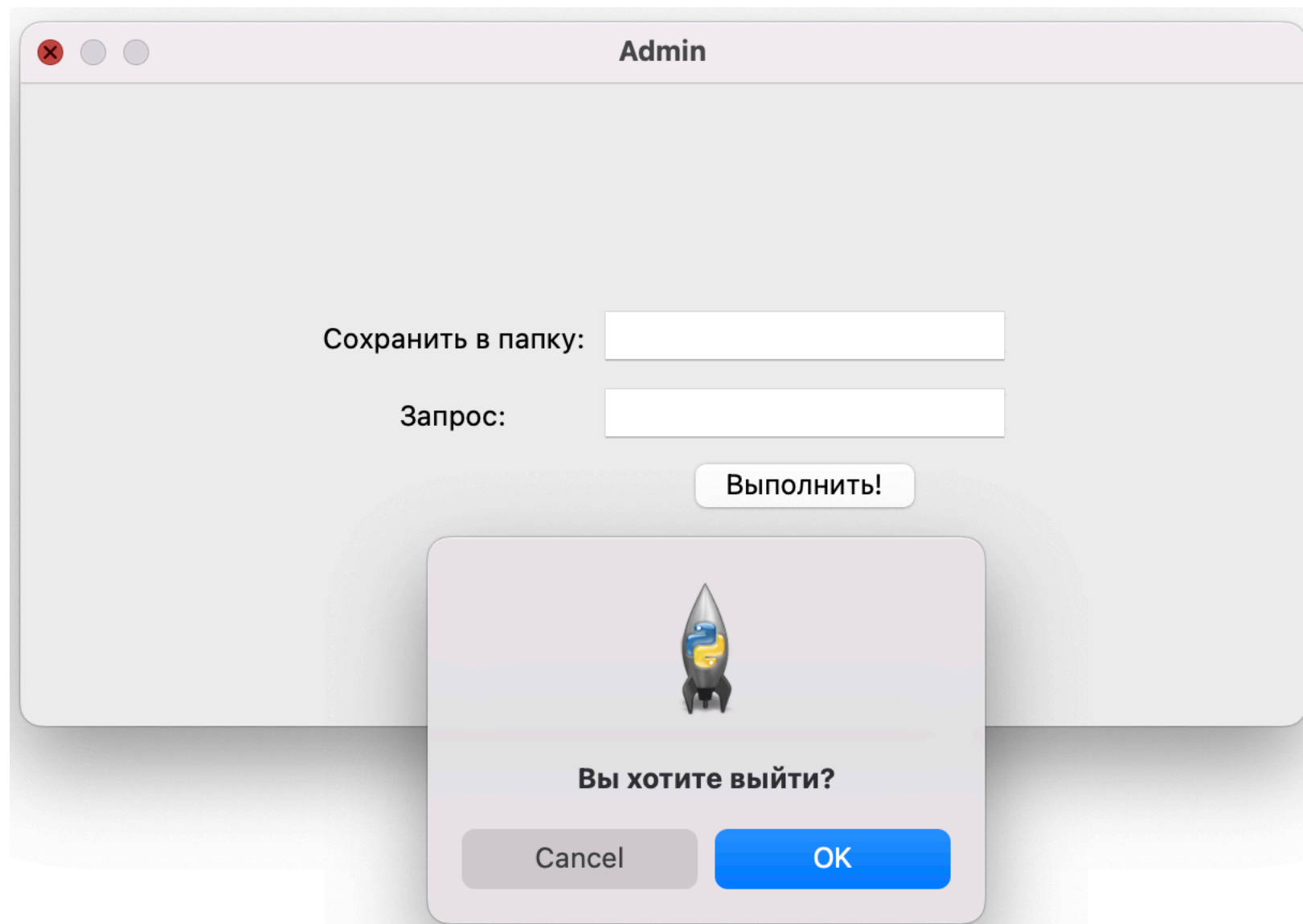
```
Create role "reader" password 'readerserver2024'  
grant select on book to reader  
grant select on new to reader  
grant select, insert, delete on formulyr to reader
```

```
Create role "admin"  
with login superuser  
password 'adminserver2024'
```

# Оконное приложение Читателя



# Оконное приложение Администратора



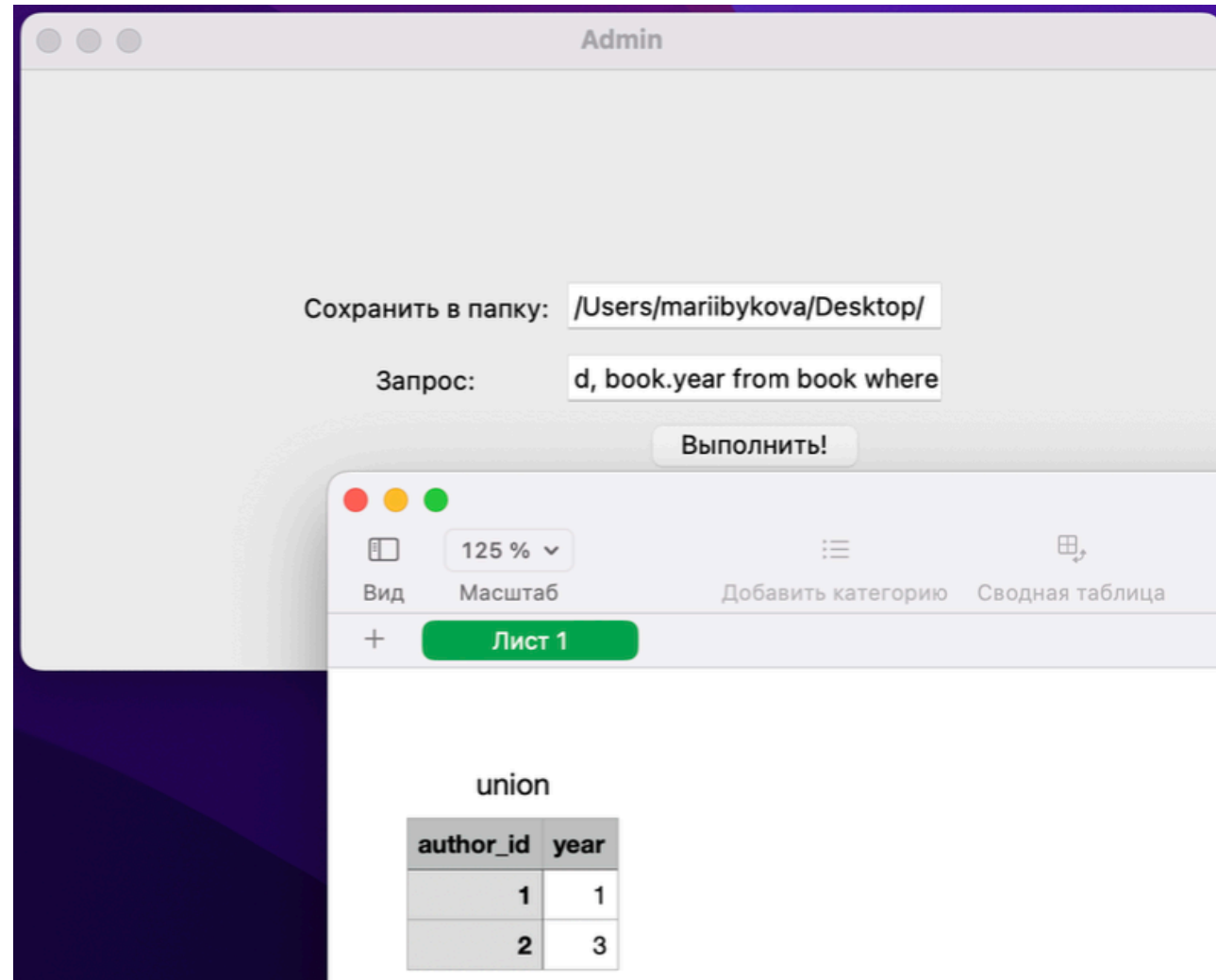


# SQL-инъекции типа UNION

```
select book.author_id, book.year  
from book where book.id= $id
```

```
select book.author_id, book.year  
from book where book.id =  
-1 UNION SELECT "new".id, "new".book_id FROM "new"
```

# SQL-инъекции типа UNION

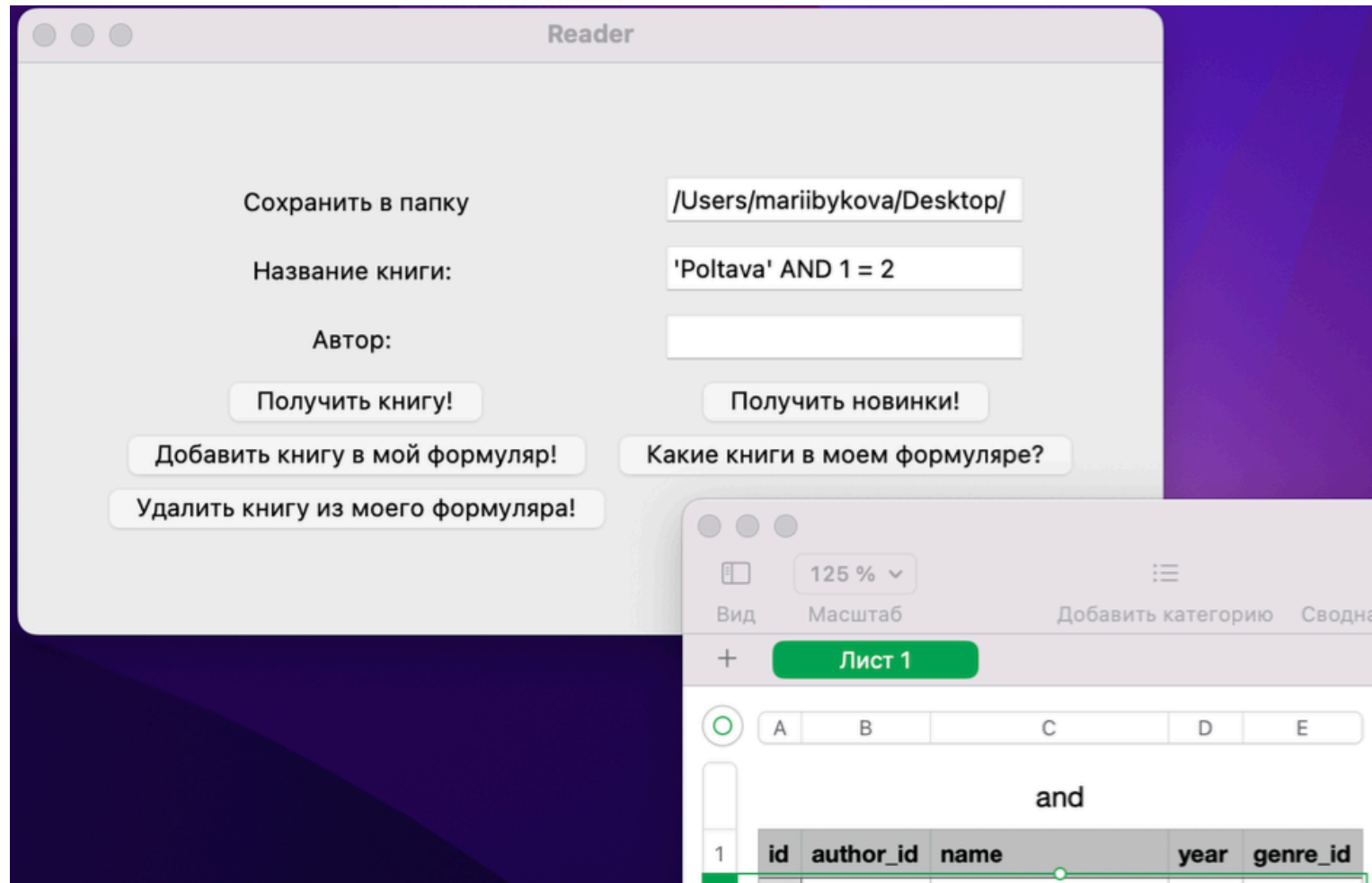


# Слепая SQL-инъекция

```
SELECT * FROM book WHERE book.id = $id|
```

```
SELECT * FROM book  
WHERE book.id = 1 AND 1 = 2
```

# Слепая SQL-инъекция

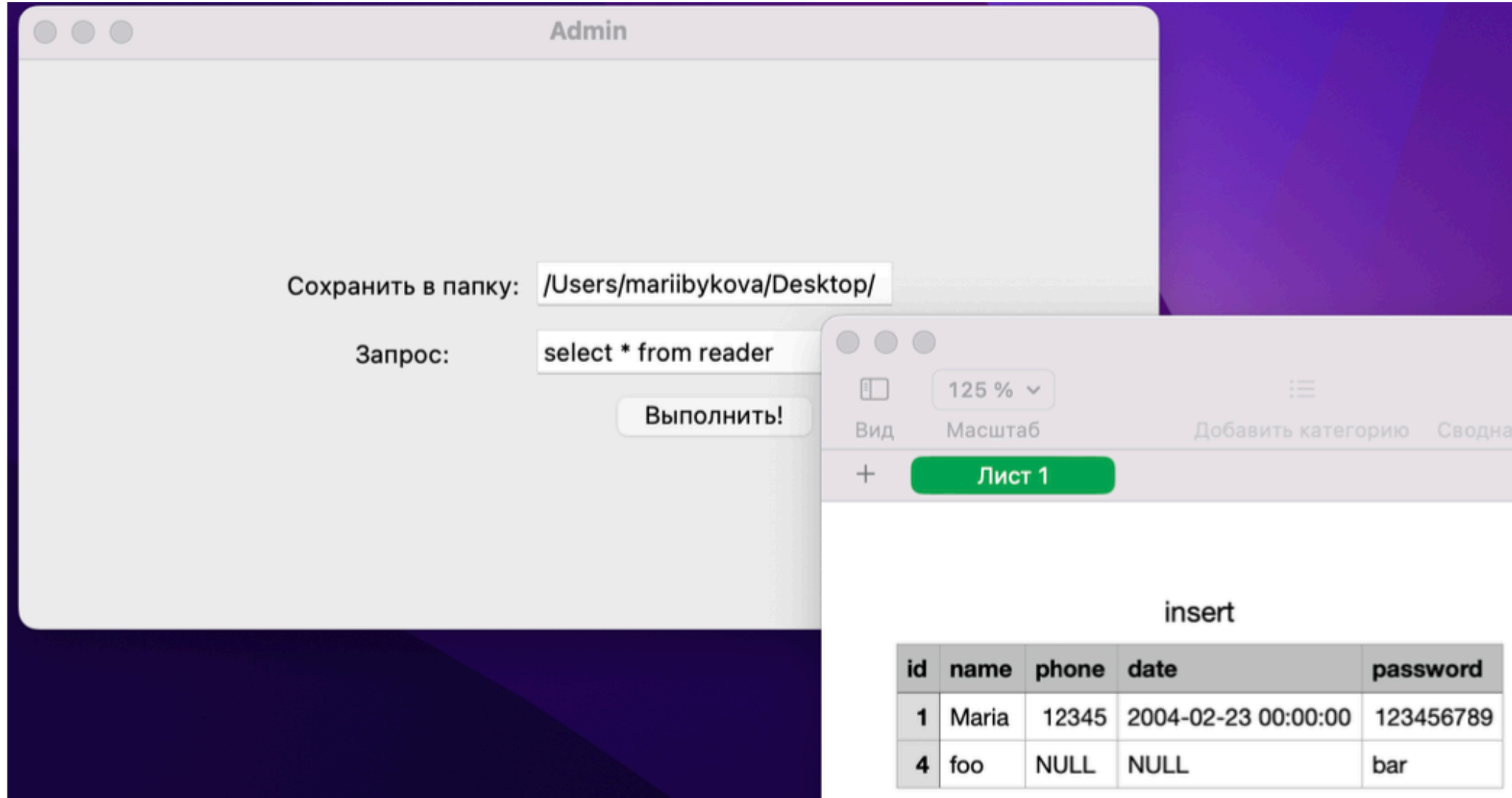


# SQL-инъекции с дополнительными символами

```
SELECT * FROM new  
WHERE new.id = $id
```

```
SELECT * FROM new  
WHERE new.id = 1;  
INSERT INTO reader(id,name, password)  
VALUES (4,'foo', 'bar')
```

# SQL-инъекции с дополнительными символами



The screenshot shows a web application titled "Admin". It features a "Сохранить в папку:" (Save to folder:) field with the path `/Users/mariibyкова/Desktop/`. Below it is a "Запрос:" (Query:) field containing the SQL statement `select * from reader`. A "Выполнить!" (Execute!) button is positioned to the right of the query field. An overlay window displays a table with the title "insert". The table has five columns: `id`, `name`, `phone`, `date`, and `password`. It contains two rows of data. The first row shows `1` for `id`, `Maria` for `name`, `12345` for `phone`, `2004-02-23 00:00:00` for `date`, and `123456789` for `password`. The second row shows `4` for `id`, `foo` for `name`, `NULL` for `phone`, `NULL` for `date`, and `bar` for `password`.

Сохранить в папку: `/Users/mariibyкова/Desktop/`

Запрос: `select * from reader`

Выполнить!

insert

id	name	phone	date	password
1	Maria	12345	2004-02-23 00:00:00	123456789
4	foo	NULL	NULL	bar

# Методы защиты от SQL-инъекций

1. Проверка вводимых данных на соответствие их типам.
2. Экранирование строк.
3. Принцип наименьших привилегий.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 Агафонов, А. БЕЗОПАСНОСТЬ СИСТЕМ БАЗ ДАННЫХ / А. Агафонов. — Самара: Издательство Самарского университета, 2023. — С. 272.
- 2 Clarke, J. SQL Injection Attacks and Defense / J. Clarke. — Waltham: Syngress, 2012. — С. 576.
- 3 SQL injection [Электронный ресурс]. — URL: <https://learn.microsoft.com/en-us/sql/relational-databases/security/sql-injection?view=sql-server-ver16> (Дата обращения 15.12.2023). Загл. с экр. Яз. англ.
- 4 PostgreSQL: Documentation: 16: CREATE DATABASE [Электронный ресурс]. — URL: <https://www.postgresql.org/docs/current/sql-createdatabase.html> (Дата обращения 22.04.2024). Загл. с экр. Яз. англ.
- 5 PostgreSQL: Documentation: 16: CREATE ROLE [Электронный ресурс]. — URL: <https://www.postgresql.org/docs/current/sql-createrole.html> (Дата обращения 25.05.2024). Загл. с экр. Яз. англ.
- 6 Информационная безопасность баз данных [Электронный ресурс]. — URL: <https://searchinform.ru/informatsionnaya-bezopasnost/osnovy-ib/informatsionnaya-bezopasnost-v-otraslyakh/informatsionnaya-bezopasnost-baz-dannykh/> (Дата обращения 21.05.2024). Загл. с экр. Яз. рус.
- 7 Графический интерфейс на Tkinter [Электронный ресурс]. — URL: <https://proglib.io/p/tkinter-2023-05-02?ysclid=lrthmh8j8k298075821> (Дата обращения 03.12.2023). Загл. с экр. Яз. англ.
- 8 Python Tkinter Image + Examples [Электронный ресурс]. — URL: <https://pythonguides.com/python-tkinter-image/> (Дата обращения 14.01.2024). Загл. с экр. Яз. рус.
- 9 Безопасность на уровне строк [Электронный ресурс]. — URL: <https://learn.microsoft.com/ru-ru/sql/relational-databases/security/row-level-security?view=sql-server-ver16> (Дата обращения 07.11.2023). Загл. с экр. Яз. рус.
- 10 Что такое SQL-инъекция? [Электронный ресурс]. — URL: <https://www.kaspersky.ru/resource-center/definitions/sql-injection> (Дата обращения 17.02.2024). Загл. с экр. Яз. рус.
- 11 Blind SQL Injection [Электронный ресурс]. — URL: <https://www.stationx.net/blind-sql-injection/> (Дата обращения 10.05.2024). Загл. с экр. Яз. англ.