Library Database Application (D1 – Repository Pattern)

1. Basic Project Information

Project Name:
Library Database Application

Author:
Milana Poljanskova

Contact:
mila.p.06@seznam.cz

School: SPŠE
Ječná

Date:
11.01.2026

Project Type:
School project - database application with graphical user interface

Notes:
This application was created as part of a school portfolio and fulfills the D1 assignment type - Repository pattern.

2. User Requirements Specification

The purpose of the application is to manage a database of books, authors, and publishers, with features for data import, report generation, and database connection configuration.

Functional requirements (Use Cases):

- Create, edit, delete, and view books
- Manage authors
- Manage publishers
- Link books and authors (M:N relationship)
- Transfer authorship between authors
- Import data from CSV, JSON, and XML
- Generate summary reports
- Configure database connection **User Roles:**
- Application user (database administrator)

3. Application Architecture and Design Patterns

Design Pattern: Repository pattern (D1)

Architecture:

- **User Interface (UI):** Tkinter
- **Repository Layer:** Encapsulates database access logic
- **Database Layer:** MS SQL Server
- **Configuration:** config.ini

Design Patterns Used:

- Repository pattern
- Separation of concerns

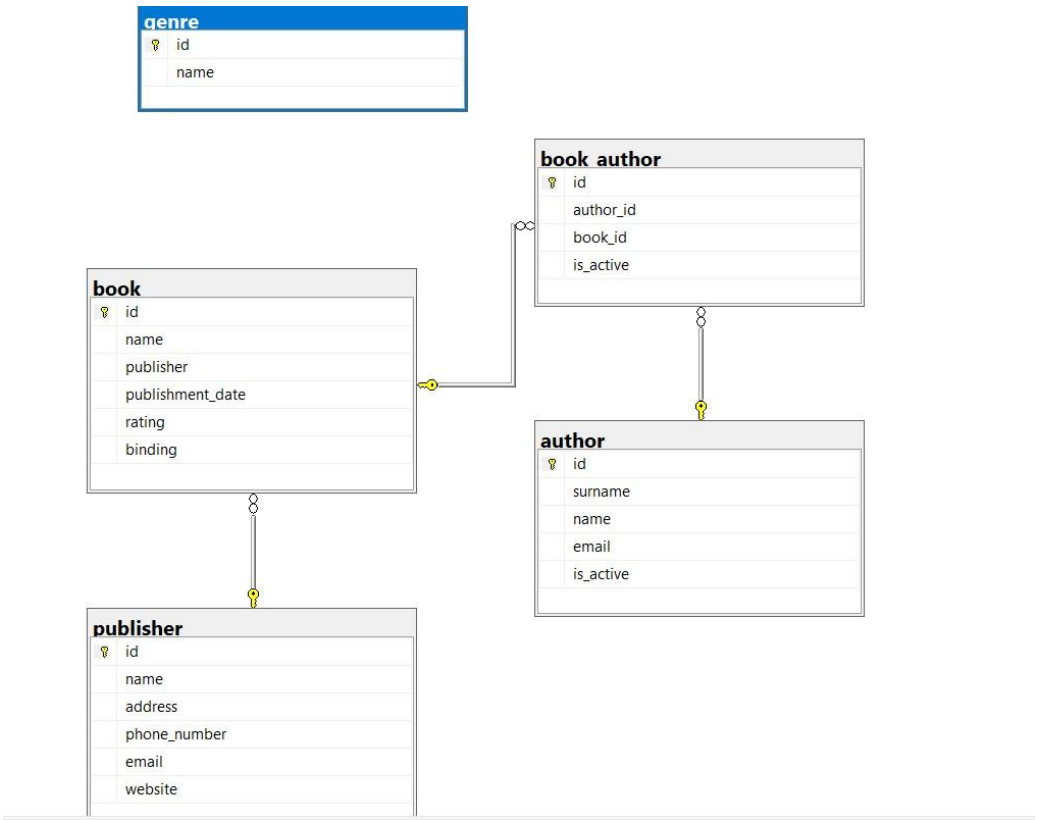
Each database table has its own repository class. The UI communicates only via the repository layer.

4. Application Behavior

Example: Adding a Book

1. The user opens the “Add book” dialog.
2. Fills in book name, publisher, date, rating, binding, and authors.
3. On confirmation, data is stored in the following tables:
    - book
    - book_author
4. The operation is performed in a transaction.
5. In case of an error, the transaction is rolled back.

5. Database Model (E-R)



Tables:

- book
- author
- publisher
- genre
- book_author (M:N relationship)

Views:

- vw_book_list - list of books with publisher names
- vw_publisher_report - aggregated publisher report

Data Types Used:

- String: names, emails
- Float: rating
- Boolean: is_active
- Enum: binding (hardcover, paperback, ebook)
- Date: publishment_date

6. Imported and Exported Files

Format	Table	Description
CSV	publisher	Import publishers
JSON	author	Import authors
XML	genre	Import genres

Fields for CSV publisher:

- name (required)
- address (optional)
- phone_number (optional)
- email (optional)
- website (optional)

Incorrect formats are caught and reported by the application.

7. Application Configuration

Configuration is stored in config.ini.

Configurable Options:

- ODBC Driver
- Server
- Database name
- Trusted connection
- Encrypt
- Trust server certificate

Configuration can also be edited in the application via the Settings tab.

8. Installation and Running the Application

1. Import database structure using SQL script (database.sql).
2. Set connection options in config.ini.
3. Run the application:

```
python main.py
```

Detailed instructions are available in README.txt.

9. Error States and Solutions

Error Type	Description	Solution
Database	Incorrect credentials, connection error, or server unavailable	Check settings in **Settings** tab or config.ini; ensure SQL Server is running and accessible
Configuration	Missing database section or required options (driver, server, database, username, password, encrypt, trust_server_certificate) in config.ini	Add the missing database section and required configuration options
Invalid input	Rating outside 0–5 or invalid binding value	Correct the input: rating must be between **0–5**, binding must be one of **hardcover / paperback / ebook**
Missing required data	Required fields are empty (e.g. book name, publisher)	Fill in all mandatory fields before saving
Import error	Wrong file format or invalid data in CSV, JSON, or XML	Check file format and correct errors; the application displays an error message

10. Third-Party Libraries

- pyodbc
- tkinter
- configparser
- csv
- json
- xml.etree.ElementTree

11. Final Summary

The application fulfills all requirements of the D1 assignment. It uses a relational database, Repository pattern, transactions, data import, reporting, and configurable connection.

12. Documentation Format

The documentation is prepared as a single PDF file, created using Microsoft Word.