Project Overview

- Purpose
    - This database is designed to replace outdated, handwritten record-keeping systems for tracking library assets. By automating the process, librarians and staff can reduce the time spent on administrative tasks and focus more on assisting students. It also minimizes human errors, ensures accurate record-keeping, and improves overall efficiency in managing library materials.

- Intended Use
    - The database serves as an automated system for tracking library assets, making it easier to manage inventory, lending, and returns. It will be used by students, teachers, librarians, school administrators, and the IT department. Students will primarily use it to search the catalog, check out books, and track their borrowing history. Librarians and administrators will oversee inventory, manage book conditions, and handle late fees or lost book charges. By streamlining these processes, the database ensures smoother library operations and better access to educational resources.

Project Scope

The School Library Database will serve as a centralized system for efficiently managing library resources, user accounts, and borrowing activities. The system will track books, their availability, and condition while maintaining comprehensive borrowing records. It will support core library functions such as inventory management, book reservations, loan processing, and overdue fee calculations. The database will also enable librarians and administrators to monitor user activity, enforce borrowing policies. This ensures seamless operations while maintaining scalability and accuracy in tracking library assets and user interactions.

Glossary

- **ISBN:** (International Standard Book Number) is a unique identifier assigned to books for tracking and cataloging purposes.
- **Medium:** The format in which a library item is available, such as DVD, VHS, Blu-ray, audiobook, eBook, or print.
- **IT:** (Information Technology) refers to the use, development, and management of computer systems, networks, and software to store, process, and transmit data.
- **Unix Timestamp:** A numerical representation of time, counting the number of seconds that have elapsed since January 1, 1970 (UTC). It is commonly used in databases and programming for date/time storage and calculations.
- **VARCHAR:** (Variable Character) A data type in SQL that stores variable-length text strings. It allows storing strings with a defined maximum length, making it more efficient than fixed-length types like char.

- **SQL:** (Structured Query Language) is a domain-specific language used to manage, query, and manipulate relational databases.
- **SQLite 3:** Lightweight, self-contained, serverless database engine that uses a single file to store data and supports SQL for efficient data management.
- **MariaDB:** An open-source relational database system, designed for high-volume transactional environments.

Platform Selection

For our project, we selected SQLite (version 3) because it delivers all the necessary functionality while remaining compact and easy to use. Unlike larger, more complex database systems, SQLite doesn't require a separate server process or extensive setup, which makes it ideal for quick development and testing.

We based our choice on a few key factors:

- Lightweight and Efficient: SQLite operates directly from a single file, which simplifies both development and distribution. It's fast, resource-friendly, and well-suited for applications that don't need the overhead of a full client-server model.
- Portability: Because the entire database lives in a single file, it's easy to share among team members, back up, and deploy without any configuration issues.
- Simplicity: SQLite offers the best balance of features and simplicity while other systems, such as MariaDB, are more suitable for high-volume environments.

Database Schema

```
-- Table: Authors
-- Stores author details for books
CREATE TABLE authors (
    author_id INTEGER PRIMARY KEY AUTOINCREMENT,
    first_name VARCHAR(255),
    middle_name VARCHAR(255),
    last_name VARCHAR(255)
);

-- Table: Items
-- General inventory for all library holdings
CREATE TABLE items (
    item_id INTEGER PRIMARY KEY AUTOINCREMENT,
    title VARCHAR(255),
    price DECIMAL(8,2),
    condition VARCHAR(255),
    CONSTRAINT validate_price CHECK (price >= 0),
    CONSTRAINT validate_condition CHECK (condition IN ('new', 'good', 'bad'))
);
```

```
-- Table: Books
-- Specific attributes for books
CREATE TABLE books (
    item_id INTEGER PRIMARY KEY,
    author_id INTEGER,
    isbn VARCHAR(13),
    genre VARCHAR(255),
    publisher_name VARCHAR(255),
    edition VARCHAR(255),
    publication_date DATE,
    faculty_only BOOLEAN,
    FOREIGN KEY (item_id) REFERENCES items(item_id),
    FOREIGN KEY (author_id) REFERENCES authors(author_id),
    CONSTRAINT ISBN_Length CHECK (LENGTH(isbn) = 10 OR LENGTH(isbn) = 13),
    CONSTRAINT validate_genre CHECK (genre IN ('action', 'adventure', 'comedy', 'drama',
'fantasy', 'horror', 'mystery', 'romance', 'scifi', 'thriller', 'other'))
);

-- Table: Digital Media
-- Metadata for non-book digital content
CREATE TABLE digital_media (
    item_id INTEGER PRIMARY KEY,
    medium VARCHAR(255),
    publication_date DATE,
    publisher_name VARCHAR(255),
    FOREIGN KEY (item_id) REFERENCES items(item_id),
    CONSTRAINT validate_medium CHECK (medium IN ('ebook', 'audiobook', 'dvd', 'vhs',
'bluray', 'scanned'))
);

-- Table: Magazines
-- Periodicals with issue information
CREATE TABLE magazines (
    item_id INTEGER PRIMARY KEY,
    publisher_name VARCHAR(255),
    issue_number INTEGER,
    FOREIGN KEY (item_id) REFERENCES items(item_id),
    CONSTRAINT validate_issue CHECK (issue_number >= 0)
);

-- Table: Users
-- Common user information for students and faculty
CREATE TABLE users (
    user_id INTEGER PRIMARY KEY AUTOINCREMENT,
```

```
    email VARCHAR(255),
    address VARCHAR(255),
    phone VARCHAR(10),
    first_name VARCHAR(255),
    last_name VARCHAR(255),
    middle_name VARCHAR(255)
);

-- Table: Students
-- Links student records to users
CREATE TABLE students (
    user_id INTEGER PRIMARY KEY,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);

-- Table: Faculty
-- Links faculty records and defines their roles
CREATE TABLE faculty (
    user_id INTEGER PRIMARY KEY,
    role VARCHAR(255),
    FOREIGN KEY (user_id) REFERENCES users(user_id),
    CONSTRAINT validate_role CHECK (role IN ('teacher', 'librarian', 'admin', 'it'))
);

-- Table: Fees
-- Tracks financial transactions related to items
CREATE TABLE fees (
    fee_id INTEGER PRIMARY KEY AUTOINCREMENT,
    amount DECIMAL(8,2),
    fee_applier INTEGER,
    fee_acceptor INTEGER,
    status VARCHAR(255),
    reason VARCHAR(255),
    FOREIGN KEY (fee_applier) REFERENCES users(user_id),
    FOREIGN KEY (fee_acceptor) REFERENCES users(user_id),
    CONSTRAINT validate_amount CHECK (amount >= 0),
    CONSTRAINT validate_status CHECK (status IN ('paid', 'unpaid', 'waived'))
);

-- Table: Reserves
-- Tracks item reservations by users
CREATE TABLE reserves (
    reserve_id INTEGER PRIMARY KEY AUTOINCREMENT,
    item_id INTEGER,
```

```
    user_id INTEGER,
    reserve_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (item_id) REFERENCES items(item_id),
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);

-- Table: Loans
-- Tracks borrowed items and their due dates
CREATE TABLE loans (
    loan_id INTEGER PRIMARY KEY AUTOINCREMENT,
    item_id INTEGER,
    user_id INTEGER,
    checkout_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    expected_time TIMESTAMP,
    FOREIGN KEY (item_id) REFERENCES items(item_id),
    FOREIGN KEY (user_id) REFERENCES users(user_id),
    CONSTRAINT validate_expected CHECK (expected_time >= checkout_time)
);
```

Data Population

```
/* ----------  AUTHORS  ---------- */
INSERT INTO authors (author_id, first_name, middle_name, last_name) VALUES
  (1,'Thomas','H','Cormen'),
  (2,'Robert','C','Martin'),
  (3,'Andrew',NULL,'Hunt'),
  (4,'Joanne','Kathleen','Rowling'),
  (5,'Frank',NULL,'Herbert'),
  (6,'George',NULL,'Orwell'),
  (7,'Harper',NULL,'Lee'),
  (8,'F','Scott','Fitzgerald'),
  (9,'Herman',NULL,'Melville'),
  (10,'Jane',NULL,'Austen');

/* ----------  USERS  (20 rows) ---------- */
INSERT INTO users (user_id,email,address,phone,first_name,last_name,middle_name)
VALUES
  (1,'alice.johnson@school.edu'   ,'123 Library St' ,'5550000001','Alice'   ,'Johnson',NULL),
  (2,'bob.smith@school.edu'       ,'124 Library St' ,'5550000002','Bob'  ,'Smith'  ,NULL),
  (3,'carol.williams@school.edu'  ,'125 Library St' ,'5550000003','Carol'   ,'Williams',NULL),
  (4,'david.brown@school.edu'     ,'126 Library St' ,'5550000004','David'   ,'Brown'  ,NULL),
  (5,'eve.davis@school.edu'   ,'127 Library St' ,'5550000005','Eve'  ,'Davis'  ,NULL),
  (6,'frank.miller@school.edu' ,'128 Library St' ,'5550000006','Frank'   ,'Miller' ,NULL),
  (7,'grace.wilson@school.edu'    ,'129 Library St' ,'5550000007','Grace'   ,'Wilson' ,NULL),
```

```
  (8,'henry.taylor@school.edu'           ,'130 Library St' ,'5550000008','Henry'   ,'Taylor' ,NULL),
  (9,'isabella.moore@student.edu' ,'201 Dorm Rd'    ,'5550000101','Isabella','Moore'  ,NULL),
  (10,'jack.anderson@student.edu' ,'202 Dorm Rd'   ,'5550000102','Jack'   ,'Anderson',NULL),
  (11,'karen.thomas@student.edu'  ,'203 Dorm Rd'   ,'5550000103','Karen'   ,'Thomas' ,NULL),
  (12,'leo.jackson@student.edu'   ,'204 Dorm Rd'     ,'5550000104','Leo'     ,'Jackson',NULL),
  (13,'mia.white@student.edu'           ,'205 Dorm Rd'           ,'5550000105','Mia'     ,'White'
,NULL),
  (14,'nathan.harris@student.edu' ,'206 Dorm Rd'    ,'5550000106','Nathan'  ,'Harris' ,NULL),
  (15,'olivia.martin@student.edu' ,'207 Dorm Rd'      ,'5550000107','Olivia'  ,'Martin' ,NULL),
  (16,'paul.thompson@student.edu' ,'208 Dorm Rd'  ,'5550000108','Paul'   ,'Thompson',NULL),
  (17,'quinn.garcia@student.edu'  ,'209 Dorm Rd'    ,'5550000109','Quinn'   ,'Garcia' ,NULL),
  (18,'ruby.martinez@student.edu' ,'210 Dorm Rd'   ,'5550000110','Ruby'   ,'Martinez',NULL),
  (19,'sam.robinson@student.edu'  ,'211 Dorm Rd'   ,'5550000111','Sam'    ,'Robinson',NULL),
  (20,'tina.clark@student.edu' ,'212 Dorm Rd'           ,'5550000112','Tina'    ,'Clark'  ,NULL);

/* ----------  FACULTY  (8 rows — roles are validated) ---------- */
INSERT INTO faculty (user_id, role) VALUES
  (1,'librarian'),
  (2,'teacher'),
  (3,'teacher'),
  (4,'admin'),
  (5,'it'),
  (6,'librarian'),
  (7,'teacher'),
  (8,'teacher');

/* ----------  STUDENTS  (12 rows) ---------- */
INSERT INTO students (user_id) VALUES
  (9),(10),(11),(12),(13),(14),(15),(16),(17),(18),(19),(20);

/* ----------  ITEMS  (20 rows) ---------- */
INSERT INTO items (item_id,title,price,condition) VALUES
  (1 ,'Introduction to Algorithms'                  ,99.95,'good'),
  (2 ,'Introduction to Algorithms'                  ,99.95,'new' ),
  (3 ,'Clean Code'                          ,39.99,'good'),
  (4 ,'The Pragmatic Programmer'                ,49.99,'good'),
  (5 ,'Harry Potter and the Sorcerers Stone'   ,29.99,'new' ),
  (6 ,'Harry Potter and the Sorcerers Stone'   ,29.99,'good'),
  (7 ,'Dune'                                ,19.99,'good'),
  (8 ,'1984'                                ,14.99,'good'),
  (9 ,'To Kill a Mockingbird'               ,24.99,'good'),
  (10,'The Great Gatsby'                    ,19.99,'good'),
  (11,'Moby Dick'                           ,17.99,'good'),
  (12,'Pride and Prejudice'                 ,15.99,'good'),
```

```
  (13,'Inception (DVD)'                  , 9.99,'new' ),
  (14,'Interstellar (Blu-ray)'           ,12.99,'new' ),
  (15,'The Hobbit (Audiobook)'                ,19.99,'new' ),
  (16,'1984 (eBook)'                     , 4.99,'new' ),
  (17,'National Geographic May 2025'     , 5.99,'new' ),
  (18,'Time Magazine April 2025'         , 4.99,'new' ),
  (19,'Scientific American March 2025'   , 6.99,'new' ),
  (20,'Forbes February 2025'             , 5.99,'new' );

/* ----------  BOOKS  (12 items, incl. duplicate copies) ---------- */
INSERT INTO books
(item_id,author_id,isbn,genre,publisher_name,edition,publication_date,faculty_only) VALUES
  (1 ,1 ,'9780262033848','other'         ,'MIT Press'              ,'3rd','2009-07-31',FALSE),
  (2 ,1 ,'9780262033848','other'         ,'MIT Press'              ,'3rd','2009-07-31',FALSE),
  (3 ,2 ,'9780132350884','other'         ,'Prentice Hall'          ,'1st','2008-08-11',FALSE),
  (4 ,3 ,'9780201616224','other'         ,'Addison-Wesley'               ,'1st','1999-10-30',FALSE),
  (5 ,4 ,'0439708184'   ,'fantasy'  ,'Scholastic'          ,'1st','1998-09-01',FALSE),
  (6 ,4 ,'0439708184'   ,'fantasy'  ,'Scholastic'          ,'1st','1998-09-01',FALSE),
  (7 ,5 ,'0441172717'   ,'scifi'    ,'Chilton'                ,'1st','1965-08-01',FALSE),
  (8 ,6 ,'9780451524935','scifi'         ,'Secker & Warburg'             ,'1st','1949-06-08',FALSE),
  (9 ,7 ,'9780061120084','drama'         ,'J. B. Lippincott & Co.'       ,'1st','1960-07-11',FALSE),
  (10,8 ,'9780743273565','drama'         ,'Charles Scribner''s Sons'  ,'1st','1925-04-10',FALSE),
  (11,9 ,'9781503280786','adventure','Harper & Brothers'     ,'1st','1851-10-18', TRUE),
  (12,10,'9781503290563','romance'  ,'T. Egerton'             ,'1st','1813-01-28',FALSE);

/* ----------  DIGITAL MEDIA  (4 distinct item_ids) ---------- */
INSERT INTO digital_media (item_id,medium,publication_date,publisher_name) VALUES
  (13,'dvd'        ,'2010-07-16','Warner Bros'),
  (14,'bluray'  ,'2014-11-07','Paramount Pictures'),
  (15,'audiobook','2012-09-18','HarperAudio'),
  (16,'ebook'   ,'2013-04-04','Houghton Mifflin Harcourt');

/* ----------  MAGAZINES  (4 distinct item_ids) ---------- */
INSERT INTO magazines (item_id,publisher_name,issue_number) VALUES
  (17,'National Geographic',202505),
  (18,'Time USA',        202504),
  (19,'Springer Nature',   202503),
  (20,'Forbes Media',   202502);

/* ----------  FEES  (fee_applier must be a librarian) ---------- */
INSERT INTO fees (fee_id,amount,fee_applier,fee_acceptor,reason,status) VALUES
  (1, 5.00 ,1 , 9 ,'Overdue book' ,'unpaid'),
  (2,10.00 ,1 ,10 ,'Lost item'    ,'unpaid'),
  (3, 2.50 ,6 , 3 ,'Late return'  ,'paid'  ),
```

```
  (4, 7.25 ,1 ,14 ,'Damage fee'   ,'unpaid'),
  (5, 3.00 ,6 ,18 ,'Overdue book' ,'waived');

INSERT INTO fees (fee_id, amount, fee_applier, fee_acceptor, reason, status) VALUES
  ( 6 ,  8.50 , 6 , 11 , 'Overdue book'   , 'unpaid'),
  ( 7 , 12.00 , 1 , 12 , 'Lost item'      , 'paid'  ),
  ( 8 ,  4.25 , 6 , 13 , 'Late return'    , 'unpaid'),
  ( 9 ,  6.75 , 1 , 16 , 'Damaged cover'  , 'unpaid'),
  (10 , 15.00 , 6 , 19 , 'Lost item'      , 'unpaid'),
  (11 ,  3.50 , 1 , 20 , 'Overdue book'   , 'paid'  ),
  (12 ,  1.75 , 6 , 10 , 'Late return'    , 'paid'  ),
  (13 ,  2.00 , 1 ,  8 , 'Overdue book'   , 'waived'),
  (14 ,  5.00 , 6 ,  5 , 'Lost DVD'       , 'unpaid'),
  (15 ,  9.50 , 1 ,  2 , 'Damaged pages'  , 'unpaid'),
  (16 ,  7.00 , 6 ,  3 , 'Overdue book'   , 'paid'  ),
  (17 , 11.25 , 1 , 14 , 'Lost item'      , 'waived'),
  (18 ,  4.00 , 6 , 17 , 'Late return'    , 'unpaid'),
  (19 ,  6.00 , 1 ,  9 , 'Overdue book'   , 'paid'  ),
  (20 , 13.50 , 6 , 18 , 'Damaged Blu-ray' , 'unpaid');

INSERT INTO reserves (reserve_id, item_id, user_id, reserve_time) VALUES
  ( 1,  3,  9,'2025-04-15 10:00:00'),
  ( 2,  8, 10,'2025-04-16 11:00:00'),
  ( 3, 17, 11,'2025-04-18 09:30:00'),
  ( 4, 14, 12,'2025-04-19 14:45:00'),
  ( 5,  3, 17,'2025-04-20 10:00:00'),
  ( 6,  3, 18,'2025-04-20 10:05:00'),
  ( 7,  7, 19,'2025-04-20 11:00:00'),
  ( 8,  7, 20,'2025-04-20 11:05:00'),
  ( 9,  1, 12,'2025-04-21 09:00:00'),
  (10,  1, 13,'2025-04-21 09:05:00'),
  (11, 11, 14,'2025-04-21 10:00:00'),
  (12, 15, 15,'2025-04-21 11:30:00'),
  (13, 16, 10,'2025-04-22 09:00:00'),
  (14, 17, 18,'2025-04-22 10:00:00'),
  (15, 18, 11,'2025-04-22 11:00:00'),
  (16, 19, 16,'2025-04-22 12:00:00'),
  (17, 20, 17,'2025-04-22 13:00:00'),
  (18, 12,  9,'2025-04-23 09:00:00'),
  (19,  4, 15,'2025-04-23 10:00:00'),
  (20,  5, 19,'2025-04-23 11:00:00');

/* ----------  LOANS  (expected_time ≥ checkout_time) ---------- */
INSERT INTO loans (loan_id,item_id,user_id,checkout_time,expected_time) VALUES
```

```
(1, 1,  9,'2025-04-10 09:00:00','2025-04-24 09:00:00'),
(2, 2,  2,'2025-04-11 10:00:00','2025-04-25 10:00:00'),
(3, 5, 13,'2025-04-12 12:00:00','2025-04-26 12:00:00'),
(4, 6,  3,'2025-04-14 13:00:00','2025-04-28 13:00:00'),
(5,13, 14,'2025-04-15 15:30:00','2025-04-22 15:30:00'),
(6,18, 15,'2025-04-18 16:00:00','2025-04-25 16:00:00');
```

Table Outputs

SELECT * FROM Items;

item_id	title	price	condition
1	Introduction to Algorithms	99.95	good
2	Introduction to Algorithms	99.95	new
3	Clean Code	39.99	good
4	The Pragmatic Programmer	49.99	good
5	Harry Potter and the Sorcerers Stone	29.99	new
6	Harry Potter and the Sorcerers Stone	29.99	good
7	Dune	19.99	good
8	1984	14.99	good
9	To Kill a Mockingbird	24.99	good
10	The Great Gatsby	19.99	good
11	Moby Dick	17.99	good
12	Pride and Prejudice	15.99	good
13	Inception (DVD)	9.99	new
14	Interstellar (Blu-ray)	12.99	new
15	The Hobbit (Audiobook)	19.99	new
16	1984 (eBook)	4.99	new
17	National Geographic May 2025	5.99	new
18	Time Magazine April 2025	4.99	new
19	Scientific American March 2025	6.99	new
20	Forbes February 2025	5.99	new

SELECT * FROM Books;

item_id	author_id	isbn	genre	publisher_name	edition	publication_date	faculty_only
1	1	9780262033848	other	MIT Press	3rd	2009-07-31	0
2	1	9780262033848	other	MIT Press	3rd	2009-07-31	0
3	2	9780132350884	other	Prentice Hall	1st	2008-08-11	0
4	3	9780201616224	other	Addison-Wesley	1st	1999-10-30	0
5	4	0439708184	fantasy	Scholastic	1st	1998-09-01	0
6	4	0439708184	fantasy	Scholastic	1st	1998-09-01	0
7	5	0441172717	scifi	Chilton	1st	1965-08-01	0
8	6	9780451524935	scifi	Secker & Warburg	1st	1949-06-08	0
9	7	9780061120084	drama	J. B. Lippincott & Co.	1st	1960-07-11	0
10	8	9780743273565	drama	Charles Scribner's Sons	1st	1925-04-10	0

```
11      9       9781503280786  adventure  Harper & Brothers        1st     1851-10-18      1
12      10      9781503290563  romance    T. Egerton               1st     1813-01-28      0
```

SELECT * FROM Digital_Media;

item_id	medium	publication_date	publisher_name
13	dvd	2010-07-16	Warner Bros
14	bluray	2014-11-07	Paramount Pictures
15	audiobook	2012-09-18	HarperAudio
16	ebook	2013-04-04	Houghton Mifflin Harcourt

SELECT * FROM Magazines;

item_id	publisher_name	issue_number
17	National Geographic	202505
18	Time USA	202504
19	Springer Nature	202503
20	Forbes Media	202502

SELECT * FROM Authors;

author_id	first_name	middle_name	last_name
1	Thomas	H	Cormen
2	Robert	C	Martin
3	Andrew		Hunt
4	Joanne	Kathleen	Rowling
5	Frank		Herbert
6	George		Orwell
7	Harper		Lee
8	F	Scott	Fitzgerald
9	Herman		Melville
10	Jane		Austen

SELECT * FROM Users;

user_id	email	address	phone	first_name	last_name	middle_name
1	alice.johnson@school.edu	123 Library St	5550000001	Alice	Johnson	
2	bob.smith@school.edu	124 Library St	5550000002	Bob	Smith	
3	carol.williams@school.edu	125 Library St	5550000003	Carol	Williams	
4	david.brown@school.edu	126 Library St	5550000004	David	Brown	
5	eve.davis@school.edu	127 Library St	5550000005	Eve	Davis	
6	frank.miller@school.edu	128 Library St	5550000006	Frank	Miller	
7	grace.wilson@school.edu	129 Library St	5550000007	Grace	Wilson	
8	henry.taylor@school.edu	130 Library St	5550000008	Henry	Taylor	
9	isabella.moore@student.edu	201 Dorm Rd	5550000101	Isabella	Moore	

```
10      jack.anderson@student.edu   202 Dorm Rd    5550000102  Jack         Anderson
11      karen.thomas@student.edu  203 Dorm Rd      5550000103  Karen        Thomas
12      leo.jackson@student.edu   204 Dorm Rd      5550000104  Leo          Jackson
13      mia.white@student.edu     205 Dorm Rd      5550000105  Mia          White
14      nathan.harris@student.edu   206 Dorm Rd    5550000106  Nathan       Harris
15      olivia.martin@student.edu   207 Dorm Rd    5550000107  Olivia       Martin
16      paul.thompson@student.edu   208 Dorm Rd    5550000108  Paul         Thompson
17      quinn.garcia@student.edu  209 Dorm Rd      5550000109  Quinn        Garcia
18      ruby.martinez@student.edu   210 Dorm Rd    5550000110  Ruby         Martinez
19      sam.robinson@student.edu  211 Dorm Rd      5550000111  Sam          Robinson
20      tina.clark@student.edu    212 Dorm Rd      5550000112  Tina         Clark
```

SELECT * FROM Students;

```
user_id
-------
9
10
11
12
13
14
15
16
17
18
19
20
```

SELECT * FROM Faculty;

```
user_id  role
-------  ---------
1      librarian
2      teacher
3      teacher
4      admin
5      it
6      librarian
7      teacher
8      teacher
```

SELECT * FROM Fees;

```
fee_id  amount  fee_applier  fee_acceptor  status  reason
------  ------  -----------  ------------  ------  ---------------
1       5       1            9             unpaid  Overdue book
2       10      1            10            unpaid  Lost item
3       2.5     6            3             paid    Late return
```

```
4      7.25   1            14           unpaid  Damage fee
5      3      6            18           waived  Overdue book
6      8.5    6            11           unpaid  Overdue book
7      12     1            12           paid   Lost item
8      4.25   6            13           unpaid  Late return
9      6.75   1            16           unpaid  Damaged cover
10     15     6            19           unpaid  Lost item
11     3.5    1            20           paid   Overdue book
12     1.75   6            10           paid   Late return
13     2      1            8            waived  Overdue book
14     5      6            5            unpaid  Lost DVD
15     9.5    1            2            unpaid  Damaged pages
16     7      6            3            paid   Overdue book
17     11.25   1           14           waived  Lost item
18     4      6            17           unpaid  Late return
19     6      1            9            paid   Overdue book
20     13.5   6            18           unpaid  Damaged Blu-ray
```

SELECT * FROM Reserves;

```
reserve_id  item_id  user_id  reserve_time
----------  -------  -------  -------------------
1            3      9      2025-04-15 10:00:00
2            8      10     2025-04-16 11:00:00
3            17     11     2025-04-18 09:30:00
4            14     12     2025-04-19 14:45:00
5            3      17     2025-04-20 10:00:00
6            3      18     2025-04-20 10:05:00
7            7      19     2025-04-20 11:00:00
8            7      20     2025-04-20 11:05:00
9            1      12     2025-04-21 09:00:00
10           1      13     2025-04-21 09:05:00
11           11     14     2025-04-21 10:00:00
12           15     15     2025-04-21 11:30:00
13           16     10     2025-04-22 09:00:00
14           17     18     2025-04-22 10:00:00
15           18     11     2025-04-22 11:00:00
16           19     16     2025-04-22 12:00:00
17           20     17     2025-04-22 13:00:00
18           12     9      2025-04-23 09:00:00
19           4      15     2025-04-23 10:00:00
20           5      19     2025-04-23 11:00:00
```

SELECT * FROM Loans;

```
loan_id  item_id  user_id  checkout_time       expected_time
-------  -------  -------  ------------------  ------------------
1        1        9        2025-04-10 09:00:00  2025-04-24 09:00:00
2        2        2        2025-04-11 10:00:00  2025-04-25 10:00:00
3        5        13       2025-04-12 12:00:00  2025-04-26 12:00:00
4        6        3        2025-04-14 13:00:00  2025-04-28 13:00:00
5        13       14       2025-04-15 15:30:00  2025-04-22 15:30:00
6        18       15       2025-04-18 16:00:00  2025-04-25 16:00:00
```

Database Functionalities

SEARCH BOOKS BY AUTHOR

WITH cte AS (SELECT author_id from authors WHERE first_name='first_name_here' AND last_name='last_name_here') SELECT * FROM books,cte WHERE cte.author_id=books.author_id;

SEARCH MAGAZINES BY TITLE

SELECT items.title,items.item_id,magazines.issue_number FROM items,magazines WHERE items.title='magazine_title_here' AND magazines.item_id = items.item_id;

SEARCH BOOKS BY TITLE

SELECT items.title,items.item_id,books.isbn,books.publisher_name FROM items,books WHERE items.title='book_title_here' AND books.item_id = items.item_id;

SEARCH BOOKS BY KEYWORD

SELECT items.item_id,items.title,books.isbn,books.publisher_name FROM items,books WHERE books.item_id = items.item_id AND (items.title LIKE '%keyword_here%' OR books.isbn LIKE '%keyword_here%' OR books.publisher_name LIKE '%keyword_here%');

RESERVE ITEM BY Item ID and User ID

INSERT INTO reserves (item_id,user_id,reserve_time) VALUES (item_id_here,user_id_here,CURRENT_TIMESTAMP);

SEE ALL OVERDUE BOOKS

WITH all_overdue AS (SELECT * FROM loans WHERE CURRENT_TIMESTAMP > loans.expected_time)
SELECT items.title,users.first_name,users.last_name,all_overdue.expected_time FROM all_overdue,users,items WHERE all_overdue.item_id=items.item_id AND all_overdue.user_id=users.user_id;

SEE OVERDUE BOOKS BY STUDENT NAME

WITH all_overdue AS (SELECT * FROM loans WHERE CURRENT_TIMESTAMP > loans.expected_time)
SELECT items.title,users.first_name,users.last_name,all_overdue.expected_time FROM all_overdue,users,items WHERE all_overdue.item_id=items.item_id AND

all_overdue.user_id=users.user_id AND users.first_name='first_name' and users.last_name ='last_name';

RETURN BOOK BY ITEM_ID

DELETE FROM loans WHERE item_id=item_id;

ADD NEW STUDENT

INSERT INTO USERS (email, address, phone, first_name, last_name, middle_name) VALUES ('email', 'address', 'phone', 'first_name', 'last_name', 'middle_name'); INSERT INTO students (user_id) VALUES ((SELECT user_id FROM users WHERE user_id NOT IN (SELECT user_id FROM faculty UNION SELECT user_id FROM students)));

ADD NEW FACULTY

INSERT INTO USERS (email, address, phone, first_name, last_name, middle_name) VALUES ('email', 'address', 'phone', 'first_name', 'last_name', 'middle_name'); INSERT INTO faculty (user_id,role) VALUES ((SELECT user_id FROM users WHERE user_id NOT IN (SELECT user_id FROM faculty UNION SELECT user_id FROM students)), 'role');

FACULTY APPLY FEE TO STUDENT

INSERT INTO fees (amount, fee_applier, fee_acceptor, status, reason) VALUES (amount,(SELECT users.user_id FROM users,faculty WHERE users.user_id=faculty.user_id AND users.first_name='faculty_first_name' AND users.last_name='faculty_last_name'),(SELECT users.user_id FROM users,students WHERE users.user_id=students.user_id AND users.first_name='student_first_name' AND users.last_name='student_last_name'), 'status', 'reason');

CHANGE FACULTY ROLE

UPDATE faculty SET role = 'role_name' WHERE user_id = (SELECT users.user_id FROM users,faculty WHERE users.first_name = 'first_name' AND users.last_name = 'last_name' AND users.user_id = faculty.user_id);