1. **Data Model Structure**

**Entities**

1. Authors

* Represents author information.

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Description** |
| Author\_id | INTEGER | Primary Key. Unique ID for each author. |
| Author\_name | TEXT | Name of the author. |

1. Books

* Represents the book itself, containing general data about the book.

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Description** |
| Book\_Id | INTEGER | Primary Key. Unique ID for each book. |
| Author\_Id | TEXT | Foreign Key, References the author of the book. |
| Title | TEXT | Title of the book. |
| Summary | TEXT | Summary of the book. |
| Release\_Date | TEXT | Book release date. |

1. Chapters

* Represents the chapters of a book, containing title and number of each chapter.

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Description** |
| Chapter\_Id | INTEGER | Primary Key. Unique ID for each chapter |
| Book\_Id | INTEGER | Foreign Key. References the corresponding book. |
| Chapter\_Number | INTEGER | Chapter number of the book. |
| Chapter\_Title | TEXT | Title of the chapter. |

1. Contents

* Represents the contents of a chapter, containing text and image contents depending on content type.

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Description** |
| Content\_Id | INTEGER | Primary Key. Unique ID for each content. |
| Chapter\_Id | INTEGER | Foreign Key. References the associated chapter. |
| Content\_Type | TEXT | Type of content to store (image/text/table) |
| Chapter\_Content | TEXT | Content of the chapter. |

**2. Entity-Relationship Diagram**

The ER Diagram depicts:

* An Author entity linked one-to-many to Book
* A Book entity linked by a one-to-many relationship to Chapter
* A Chapter entity linked by a one-to-many relationship to Content

A screenshot of a computer

Description automatically generated

**3. SQL Queries for Data Insertion and Retrieval**

1. This query fetches the book's metadata, including the author, title, summary, and release date.

**SELECT**

**b.book\_id, b.title, b.summary, b.release\_date, a.author\_name**

**FROM**

**Books b**

**JOIN**

**Authors a ON b.author\_id = a.author\_id**

**WHERE**

**b.book\_id = ?; -- Replace '?' with the specific book ID**

1. This query retrieves the chapters of a specific book, ordered by chapter number.

**SELECT**

**c.chapter\_id, c.chapter\_number, c.chapter\_title**

**FROM**

**Chapters c**

**WHERE**

**c.book\_id =? -- Replace '?' with the specific book ID**

**ORDER BY**

**c.chapter\_number;**

1. This query fetches all the content for a specific chapter, ordered by its insertion sequence.

**SELECT**

**ct.content\_id, ct.content\_type, ct.chapter\_content**

**FROM**

**Contents ct**

**WHERE**

**ct.chapter\_id = ? -- Replace '?' with the specific chapter ID**

**ORDER BY**

**ct.content\_id; -- Ensure consistent order of content**

1. This query allows users to search for specific keywords within the content of all chapters of a book.

**SELECT**

**ct.content\_id, ct.content\_type, ct.chapter\_content, ch.chapter\_title, ch.chapter\_number**

**FROM**

**Contents ct**

**JOIN**

**Chapters ch ON ct.chapter\_id = ch.chapter\_id**

**WHERE**

**ch.book\_id = ? AND ct.chapter\_content LIKE '%' || ? || '%'; -- Replace '?' with book ID and keyword**

1. This query provides a summary of the chapters in the book for the Table of Contents screen.

**SELECT**

**c.chapter\_id, c.chapter\_number, c.chapter\_title**

**FROM**

**Chapters c**

**WHERE**

**c.book\_id =? -- Replace '?' with the specific book ID**

**ORDER BY**

**c.chapter\_number;**

**INSERT INTO Authors (author\_id, author\_name)**

**VALUES (1, 'Jane Austen'), (2, 'Mark Twain'), (3, 'J.K. Rowling');**

**INSERT INTO Books (book\_id, author\_id, title, summary, release\_date)**

**VALUES**

**(1, 1, 'Pride and Prejudice', 'A classic novel of manners.', '1813-01-28'),**

**(2, 2, 'Adventures of Huckleberry Finn', 'A story of friendship and freedom.', '1884-12-10'),**

**(3, 3, 'Harry Potter and the Sorcerer's Stone', 'The beginning of the Harry Potter saga.', '1997-06-26');**

**INSERT INTO Chapters (chapter\_id, book\_id, chapter\_number, chapter\_title)**

**VALUES**

**(1, 1, 1, 'Introduction to the Bennets'),**

**(2, 1, 2, 'Mr. Darcy''s Arrival'),**

**(3, 2, 1, 'The Journey Begins');**

**INSERT INTO Contents (content\_id, chapter\_id, content\_type, chapter\_content)**

**VALUES**

**(1, 1, 'Text', 'It is a truth universally acknowledged...'),**

**(2, 2, 'Text', 'Mr. Darcy entered the room with an air of disdain...'),**

**(3, 3, 'Text', 'Huck and Jim set out on their raft adventure...');**

**4. Data Choice and Rationality**

1. Authors Table

* Rationale: The Authors table is necessary to provide metadata about the authors of the books. This entity allows for:
  + Representing the relationship between books and their authors.
  + Searching and filtering books by specific authors.
* Design: The one-to-many relationship between Authors and Books enables a single author to write multiple books, a common real-world scenario.

2. Books Table

* Rationale: The Books table holds the essential details of each book, such as its title, summary, and release date. This table:
  + Provides the foundational metadata needed for the app's Library Screen.
  + Facilitates navigation to related entities like Chapters.
* Design:
  + The author\_id foreign key connects books to their respective authors.
  + The one-to-many relationship between Books and Chapters organizes the book's content hierarchically.

3. Chapters Table

* Rationale: The Chapters table represents the structure of the book. This design:
  + Supports the Table of Contents Screen, enabling users to navigate to specific chapters.
  + Organizes content logically for easy access and retrieval.
* Design:
  + The one-to-many relationship with Books allows each book to have multiple chapters.
  + Attributes such as chapter\_number and chapter\_title enable sorting and user-friendly navigation.

4. Contents Table

* Rationale: The Contents table stores the actual content of each chapter. By breaking down the content into smaller sections, the table:
  + Facilitates efficient searching (e.g., through the Search Screen) by focusing on smaller content segments.
  + Accommodates diverse content types (e.g., text, images, tables) with the content\_type attribute.
* Design:
  + The one-to-many relationship with Chapters enables chapters to hold multiple content pieces.
  + A large chapter\_content field ensures that the app can store substantial content while supporting varied content formats.

**5. Testing Strategy**

**Unit Testing**

1. HTML Parsing:
   * Test that Jsoup correctly extracts book titles, chapters, and content from a sample HTML file.
   * Test that images and tables are parsed and formatted properly.
   * Handle edge cases such as malformed HTML or missing elements.
2. Compression and Decompression:
   * Test that the compressed ZIP file saves correctly to local storage.
   * Test that the ZIP file is decompressed and all files are extracted.
   * Verify error handling for missing or corrupted ZIP files.
3. Content Retrieval:
   * Test database queries for retrieving books, chapters, and content (as per the ERD).
   * Verify data consistency between relationships, such as books linked to authors and chapters.
4. Search Functionality:
   * Test the search logic to ensure accurate results for keywords.
   * Handle special cases like case-insensitivity, multi-word searches, and no matches found.

**Instrumentation Testing**

1. Adaptive Navigation Components

Test Scenarios

1. Bottom Navigation Bar (Compact screens):
   * Verify that all navigation items are visible and clickable.
   * Test navigation between screens using the Bottom Navigation Bar.
   * Handle cases where a user tries to click a disabled navigation item.
2. Navigation Rail (Medium-sized screens):
   * Verify that the Navigation Rail displays correctly on medium-sized screens.
   * Test that icons and labels appear as expected.
   * Test that navigation transitions are seamless.
3. Navigation Drawer (Large screens):
   * Test that the Navigation Drawer opens and closes properly.
   * Verify that selecting a menu item navigates to the correct screen.
   * Handle cases where the drawer remains open in landscape mode.

2. UI Components

Test Scenarios

1. Home Screen:
   * Verify that the app information is displayed correctly.
2. Library Screen:
   * Test that book images, titles, and metadata are displayed correctly.
3. Search Screen:
   * Test the search field for input validation and edge cases.
   * Verify that results update dynamically as the user types.
   * Handle empty or invalid search terms.
4. Table of Contents Screen:
   * Test that the chapters list updates correctly when a new book is selected.
   * Verify click ability of chapter items to navigate to the Reading Screen.
5. Reading Screen:
   * Test the rendering of book content, including text, images, and tables.

3. Additional Edge Cases

Test Scenarios

1. Offline Mode:
   * Test app behavior when there’s no internet for downloading books.
   * Verify that already downloaded books are accessible offline.
2. Localization:
   * Verify UI elements and book content display correctly for multiple languages.
3. Screen Size and Orientation Changes:
   * Verify data retention when switching between portrait and landscape modes.