Assignment

Luca Mawyin

Professor

February 15, 2026

McMaster University

Student Number: 400531739

MacID: mawyinl

Problem 1.

Constraints

Person

- pid is unique
- pid is primary key

Publisher

- name is primary key

Book

- ISBN is primary key
- Publisher name is foreign key referencing Publisher(name)

Website

- Weak entity
- ISBN is a foreign key referencing Book(ISBN)
- url & ISBN together form primary key

EBook

- ISA Book
- ISBN is primary key
- ISBN is foreign key referencing Book(ISBN)

PrintBook

- ISA Book
- ISBN is primary key
- ISBN is foreign key referencing Book(ISBN)
- EVersion is foreign key referencing EBook(ISBN)

authorOf

- Relation between Person and Book with attribute rank
- pid is foreign key referencing Person(pid)
- ISBN is foreign key referencing Book(ISBN)
- pid and ISBN together form primary key

Recipe

- Weak entity
- ISBN is foreign key referencing Book(ISBN)
- title and ISBN together form primary key

usedBy

- Recursive relation between Recipe and itself

- recipe_ISBN, recipe_title is foreign key referencing Recipe(ISBN, title)

- ingredient_ISBN, ingredient_title is foreign key referencing Recipe(ISBN, title)

- recipe_ISBN, recipe_title, ingredient_ISBN, ingredient_title together form primary key

```
Person(
    pid PRIMARY KEY,
    name
)

Publisher(
    name PRIMARY KEY
)

Book(
    ISBN PRIMARY KEY,
    title,
    year,
    edition,
    publisher_name,
    FOREIGN KEY (publisher_name) REFERENCES Publisher(name)
)

Website(
    ISBN,
    url,
    PRIMARY KEY (ISBN, url),
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN)
)

EBook(
    ISBN PRIMARY KEY,
    format,
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN)
)

PrintBook(
    ISBN PRIMARY KEY,
    weight,
    type,
    EVersion,
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN),
    FOREIGN KEY (EVersion) REFERENCES EBook(ISBN)
)

authorOf(
    pid,
    ISBN,
    rank,
    PRIMARY KEY (pid, ISBN),
```

```
    FOREIGN KEY (pid) REFERENCES Person(pid),
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN)
)

Recipe(
    ISBN,
    title,
    time,
    PRIMARY KEY (ISBN, title),
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN)
)

usedBy(
    recipe_ISBN,
    recipe_title,
    ingredient_ISBN,
    ingredient_title,
    PRIMARY KEY (recipe_ISBN, recipe_title, ingredient_ISBN, ingredient_title),
    FOREIGN KEY (recipe_ISBN, recipe_title) REFERENCES Recipe(ISBN, title),
    FOREIGN KEY (ingredient_ISBN, ingredient_title) REFERENCES Recipe(ISBN, title)
)
```

Problem 2.

```
CREATE TABLE Person (
    pid INT PRIMARY KEY,
    name VARCHAR(255) NOT NULL
);

CREATE TABLE Publisher (
    name VARCHAR(255) PRIMARY KEY
);

CREATE TABLE Book (
    ISBN VARCHAR(20) PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    year INT,
    edition SMALLINT,
    publisher_name VARCHAR(255),
    FOREIGN KEY (publisher_name) REFERENCES Publisher(name)
);

-- Cannot represent weak entity in SQL
-- Representing Website and Recipe as separate tables with foreign keys referencing Book(ISBN)
CREATE TABLE Website (
    ISBN VARCHAR(20) NOT NULL,
    url VARCHAR(255) NOT NULL,
    PRIMARY KEY (ISBN, url),
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN)
);
```

```
-- Cannot represent ISA relationship in SQL
-- Representing EBook and PrintBook as separate tables with foreign keys referencing Book(ISBN)
CREATE TABLE EBook (
    ISBN VARCHAR(20) PRIMARY KEY,
    format VARCHAR(50),
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN)
);

CREATE TABLE PrintBook (
    ISBN VARCHAR(20) PRIMARY KEY,
    weight INT,
    type VARCHAR(255),
    EVersion VARCHAR(20),
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN),
    FOREIGN KEY (EVersion) REFERENCES EBook(ISBN)
);

CREATE TABLE authorOf (
    pid INT NOT NULL,
    ISBN VARCHAR(20) NOT NULL,
    rank INT,
    PRIMARY KEY (pid, ISBN),
    FOREIGN KEY (pid) REFERENCES Person(pid),
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN)
);

CREATE TABLE Recipe (
    ISBN VARCHAR(20) NOT NULL,
    title VARCHAR(255) NOT NULL,
    time INT,
    PRIMARY KEY (ISBN, title),
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN)
);

CREATE TABLE usedBy (
    recipe_ISBN VARCHAR(20) NOT NULL,
    recipe_title VARCHAR(255) NOT NULL,
    ingredient_ISBN VARCHAR(20) NOT NULL,
    ingredient_title VARCHAR(255) NOT NULL,
    PRIMARY KEY (recipe_ISBN, recipe_title, ingredient_ISBN, ingredient_title),
    FOREIGN KEY (recipe_ISBN, recipe_title) REFERENCES Recipe(ISBN, title),
    FOREIGN KEY (ingredient_ISBN, ingredient_title) REFERENCES Recipe(ISBN, title)
);
```

Problem 3.

The best way to make recipes depend only on recipes of the same book is to only include a single ISBN in the usedBy relation instead of two. As both the recipe and ingredient are from the same book, the ISBN is implied for both attributes. Then, to validate that the ingredient recipe is from the same book as the recipe, it can be queried from the ingredient_title and the shared ISBN.

Problem 4.

In the resulting relational schema, it could be important to create some ISBN standard for the database. This would ensure that all ISBNs within the database are the same length and structure, making any sort of validation or querying easier and less error prone.

Problem 5.

As the relational model and SQL statements stand, despite Book being subjected to two subclasses, it is technically able to exist independently, which is not ideal. We want to restrict books to being exclusively either an EBook or a PrintBook. To enforce this constraint in the SQL, we can add a column to the Book table called type, which can only have values of either EBook or PrintBook. Then, a CHECK constraint can be added to the Book table to ensure that the entered values in the type column are valid.

In addition to the Book issues, there is also an issue with the usedBy table. As the usedBy table is a recursive relation on the Recipe table, it is possible for a recipe to reference an ingredient that references the recipe, which creates a cycle. In order to prevent looping, a CHECK constraint can be added to the usedBy table to ensure that the recipe_title is not the same as the ingredient_title. Furthermore, there were no established cardinality constraints for the usedBy relation. This means that a recipe could use or be used by other recipes an unlimited number of times.

Problem 6.

```
CREATE TABLE Review(
    pid INT NOT NULL,
    ISBN VARCHAR(20) NOT NULL,
    score SMALLINT CHECK (score >= 0 and score <= 10),
    last_updated TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    description TEXT,
    PRIMARY KEY (pid, ISBN),
    FOREIGN KEY (pid) REFERENCES Person(pid),
    FOREIGN KEY (ISBN) REFERENCES Book(ISBN)
);

CREATE TABLE ReviewVote(
    pid INT NOT NULL,
    rpid INT NOT NULL,
    risbn VARCHAR(20) NOT NULL,
    up BOOLEAN DEFAULT FALSE,
    FOREIGN KEY (rpid, risbn) REFERENCES Review(pid, ISBN),
    FOREIGN KEY (pid) REFERENCES Person(pid),
    PRIMARY KEY (rpid, risbn)
);

CREATE TABLE Recommend(
    pid INT NOT NULL,
    friend_pid INT NOT NULL,
    recipe_ISBN VARCHAR(20) NOT NULL,
    description TEXT,
    FOREIGN KEY (pid) REFERENCES Person(pid),
    FOREIGN KEY (friend_pid) REFERENCES Person(pid),
    FOREIGN KEY (recipe_ISBN) REFERENCES Recipe(ISBN),
    PRIMARY KEY (pid, friend_pid, recipe_ISBN)
);
```

Problem 7.

Given the SQL constructions that we have seen in the course so far, it is not possible to implement the constraint of a author not being able to review their own book. Subsequently, it is also not possible to implement the constraint of a user not being able to recommend a recipe to themselves. Given the SQL constructions we have seen in the course, it is possible to query the database for user information and book information to check if a used is the author of the book that they are reviewing, or if a user is recommending a recipe to themselves. However, it is not possible to enforce these constraints as the data is being entered into the database. This is because there is no conditional logic that would allow us to add a CHECK constraint to the Review and Recommend tables to check if the pid of the review or recommendation is the same as the pid of the author of the book being reviewed or recommended.