WolfCity Publishing Management System

for WolfCity Publishing House

Project Report 2
CSC 540 Spring 2022

Andrew Abate, Chaitanya Patel, Morgan Chapman, Ilya Arakelyan
March 11, 2022

Assumptions

- For Distributor, a unique account number is assigned to each distributor, identifying them. Each distributor has a unique phone number, distributors can have the same name, type, and a street address, but not the same {street address, city} combination.
- For each order placed on behalf of a distributor, a unique order number, OrderID, is assigned. It determines the distributor name, a publication ordered, the number of copies, production date, price, and the shipping costs. All the dependent attributes are not related to each other and any combination of those might occur.
- A distributor is billed on the first of each month. The corresponding invoice gets an ID number, InvoiceID, assigned that uniquely identifies the distributor to bill, invoice amount, billing and payment dates. The billing amount is calculated based on all the orders placed by the distributor in the previous month using the Orders table. All the dependent attributes are not related to each other and any combination of those might occur.
- If a distributor is removed from the Distributors table, its corresponding records are preserved in other tables to keep order history and financial information.
- For each Issue, there is a new Publication that is created, this is done because Issues cannot have a key by itself as it is only a subclass of Publication.
- ArticleText will be a link to the actual text location in the file system (read-only).
- If an issue does not have a specific title, an issue number or issue date should be used.
- Record deletion is allowed (for Publication/Issues/Books, Distributors, Employees) as per the project narrative. However, deletion of these items may affect some calculations in the monthly reports. Deletion of any Publication, Distributor, or Employee record that is referenced by the Payments, Orders, and Invoices relations is not recommended.
- Following the narrative, the Distribution Team can change all Distributor's attributes, including its outstanding balance. Being a more frequent and explicitly requested task, the Distribution Team changing only the Distributor's balance is included as an additional (though redundant) operation.
- A publication must be a periodical issue or a book edition.
- The Production section of the API handles the creation, updating, and deletion of all publications. Overlapping publication operations were excluded from Editing/Publishing.
- Periodicity is specified by the Publication type attribute. Books are published by edition; all journals are published monthly; all magazines are published weekly.
- A publication has an overall topic of interest. Journals and magazines also include articles with their own topic, which can be the same as the publication topic or different. A book only has a single topic.
- Authors of books are associated with the book edition as a whole, not just the chapter(s) for which the author is responsible.
- No chapters within a book edition may have the same title.
- No articles within a periodical issue may have the same title.
- Journalists are included in the authors table.
- Employees can't view the payment table. Their payment information will be on physical paychecks.

1. Global Relational Database Schema

Publications(<u>PublicationID</u>, Title, Type, Topic)

PublicationID -> Title, Type, Topic, PublicationID is the only functional dependency for the publication relation. Since each publicationID is associated with a certain publication, its relation will be unique to the title, type and topic of the publication. Even when a publication has duplicate titles, the publicationID will be unique and can be used to identify the relation subclasses. This relation satisfies the 3NF criteria

Issues(<u>PublicationID</u>, IssueDate, IssueTitle)

PublicationID -> PublicationID, IssueTitle, IssueDate holds when a publication such as a magazine can have a unique publicationID for each of its Issues. For example, Time magazine can have 12 issues a year and for each issue there is a publication with a different publicationsID for each issue(see assumption). This relation satisfied the 3NF criteria.

Articles(<u>PublicationID</u>, <u>ArticleTitle</u>, ArticleTopic, ArticleText)

PublicationID, ArticleTitle -> ArticleTopic, ArticleText, PublicationID, ArticleTitle holds because each article is part of an Issue and has a unique title to it. Knowing the title and the publicationID for the issue it was part of can determine the topic, and text of the article. Because of this functional dependency's left hand side is a superkey, this relation is in BCNF and therefore in 3NF.

Books(<u>PublicationID</u>, EditionNumber, ISBN, PublicationDate)

PublicationID -> EditionNumber, ISBN, PublicationDate, PublicationID holds because each book has a unique publicationID. Using this ID, the editionNumber, ISBN and publication date can be found. Since publicationID is a superkey, the relation is in BCNF and therefore in 3NF.

ISBN -> EditionNumber, ISBN, PublicationDate, PublicationID also holds because ISBN is an international standard book number that is used as a commercial book identifier around the world. The ISBN is unique for each book even if the edition number is the only other difference between books. Because of this, the ISBN can be seen as a publicationID that is standard to the world and so it can also be used to identify the editionNumber, publicationID and publication date in this relation. Since ISBN is a superkey, this relation is in BCNF and therefore in 3NF.

Chapters(<u>PublicationID</u>, <u>ChapterTitle</u>)

PublicationID, ChapterTitle -> PublicationID, ChapterTitle is the only functional dependency for this relation because Chapters, as a weak entity, are uniquely identified by their chapter title and the publication ID to which it belongs. Since this is a trivial functional dependency, which includes a superkey, it satisfies BCNF and 3NF criteria.

Employees(<u>EmpID</u>, Name, Type, Active)

EmpID → EmpID, Name, Type, Active holds because each EmpID is unique and identifies an employee that has a name, type, and is either active or inactive. No other attribute functionally determines any of the others. This functional dependency is in BCNF (and therefore 3NF) because the left hand side is a superkey.

Editors(<u>EmpID</u>)

EmpID → EmpID is in BCNF and 3NF because there is only one attribute, and it is the key.

Authors(<u>EmpID</u>)

EmpID → EmpID is in BCNF and 3NF because there is only one attribute, and it is the key.

Edits(<u>PublicationID</u>, <u>EmpID</u>)

PublicationID, EmpID → PublicationID, EmpID holds because the relationship “Edits” connects two entities: a publication and an employee. Without either ID we cannot appropriately describe the relationship. Because the left hand side contains all attributes, it is a superkey and is therefore in BCNF and 3NF.

WritesBook(<u>PublicationID</u>, <u>EmpID</u>)

PublicationID, EmpID -> PublicationID, EmpID is in BCNF and 3NF because it is a trivial functional dependency containing only 2 attributes, which also make a superkey.

WritesArticle(<u>PublicationID</u>, <u>ArticleTitle</u>, <u>EmpID</u>)

PublicationID, ArticleTitle, EmpID -> PublicationID, ArticleTitle, EmpID is in BCNF and 3NF because it is a trivial functional dependency containing only attributes that also make a superkey. This relation results from a relationship with the key attributes from the 2 connecting entity sets. Without all three attributes, the entities and the relationship cannot be defined.

Payments(<u>CheckNumber</u>, EmpID, Amount, SubmitDate, ClaimDate)

CheckNumber -> EmpID, Amount, SubmitDate, ClaimDate is the only non-trivial functional dependency to hold on this relation. The relation is defined to include a unique check number for each payment, representing the only key. Therefore no other attribute combination could conceivably functionally determine the other attributes. There are many payments that could have the same employee ID, amount, or dates. The left-hand side of the functional dependency represents a superkey, meaning it satisfies BCNF and 3NF criteria.

Distributors(DistAccountNum, Name, Type, Address, City, PhoneNumber, Contact, Balance)

Per distributor assumptions, a unique account number is assigned to each distributor, identifying the distributor. Thus, a straightforward FD follows: DistAccountNum → all other attributes. Per other assumptions, PhoneNumber → all others, {Address, City} → All others. As the left-hand-sides of all FDs are keys, no new non-trivial dependencies can be derived, and the schema is in BCNF, and therefore in 3NF.

Orders(OrderID, DistAccountNum, PublicationID, NumCopies, ProduceByDate, Price, ShippingCosts)

Per distributor assumptions, an unique order identifier, OrderID, determines all other attributes: OrderID → all others. The rest of the attributes are independent of each other, *i.e.*, there are no constraints on any instances of the tuple {DistAccountNum, PublicationID, NumCopies, ProduceByDate, Price, ShippingCosts}, and no additional FD are available. Since the LHS of the is a key, no derived FDs are possible, and the schema is in BCNF and 3NF.

Invoices(InvoiceID, DistAccountNum, Amount, BillingDate, PaymentDate)

Per distributor assumptions, an unique order identifier, InvoiceID, determines all other attributes: InvoiceID → all others. The rest of the attributes are independent of each other, *i.e*., there are no constraints on any instances of the tuple {DistAccountNum, Amount, BillingDate, PaymentDate}, and no additional FD are available. Since the LHS of the is a key, no derived FDs are possible, and the schema is in BCNF and 3NF.

2. Description of Design for Global Schema

Design Decisions for Global Schema

For the global database schema, entity sets involved in subclass hierarchies were converted using an entity-relationship approach. Relations were created for the main superclass entity sets with superclass key and non-key attributes. Each subclass was converted into a relation with any subclass-specific attributes along with the key attribute of the superclass. This approach was applied to Publications with subclasses Issues and Books, and Employees with subclasses Editors and Authors.

Normal entity sets were converted into relations containing all of its attributes. This resulted in relations for Distributors, Order, Invoices, and Payments. In cases involving many-to-one relationships, relationships were combined by including the key attribute of the "one" side entity set(s) within the relation schema of the "many" side entity set. This was applied to Orders, Invoices, Payments. This reduces the required space by eliminating redundant relations and enhances database efficiency by allowing simplified queries.

Weak entity sets were converted into relations containing its own attributes in addition to the key attribute from the supporting entity set. This was applied to Articles and Chapters. Supporting relationships were not converted to relations.

Other relationships were converted into relations containing the key attribute(s) from each connected entity set. This resulted in relations for Edits, WritesBook, and WritesArticle.

Publication(<u>PublicationID</u>, Title, Type, Topic)

PublicationID is the primary key
Title, Type, Topic are not allowed to be null.

Issues(<u>PublicationID</u>, IssueDate, IssueTitle)

PublicationID is the primary key.
IssueDate is not allowed to be null.
IssueTitle is not allowed to be null.
PublicationID must exist in the Publication relation.

Articles(<u>PublicationID</u>, <u>ArticleTitle</u>, ArticleTopic, ArticleText)

PublicationID is one of two primary keys
ArticleTitle is the second of two primary keys
ArticleTopic is not allowed to be null.
ArticleText can be Null, or will be a link to the actual text location in the file system(read-only). This is done because each article can vary in length so storing the whole text can be inefficient. The ArticleText can be updated after creation.
PublicationID must exist in the Issues relation.

Books(PublicationID, EditionNumber, ISBN, PublicationDate)

PublicationID is the primary key.
ISBN and PublicationDate are not allowed to be null.
EditionNumber may be null if the book has not specified any edition numbers at all.
PublicationID must exist in the Publication relation.

Chapters(PublicationID, ChapterTitle)

PublicationID is one of two primary keys
ChapterTitle is the second of two primary keys
PublicationID must exist in the Books relation

Employees(EmpID, Name, Type, Active)

EmpID is the primary key
Name, type, and active are not allowed to be null

Editors(EmpID)

EmpID is the primary key
EmpID must exist in the Employees relation

Authors(EmpID)

EmpID is the primary key
EmpID must exist in the Employees relation

Edits(PublicationID, EmpID)

PublicationID is one of the two primary keys
EmpID is the second of the two primary keys
PublicationID must exist in the Publication relation
EmpID must exist in the Editors relation

WritesBook(PublicationID, EmpID)

PublicationID is one of two primary keys
EmpID is the second of two primary keys
PublicationID must exist in the Books relation
EmpID must exist in the Authors relation

WritesArticle(PublicationID, ArticleTitle, EmpID)

PublicationID is one of three primary keys
ArticleTitle is the second of three primary keys
EmpID is the third of three primary keys
PublicationID and ArticleTitle must exist in the Articles relation
EmpID must exist in the Authors relation

Payments(CheckNumber, EmpID, Amount, SubmitDate, ClaimDate)

CheckNumber is the primary key
EmpID must exist in the Employees relation
Payments, Amount, and SubmitDate are not allowed to be NULL.
EmpID is allowed to be NULL. Employees are allowed to be deleted from the database. However, we wish to preserve any Payments information for the financial reports. Upon deletion of an employee from the Employee relation, any payments to that employee will have a NULL value for EmpID.
ClaimDate is allowed to be NULL. To provide payment tracking, upon creation and submission of a Payment, ClaimDate will be set to NULL by default. A NULL value here represents a payment that has not been claimed by the recipient. Once the payment has been claimed by its addressee, the ClaimDate should be updated to reflect this date.

Distributors(DistAccountNum, Name, Type, Address, City, PhoneNumber, Contact, Balance)

The following set of attributes functionally determine all other attributes and, if entered in the DB, can be used as keys: {DistAccountNum}, {PhoneNumber}, {Address, City}.
Following assumptions, we picked DistAccountNum as the primary key.
DistAccountNum is a foreign key for tables Orders and Invoices.
None of the attributes can be NULL: Address, City, PhoneNumber, Contact should be filled in for shipping purposes; Name, Type, Balance - for generating reports.

Orders(OrderID, DistAccountNum, PublicationID, NumCopies, ProduceByDate, Price, ShippingCosts)

OrderID is the primary key.
Integrity constraints: The Orders table refers to tables Distributors(DistAccountNum, …) and Publications(PublicationID, ...). Updating Distributors and Publications results in the corresponding updates in Orders.
Upon adding a new order all attributes must be specified to complete the order and cannot be NULL except for DistAccountNum that can be updated to NULL if it is removed from the parent table Distributors.

Invoices(InvoiceID, DistAccountNum, Amount, BillingDate, PaymentDate)

InvoiceID is the primary key. To generate an invoice, attributes DistAccountNum, Amount, BillingDate must be specified.
Integrity constraints: The Invoices table refers to table Distributors(DistAccountNum, …). Updating Distributors results in the corresponding updates in Orders.
DistAccountNum can be updated to NULL if it is removed from the parent table Distributors.
Attribute PaymentDate can be NULL at invoice generation, the NULL value stands for no payment made. The Payment Date can be updated when the payment is received.

3. Base Relations

```
CREATE TABLE Publication(
    PublicationID INT,
    Title VARCHAR(128) NOT NULL,
    Type VARCHAR(128) NOT NULL,
    Topic VARCHAR(128) NOT NULL,
    PRIMARY KEY (PublicationID)
);

> SELECT * FROM Publication;
+---------------+---------------------------+----------+----------------+
| PublicationID | Title                     | Type     | Topic          |
+---------------+---------------------------+----------+----------------+
|             1 | Don Quixote               | Novel    | Adventure      |
|             2 | People                    | Magazine | Celebrity News |
|             3 | People                    | Magazine | Celebrity News |
|             4 | The Count of Monte Cristo | Novel    | Adventure      |
|             5 | Science                   | Journal  | Science        |
|             6 | Science                   | Journal  | Science        |
|             7 | Database Systems          | Book     | Databases      |
|             8 | Database Systems          | Book     | Databases      |
+---------------+---------------------------+----------+----------------+
```

```
CREATE TABLE Issues(
    PublicationID INT,
    IssueTitle VARCHAR(128) NOT NULL,
    IssueDate DATE NOT NULL,
    PRIMARY KEY (PublicationID),
    FOREIGN KEY (PublicationID) REFERENCES Publication(PublicationID) ON
UPDATE CASCADE ON DELETE CASCADE
);

> SELECT * FROM Issues;
+---------------+----------------------+------------+
| PublicationID | IssueTitle           | IssueDate  |
+---------------+----------------------+------------+
|             2 | February-21          | 2022-02-21 |
|             3 | February-14          | 2022-02-14 |
|             5 | Volume 375 Issue 6584 | 2022-03-04 |
|             6 | Volume 375 Issue 6583 | 2022-02-25 |
+---------------+----------------------+------------+
```

```
CREATE TABLE Articles(
    PublicationID INT,
    ArticleTitle VARCHAR(128) NOT NULL,
    ArticleTopic VARCHAR(128) NOT NULL,
    ArticleText VARCHAR(128) NOT NULL,
    PRIMARY KEY (PublicationID, ArticleTitle),
    FOREIGN KEY (PublicationID) REFERENCES Issues(PublicationID) ON UPDATE
CASCADE ON DELETE CASCADE
);

> SELECT * FROM Articles;
+--------------+---------------------+---------------+---------------------------------+
| PublicationID | ArticleTitle        | ArticleTopic   | ArticleText                     |
+--------------+---------------------+---------------+---------------------------------+
|            2 | Title1              | Celebrity News | /Issues/1/February-21/Title1.pdf |
|            2 | Title2              | Stories        | /Issues/1/February-21/Title2.pdf |
|            3 | Title1              | Stories        | /Issues/1/February-14/Title1.pdf |
|            6 | Fly Cell Atlas      | Genetics       | fly_cell_atlas.txt               |
|            6 | Structure of Omicron | Coronavirus    | structure_of_omicron.txt         |
+--------------+---------------------+---------------+---------------------------------+
```

```
CREATE TABLE Books(
    PublicationID INT,
    EditionNumber VARCHAR(16),
    ISBN VARCHAR(32) NOT NULL,
    PublicationDate DATE NOT NULL,
    PRIMARY KEY (PublicationID),
    UNIQUE(ISBN),
    FOREIGN KEY (PublicationID) REFERENCES Publication(PublicationID) ON
UPDATE CASCADE ON DELETE CASCADE
);

> SELECT * FROM Books;
+--------------+--------------+------------+-----------------+
| PublicationID | EditionNumber | ISBN       | PublicationDate |
+--------------+--------------+------------+-----------------+
|            1 | 1            | 0142437239 | 2003-02-25      |
|            4 | 1            | 0140449264 | 2003-05-27      |
|            7 | 1            | 0167120441 | 2022-01-31      |
|            8 | 2            | 0335009913 | 2022-02-26      |
+--------------+--------------+------------+-----------------+
```

```
CREATE TABLE Chapters(
    PublicationID INT,
    ChapterTitle VARCHAR(128),
    PRIMARY KEY (PublicationID, ChapterTitle),
    FOREIGN KEY (PublicationID) REFERENCES Books(PublicationID) ON UPDATE
CASCADE ON DELETE CASCADE
);

> SELECT * FROM Chapters;
+---------------+---------------------------------+
| PublicationID | ChapterTitle                    |
+---------------+---------------------------------+
|             1 | Chapter 1                       |
|             1 | Chapter 2                       |
|             4 | Chapter 1                       |
|             7 | DBMS Overview                   |
|             7 | The Relational Model            |
|             8 | DBMS Overview and More          |
|             8 | The Relational Model - Expanded |
+---------------+---------------------------------+
7 rows in set (0.00 sec)
```

```
CREATE TABLE Employees(
    EmpID INT,
    Name VARCHAR(128) NOT NULL,
    Type VARCHAR(10) NOT NULL,
    Active BOOLEAN NOT NULL,
    PRIMARY KEY (EmpID)
);
> SELECT * FROM Employees;
+-------+--------+---------+--------+
| EmpID | Name   | Type    | Active |
+-------+--------+---------+--------+
|     1 | Sam T  | Staff   |      1 |
|     2 | Mary S | Staff   |      1 |
|     3 | John D | Invited |      1 |
|     4 | Pam L  | Invited |      1 |
|     5 | Don D  | Staff   |      1 |
|     6 | Tom A  | Invited |      1 |
|     7 | Jan P  | Staff   |      0 |
|     8 | Nina T | Invited |      1 |
+-------+--------+---------+--------+
```

```
CREATE TABLE Editors(
    EmpID INT,
    PRIMARY KEY (EmpID),
    FOREIGN KEY (EmpID) REFERENCES Employees(EmpID) ON UPDATE CASCADE ON
DELETE CASCADE
);

> SELECT * FROM Editors;
+-------+
| EmpID |
+-------+
|     5 |
|     6 |
|     7 |
|     8 |
+-------+
```

```
CREATE TABLE Authors(
    EmpID INT,
    PRIMARY KEY (EmpID),
    FOREIGN KEY (EmpID) REFERENCES Employees(EmpID) ON UPDATE CASCADE ON
DELETE CASCADE
);

> SELECT * FROM Authors;
+-------+
| EmpID |
+-------+
|     1 |
|     2 |
|     3 |
|     4 |
+-------+
```

```
CREATE TABLE Edits(
    PublicationID INT,
    EmpID INT,
    PRIMARY KEY (PublicationID, EmpID),
    FOREIGN KEY (PublicationID) REFERENCES Publication(PublicationID) ON
UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (EmpID) REFERENCES Editors(EmpID) ON UPDATE CASCADE ON
DELETE CASCADE
);

> SELECT * FROM Edits;
+---------------+-------+
| PublicationID | EmpID |
+---------------+-------+
|             1 |     5 |
|             2 |     6 |
|             5 |     7 |
|             8 |     8 |
+---------------+-------+
```

```
CREATE TABLE WritesBook (
    PublicationID INT,
    EmpID INT,
    PRIMARY KEY (PublicationID, EmpID),
    FOREIGN KEY (PublicationID) REFERENCES Books(PublicationID) ON UPDATE
CASCADE ON DELETE CASCADE,
    FOREIGN KEY (EmpID) REFERENCES Authors(EmpID) ON UPDATE CASCADE ON
DELETE CASCADE
);

> SELECT * FROM WritesBook;
+---------------+-------+
| PublicationID | EmpID |
+---------------+-------+
|             7 |     1 |
|             7 |     4 |
|             8 |     1 |
|             8 |     2 |
|             8 |     4 |
|            11 |     9 |
+---------------+-------+
```

```
CREATE TABLE WritesArticle (
    PublicationID INT,
    ArticleTitle VARCHAR(128),
    EmpID INT,
    PRIMARY KEY (PublicationID, ArticleTitle, EmpID),
    FOREIGN KEY (PublicationID, ArticleTitle) REFERENCES
Articles(PublicationID, ArticleTitle) ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (EmpID) REFERENCES Authors(EmpID) ON UPDATE CASCADE ON
DELETE CASCADE
);
> SELECT * FROM WritesArticle;
+---------------+----------------------+-------+
| PublicationID | ArticleTitle         | EmpID |
+---------------+----------------------+-------+
|             2 | Title1               |     1 |
|             2 | Title2               |     4 |
|             6 | Fly Cell Atlas       |     2 |
|             6 | Structure of Omicron |     3 |
+---------------+----------------------+-------+
```

```
CREATE TABLE Payments (
    CheckNumber INT,
    EmpID INT,
    Amount DECIMAL(8,2) NOT NULL,
    SubmitDate DATE NOT NULL,
    ClaimDate DATE,
    PRIMARY KEY (CheckNumber),
    FOREIGN KEY (EmpID) REFERENCES Employees(EmpID) ON UPDATE CASCADE ON
DELETE SET NULL
);
> SELECT * FROM Payments;
+-------------+-------+---------+------------+------------+
| CheckNumber | EmpID | Amount  | SubmitDate | ClaimDate  |
+-------------+-------+---------+------------+------------+
|        1001 |     1 | 1000.00 | 2022-01-31 | 2022-02-05 |
|        1002 |     2 | 1200.00 | 2022-01-31 | 2022-02-08 |
|        1003 |     4 |  800.00 | 2022-01-31 | 2022-02-10 |
|        1004 |     1 |  100.00 | 2022-02-28 | NULL       |
|        1005 |     2 | 1200.00 | 2022-02-28 | NULL       |
|        1006 |     4 |  800.00 | 2022-02-28 | NULL       |
|        1234 |     3 | 1000.00 | 2022-03-01 | 2022-03-07 |
+-------------+-------+---------+------------+------------+
```

```
CREATE TABLE Distributors (
    DistAccountNum INT,
    Name VARCHAR(128) NOT NULL,
    Type VARCHAR(128) NOT NULL,
    Address VARCHAR(128) NOT NULL,
    City VARCHAR(128) NOT NULL,
    PhoneNumber VARCHAR(16) NOT NULL,
    Contact VARCHAR(128) NOT NULL,
    Balance DECIMAL(8,2) NOT NULL,
    PRIMARY KEY (DistAccountNum)
);

> SELECT * FROM Distributors;
+---------------+----------+---------+---------------+---------+--------------+---------+---------+
| DistAccountNum | Name     | Type    | Address       | City    | PhoneNumber  | Contact | Balance |
+---------------+----------+---------+---------------+---------+--------------+---------+---------+
|             1 | Library1 | Library | 100 New Street | Raleigh | 919-xxx-xxxx | John    |   10.00 |
|             2 | Library2 | Library | 200 New Street | Durham  | 919-xxx-xxxx | John    |   20.00 |
|             3 | Books    | Store   | 300 New Street | Cary    | 919-xxx-xxxx | John    |   30.00 |
|             4 | BooksEtc | Store   | 400 New Street | Durham  | 919-xxx-xxxx | John    |   40.00 |
+---------------+----------+---------+---------------+---------+--------------+---------+---------+
```

```
CREATE TABLE Orders (
    OrderID INT,
    DistAccountNum INT,
    PublicationID INT NOT NULL,
    NumCopies INT NOT NULL,
    ProduceByDate DATE NOT NULL,
    Price DECIMAL(8,2) NOT NULL,
    ShippingCosts DECIMAL(8,2) NOT NULL,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (DistAccountNum) REFERENCES Distributors(DistAccountNum) ON
UPDATE CASCADE ON DELETE SET NULL,
    FOREIGN KEY (PublicationID) REFERENCES Publication(PublicationID) ON
UPDATE CASCADE
  );
> SELECT * FROM Orders;
+---------+---------------+---------------+-----------+---------------+-------+---------------+
| OrderID | DistAccountNum | PublicationID | NumCopies | ProduceByDate | Price | ShippingCosts |
+---------+---------------+---------------+-----------+---------------+-------+---------------+
|       1 |             1 |             2 |         1 | 2022-05-10    |  5.00 |          1.00 |
|       2 |             4 |             2 |         2 | 2022-05-12    | 10.00 |          1.00 |
|       3 |             4 |             1 |         3 | 2022-05-23    | 30.00 |          1.00 |
|       4 |             4 |             4 |         4 | 2022-05-25    | 40.00 |          1.00 |
+---------+---------------+---------------+-----------+---------------+-------+---------------+
```

Invoices(InvoiceID, DistAccountNum, Amount, BillingDate, PaymentDate)

```
CREATE TABLE Invoices (
    InvoiceID INT,
    DistAccountNum INT,
    Amount DECIMAL(8,2) NOT NULL,
    BillingDate DATE NOT NULL,
    PaymentDate DATE,
    PRIMARY KEY (InvoiceID),
    FOREIGN KEY (DistAccountNum) REFERENCES Distributors(DistAccountNum) ON
UPDATE CASCADE ON DELETE SET NULL
  );
> SELECT * FROM Invoices;
+-----------+----------------+--------+-------------+-------------+
| InvoiceID | DistAccountNum | Amount | BillingDate | PaymentDate |
+-----------+----------------+--------+-------------+-------------+
|         1 |              3 |   5.15 | 2022-05-25  | NULL        |
|         2 |              3 |   5.15 | 2022-05-25  | 2022-08-25  |
|         3 |              3 |   5.15 | 2022-05-25  | 2022-08-25  |
|         4 |              3 |   5.15 | 2022-05-25  | NULL        |
+-----------+----------------+--------+-------------+-------------+
```

4. SQL Queries

4.1 Operations as SQL Statements

Editing and Publishing:

Assign Editor to Publication

```
> INSERT INTO Edits VALUES (9, 1);
Query OK, 1 row affected (0.01 sec)
```

View Editor Responsibilities

```
> SELECT * FROM Publication WHERE PublicationID IN
    -> (SELECT PublicationID FROM Edits WHERE EmpID = 1);
+---------------+-------+------+-----------------+
| PublicationID | Title | Type | Topic           |
+---------------+-------+------+-----------------+
|             9 | Dune  | Book | Science Fiction |
+---------------+-------+------+-----------------+
```

Add Editor

```
> INSERT INTO Employees VALUES (1, 'Andrew Abate', 'staff', true);
Query OK, 1 row affected (0.00 sec)

> INSERT INTO Editors VALUES (1);
Query OK, 1 row affected (0.01 sec)
```

Update Editor

```
> UPDATE Employees SET Active = false WHERE EmpID = 1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Delete Editor

```
> DELETE FROM Employees WHERE EmpID = 1;
Query OK, 1 row affected (0.00 sec)
```

Add Author

```
> INSERT INTO Employees VALUES (2, 'Morgan Chapman', 'contractor', true);
Query OK, 1 row affected (0.01 sec)

> INSERT INTO Authors VALUES (2);
Query OK, 1 row affected (0.01 sec)
```

Update Author

```
> UPDATE Employees SET Type = 'staff' WHERE EmpID = 2;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Delete Author

```
> DELETE FROM Employees WHERE EmpID = 2;
Query OK, 1 row affected (0.00 sec)
```

Production:

Enter New Book

```
> INSERT INTO Publication VALUES (9, 'Dune', 'Book', 'Science Fiction');
Query OK, 1 row affected (0.00 sec)

> INSERT INTO Books VALUES (9, 1, '0143111582', '2016-10-25');
Query OK, 1 row affected (0.00 sec)

> INSERT INTO WritesBook VALUES (9, 9);
Query OK, 1 row affected (0.01 sec)
Records: 1  Duplicates: 0  Warnings: 0
```

Update Book information

```
> UPDATE Books SET ISBN = '044100590X', PublicationDate = '1999-10-01' WHERE
PublicationID = 9;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Delete Book

```
> DELETE FROM Publication WHERE PublicationID = 9;
Query OK, 1 row affected (0.00 sec)
```

Enter New Periodical Issue

```
> INSERT INTO Publication VALUES (10, 'New Yorker', 'Magazine', 'News');
Query OK, 1 row affected (0.01 sec)

> INSERT INTO Issues VALUES (10, 'February-28', '2022-02-28');
Query OK, 1 row affected (0.00 sec)
```

Update Issue Information

```
> UPDATE Issues SET IssueDate = '2022-02-27', IssueTitle = 'February-27' WHERE
PublicationID = 10;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Delete Issue

```
> DELETE FROM Publication WHERE PublicationID = 10;
Query OK, 1 row affected (0.02 sec)
```

Add Chapter to Book

```
> INSERT INTO Chapters VALUES (9, 'Chapter 1');
Query OK, 1 row affected (0.01 sec)
```

Update Chapter Information

```
> UPDATE Chapters SET ChapterTitle = 'Chapter 1 - Introduction' WHERE PublicationID
= 9 AND ChapterTitle = 'Chapter 1';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Add Article to Issue

```
> INSERT INTO Articles VALUES (10, 'Book Review', 'Books', NULL);
Query OK, 1 row affected (0.00 sec)

> INSERT INTO WritesArticle VALUES (10, 'Book Review', 9);
Query OK, 1 row affected (0.01 sec)
Records: 1  Duplicates: 0  Warnings: 0
```

Update Issue Article

```
> UPDATE Articles SET ArticleTitle = 'Non-Fiction Book Review' WHERE PublicationID
= 10 AND ArticleTitle = 'Book Review';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

> UPDATE WritesArticle SET EmpID = 10 WHERE PublicationID = 10 AND ArticleTitle =
'Non-Fiction Book Review';
Query OK, 0 rows affected (0.01 sec)
Rows matched: 1  Changed: 0  Warnings: 0
```

Add Article Text

```
> UPDATE Articles SET ArticleText = 'nonfic_book_review_draft.txt' WHERE
```

```
PublicationID = 10 AND ArticleTitle = 'Non-Fiction Book Review';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Update Article Text

```
> UPDATE Articles SET ArticleText = 'nonfic_book_review_final.txt' WHERE
PublicationID = 10 AND ArticleTitle = 'Non-Fiction Book Review';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Get Book Information by author, publication date, and topic

```
> SELECT * FROM Publication NATURAL JOIN Books WHERE PublicationID IN
    -> (SELECT PublicationID FROM WritesBook WHERE EmpID IN (SELECT EmpID FROM Employees
WHERE Name = 'Frank Herbert')) AND PublicationDate = '1999-10-01' AND Topic = 'Science
Fiction';
+---------------+-------+------+-----------------+---------------+------------+-----------------+
| PublicationID | Title | Type | Topic           | EditionNumber | ISBN       | PublicationDate |
+---------------+-------+------+-----------------+---------------+------------+-----------------+
|             9 | Dune  | Book | Science Fiction | 1             | 044100590X | 1999-10-01      |
+---------------+-------+------+-----------------+---------------+------------+-----------------+
```

Get Book Information by author

```
> SELECT * FROM Publication NATURAL JOIN Books WHERE PublicationID IN
    -> (SELECT PublicationID FROM WritesBook WHERE EmpID IN (SELECT EmpID FROM Employees
WHERE Name = 'Frank Herbert'));
+---------------+------------------+------+-----------------+---------------+------------+-----------------+
| PublicationID | Title            | Type | Topic           | EditionNumber | ISBN       | PublicationDate |
+---------------+------------------+------+-----------------+---------------+------------+-----------------+
|             9 | Dune             | Book | Science Fiction | 1             | 044100590X | 1999-10-01      |
|            11 | Children of Dune | Book | Science Fiction | 1             | 0593098242 | 2019-06-04      |
+---------------+------------------+------+-----------------+---------------+------------+-----------------+
```

Get Book Information by publication date

```
> SELECT * FROM Publication NATURAL JOIN Books WHERE PublicationDate = '1999-10-01';
+---------------+-------+------+-----------------+---------------+------------+-----------------+
| PublicationID | Title | Type | Topic           | EditionNumber | ISBN       | PublicationDate |
+---------------+-------+------+-----------------+---------------+------------+-----------------+
|             9 | Dune  | Book | Science Fiction | 1             | 044100590X | 1999-10-01      |
+---------------+-------+------+-----------------+---------------+------------+-----------------+
```

Get Book Information by topic

```
> SELECT * FROM Publication NATURAL JOIN Books WHERE Topic = 'Science Fiction';
+---------------+------------------+------+-----------------+---------------+------------+-----------------+
| PublicationID | Title            | Type | Topic           | EditionNumber | ISBN       | PublicationDate |
+---------------+------------------+------+-----------------+---------------+------------+-----------------+
|             9 | Dune             | Book | Science Fiction | 1             | 044100590X | 1999-10-01      |
|            11 | Children of Dune | Book | Science Fiction | 1             | 0593098242 | 2019-06-04      |
+---------------+------------------+------+-----------------+---------------+------------+-----------------+
```

Get Article Information by author, creation date, and topic

```
> SELECT * FROM Issues NATURAL JOIN Articles WHERE (PublicationID, ArticleTitle) IN (SELECT
PublicationID, ArticleTitle FROM WritesArticle WHERE EmpID IN (SELECT EmpID FROM Employees
WHERE Name = 'Mary Brooklyn')) AND IssueDate = '2022-02-27' AND ArticleTopic = 'Books';
+---------------+-------------+------------+------------------------+-------------+------------------------------+
| PublicationID | IssueTitle  | IssueDate  | ArticleTitle           | ArticleTopic | ArticleText                 |
+---------------+-------------+------------+------------------------+-------------+------------------------------+
|            10 | February-27 | 2022-02-27 | Non-Fiction Book Review | Books       | nonfic_book_review_final.txt |
+---------------+-------------+------------+------------------------+-------------+------------------------------+
```

Get Article Information by author

```
> SELECT * FROM Issues NATURAL JOIN Articles WHERE (PublicationID, ArticleTitle) IN (SELECT
PublicationID, ArticleTitle FROM WritesArticle WHERE EmpID IN (SELECT EmpID FROM Employees
WHERE Name = 'Mary Brooklyn'));
+---------------+-------------+------------+------------------------+-------------+------------------------------+
| PublicationID | IssueTitle  | IssueDate  | ArticleTitle           | ArticleTopic | ArticleText                 |
+---------------+-------------+------------+------------------------+-------------+------------------------------+
|            10 | February-27 | 2022-02-27 | Best Books of 2022     | Books       | NULL                         |
|            10 | February-27 | 2022-02-27 | Non-Fiction Book Review | Books       | nonfic_book_review_final.txt |
+---------------+-------------+------------+------------------------+-------------+------------------------------+
```

Get Article Information by creation date

```
> SELECT * FROM Issues NATURAL JOIN Articles WHERE IssueDate = '2022-02-27';
+---------------+-------------+------------+------------------------+-------------+------------------------------+
| PublicationID | IssueTitle  | IssueDate  | ArticleTitle           | ArticleTopic | ArticleText                 |
+---------------+-------------+------------+------------------------+-------------+------------------------------+
|            10 | February-27 | 2022-02-27 | Best Books of 2022     | Books       | NULL                         |
|            10 | February-27 | 2022-02-27 | Non-Fiction Book Review | Books       | nonfic_book_review_final.txt |
+---------------+-------------+------------+------------------------+-------------+------------------------------+
```

Get Article Information by topic

```
> SELECT * FROM Issues NATURAL JOIN Articles WHERE ArticleTopic = 'Books';
+---------------+-------------+------------+------------------------+-------------+------------------------------+
| PublicationID | IssueTitle  | IssueDate  | ArticleTitle           | ArticleTopic | ArticleText                 |
+---------------+-------------+------------+------------------------+-------------+------------------------------+
|            10 | February-27 | 2022-02-27 | Best Books of 2022     | Books       | NULL                         |
|            10 | February-27 | 2022-02-27 | Non-Fiction Book Review | Books       | nonfic_book_review_final.txt |
+---------------+-------------+------------+------------------------+-------------+------------------------------+
```

Make Payment to Employee

```
> INSERT INTO Payments VALUES (1234, 3, 1000.00, '2022-03-01', NULL);
Query OK, 1 row affected (0.00 sec)
```

Update Payment to Employee

```
> UPDATE Payments SET ClaimDate = '2022-03-07' WHERE CheckNumber = 1234 AND EmpID =
3;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Track Payment to Employee

```
> SELECT * FROM Payments WHERE CheckNumber = 1234 AND EmpID = 3;
+-------------+-------+---------+------------+------------+
| CheckNumber | EmpID | Amount  | SubmitDate | ClaimDate  |
+-------------+-------+---------+------------+------------+
|        1234 |     3 | 1000.00 | 2022-03-01 | 2022-03-07 |
+-------------+-------+---------+------------+------------+
```

Distribution:

Add Distributor

```
> INSERT INTO Distributors VALUES (5, 'ReadThis','Library', '500 New Street',
'Raleigh', '919-xxx-xxxx',  'Jane', 1.00);
Query OK, 1 row affected (0.00 sec)
```

Delete Distributor

```
> DELETE FROM Distributors WHERE DistAccountNum = 5;
Query OK, 1 row affected (0.00 sec)
```

Update Distributor: Change the Contact attribute of a distributor specified by Name.

```
UPDATE Distributors SET Contact = 'Jane' WHERE Name = 'Library1';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Place Publication Order

```
> INSERT INTO Orders VALUES (5, 3,3, 500, '2022-12-12', 5-15, 1.00);
Query OK, 1 row affected (0.00 sec)
```

Bill Distributor: Create a new Invoice row with the Amount equal to the sum of Price column from Orders placed in the last month for a given distributor.

```
> INSERT INTO Invoices VALUES (8, 1, (select sum(Price) FROM Orders WHERE
DistAccountNum = 4 AND ProduceByDate=>'2022-05-01' AND ProduceByDate<'2022-06-1'),
'2022-06-01',NULL);
Query OK, 1 row affected (0.00 sec)
```

Update Invoice Payment Status: by changing the PaymentDate of a given InvoiceID

```
> UPDATE Invoices SET PaymentDate = '2022-03-08' WHERE InvoiceID = 1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Change Distributor Balance

```
> UPDATE Distributors SET Balance = 77.51 WHERE DistAccountNum = 4;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Reports:

Generate Report: number and total price of copies of each publication bought per distributor per Time period specified.

```
> SELECT Name AS Distributor,DistAccountNum AS Acc_Num , p.PublicationID, p.Title,
SUM(NumCopies), SUM(Price) AS TotalPrice
FROM Publication p, Distributors d NATURAL JOIN Orders o
WHERE p.PublicationID = o.PublicationID AND ProduceByDate >= '2022-04-01' AND
ProduceByDate <= '2022-04-28'
GROUP BY DistAccountNum, p.PublicationID;

+-------------+---------+--------------+--------------------------+----------------+------------+
| Distributor | Acc_Num | PublicationID | Title                    | SUM(NumCopies) | TotalPrice |
+-------------+---------+--------------+--------------------------+----------------+------------+
| Library1    |       1 |            3 | People                   |              1 |       5.15 |
| Books       |       3 |            1 | Don Quixote              |              4 |       5.15 |
| Books       |       3 |            4 | The Count of Monte Cristo |              4 |       5.15 |
| BooksEtc    |       4 |            1 | Don Quixote              |              3 |       5.15 |
| BooksEtc    |       4 |            2 | People                   |              2 |       5.15 |
+-------------+---------+--------------+--------------------------+----------------+------------+
```

Total Monthly Publications Sold: Total price and number of copies of a publication sold for a given Month

```
> SELECT PublicationID, Title, SUM(NumCopies) AS Total_Copies, SUM(Price) AS
Total_Price
FROM Publication p NATURAL JOIN Orders o
WHERE ProduceByDate >= '2022-04-01' AND ProduceByDate <= '2022-04-28' AND
PublicationID= 1
GROUP BY PublicationID;

+---------------+-------------+-----------------------+-----------------------+
| PublicationID | Title       | Total_Copies_For_Month | Total_Price_For_Month |
+---------------+-------------+-----------------------+-----------------------+
```

```
|             1 | Don Quixote |                       7 |                  10.30 |
+---------------+-------------+-------------------------+------------------------+
```

Total Monthly Publication Revenue: total monthly revenue of the publishing house

```
> SELECT SUM(Price) AS Total_Revenue_For_Month
FROM Orders
WHERE ProduceByDate >= '2022-04-01' AND ProduceByDate <= '2022-04-28';

+------------------------+
| Total_Revenue_For_Month |
+------------------------+
|                  25.75 |
+------------------------+
```

Total Monthly Expenses: total expenses of the publishing house depending on the type of expense (shipping cost, salaries, Total)

```
> SELECT SUM(ShippingCosts) AS Total_ShippingCosts_For_Month
FROM Orders
WHERE ProduceByDate >= '2022-04-01' AND ProduceByDate <= '2022-04-28';
+------------------------------+
| Total_ShippingCosts_For_Month |
+------------------------------+
|                         5.00 |
+------------------------------+

> SELECT SUM(Amount) AS Total_EmployeeCosts_For_Month
FROM Payments
WHERE SubmitDate >= '2022-01-01' AND SubmitDate <= '2022-01-31';
+------------------------------+
| Total_EmployeeCosts_For_Month |
+------------------------------+
|                      3000.00 |
+------------------------------+
```

Total Distributors

```
> SELECT COUNT(DistAccountNum) AS Total_Distributors
FROM Distributors;

+--------------------+
| Total_Distributors |
+--------------------+
|                  4 |
+--------------------+
```

Total City Revenue

```
> SELECT SUM(Price) AS Total_Revenue_For_City
FROM Orders o, Distributors d
WHERE o.DistAccountNum = d.DistAccountNum AND City = 'Durham';

+------------------------+
| Total_Revenue_For_City |
+------------------------+
|                  40.60 |
+------------------------+
```

Total Distributor Revenue

```
> SELECT SUM(Price) AS Total_Revenue_For_Distributor
FROM Orders o, Distributors d
WHERE o.DistAccountNum = d.DistAccountNum AND d.DistAccountNum = 1;

+-------------------------------+
| Total_Revenue_For_Distributor |
+-------------------------------+
|                         20.30 |
+-------------------------------+
```

Total Address Revenue

```
> SELECT SUM(Price) AS Total_Revenue_For_Address
FROM Orders o, Distributors d
WHERE o.DistAccountNum = d.DistAccountNum AND d.Address = '300 New Street';

+---------------------------+
| Total_Revenue_For_Address |
+---------------------------+
|                     25.45 |
+---------------------------+
```

Total Payments By Time and Type of Employee

Editors VS Authors: For a Given Month

```
> SELECT SUM(Amount) AS Total_Editor_Pay
FROM Payments p, Employees e, Editors ed
WHERE p.EmpID = e.EmpID AND p.EmpID = ed.EmpID AND SubmitDate >= '2022-02-01' AND
SubmitDate <= '2022-02-28';
+------------------+
| Total_Editor_Pay |
+------------------+
|          1900.00 |
```

```
+------------------+

> SELECT SUM(Amount) AS Total_Author_Pay
FROM Payments p, Employees e, Authors a
WHERE p.EmpID = e.EmpID AND p.EmpID = a.EmpID AND SubmitDate >= '2022-02-01' AND
SubmitDate <= '2022-02-28';
+------------------+
| Total_Author_Pay |
+------------------+
|          2100.00 |
+------------------+
```

Invited VS Staff: For a Given Month

```
> SELECT SUM(Amount) AS Total_Staff_Pay
FROM Payments p, Employees e
WHERE p.EmpID = e.EmpID AND Type = 'Staff' AND SubmitDate >= '2022-02-01' AND
SubmitDate <= '2022-02-28';
+-----------------+
| Total_Staff_Pay |
+-----------------+
|         2100.00 |
+-----------------+
```

Authors: For a Given Time Length

```
> SELECT '2022-01-01 to 2022-01-31' AS PayPeriod, SUM(Amount) AS Total_Editor_Pay
FROM Payments p, Employees e, Authors a
WHERE p.EmpID = e.EmpID AND p.EmpID = a.EmpID AND SubmitDate >= '2022-01-01' AND
SubmitDate <= '2022-02-28';
+--------------------------+------------------+
| PayPeriod                | Total_Editor_Pay |
+--------------------------+------------------+
| 2022-01-01 to 2022-01-31 |          5100.00 |
+--------------------------+------------------+
```

Total Payments By Type of Employee and Work Type

Editor Pay: for Invited and Staff

```
> SELECT '2022-02-01 to 2022-02-28' AS PayPeriod, Type, SUM(Amount) AS
Total_Pay_For_WorkType
FROM Payments p, Employees e, Editors ed, Edits et
WHERE p.EmpID = e.EmpID AND p.EmpID = ed.EmpID AND p.EmpID = et.EmpID AND
SubmitDate >= '2022-02-01' AND SubmitDate <= '2022-02-28'
GROUP BY Type;
```

```
+--------------------------+---------+------------------------+
| PayPeriod                | Type    | Total_Pay_For_WorkType |
+--------------------------+---------+------------------------+
| 2022-02-01 to 2022-02-28 | Invited |                1100.00 |
| 2022-02-01 to 2022-02-28 | Staff   |                 800.00 |
+--------------------------+---------+------------------------+
```

Author Pay: for Invited and Staff

```
> SELECT '2022-02-01 to 2022-02-28' AS PayPeriod, p.EmpID, Type, Amount AS
Total_Pay_For_WorkType
FROM Payments p NATURAL JOIN Employees e
WHERE SubmitDate >= '2022-02-01' AND SubmitDate <= '2022-02-28' AND p.EmpID IN
(SELECT EmpID from WritesArticle);

+--------------------------+-------+---------+------------------------+
| PayPeriod                | EmpID | Type    | Total_Pay_For_WorkType |
+--------------------------+-------+---------+------------------------+
| 2022-02-01 to 2022-02-28 |     1 | Staff   |                 100.00 |
| 2022-02-01 to 2022-02-28 |     2 | Staff   |                1200.00 |
| 2022-02-01 to 2022-02-28 |     4 | Invited |                 800.00 |
+--------------------------+-------+---------+------------------------+
```

4.2 SQL Execution Plans and Index Creation

1. Get Book Information by Publication Date

```
> EXPLAIN SELECT * FROM Publication NATURAL JOIN Books
    -> WHERE PublicationDate = '2003-02-25';
+------+-------------+-------------+-------+---------------+---------+---------+--------------------------+------+-------------+
| id   | select_type | table       | type  | possible_keys | key     | key_len | ref                      | rows | Extra       |
+------+-------------+-------------+-------+---------------+---------+---------+--------------------------+------+-------------+
|    1 | SIMPLE      | Books       | ALL   | PRIMARY       | NULL    | NULL    | NULL                     |    4 | Using where |
|    1 | SIMPLE      | Publication | eq_ref | PRIMARY      | PRIMARY | 4       | aabate.Books.PublicationID |    1 |             |
+------+-------------+-------------+-------+---------------+---------+---------+--------------------------+------+-------------+
2 rows in set (0.01 sec)

> CREATE INDEX pubDateIndex ON Books(PublicationDate);
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0

> EXPLAIN SELECT * FROM Publication NATURAL JOIN Books
    -> WHERE PublicationDate = '2003-02-25';
+------+-------------+-------------+-------+----------------------+--------------+---------+--------------------------+------+-------+
| id   | select_type | table       | type  | possible_keys        | key          | key_len | ref                      | rows | Extra |
+------+-------------+-------------+-------+----------------------+--------------+---------+--------------------------+------+-------+
|    1 | SIMPLE      | Books       | ref   | PRIMARY,pubDateIndex | pubDateIndex | 3       | const                    |    1 |       |
|    1 | SIMPLE      | Publication | eq_ref | PRIMARY             | PRIMARY      | 4       | aabate.Books.PublicationID |    1 |       |
+------+-------------+-------------+-------+----------------------+--------------+---------+--------------------------+------+-------+
2 rows in set (0.00 sec)
```

2. Get Article Information by Topic

```
> EXPLAIN SELECT * FROM Issues NATURAL JOIN Articles WHERE ArticleTopic = 'Books';
+------+-------------+----------+------+---------------+------+---------+------+------+-----------------------------------------------+
| id   | select_type | table    | type | possible_keys | key  | key_len | ref  | rows | Extra                                         |
+------+-------------+----------+------+---------------+------+---------+------+------+-----------------------------------------------+
|    1 | SIMPLE      | Issues   | ALL  | PRIMARY       | NULL | NULL    | NULL |    5 |                                               |
|    1 | SIMPLE      | Articles | ALL  | PRIMARY       | NULL | NULL    | NULL |    7 | Using where; Using join buffer (flat, BNL join) |
+------+-------------+----------+------+---------------+------+---------+------+------+-----------------------------------------------+
2 rows in set (0.00 sec)

> CREATE INDEX ArticleTopicIndex ON Articles(ArticleTopic);
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0

> EXPLAIN SELECT * FROM Issues NATURAL JOIN Articles WHERE ArticleTopic = 'Books';
+------+-------------+----------+--------+-------------------------+-----------------+---------+-------------------------------+------+-----------------------+
| id   | select_type | table    | type   | possible_keys           | key             | key_len | ref                           | rows | Extra                 |
+------+-------------+----------+--------+-------------------------+-----------------+---------+-------------------------------+------+-----------------------+
|    1 | SIMPLE      | Articles | ref    | PRIMARY,ArticleTopicIndex | ArticleTopicIndex | 130     | const                         |    2 | Using index condition |
|    1 | SIMPLE      | Issues   | eq_ref | PRIMARY                 | PRIMARY         | 4       | mchapma6.Articles.PublicationID |    1 |                       |
+------+-------------+----------+--------+-------------------------+-----------------+---------+-------------------------------+------+-----------------------+
2 rows in set (0.00 sec)
```

4.3 SQL Query Correctness

Query1: **Get Book Information by topic**

```
> SELECT * FROM Publication NATURAL JOIN Books WHERE Topic = 'Science Fiction';
+---------------+------------------+------+-----------------+---------------+------------+-----------------+
| PublicationID | Title            | Type | Topic           | EditionNumber | ISBN       | PublicationDate |
+---------------+------------------+------+-----------------+---------------+------------+-----------------+
|             9 | Dune             | Book | Science Fiction | 1             | 044100590X | 1999-10-01      |
|            11 | Children of Dune | Book | Science Fiction | 1             | 0593098242 | 2019-06-04      |
+---------------+------------------+------+-----------------+---------------+------------+-----------------+
```

Relational algebra as a sequence

$$R1 = Publication \bowtie Books$$
$$R2 = \sigma_{topic="Science\ Fiction"}(R1)$$
$$R3 = \pi_{*}(R2)$$

Query validity

Consider Publication and Books tables:

```
> SELECT * FROM Publication;
```

```
+---------------+---------------------------+----------+----------------+
| PublicationID | Title                     | Type     | Topic          |
+---------------+---------------------------+----------+----------------+
|             1 | Don Quixote               | Novel    | Adventure      |
|             2 | People                    | Magazine | Celebrity News |
|             3 | People                    | Magazine | Celebrity News |
|             4 | The Count of Monte Cristo | Novel    | Adventure      |
+---------------+---------------------------+----------+----------------+

> SELECT * FROM Books;
+---------------+---------------+------------+-----------------+
| PublicationID | EditionNumber | ISBN       | PublicationDate |
+---------------+---------------+------------+-----------------+
|             1 | 1             | 0142437239 | 2003-02-25      |
|             4 | 1             | 0140449264 | 2003-05-27      |
|             7 | 1             | 0167120441 | 2022-01-31      |
|             8 | 2             | 0335009913 | 2022-02-26      |
+---------------+---------------+------------+-----------------+
```

Step 1. Consider tuples *p* and *b* from Publication and Books relations. The corresponding tables have a common attribute PublicationID, their natural joint exists and results in concatenating tuples from *p* and *b* with the same PublicationID value, dropping the duplicate attribute PublicationID. The resulting table R1 returns all publications that are books and contains the following attributes:

```
| PublicationID | Title | Type | Topic  | EditionNumber | ISBN  | PublicationDate |
```

Step 2. Applying selection Topic = Desired Topic (e.g., “Science Fiction”) results in a table R2 that only contains rows with the Desired Topic in the Topic column.

Step 3. Project All operation returns relations R2 as R3. The result contains all possible attributes (columns) available for a book publication (PublicationID, Title, Type, Topic, EditionNumber, ISBN, PublicationDate) with rows whose Type = Desired Topic. In other words, we found all the publications that are books with a desired topic. This is exactly what we were required to retrieve.

Query2: Total Monthly Publications Sold: Total price and number of copies of a publication sold for a given Month

```
> SELECT PublicationID, Title, SUM(NumCopies) AS Total_Copies_For_Month, SUM(Price)
AS Total_Price_For_Month
FROM Publication NATURAL JOIN Orders
WHERE ProduceByDate >= '2022-01-01' AND ProduceByDate <= '2022-01-31' AND
PublicationID= 1
GROUP BY PublicationID;
```

```
+---------------+-------------+------------------------+-----------------------+
| PublicationID | Title       | Total_Copies_For_Month | Total_Price_For_Month |
+---------------+-------------+------------------------+-----------------------+
|             1 | Don Quixote |                      7 |                 10.30 |
+---------------+-------------+------------------------+-----------------------+
```

$$\rho_{\text{PublicationID, Title, Total\_Copies\_For\_Month, Total\_Price\_For\_Month}}(\gamma_{\text{PublicationID, Title, SUM(NumCopies), SUM(Price)}}(\sigma_{\text{MONTH(ProduceByDate) = MONTH('2019-01-03') AND PublicationID= 1}}(\text{Publication} \bowtie \text{Orders})))$$

Suppose p is any tuple in the Publication Relation, and o is any tuple in the Orders relation, such that the Natural join exists where the PublicationID value is used to join. The joining of the relations will give all possible tuples o natural joined with all tuples p such that all orders of each publication will be shown. By looking at the returned join, we choose all tuples where PublicationID is of the specified integer and where the month of the ProduceByDate is also specified. The selected tuples are then grouped together by Publication and aggregation is done on them for the SUM of NumCopies and SUM of Price. This returns a single tuple with the attributes PublicationID, Title, SUM(NumCopies), SUM(Price). This tuple's Attributes are then renamed for clarity and now become PublicationID, Title, Total_Copies_For_Month, Total_Price_For_Month. The example above shows that we wanted the total price and number of copies sold for the publication with the ID of 1 for the month of February and that is what was returned.