CSC 540 DATABASE MANAGEMENT CONCEPTS AND SYSTEMS

WOLFCITY PUBLICATION HOUSE

Project Report 2

Team Members (Q):
Alisha Shahane (asshahan)
Shruti Kangle (sskangle)
Poorva Kulkarni (pnkulkar)

Assumptions:

a. Publications:
   i. Publications can either be books or periodic publications.
   ii. Books have chapters and periodic publications have articles.
   iii. Each publication is identified by its isbn.
   iv. Periodic publications have a type which indicate whether it is a magazine or a journal.
   v. All editions of a book or issues of a periodic publication have the same isbn.
   vi. Price is associated with every edition or issue.

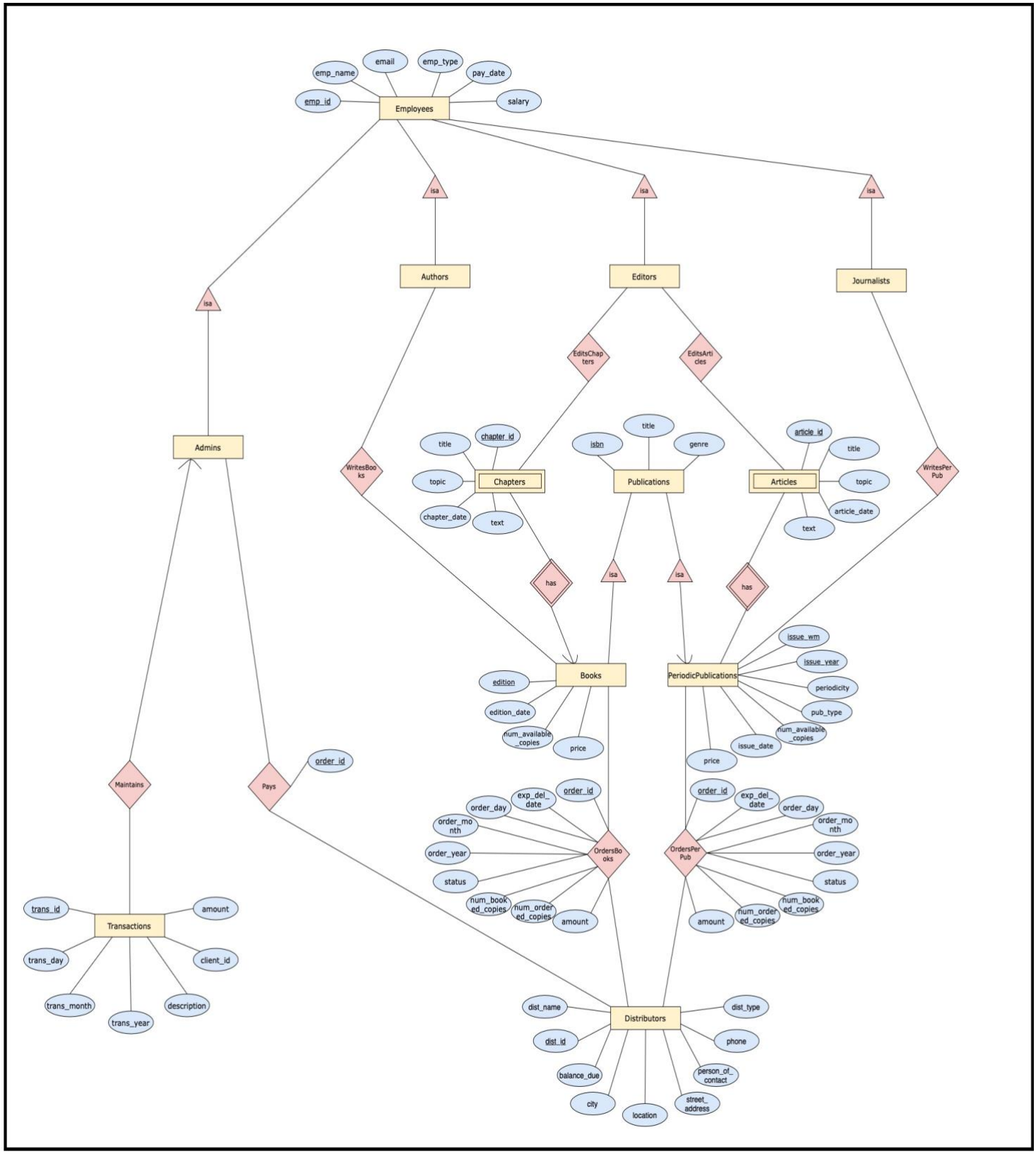
b. Employees:
   i. Every employee of the publishing house is either an admin, author, editor or a journalist.
   ii. Authors have access to all chapters of the book edition he/she has contributed to.
   iii. Similarly, journalists have access to all articles of the magazine/journal issue he/she has contributed to.
   iv. Editors are allowed to edit chapters and articles only of books or journals/magazines assigned to them.
   v. Pay_date denotes the last payment date of every employee.
   vi. Type indicates whether the author, editor or journalist is a staff or invited member. Type for admins of the publishing house is 'admin'.

c. Order:
   i. We maintain the status of every order, and it can be accepted, processing, completed or discarded.
   ii. A single order can contain only one issue/edition of a book/magazine. An order is completed only when all its requirements are met before the expected delivery date. After the expected delivery date is passed, the order can no longer be completed, and it is discarded.
   iii. Amount in an order is calculated based on the number of copies ordered and price of the publication.

d. Transactions:
   i. There are 3 kinds of transactions recorded namely, salary payments, shipping expenses and payments from distributors.
   ii. Shipping cost is equal to 10% of the total order amount. It is deducted from the publishing houses' treasury once an order is completed.

e. Miscellaneous:
   i. The system's scope is limited to a single publication house.
   ii. Any add/update/delete operation will take emp_id as one of the parameters and will be performed only if that is a permitted operation for that employee type.
   iii. No transaction can be deleted or updated. A transaction can only be inserted.
   iv. When a new distributor is added, balance_due is always 0 till he places an order.
   v. While generating reports, the total revenue takes only the income from Payment of Distributors into consideration.

Global ER:


Employees
emp_id
emp_name
email
emp_type
pay_date
salary
isa
Admins
Authors
Editors
Journalists
WritesBooks
EditsChapters
EditsArticles
WritesPerPub
Chapters
chapter_id
title
topic
chapter_date
text
Publications
isbn
title
genre
Articles
article_id
title
topic
article_date
text
has
Books
edition
edition_date
num_available_copies
price
PeriodicPublications
issue_wm
issue_year
periodicity
pub_type
num_available_copies
issue_date
price
Maintains
Pays
order_id
Transactions
trans_id
amount
trans_day
client_id
trans_month
description
trans_year
OrdersBooks
exp_del_date
order_day
order_month
order_year
status
num_booked_copies
num_ordered_copies
amount
OrdersPerPub
Distributors
dist_name
dist_id
dist_type
phone
balance_due
person_of_contact
city
location
street_address

Database Schemas:

1. Publications (isbn, title, genre)
   FDs:
   isbn -> title, genre

   The functional dependency holds because isbn uniquely determines title and genre. That is, LHS of the FD if a superkey. Thus, the relation is in 3NF.

2. PeriodicPublications (isbn, issue_wm, issue_year, periodicity, pub_type, num_available_copies, issue_date, price)
   FDs:
   isbn, issue_wm, issue_year. -> periodicity, pub_type, num_available_copies, issue_date, price

   The functional dependency holds because we need isbn, issue_wm, issue_year together to uniquely identify periodicity, pub_type, num_available_copies, issue_date. That is, LHS of the FD if a superkey. Thus, the relation is in 3NF.

3. Articles (isbn, issue_wm, issue_year, article_id, article_date, title, topic, text)
   FDs:
   isbn, issue_wm, issue_year, article_id -> article_date, title, topic, text

   The functional dependency holds because we need article_id along with isbn, issue_wm, issue_year to uniquely identify article_date, title, topic, text.
   Thus the FD is in 3NF as LHS is a superkey.

4. Books (isbn, edition, num_available_copies, edition_date, price)
   FDs:
   isbn, edition -> num_available_copies, edition_date, price

   The dependency holds true because in order to identify num_available_copies, price and edition_date we need isbn and edition.
   Thus, the FD is in 3NF.

5. Chapters (isbn, edition, chapter_id, chapter_date, title, topic, text)
   FDs:
   isbn, edition, chapter_id-> chapter_date, title, topic, text

   The functional dependency holds because we need chapter_id along with isbn, edition to uniquely identify chapter_date, title, topic, text.
   Thus the FD is in 3NF as LHS is a superkey.

6. Employees (emp_id, emp_name, email, salary, pay_date, emp_type)
   FDs:
   emp_id -> emp_name, email, salary, pay_date, emp_type

The functional dependency holds true as emp_id is enough to identify emp_name, salary, pay_date and emp_type.

7. Admins (emp_id)

   Since the relation has only one attribute, it is in 3NF.

8. Authors (emp_id)

   Since the relation has only one attribute, it is in 3NF.

9. Editors (emp_id)

   Since the relation has only one attribute, it is in 3NF.

10. Journalists (emp_id)

    Since the relation has only one attribute, it is in 3NF.

11. Transactions (trans_id, trans_day, trans_month, trans_year, client_id, amount, description)

    FDs:

    trans_id -> trans_day, trans_month, trans_year, client_id, amount, description

    The functional dependency holds true because trans_id is sufficient to uniquely identify trans_day, trans_month, trans_year, client_id, amount and description for that given transaction. Thus the FD is in 3NF as LHS is a superkey.

12. Maintains (emp_id, trans_id)

    FDs:

    emp_id -> emp_id
    trans_id -> trans_id

    The functional dependency holds true because emp_id determines emp_id and trans_id determines trans_id uniquely. Hence, both are keys of that relation. Thus, the FD is in 3NF as LHS is a superkey.

13. Distributors (dist_id, dist_name, dist_type, location, city, street_addr, phone, person_of_contact, balance_due)

    FDs:

    dist_id -> dist_name, dist_type, location, city, street_addr, phone, person_of_contact, balance_due

    The functional dependency holds true because dist_id is the key of the above relation which uniquely determines dist_name, dist_type, location, city, street_addr, phone, person_of_contact and balance_due. Thus, the FD is in 3NF as LHS is a superkey.

14. Pays (dist_id, emp_id, order_id)

FDs:

dist_id	->	dist_id
emp_id	->	emp_id
order_id	->	order_id

The functional dependency holds true because all the attributes are keys and thus FD is in 3NF as LHS is a superkey. None of the keys are individually capable of determining an entire row of the relation uniquely.

15. WritesBooks (emp_id, isbn, edition)

FDs:

isbn	->	isbn
emp_id	->	emp_id
edition	->	edition

The functional dependency holds true because all the attributes are keys and thus FD is in 3NF as LHS is a superkey. None of the keys are individually capable of determining an entire row of the relation uniquely.

16. WritesPerPubs (emp_id, isbn, issue_wm, issue_year)

FDs:

emp_id	->	emp_id
isbn	->	isbn
issue_wm	->	issue_wm
issue_year	->	issue_year

The functional dependency holds true because all the attributes are keys and thus FD is in 3NF as LHS is a superkey. None of the keys are individually capable of determining an entire row of the relation uniquely.

17. EditsChapters (emp_id, isbn, edition, chapter_id)

FDs:

emp_id	->	emp_id
esbn	->	isbn
edition	->	edition
chapter_id	->	chapter_id

The functional dependency holds true because all the attributes are keys and thus FD is in 3NF as LHS is a superkey. None of the keys are individually capable of determining an entire row of the relation uniquely.

18. EditsArticles (emp_id, isbn, issue_wm, issue_year, article_id)

FDs:

emp_id -> emp_id
isbn -> isbn
issue_wm -> issue_wm
issue_year -> issue_year
article_id -> article_id

The functional dependency holds true because all the attributes are keys and thus FD is in 3NF as LHS is a superkey. None of the keys are individually capable of determining an entire row of the relation uniquely.

19. OrdersBooks (order_id, isbn, edition, dist_id, order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount)

FDs:

order_id, isbn, edition, dist_id -> order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount

The functional dependency holds because we need order_id, isbn, edition, dist_id to uniquely identify order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount. Thus the FD is in 3NF as LHS is a superkey.

20. OrdersPerPub (order_id, isbn, issue_wm, issue_year, dist_id, order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount)

FDs:

order_id, isbn, issue_wm, issue_year, dist_id -> order_day, order_month, order_year, exp_del_date, status, amount num_ordered_copies, num_booked_copies,

The functional dependency holds because we need order_id, isbn, issue_wm, issue_year, dist_id to uniquely identify order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount. Thus the FD is in 3NF as LHS is a superkey.

Design Decisions:

1. Publications (isbn, title, genre)
    a. isbn: NOT NULL, PRIMARY KEY - Identifying the publication uniquely.
    b. title: NOT NULL : Name of publication cannot be NULL.
    c. genre: NULL - Genre of a publication could be NULL if it cannot be classified into one particular genre.

2. PeriodicPublications (isbn, issue_wm, issue_year, periodicity, pub_type, num_available_copies, issue_date, price)
    a. isbn: NOT NULL, PRIMARY KEY - Identifying the publication uniquely along with issue_wm and issue_year. This attribute is referenced from Publications.
    b. issue_wm: NOT NULL, PRIMARY KEY - It is the week number or month number of an issue. Can be determined using periodicity.
    c. issue_year: NOT NULL, PRIMARY KEY - Identifying the year in which the issue is published.
    d. periodicity: NOT NULL - A periodic publication must have a particular time interval for each release.
    e. pub_type: NOT NULL - Each periodic publication must be either a magazine or journal.
    f. num_available_copies: NOT NULL - The number of copies available in the publishing house.
    g. issue_date: NOT NULL - Identifying the date on which that particular issue is published.
    h. Price: NOT NULL - Denotes the price of every periodic publication.

3. Articles (isbn, issue_wm, issue_year, article_id, article_date, title, topic, text)
    a. isbn: NOT NULL, PRIMARY KEY - Identifying the publication uniquely along with issue_wm, issue_year, article_id. This attribute is referenced from PeriodicPublications.
    b. issue_wm: NOT NULL, PRIMARY KEY - It is the week number or month number of an issue. Can be determined using periodicity.
    c. issue_year: NOT NULL, PRIMARY KEY - Identifying the year in which the issue is published.
    d. article_id: NOT NULL, PRIMARY KEY - Cannot be NULL for identifying article (Unique identifier).
    e. article_date: NOT NULL - Identifies the date on which the article is published.
    f. title: NOT NULL - Denotes the title of an article which cannot be null.
    g. topic: NOT NULL - Denotes the topic of an article which cannot be null.
    h. text: NULL - Can be NULL if editor/journalist has not started work on that article.

4. Books (isbn, edition, num_available_copies, edition_date, price)
    a. isbn: NOT NULL, PRIMARY KEY - Identifying the publication uniquely along with edition. This attribute is referenced from Publications.
    b. edition: NOT NULL, PRIMARY KEY - Identifying the publication uniquely along with isbn (Unique identifier).

c. num_available_copies: NOT NULL - The number of copies available in the publishing house.
d. edition_date: NOT NULL - Identifying the date on which that particular book is published.
e. Price: NOT NULL - Denotes the price of every book.

5. Chapters (isbn, edition, chapter_id, chapter_date, title, topic, text)
    a. isbn: NOT NULL, PRIMARY KEY - Identifying the publication uniquely along with edition and chapter_id. This attribute is referenced from Books.
    b. edition: NOT NULL, PRIMARY KEY - Identifying the publication uniquely along with isbn.
    c. chapter_id: NOT NULL, PRIMARY KEY - Cannot be NULL for identifying chapter.
    d. chapter_date: NOT NULL - Identifies the date on which the chapter is published.
    e. title: NOT NULL - Denotes the title of a chapter which cannot be null.
    f. topic: NOT NULL - Denotes the topic of a chapter which cannot be null.
    g. text: NULL - Can be NULL if author/editor has not started work on that chapter.

6. Employees (emp_id, emp_name, email, salary, pay_date, emp_type)
    a. emp_id: NOT NULL, AUTO INCREMENT, PRIMARY KEY - Unique identifier for every employee.
    b. emp_name: NOT NULL - Employee must have a name for identification.
    c. email: NULL - This field can be null if the employee does not have an email address
    d. salary: NOT NULL - Every employee must have a salary.
    e. pay_date: NOT NULL - Every employee must have a salary pay date.
    f. emp_type: NOT NULL - Must identify between staff and invited employees.

7. Admins (emp_id)
    a. emp_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier for every admin, FOERIGN KEY referenced from Employees.

8. Authors (emp_id)
    a. emp_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier for every author, FOERIGN KEY referenced from Employees.

9. Editors (emp_id)
    a. emp_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier for every editor, FOERIGN KEY referenced from Employees

10. Journalists (emp_id)
    a. emp_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier for every journalist, FOERIGN KEY referenced from Employees

11. Transactions (trans_id, trans_day, trans_month, trans_year, client_id, amount, description)
    a. trans_id: NOT NULL, PRIMARY KEY, AUTO INCREMENT - Unique identifier
    b. trans_day: NOT NULL - Transaction takes place on a date, so cannot be NULL.
    c. Trans_month: NOT NULL - Transaction takes place on a date, so cannot be NULL.
    d. trans_year: NOT NULL - Transaction takes place on a date, so cannot be NULL.

e. client_id: NOT NULL - Identifies with whom the transaction was done.
f. amount: NOT NULL - Transaction always involves at least one amount.
g. description: NOT NULL - Identifies the kind of transaction.

12. Maintains (emp_id, trans_id)
    a. emp_id: NOT NULL, PRIMARY KEY - Unique identifier referenced from Admins
    b. trans_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Transactions.

13. Distributors (dist_id, dist_name, dist_type, location, city, street_addr, phone, person_of_contact, balance_due)
    a. dist_id: NOT NULL, AUTO INCREMENT, PRIMARY KEY - Unique identifier
    b. dist_name: NOT NULL - Distributor must have a name for identification.
    c. dist_type: NOT NULL - Distributor must have a type such as library or wholesaler, etc.
    d. location: NOT NULL
    e. city: NOT NULL - Distributor must have a city where he is located.
    f. street_Address: NOT NULL
    g. phone: NOT NULL - Distributor must have a phone for contact.
    h. person_of_contact: NULL - Identifies the person to contact.
    i. balance_due: NOT NULL - Indicates the amount which a distributor owes the publishing house.

14. Pays (dist_id, emp_id, order_id)
    a. dist_id: NOT NULL, PRIMARY KEY - Unique identifier referenced from Distributors.
    b. emp_id: NOT NULL, PRIMARY KEY - Unique identifier referenced from Admins.
    c. order_id: NOT NULL, PRIMARY KEY - Unique identifier for identifying the order for which a payment is being registered.

15. WritesBooks (emp_id, isbn, edition)
    a. emp_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Authors.
    b. isbn: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Books
    c. edition: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Books

16. WritesPerPubs (emp_id, isbn, issue_wm, issue_year)
    a. emp_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Journalists.
    b. issue_wm: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from PeriodicPublications.
    c. issue_year: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from PeriodicPublications.
    d. isbn: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from PeriodicPublications.

17. EditsChapters (emp_id, isbn, edition, chapter_id)
    a. emp_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Editors.
    b. isbn: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Chapters.
    c. edition: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Chapters.
    d. chapter_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Chapters.

18. EditsArticles (emp_id, isbn, issue_wm, issue_year, article_id)
    a. emp_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Editors
    b. isbn: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Articles
    c. issue_wm: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Articles
    d. issue_year: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Articles
    e. article_id: NOT NULL, ON DELETE CASCADE, PRIMARY KEY - Unique identifier referenced from Articles

19. OrdersBooks (order_id, isbn, edition, dist_id, order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount)
    a. order_id: NOT NULL, PRIMARY KEY, AUTO INCREMENT - Unique identifier for every order.
    b. isbn: NOT NULL, PRIMARY KEY - Unique identifier referenced from Books.
    c. edition: NOT NULL, PRIMARY KEY - Unique identifier referenced from Books.
    d. dist_id: NOT NULL, PRIMARY KEY - Unique identifier referenced from Distributors.
    e. order_day: NOT NULL - Date on which the order is placed.
    f. order_month: NOT NULL - Date on which the order is placed.
    g. order_year: NOT NULL - Date on which the order is placed.
    h. exp_del_Date: NOT NULL - Expected delivery date of the order.
    i. status: NOT NULL - Denotes the status of the order.
    j. num_ordered_copies: NOT NULL - Number of copies in the order
    k. num_booked_copies: NOT NULL - Number of copies booked for the order
    l. amount: NOT NULL - Total amount of the order

20. OrdersPerPubs (order_id, isbn, issue_wm, issue_year, dist_id, order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount)
    a. order_id: NOT NULL, PRIMARY KEY, AUTO INCREMENT - Unique identifier
    b. isbn: NOT NULL, PRIMARY KEY - Unique identifier referenced from PeriodicPublications.
    c. issue_wm: NOT NULL, PRIMARY KEY - Unique identifier referenced from PeriodicPublications.
    d. issue_year: NOT NULL, PRIMARY KEY - Unique identifier referenced from PeriodicPublications.
    e. dist_id: NOT NULL, PRIMARY KEY - Unique identifier referenced from Distributors.
    f. order_day: NOT NULL - Date on which the order is placed.
    g. order_month: NOT NULL - Date on which the order is placed.
    h. order_year: NOT NULL - Date on which the order is placed.
    i. exp_del_date: NOT NULL - Expected delivery date of the order.
    j. status: NOT NULL - Denotes the status of the order.
    k. num_ordered_copies: NOT NULL - Number of copies in the order
    l. num_booked_copies: NOT NULL - Number of copies booked for the order
    m. amount: NOT NULL - Total amount of the order

Table Creation

Create Table Queries

1. Publications

```
CREATE TABLE Publications(
isbn INT,
title VARCHAR(30) NOT NULL,
genre VARCHAR(30),
PRIMARY KEY(isbn)
);
```

2. PeriodicPublications

```
CREATE TABLE PeriodicPublications(
isbn INT,
issue_wm INT,
issue_year INT,
periodicity VARCHAR(30) NOT NULL,
pub_type VARCHAR(30) NOT NULL,
num_available_copies INT NOT NULL,
issue_date DATE NOT NULL,
price FLOAT NOT NULL,
PRIMARY KEY (isbn, issue_wm, issue_year),
FOREIGN KEY(isbn) REFERENCES Publications(isbn) ON DELETE CASCADE
);
```

3. Articles

```
CREATE TABLE Articles(
isbn INT,
issue_wm INT,
issue_year INT,
article_id INT,
article_date DATE NOT NULL,
title VARCHAR(30) NOT NULL,
topic VARCHAR(30) NOT NULL,
text VARCHAR(500),
PRIMARY KEY(isbn, issue_wm, issue_year, article_id),
FOREIGN KEY(isbn, issue_wm, issue_year) REFERENCES PeriodicPublications(isbn,
issue_wm, issue_year) ON DELETE CASCADE
);
```

4. Books

```
CREATE TABLE Books(
isbn INT,
edition INT,
num_available_copies INT NOT NULL,
edition_date DATE NOT NULL,
price FLOAT NOT NULL,
PRIMARY KEY (isbn, edition),
FOREIGN KEY(isbn) REFERENCES Publications(isbn) ON DELETE CASCADE
);
```

5. Chapters

```
CREATE TABLE Chapters(
isbn INT,
edition INT,
chapter_id INT,
chapter_date DATE NOT NULL,
title VARCHAR(30) NOT NULL,
topic VARCHAR(30) NOT NULL,
text VARCHAR(500),
PRIMARY KEY(isbn, edition, chapter_id),
FOREIGN KEY(isbn, edition) REFERENCES Books(isbn, edition) ON DELETE
CASCADE
);
```

6. Employees

```
CREATE TABLE Employees(
emp_id INT AUTO_INCREMENT,
emp_name VARCHAR(50) NOT NULL,
email VARCHAR(30),
salary FLOAT NOT NULL,
pay_date DATE NOT NULL,
emp_type VARCHAR(30) NOT NULL,
PRIMARY KEY(emp_id)
);
```

7. Admins

```
CREATE TABLE Admins(
emp_id INT,
PRIMARY KEY(emp_id),
FOREIGN KEY(emp_id) REFERENCES Employees(emp_id) ON DELETE CASCADE
);
```

8. Authors

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

9. Editors

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

10. Journalists

```
CREATE TABLE Journalists(
emp_id INT,
PRIMARY KEY(emp_id),
FOREIGN KEY(emp_id) REFERENCES Employees(emp_id) ON DELETE CASCADE
);
```

11. Transactions

```
CREATE TABLE Transactions(
trans_id INT AUTO_INCREMENT,
trans_day INT NOT NULL,
trans_month INT NOT NULL,
trans_year INT NOT NULL,
client_id INT NOT NULL,
amount FLOAT NOT NULL,
description VARCHAR(30) NOT NULL,
PRIMARY KEY(trans_id)
);
```

12. Maintains

```
CREATE TABLE Maintains(
trans_id INT,
emp_id INT,
PRIMARY KEY(trans_id, emp_id),
FOREIGN KEY(emp_id) REFERENCES Admins(emp_id),
FOREIGN KEY(trans_id) REFERENCES Transactions(trans_id) ON DELETE CASCADE
);
```

13. Distributors

```
CREATE TABLE Distributors(
dist_id INT AUTO_INCREMENT,
dist_name VARCHAR(30) NOT NULL,
dist_type VARCHAR(30) NOT NULL,
location VARCHAR(30) NOT NULL,
city VARCHAR(30) NOT NULL,
street_addr VARCHAR(30) NOT NULL,
phone VARCHAR(30) NOT NULL,
person_of_contact VARCHAR(30),
balance_due FLOAT NOT NULL,
PRIMARY KEY(dist_id)
);
```

14. Pays

```
CREATE TABLE Pays(
dist_id INT,
emp_id INT,
order_id INT,
PRIMARY KEY(dist_id, emp_id, order_id),
FOREIGN KEY(emp_id) REFERENCES Admins(emp_id),
FOREIGN KEY(dist_id) REFERENCES Distributors(dist_id) ON DELETE CASCADE
);
```

15. WritesBooks

```
CREATE TABLE WritesBooks(
emp_id INT,
isbn INT,
edition INT,
PRIMARY KEY(emp_id, edition, isbn),
FOREIGN KEY(emp_id) REFERENCES Authors(emp_id) ON DELETE CASCADE,
FOREIGN KEY(isbn, edition) REFERENCES Books(isbn, edition) ON DELETE
CASCADE
 );
```

16. WritesPerPub

```
CREATE TABLE WritesPerPub(
emp_id INT,
isbn INT,
issue_wm INT,
issue_year INT,
PRIMARY KEY(emp_id, issue_wm, issue_year, isbn),
FOREIGN KEY(isbn, issue_wm, issue_year) REFERENCES PeriodicPublications(isbn,
issue_wm, issue_year) ON DELETE CASCADE,
FOREIGN KEY(emp_id) REFERENCES Journalists(emp_id) ON DELETE CASCADE
);
```

17. EditsChapters

```
CREATE TABLE EditsChapters(
emp_id INT,
isbn INT,
edition INT,
chapter_id INT,
PRIMARY KEY(emp_id, edition, isbn, chapter_id),
FOREIGN KEY(isbn, edition, chapter_id) REFERENCES Chapters(isbn, edition,
chapter_id) ON DELETE CASCADE,
FOREIGN KEY(emp_id) REFERENCES Editors(emp_id) ON DELETE CASCADE
);
```

18. EditsArticles

```
CREATE TABLE EditsArticles(
emp_id INT,
isbn INT,
issue_wm INT,
issue_year INT,
article_id INT,
PRIMARY KEY(emp_id, issue_wm, issue_year, isbn, article_id),
FOREIGN KEY(isbn, issue_wm, issue_year, article_id) REFERENCES Articles(isbn,
issue_wm, issue_year, article_id) ON DELETE CASCADE,
FOREIGN KEY(emp_id) REFERENCES Editors(emp_id) ON DELETE CASCADE
);
```

19. OrdersBooks

```
CREATE TABLE OrdersBooks(
order_id INT AUTO_INCREMENT,
isbn INT,
edition INT,
dist_id INT,
order_day INT NOT NULL,
order_month INT NOT NULL,
order_year INT NOT NULL,
exp_del_date DATE NOT NULL,
status VARCHAR(30) NOT NULL,
num_ordered_copies INT NOT NULL,
num_booked_copies INT NOT NULL,
amount FLOAT NOT NULL,
PRIMARY KEY(order_id, isbn, edition, dist_id),
FOREIGN KEY(isbn, edition) REFERENCES Books(isbn, edition),
FOREIGN KEY(dist_id) REFERENCES Distributors(dist_id)
)AUTO_INCREMENT=1;
```

20. OrdersPerPub

```
CREATE TABLE OrdersPerPub(
order_id INT AUTO_INCREMENT,
isbn INT,
issue_wm INT,
issue_year INT,
dist_id INT,
order_day INT NOT NULL,
order_month INT NOT NULL,
order_year INT NOT NULL,
exp_del_date DATE NOT NULL,
status VARCHAR(30) NOT NULL,
num_ordered_copies INT NOT NULL,
num_booked_copies INT NOT NULL,
amount FLOAT NOT NULL,
PRIMARY KEY(order_id, isbn, issue_wm, issue_year, dist_id),
FOREIGN KEY(isbn, issue_wm, issue_year) REFERENCES PeriodicPublications(isbn,
issue_wm, issue_year),
FOREIGN KEY(dist_id) REFERENCES Distributors(dist_id)
)AUTO_INCREMENT=501;
```

```
MariaDB [sskangle]> SHOW TABLES;
+----------------------+
| Tables_in_sskangle   |
+----------------------+
| Admins               |
| Articles             |
| Authors              |
| Books                |
| Chapters             |
| Distributors         |
| Editors              |
| EditsArticles        |
| EditsChapters        |
| Employees            |
| Journalists          |
| Maintains            |
| OrdersBooks          |
| OrdersPerPub         |
| Pays                 |
| PeriodicPublications |
| Publications         |
| Transactions         |
| WritesBooks          |
| WritesPerPub         |
+----------------------+
20 rows in set (0.00 sec)
```

Insert Table Queries

Publications

```
INSERT INTO Publications VALUES (100001, "The Da Vinci Code", "Mystery");
INSERT INTO Publications VALUES (100002, "Introduction to Algorithms", "Textbook");
INSERT INTO Publications VALUES (100003, "IEEE Explore", "Science and Technology");
INSERT INTO Publications VALUES (100004, "Springer", "Scientific Research");
INSERT INTO Publications VALUES (100005, "Becoming", "Autobiography");
```

```
MariaDB [sskangle]> SELECT * FROM Publications;
+--------+-----------------------------------+-----------------------+
| isbn   | title                             | genre                 |
+--------+-----------------------------------+-----------------------+
| 100001 | The Da Vinci Code                 | Mystery               |
| 100002 | Introduction to Algorithms        | Textbook              |
| 100003 | IEEE Explore                      | Science and Technology |
| 100004 | Springer                          | Scientific Research   |
| 100005 | Becoming                          | Autobiography         |
+--------+-----------------------------------+-----------------------+
5 rows in set (0.00 sec)
```

PeriodicPublications

```
INSERT INTO PeriodicPublications VALUES(100003, 03, 2020, "monthly", "journal", 100, "2019-09-05", 10.50);
INSERT INTO PeriodicPublications VALUES(100004, 09, 1999, "monthly", "journal", 500, "1999-09-05", 30.5);
INSERT INTO PeriodicPublications VALUES(100008, 09, 2019, "weekly", "magazine", 10, "2019-09-05", 13.54);
INSERT INTO PeriodicPublications VALUES(100010, 46, 2012, "weekly", "magazine", 45, "2012-07-28", 7.64);
INSERT INTO PeriodicPublications VALUES(100009, 27, 2013, "weekly", "magazine", 38, "2013-07-28", 20.03);
```

```
MariaDB [sskangle]> SELECT * FROM PeriodicPublications;
+--------+----------+------------+-------------+----------+----------------------+------------+-------+
| isbn   | issue_wm | issue_year | periodicity | pub_type | num_available_copies | issue_date | price |
+--------+----------+------------+-------------+----------+----------------------+------------+-------+
| 100003 |        3 |       2020 | monthly     | journal  |                  100 | 2019-09-05 |  10.5 |
| 100004 |        9 |       1999 | monthly     | journal  |                  500 | 1999-09-05 |  30.5 |
| 100008 |        9 |       2019 | weekly      | magazine |                   10 | 2019-09-05 | 13.54 |
| 100009 |       27 |       2013 | weekly      | magazine |                   38 | 2013-07-28 | 20.03 |
| 100010 |       46 |       2012 | weekly      | magazine |                   45 | 2012-07-28 |  7.64 |
+--------+----------+------------+-------------+----------+----------------------+------------+-------+
5 rows in set (0.00 sec)
```

Articles

```
INSERT INTO Articles VALUES(100003, 03, 2020, 3, "2019-12-12", "Fault Diagnosis", "Distributed Systems", "This is a static technique");
INSERT INTO Articles VALUES(100004, 09, 1999, 3, "1999-08-12", "Neural Networks", "Deep Learning", "Uses neural networks.");
INSERT INTO Articles VALUES(100008, 09, 2019, 3, "2019-08-07", "Shoes for summer", "Shoes", "Ballet flats are amazing.");
INSERT INTO Articles VALUES(100010, 46, 2012, 3, "2012-07-19", "Personalities", "People", "Person of the year.");
INSERT INTO Articles VALUES(100009, 27, 2013, 3, "2013-06-16", "Market Statistics", "Business", "Stock market looks good.");
```

MariaDB [sskangle]> SELECT * FROM Articles;

```
+--------+----------+------------+------------+--------------+------------------+------------------------+-----------------------------------
-------------------------------------------------------------------------------+
| isbn   | issue_wm | issue_year | article_id | article_date | title            | topic              | text            |
+--------+----------+------------+------------+--------------+------------------+------------------------+-----------------------------------
-------------------------------------------------------------------------------+
| 100003 |       3 |      2020 |       3 | 2019-12-12   | Fault Diagnosis| Distributed Systems  |This is a static technique|
| 100004 |       9 |      1999 |       3 | 1999-08-12   | Neural Networks   | Deep Learning          | Neural networks
are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. |
| 100008 |       9 |      2019 |       3 | 1999-08-12   | Shoes for summer  | Shoes      | Ballet flats are amazing.    |
| 100008 |       9 |      2019 |       4 | 2019-08-12   | Formal Wear       | Formal Shoes and Clothes | Dress up to
make an impression                                                                                                    |
| 100009 |      27 |      2013 |       3 | 2013-06-16   | Market Statistics | Business   | Stock market looks good.   |
| 100010 |      46 |      2012 |       3 | 2012-07-19   | Personalities     | People               | Person of the year.   |
+--------+----------+------------+------------+--------------+------------------+------------------------+-----------------------------------
-------------------------------------------------------------------------------+
6 rows in set (0.00 sec)
```

Books

```
INSERT INTO Books VALUES(100001, 1, 150, "2019-05-14", 30.4);
INSERT INTO Books VALUES(100002, 2, 10, "1996-05-14", 21.24);
INSERT INTO Books VALUES(100005, 1, 50, "2003-05-23", 56.6);
INSERT INTO Books VALUES(100006, 2, 15, "2008-10-07", 9.8);
INSERT INTO Books VALUES(100007, 1, 100, "2019-10-07", 10.5);
```

MariaDB [sskangle]> SELECT * FROM Books;

```
+--------+---------+---------------------+--------------+-------+
| isbn   | edition | num_available_copies | edition_date | price |
+--------+---------+---------------------+--------------+-------+
| 100001 |       1 |                 150 | 2019-05-14   |  30.4 |
| 100002 |       2 |                  10 | 1996-05-14   | 21.24 |
| 100005 |       1 |                  50 | 2003-05-23   |  56.6 |
| 100006 |       2 |                  15 | 2008-10-07   |   9.8 |
| 100007 |       1 |                 100 | 2019-10-07   |  10.5 |
+--------+---------+---------------------+--------------+-------+
5 rows in set (0.00 sec)
```

Chapter

```
INSERT INTO Chapters VALUES(100001, 1, 2, "2018-03-02", "Introduction", "topic1", "While in Paris on business.");
INSERT INTO Chapters VALUES(100002, 2, 3, "2017-06-15", "Sorting", "topic2", "Quick sort is a fast algorithm.");
INSERT INTO Chapters VALUES(100005, 1, 4, "2018-11-29", "Childhood", "topic3", "Happy childhood is essential.");
INSERT INTO Chapters VALUES(100006,2,6,"2019-01-25", "Weak Entity Sets", "topic4", "Denote by double border.");
INSERT INTO Chapters VALUES(100007,1,7,"2016-05-24","Robots in everyday life","topic5","AI has reached new
levels.");
```

MariaDB [sskangle]> SELECT * FROM Chapters;

```
+--------+---------+------------+--------------+-----------------------+----------------------+-----------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------+
| isbn   | edition | chapter_id | chapter_date | title                 | topic                | text                                    |
+--------+---------+------------+--------------+-----------------------+----------------------+-----------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------+
| 100001 |       1 |          2 | 2018-03-02   | Introduction          | topic1          | While in Paris on business.       |
```

```
| 100002 |      2 |        3 | 2017-06-15  | Sorting              | topic2              | Quick sort is a fast algorithm.  |
| 100005 |      1 |        4 | 2018-11-29  | Childhood            | topic3              | Happy childhood is essential.    |
| 100006 |      2 |        6 | 2019-01-25  | Weak Entity Sets     | topic4              | Denote by double border.    |
| 100007 |      1 |        7 | 2016-05-24  | Robots in everyday life | topic5           | AI has reached new levels.   |
+--------+---------+------------+--------------+-----------------------+-----------------------+-----------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------+
5 rows in set (0.00 sec)
```

Employees

```
INSERT INTO Employees(emp_id, emp_name, email, salary, pay_date, emp_type) VALUES( "Dan Brown",
"brown@abc.com", 115, "2020-02-29", "staff");
INSERT INTO Employees(emp_id, emp_name, email, salary, pay_date, emp_type) VALUES( "Thomas Cormen",
"cormen@abc.com", 150, "2020-02-29", "staff");
INSERT INTO Employees(emp_id, emp_name, email, salary, pay_date, emp_type) VALUES( "Steve Forbes",
"forbes@abc.com", 150, "2020-02-29", "staff");
INSERT INTO Employees(emp_name, email, salary, pay_date, emp_type) VALUES("Ronald Rivest ",
"rivest@abc.com", 115, "2019-11-30", "invited");
INSERT INTO Employees(emp_name, email, salary, pay_date, emp_type) VALUES("Michelle Obama",
"obama@abc.com", 115, "2020-01-31", "invited");
```

```
MariaDB [sskangle]> SELECT * FROM Employees;
+--------+---------------------+------------------+--------+------------+----------+
| emp_id | emp_name            | email            | salary | pay_date   | emp_type |
+--------+---------------------+------------------+--------+------------+----------+
|      1 | Dan Brown           | brown@abc.com    |    115 | 2020-02-29 | staff    |
|      2 | Thomas Cormen       | cormen@abc.com   |    150 | 2020-02-29 | staff    |
|      3 | Steve Forbes        | forbes@abc.com   |    150 | 2020-02-29 | staff    |
|      4 | Ronald Rivest       | rivest@abc.com   |    115 | 2019-11-30 | invited  |
|      5 | Michelle Obama      | obama@abc.com    |    115 | 2020-01-31 | invited  |
+--------+---------------------+------------------+--------+------------+----------+
5 rows in set (0.00 sec)
```

Admins

```
INSERT INTO Admins(emp_id) values(12);
INSERT INTO Admins(emp_id) values(13);
INSERT INTO Admins(emp_id) values(14);
```

```
MariaDB [sskangle]> SELECT * FROM Admins;
+--------+
| emp_id |
+--------+
|     12 |
|     13 |
|     14 |
+--------+
3 rows in set (0.00 sec)
```

Authors

```
INSERT INTO Authors(emp_id) values(6);
INSERT INTO Authors(emp_id) values(5);
INSERT INTO Authors(emp_id) values(4);
INSERT INTO Authors(emp_id) values(7);
```

MariaDB [sskangle]> SELECT * FROM Authors;
+--------+
| emp_id |
+--------+
| 4 |
| 5 |
| 6 |
| 7 |
+--------+
4 rows in set (0.01 sec)

Editors

INSERT INTO Editors(emp_id) values(8);
INSERT INTO Editors(emp_id) values(9);

MariaDB [sskangle]> SELECT * FROM Editors;
+--------+
| emp_id |
+--------+
| 8 |
| 9 |
+--------+
2 rows in set (0.00 sec)

Journalists

INSERT INTO Journalists(emp_id) values(3);
INSERT INTO Journalists(emp_id) values(10);
INSERT INTO Journalists(emp_id) values(11);

MariaDB [sskangle]> SELECT * FROM Journalists;
+--------+
| emp_id |
+--------+
| 3 |
| 10 |
| 11 |
+--------+
3 rows in set (0.00 sec)

Transactions

INSERT INTO Transactions(trans_day, trans_month, trans_year, client_id, amount, description) VALUES (29, 02, 2020, 12, -200, "Salary Paid");
INSERT INTO Transactions(trans_day, trans_month, trans_year, client_id, amount, description) VALUES (29, 02, 2020, 13, -200, "Salary Paid");
INSERT INTO Transactions(trans_day, trans_month, trans_year, client_id, amount, description) VALUES (29, 02, 2020, 14, -200, "Salary Paid");
INSERT INTO Transactions(trans_day, trans_month, trans_year, client_id, amount, description) VALUES (10, 03, 2020, 6, 2264, "Payment Received");
INSERT INTO Transactions(trans_day, trans_month, trans_year, client_id, amount, description) VALUES (10, 03, 2020, 6, -226.4, "Shipping Cost");

MariaDB [sskangle]> SELECT * FROM Transactions;

trans_id	trans_day	trans_month	trans_year	client_id	amount	description
1	29	2	2020	12	-100	Salary Paid
2	29	2	2020	13	-100	Salary Paid
3	29	2	2020	14	-100	Salary Paid
4	10	3	2020	6	2264	Payment Received
5	10	3	2020	6	-226.4	Shipping Cost

5 rows in set (0.00 sec)

Maintains

```
INSERT INTO Maintains values(1, 12);
INSERT INTO Maintains values(2, 13);
INSERT INTO Maintains values(3, 14);
INSERT INTO Maintains values(4, 12);
INSERT INTO Maintains values(5, 12);
```

MariaDB [sskangle]> SELECT * FROM Maintains;

trans_id	emp_id
1	12
2	13
3	14
4	12
5	12

5 rows in set (0.00 sec)

Distributors

```
INSERT INTO Distributors(dist_name, dist_type, location, city, street_addr, phone, person_of_contact, balance_due)
VALUES ("Pearson", "wholesaler", "Wake County", "Raleigh", "Avery Close", "9191234567", NULL, 0.0);
INSERT INTO Distributors(dist_name, dist_type, location, city, street_addr, phone, person_of_contact, balance_due)
VALUES ("Penguin", "wholesaler", "Prince George", "College Park", "Baltimore Avenue", "9191334567", NULL, 0.0);
INSERT INTO Distributors(dist_name, dist_type, location, city, street_addr, phone, person_of_contact, balance_due)
VALUES ("Hunt", "library", "Wake County", "Raleigh", "Partners Way", "9191434567", "Max Brown", 0.0);
INSERT INTO Distributors(dist_name, dist_type, location, city, street_addr, phone, person_of_contact, balance_due)
VALUES ("Hill", "library", "Prince George", "College Park", "Baltimore", "9191435567", "Jamie Brown", 0.0);
INSERT INTO Distributors(dist_name, dist_type, location, city, street_addr, phone, person_of_contact, balance_due)
VALUES ("Mason's Books", "bookstore", "Ashville", "Raleigh", "Varsity Drive", "9191235567", "NULL", 0.0);
```

MariaDB [sskangle]> SELECT * FROM Distributors;

dist_id	dist_name	dist_type	location	city	street_addr	phone	person_of_contact	balance_due
1	Pearson	wholesaler	Wake County	Raleigh	Avery Close	9191234567	NULL	3040
2	Penguin	wholesaler	Prince George	College Park	Baltimore Avenue	9191334567	NULL	0
3	Hunt	library	Wake County	Raleigh	Partners Way	9191434567	Max Brown	0
4	Hill	library	Prince George	College Park	Baltimore	9191435567	Jamie Brown	0
5	Mason's Books	bookstore	Ashville	Raleigh	Varsity Drive	9191235567	NULL	735

5 rows in set (0.00 sec)

Pays

```
INSERT INTO Pays(dist_id, emp_id, order_id) values(6, 12, 3)
INSERT INTO Pays(dist_id, emp_id, order_id) values(2, 13, 9)
INSERT INTO Pays(dist_id, emp_id, order_id) values(5, 13, 8)
INSERT INTO Pays(dist_id, emp_id, order_id) values(5, 13, 503);
INSERT INTO Pays(dist_id, emp_id, order_id) values(2, 12, 507);
```

MariaDB [sskangle]> SELECT * FROM Pays;

dist_id	emp_id	order_id
2	12	507
2	13	9
5	13	8
5	13	503
6	12	3

5 rows in set (0.01 sec)

WritesBooks

```
INSERT INTO WritesBooks values(1, 100001, 1);
INSERT INTO WritesBooks values(2, 100002, 2);
INSERT INTO WritesBooks values(5, 100005, 1);
INSERT INTO WritesBooks values(6, 100006, 2);
INSERT INTO WritesBooks values(7, 100007, 1);
```

MariaDB [sskangle]> SELECT * FROM WritesBooks;

emp_id	isbn	edition
1	100001	1
2	100002	2
5	100005	1
6	100006	2
7	100007	1

5 rows in set (0.00 sec)

WritesPerPub

```
INSERT INTO WritesPerPub values(3, 100003, 3, 2020);
INSERT INTO WritesPerPub values(3, 100004, 9, 1999);
INSERT INTO WritesPerPub values(10, 100008, 9, 2019);
INSERT INTO WritesPerPub values(10, 100009, 27, 2013);
INSERT INTO WritesPerPub values(11, 100010, 46, 2012);
```

```
MariaDB [sskangle]> SELECT * FROM WritesPerPub;
+--------+--------+----------+------------+
| emp_id | isbn   | issue_wm | issue_year |
+--------+--------+----------+------------+
|      3 | 100003 |        3 |       2020 |
|      3 | 100004 |        9 |       1999 |
|     10 | 100008 |        9 |       2019 |
|     10 | 100009 |       27 |       2013 |
|     11 | 100010 |       46 |       2012 |
+--------+--------+----------+------------+
5 rows in set (0.00 sec)
```

EditsChapters

```
INSERT INTO EditsChapters Values(8, 100001, 1, 2);
INSERT INTO EditsChapters Values(8, 100002, 2, 3);
INSERT INTO EditsChapters Values(8, 100005, 1, 4);
INSERT INTO EditsChapters Values(9, 100006, 2, 6);
INSERT INTO EditsChapters Values(9, 100007, 1, 7);
```

```
MariaDB [sskangle]> SELECT * FROM EditsChapters;
+--------+--------+---------+------------+
| emp_id | isbn   | edition | chapter_id |
+--------+--------+---------+------------+
|      8 | 100001 |       1 |          2 |
|      8 | 100005 |       1 |          4 |
|      8 | 100002 |       2 |          3 |
|      9 | 100007 |       1 |          7 |
|      9 | 100006 |       2 |          6 |
+--------+--------+---------+------------+
5 rows in set (0.00 sec)
```

EditsArticles

```
INSERT INTO EditsArticles values(8, 100003, 3, 2020, 3);
INSERT INTO EditsArticles values(9, 100004, 9, 1999, 3);
INSERT INTO EditsArticles values(9, 100008, 9, 2019, 3);
INSERT INTO EditsArticles values(9, 100009, 27, 2013, 3);
INSERT INTO EditsArticles values(8, 100010, 46, 2012, 3);
```

MariaDB [sskangle]> SELECT * FROM EditsArticles;

emp_id	isbn	issue_wm	issue_year	article_id
8	100003	3	2020	3
8	100010	46	2012	3
9	100004	9	1999	3
9	100008	9	2019	3
9	100009	27	2013	3

5 rows in set (0.00 sec)

OrdersBooks

INSERT INTO OrdersBooks(isbn, edition, dist_id, order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount) values(100001, 1, 1, 25, 02, 2020, "2020-04-10", "accepted", 100, 0,3040);
INSERT INTO OrdersBooks(isbn, edition, dist_id, order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount) values(100002, 2, 2, 01, 01, 2020, "2020-01-04", "discarded", 20, 0,424.8);
INSERT INTO OrdersBooks(isbn, edition, dist_id, order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount) values(100005,1, 6, 04, 03, 2020, "2020-03-15", "completed", 40, 0,2264);
INSERT INTO OrdersBooks(isbn, edition, dist_id, order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount) values(100006, 2, 4, 23, 12, 2019, "2020-01-20", "discarded", 15, 0, 147);
INSERT INTO OrdersBooks(isbn, edition, dist_id, order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount) values(100007, 1, 5, 04, 03, 2020, "2020-04-25", "accepted", 70, 0, 735);

MariaDB [sskangle]> SELECT * FROM OrdersBooks;

order_id	isbn	edition	dist_id	order_day	order_month	order_year	exp_del_date	status	num_ordered_copies	num_booked_copies	amount
1	100001	1	1	25	2	2020	2020-04-10	accepted	100	0	3040
2	100002	2	2	1	1	2020	2020-01-04	discarded	20	0	424.8
3	100005	1	6	4	3	2020	2020-03-15	completed	40	0	2264
4	100006	2	4	23	12	2019	2020-01-20	discarded	15	0	147
5	100007	1	5	4	3	2020	2020-04-25	accepted	70	0	735

5 rows in set (0.01 sec)

OrdersPerPub

INSERT INTO OrdersPerPub (isbn, issue_wm, issue_year, dist_id, order_day, order_month, order_year, exp_del_date, status, num_ordered_copies, num_booked_copies, amount) values (100008, 9, 2019, 3, 05, 03, 2020, "2020-03-13", "discarded", 6, 0, 81.24);

```
INSERT INTO OrdersPerPub (isbn, issue_wm, issue_year, dist_id, order_day, order_month, order_year,
exp_del_date, status, num_ordered_copies, num_booked_copies, amount) values (100004, 9, 1999, 2, 05, 03, 2020,
"2020-04-13", "accepted", 100, 0, 3050);
INSERT INTO OrdersPerPub (isbn, issue_wm, issue_year, dist_id, order_day, order_month, order_year,
exp_del_date, status, num_ordered_copies, num_booked_copies, amount) values (100009, 27, 2013, 5, 05, 03,
2020, "2020-03-13", "completed", 6, 0, 120.18);
INSERT INTO OrdersPerPub(isbn, issue_wm, issue_year, dist_id, order_day, order_month, order_year,
exp_del_date, status, num_ordered_copies, num_booked_copies, amount) values (100009, 27, 2013, 4, 12, 03,
2020, "2020-04-24", "accepted", 32, 0, 640.96);
INSERT INTO OrdersPerPub(isbn, issue_wm, issue_year, dist_id, order_day, order_month, order_year,
exp_del_date, status, num_ordered_copies, num_booked_copies, amount) values (100010, 46, 2012, 4, 16, 01,
2020, "2020-02-21", "discarded", 50, 0, 382);
```

MariaDB [sskangle]> SELECT * FROM OrdersPerPub;

```
+----------+--------+----------+------------+---------+-----------+-------------+------------+--------------+-----------+--------------------+---
----------------+--------+
| order_id | isbn   | issue_wm | issue_year | dist_id | order_day | order_month | order_year | exp_del_date |
status    | num_ordered_copies | num_booked_copies | amount |
+----------+--------+----------+------------+---------+-----------+-------------+------------+--------------+-----------+--------------------+---
----------------+--------+
|     501 | 100008 |      9 |      2019 |      3 |       5 |   3 |      2020 | 2020-03-13   | discarded |        6 |   0 |  81.24 |
|     502 | 100004 |      9 |      1999 |      2 |       5 |   3 |      2020 | 2020-04-13   | accepted  |    100 |    0 |   3050 |
|     503 | 100009 |     27 |      2013 |      5 |       5 |   3 |      2020 | 2020-03-13   | completed |      6 |   0 | 120.18 |
|     504 | 100009 |     27 |      2013 |      4 |      12 |   3 |      2020 | 2020-04-24   | accepted  |     32 |   0 | 640.96 |
|     505 | 100010 |     46 |      2012 |      4 |      16 |   1 |      2020 | 2020-02-21   | discarded |     50 |   0 |    382 |
+----------+--------+----------+------------+---------+-----------+-------------+------------+--------------+-----------+--------------------+---
----------------+--------+
5 rows in set (0.01 sec)
```

Interactive Queries for Operations and Tasks

1. Editing and Publishing

- **Enter basic information on a new publication**
  Assumption : There does not previously exist a publication with isbn equal to the value being inserted

  INSERT INTO Publications VALUES(100011,"If Tomorrow Comes","Crime Fiction");
  Query OK, 1 row affected (0.00 sec)

  MariaDB [sskangle]> SELECT * FROM Publications where isbn = 100011;
  +--------+------------------+---------------+
  | isbn | title | genre |
  +--------+------------------+---------------+
  | 100011 | If Tomorrow Comes | Crime Fiction |
  +--------+------------------+---------------+
  1 row in set (0.00 sec)

- **Update information**
  Assumption: There exists a publication with isbn equal to the value being updated.

  UPDATE Publications SET genre = "News" where isbn = 100010;
  Query OK, 1 row affected (0.01 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

  MariaDB [sskangle]> SELECT * FROM Publications where isbn = 100010;
  +--------+---------------+-------+
  | isbn | title | genre |
  +--------+---------------+-------+
  | 100010 | Time Magazine | News |
  +--------+---------------+-------+
  1 row in set (0.00 sec)

- **Assign editor(s) to articles(publications)**
  Assumption: There exist an editor in the Employees table having emp_id = 8 which is getting assigned to articles hes/he is supposed to edit.

  INSERT INTO EditsArticles(emp_id,isbn,issue_wm,issue_year,article_id) VALUES (8,100004,9,1999,3);
  Query OK, 1 row affected (0.01 sec)

  MariaDB [sskangle]> SELECT * FROM EditsArticles WHERE isbn = 100004 and emp_id = 8;
  +--------+--------+----------+------------+------------+
  | emp_id | isbn | issue_wm | issue_year | article_id |
  +--------+--------+----------+------------+------------+
  | 8 | 100004 | 9 | 1999 | 3 |
  +--------+--------+----------+------------+------------+
  1 row in set (0.00 sec)

- **Assign editor(s) to chapters(publications)**
  Assumption: There exist an editor in the Employees table having emp_id = 9 which is getting assigned to chapters he/she is supposed to edit.

  INSERT INTO EditsChapters(emp_id,isbn,edition,chapter_id) VALUES(9,100002,2,3);
  Query OK, 1 row affected (0.01 sec)

  MariaDB [sskangle]> SELECT * FROM EditsChapters WHERE isbn = 100002 and emp_id = 9;

```
+--------+--------+---------+------------+
| emp_id | isbn   | edition | chapter_id |
+--------+--------+---------+------------+
|      9 | 100002 |       2 |          3 |
+--------+--------+---------+------------+
1 row in set (0.00 sec)
```

- **Let each editor view the information on the publications he/she is responsible for**
  Assumption: There exists an isbn column in Publications as well as EditsChapters/EditsArticles in order to allow the join to get executed correctly.

  - View Articles

  SELECT Publications.isbn,title,genre,EditsArticles.issue_wm,EditsArticles.issue_year,EditsArticles.article_id FROM Publications INNER JOIN EditsArticles WHERE emp_id = 8 AND EditsArticles.isbn = Publications.isbn;

```
+--------+---------------+-----------------------+----------+------------+------------+
| isbn   | title         | genre                 | issue_wm | issue_year | article_id |
+--------+---------------+-----------------------+----------+------------+------------+
| 100003 | IEEE Explore  | Science and Technology |        3 |       2020 |          3 |
| 100004 | Springer      | Scientific Research   |        9 |       1999 |          3 |
| 100010 | Time Magazine | News                  |       46 |       2012 |          3 |
+--------+---------------+-----------------------+----------+------------+------------+
```

  3 rows in set (0.00 sec)

  - View Chapters

  SELECT Publications.isbn,title,genre,EditsChapters.edition,EditsChapters.chapter_id FROM Publications INNER JOIN EditsChapters WHERE emp_id = 9 AND EditsChapters.isbn = Publications.isbn;

```
+--------+-------------------------------------+-----------------+---------+------------+
| isbn   | title                               | genre           | edition | chapter_id |
+--------+-------------------------------------+-----------------+---------+------------+
| 100007 | The Complete Robot                  | Science Fiction | 1       | 7          |
| 100002 | Introduction to Algorithms          | Textbook        | 2       | 3          |
| 100006 | Database Systems: The Complete Book | Textbook        | 2       | 6          |
+--------+-------------------------------------+-----------------+---------+------------+
```

  3 rows in set (0.00 sec)

- **Edit table of contents of a publication, by adding/deleting articles (for periodic publications) or chapter/sections (for books)**
  Assumption: There does not previously exist an article with the same isbn, issue_wm, issue_year in the table Articles.

- Inserting into Chapters

INSERT INTO Chapters(isbn,edition,chapter_id,chapter_date,title,topic,text) VALUES (100002,2,5,"2020-03-21","Searching","topic3","Binary Search");
Query OK, 1 row affected (0.01 sec)

MariaDB [sskangle]> SELECT * FROM Chapters WHERE isbn = 100002 AND edition = 2 AND chapter_id = 5;

```
+--------+---------+------------+--------------+-----------+--------+---------------+
| isbn   | edition | chapter_id | chapter_date | title     | topic | text |
+--------+---------+------------+--------------+-----------+--------+---------------+
| 100002 |       2 | 5 | 2020-03-21   | Searching | topic3 | Binary Search |
+--------+---------+------------+--------------+-----------+--------+---------------+
```

1 row in set (0.00 sec)

- Inserting into Articles

INSERT INTO Articles(isbn,issue_wm,issue_year,article_id,article_date,title,topic,text) VALUES (100008,9,2019,4,"2020-03-21","Winter Shoes", "Fashion", "Boots are winter friendly.");
Query OK, 1 row affected (0.01 sec)

MariaDB [sskangle]> SELECT * FROM Articles WHERE isbn = 100008 AND issue_wm = 9 AND issue_year = 2019 AND article_id = 4;

```
+--------+----------+------------+------------+--------------+--------------+---------+---------------------------+
| isbn   | issue_wm | issue_year | article_id | article_date | title        | topic | text |
+--------+----------+------------+------------+--------------+--------------+---------+---------------------------+
| 100008 |       9 | 2019 |        4 | 2020-03-21 | Winter Shoes | Fashion | Boots are winter friendly. |
+--------+----------+------------+------------+--------------+--------------+---------+---------------------------+
```

1 row in set (0.00 sec)

Assumption: There exists a publication with isbn equal to the value being deleted.

- Delete Articles

DELETE FROM Articles where isbn = 100008 and article_id = 4;
Query OK, 1 row affected (0.00 sec)

MariaDB [sskangle]> SELECT * FROM Articles WHERE isbn = 100008 AND issue_wm = 9 AND issue_year = 2019 AND article_id = 4;
Empty set (0.00 sec)

- Delete Chapters

DELETE from Chapters where isbn = 100002 and chapter_id = 4;
Query OK, 1 row affected (0.00 sec)

MariaDB [sskangle]> SELECT * FROM Chapters WHERE isbn = 100002 AND edition = 2 AND chapter_id = 5;
Empty set (0.00 sec)

2. Publications of a book edition or an issue of an publication

- **Enter a new book edition of a publication**
  Assumptions: There exists a Publication with isbn = 1000002. And there exists a table Books in the database.

  INSERT INTO Books(isbn, edition, num_available_copies, edition_date, price)
  VALUES (100002, 4, 200, "2020-03-14", 25);
  Query OK, 1 row affected (0.01 sec)

  MariaDB [sskangle]> SELECT * FROM Books WHERE isbn=100002 AND edition=4;

```
+--------+---------+----------------------+--------------+-------+
| isbn   | edition | num_available_copies | edition_date | price |
+--------+---------+----------------------+--------------+-------+
| 100002 |       4 |    200 | 2020-03-14 |   25 |
+--------+---------+----------------------+--------------+-------+
```

  1 row in set (0.00 sec)

- **Enter a new issue of a publication**
  Assumptions: There exists a Publication with isbn = 1000008. And there exists a table PeriodicPublications in the database.

  INSERT INTO PeriodicPublications(isbn, issue_wm, issue_year, periodicity, pub_type,
  num_available_copies, issue_date, price)
  VALUES (100008, 12, 2019, "monthly", "magazine", 100, "2019-09-06", 30.0);
  Query OK, 1 row affected (0.00 sec)
  MariaDB [sskangle]> SELECT * FROM PeriodicPublications WHERE isbn=100008 AND issue_wm=12 AND issue_year=2019;

```
+--------+----------+------------+-------------+----------+----------------------+------------+-------+
| isbn   | issue_wm | issue_year | periodicity | pub_type | num_available_copies | issue_date | price |
+--------+----------+------------+-------------+----------+----------------------+------------+-------+
| 100008 |       12 | 2019 | monthly     | magazine | 100 | 2019-09-06 |    30 |
+--------+----------+------------+-------------+----------+----------------------+------------+-------+
```

  1 row in set (0.00 sec)

- **Update a publication issue**
  Assumptions: There exists a table PeriodicPublications in the database having an entry with isbn=100008, issue_wm=12, issue_year=2019.

  UPDATE PeriodicPublications
  SET num_available_copies=75, periodicity="weekly"
  WHERE isbn=100008 AND issue_wm=12 AND issue_year=2019;
  Query OK, 1 row affected (0.00 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

MariaDB [sskangle]> SELECT * FROM PeriodicPublications WHERE isbn=100008 AND issue_wm=12 AND issue_year=2019;

```
+--------+----------+------------+-------------+----------+---------------------+------------+-------+
| isbn   | issue_wm | issue_year | periodicity | pub_type | num_available_copies | issue_date | price |
+--------+----------+------------+-------------+----------+---------------------+------------+-------+
| 100008 |       12 | 2019 | weekly      | magazine | 75 | 2019-09-06 |    30 |
+--------+----------+------------+-------------+----------+---------------------+------------+-------+
```

1 row in set (0.01 sec)

- **Update a book edition**

  Assumptions: There exists a table Books in the database having an entry with isbn=100002 and edition=4.

  UPDATE Books
  SET edition_date = "2020-03-10", price=35.0
  WHERE isbn=100002 AND edition=4;
  Query OK, 1 row affected (0.01 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

  MariaDB [sskangle]> SELECT * FROM Books WHERE isbn=100002 AND edition=4;

```
+--------+---------+----------------------+--------------+-------+
| isbn   | edition | num_available_copies | edition_date | price |
+--------+---------+----------------------+--------------+-------+
| 100002 |       4 |    200 | 2020-03-10 |   35 |
+--------+---------+----------------------+--------------+-------+
```

  1 row in set (0.00 sec)

- **Delete book edition**

  Assumptions: There exists a table Books in the database having an entry with isbn=100002 and edition=4.

  DELETE FROM Books
  WHERE isbn=100002 AND edition=4;
  Query OK, 1 row affected (0.00 sec)

  MariaDB [sskangle]> SELECT * FROM Books WHERE isbn=100002 AND edition=4;
  Empty set (0.00 sec)

- **Delete publication issue**

  Assumptions: There exists a table PeriodicPublications in the database having an entry with isbn=100008, issue_wm=12, issue_year=2019.

  DELETE FROM PeriodicPublications
  WHERE isbn=100008 AND issue_wm=12 AND issue_year=2019;
  Query OK, 1 row affected (0.01 sec)

  MariaDB [sskangle]> SELECT * FROM PeriodicPublications WHERE isbn=100008 AND issue_wm=12 AND issue_year=2019;
  Empty set (0.00 sec)

- **Enter an article**

  Assumptions: There exists a PeriodicPublication with isbn= 100008, issue_wm=09 and issue_year= 2019. And there exists a table Articles in the database.

```
INSERT INTO Articles(isbn, issue_wm, issue_year, article_id, article_date, title, topic, text)
VALUES
(100008, 09, 2019, 4, "2019-08-10", "Formal Wear", "Clothes and Shoes for Work", "Dress up to make an impression");
Query OK, 1 row affected (0.00 sec)

MariaDB [sskangle]> SELECT * FROM Articles WHERE isbn=100008 AND issue_wm=09 AND issue_year=2019 AND article_id=4;
+--------+----------+------------+------------+-------------+-----------------+-----------------------+------------------------+
| isbn   | issue_wm | issue_year | article_id | article_date | title           | topic | text |
+--------+----------+------------+------------+-------------+-----------------+-----------------------+------------------------+
| 100008 |       9 | 2019 |        4 | 2019-08-10 | Formal Wear      | Clothes and Shoes for Work | Dress up to make an impression |
+--------+----------+------------+------------+-------------+-----------------+-----------------------+------------------------+
1 row in set (0.00 sec)
```

- **Enter a chapter**

Assumptions: There exists a Book with isbn= 100002 and edition=2. And there exists a table Chapters in the database.

```
INSERT INTO Chapters(isbn, edition, chapter_id, chapter_date, title, topic, text)
VALUES
(100002, 2, 4, "2017-07-17", "Dynamic Programming", "Memoization", "Dynamic programming is a very powerful technique to solve a particular class of problems.");
Query OK, 1 row affected (0.00 sec)

MariaDB [sskangle]> SELECT * FROM Chapters WHERE isbn=100002 AND edition=2 AND chapter_id=4;
+--------+---------+------------+--------------+-----------------------+-------------+-------------------------------------+
| isbn   | edition | chapter_id | chapter_date | title                 | topic |
text                                                                                                                 |
+--------+---------+------------+--------------+-----------------------+-------------+-------------------------------------+
| 100002 |       2 | 4 | 2017-07-17   | Dynamic Programming | Memoization  | Dynamic programming is a very powerful technique to solve a particular class of
problems.                                                                                                            |
+--------+---------+------------+--------------+-----------------------+-------------+-------------------------------------+
1 row in set (0.00 sec)
```

- **Update an article**

Assumptions: There exists an article with isbn=100008, issue_wm=09, issue_year=2019, article_id = 4 in the Articles table.

```
UPDATE Articles
SET topic="Formal Shoes and Clothes", article_date="2019-08-12"
WHERE isbn=100008 AND issue_wm=09 AND issue_year=2019 AND article_id=4;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
MariaDB [sskangle]> SELECT * FROM Articles WHERE  isbn=100008 AND issue_wm=09 AND
issue_year=2019 AND article_id=4;
+--------+----------+------------+------------+--------------+-------------------+--------------------------+------------------------+
| isbn   | issue_wm | issue_year | article_id | article_date | title             | topic | text |
+--------+----------+------------+------------+--------------+-------------------+--------------------------+-------------------------+
| 100008 |        9 | 2019 |          4 | 2019-08-12 | Formal Wear       | Formal Shoes and Clothes | Dress up to
make an impression |
+--------+----------+------------+------------+--------------+-------------------+--------------------------+-------------------------+
1 row in set (0.00 sec)
```

- **Update a chapter**

Assumptions: There exists a chapter with isbn=100002, edition=2, chapter_id=4 in the Chapters table.

```
UPDATE Chapters
SET topic="Overlapping Subproblems", chapter_date="2019-06-17"
WHERE isbn=100002 AND edition=2 AND chapter_id=4;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [sskangle]> SELECT * FROM Chapters WHERE  isbn=100002 AND edition=2 AND chapter_id=4;
+--------+---------+------------+--------------+------------------------+------------------------+-------------------------------+
| isbn   | edition | chapter_id | chapter_date | title                  | topic |   text                        |
+--------+---------+------------+--------------+------------------------+------------------------+---------------------------------+
| 100002 |       2 | 4 | 2019-06-17   | Dynamic Programming | Overlapping Subproblems | Dynamic
programming is a very powerful technique to solve a particular class of
problems.                                                                        |
+--------+---------+------------+--------------+------------------------+------------------------+-------------------------------+
1 row in set (0.00 sec)
```

- **Update text of an article**

Assumptions: There exists an article with isbn=100004, issue_wm=9, issue_year=1999, article_id=3 in the Articles table.

```
UPDATE Articles
SET text="Neural networks are a set of algorithms, modeled loosely after the human brain, that are
designed to recognize patterns."
WHERE isbn=100004 AND issue_wm=9 AND issue_year=1999 AND article_id=3;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [sskangle]> SELECT * FROM Articles WHERE isbn=100004 AND issue_wm=9 AND
issue_year=1999 AND article_id=3;
+--------+----------+------------+------------+--------------+------------------+--------------------+----------------------------+
| isbn   | issue_wm | issue_year | article_id | article_date | title            | topic |
text                                                    |
+--------+----------+------------+------------+--------------+------------------+--------------------+----------------------------+
| 100004 |        9 | 1999 |          3 | 1999-08-12 | Neural Networks   | Deep Learning | Neural networks are a
set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns.
|                                                                          |
+--------+----------+------------+------------+--------------+------------------+--------------------+----------------------------+
1 row in set (0.00 sec)
```

- **Find books by author's name**

```
MariaDB [sskangle]> SELECT p.isbn,p.title,b.edition FROM Books b
INNER JOIN Publications p
ON p.isbn=b.isbn
INNER JOIN WritesBooks w
ON w.isbn = b.isbn AND w.edition=b.edition
AND w.emp_id = (SELECT emp_id FROM Employees where emp_name =  "Thomas Cormen");

+--------+--------------------------+---------+
| isbn   | title                    | edition |
+--------+--------------------------+---------+
| 100002 | Introduction to Algorithms |       2 |
| 100002 | Introduction to Algorithms |       3 |
+--------+--------------------------+---------+
2 rows in set (0.00 sec)
```

- **Find articles by journalist's name**

```
MariaDB [sskangle]> SELECT a.title,a.topic,a.text FROM Articles a
INNER JOIN
WritesPerPub j
ON a.isbn=j.isbn AND a.issue_wm=j.issue_wm AND a.issue_year=j.issue_year
AND j.emp_id = (SELECT emp_id FROM Employees WHERE emp_name="Steve Forbes");

+----------------+--------------------+-----------------------------------------------------------------------------------------------------------------------+
| title          | topic     | text                                                                                                                  |
+----------------+--------------------+-----------------------------------------------------------------------------------------------------------------------+
| Fault Diagnosis | Distributed Systems | This is a static technique                                                                                            |
| Neural Networks | Deep Learning      | Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. |
+----------------+--------------------+-----------------------------------------------------------------------------------------------------------------------+
2 rows in set (0.00 sec)
```

- **Find articles by topic**

```
MariaDB [sskangle]> SELECT * FROM Articles WHERE topic="Deep Learning";

+--------+----------+------------+------------+-------------+----------------+--------------+-----------------------------------------------------------------------------------------------------------------------+
| isbn   | issue_wm | issue_year | article_id | article_date | title          | topic |
text                                                                                                                  |
+--------+----------+------------+------------+-------------+----------------+--------------+-----------------------------------------------------------------------------------------------------------------------+
| 100004 |       9 | 1999 |        3 | 1999-08-12 | Neural Networks | Deep Learning | Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. |
+--------+----------+------------+------------+-------------+----------------+--------------+-----------------------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

- **Find articles by date**

MariaDB [sskangle]> SELECT * FROM Articles WHERE article_date BETWEEN '2013-01-01' AND '2019-12-31';

```
+--------+----------+------------+------------+--------------+------------------+--------------------------+-----------------------+
| isbn   | issue_wm | issue_year | article_id | article_date | title            | topic | text |
+--------+----------+------------+------------+--------------+------------------+--------------------------+-----------------------+
| 100003 |       3 | 2020 |        3 | 2019-12-12 | Fault Diagnosis   | Distributed Systems | This is a static technique     |
| 100008 |       9 | 2019 |        4 | 2019-08-12 | Formal Wear       | Formal Shoes and Clothes | Dress up to make an impression |
| 100009 |      27 | 2013 |        3 | 2013-06-16 | Market Statistics | Business                 | Stock market looks good. |
+--------+----------+------------+------------+--------------+------------------+--------------------------+-----------------------+
3 rows in set (0.00 sec)
```

- **Enter payment for author**

Assumptions:

- While adding salary transaction, the amount is added as a negative value and description as “Salary Paid”.
- After that the “pay_date” for that employee is updated with the transaction date as the last payment date.

```
INSERT INTO Transactions(trans_day,trans_month,trans_year,client_id,amount,description)
VALUES(29,02,2020,9,-180,"Salary Paid");
Query OK, 1 row affected (0.00 sec)

MariaDB [sskangle]> SELECT * FROM Transactions WHERE client_id=9;
+----------+-----------+-------------+------------+-----------+---------+------------------+
| trans_id | trans_day | trans_month | trans_year | client_id | amount  | description |
+----------+-----------+-------------+------------+-----------+---------+------------------+
|       24 |     29 |   2 | 2020 |         9 | -180 | Salary Paid      |
+----------+-----------+-------------+------------+-----------+---------+------------------+
1 row in set (0.00 sec)

UPDATE Employees
SET pay_date="2020-02-29"
WHERE emp_id=9;
Query OK, 0 rows affected (0.01 sec)
Rows matched: 1  Changed: 0 Warnings: 0

MariaDB [sskangle]> SELECT * FROM Employees WHERE emp_id=9;
+--------+---------------------+-----------------+--------+------------+----------+
| emp_id | emp_name            | email | salary | pay_date   | emp_type |
+--------+---------------------+-----------------+--------+------------+----------+
|      9 | John Smith          | smith@abc.com | 180 | 2020-02-29 | staff    |
+--------+---------------------+-----------------+--------+------------+----------+
1 row in set (0.00 sec)
```

- **Keep track of when each payment was claimed by its addressee**

  Assumptions:
  - The transactions for payment of authors/editors/journalists has description “Salary Paid”.
  - The emp_type for authors/editors/journalists has is staff/invited.

```
SELECT t.trans_day,t.trans_month,t.trans_year,e.emp_name,e.emp_type,t.amount,t.description
FROM Transactions t
INNER JOIN
Employees e
ON t.client_id = e.emp_id AND t.description="Salary Paid" AND e.emp_type IN("staff","invited");
```

```
+-----------+-------------+------------+---------------------+----------+--------+-------------+
| trans_day | trans_month | trans_year | emp_name            | emp_type | amount | description |
+-----------+-------------+------------+---------------------+----------+--------+-------------+
|       29 |      2 | 2020 | Dan Brown           | staff | -115 | Salary Paid |
|       29 |      2 | 2020 | Thomas Cormen       | staff | -150 | Salary Paid |
|       29 |      2 | 2020 | Steve Forbes        | staff | -150 | Salary Paid |
|       30 |     11 | 2019 | Ronald Rivest       | invited | -115 | Salary Paid |
|       31 |      1 | 2020 | Michelle Obama      | invited | -115 | Salary Paid |
|       29 |      2 | 2020 | Isaac Asimov        | invited | -165 | Salary Paid |
|       29 |      2 | 2020 | Hector Garcia-Molina | staff    | -115 | Salary Paid |
|       30 |     11 | 2019 | Nancy Gibbs         | invited | -180 | Salary Paid |
|       29 |      2 | 2020 | John Smith          | staff | -180 | Salary Paid |
|       29 |      2 | 2020 | Adam Sandy          | staff | -150 | Salary Paid |
|       29 |      2 | 2020 | Barry Allen         | invited | -115 | Salary Paid |
+-----------+-------------+------------+---------------------+----------+--------+-------------+
11 rows in set (0.01 sec)
```

3. Distribution

- **Enter new distributor**

  Assumption: The distributor table is already created.

```
INSERT INTO
Distributors (dist_name, dist_type, location, city, street_addr, phone, person_of_contact, balance_due)
VALUES
("Grand Central Distributor", "wholesaler", "Jersey City", "New Jersey", "Hudson Street", "9191567879",
"John", 0.0);
Query OK, 1 row affected (0.01 sec)

MariaDB [sskangle]> SELECT * FROM Distributors where dist_id=7;
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------
+-------------+
| dist_id | dist_name                | dist_type  | location  | city        | street_addr     | phone |
person_of_contact | balance_due |
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------
+-------------+
|       7 | Grand Central Distributor | wholesaler | Jersey City   | New Jersey   | Hudson Street    |
9191567879 | John  |           0 |
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------
+-------------+
1 rows in set (0.00 sec)
```

- **Update distributor information**

  Assumption: Entry for dist_id = 7 already exists.

```
UPDATE Distributors
SET phone = "9191567779", person_of_contact="Chris"
WHERE dist_id=7;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [sskangle]> SELECT * FROM Distributors;
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------
+-------------+
| dist_id | dist_name           | dist_type  | location      | city        | street_addr     | phone    |
person_of_contact | balance_due |
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------
+-------------+
|       7 | Grand Central Distributor | wholesaler | Jersey City   | New Jersey   | Hudson Street    |
9191567779 | Chris     |           0 |
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------
+-------------+
1 row in set (0.01 sec)
```

- **Delete a distributor**

  DELETE FROM Distributors where dist_id = 7;

```
MariaDB [sskangle]> SELECT * FROM Distributors WHERE dist_id=7;
Empty set (0.00 sec)
```

- **Input orders from distributors, for a book edition per distributor, for a certain date**
  Assumption: The OrdersBooks table is already created. Entries for corresponding distributor and book, edition must be present.

```
INSERT INTO OrdersBooks (isbn, edition, dist_id, order_day, order_month, order_year, exp_del_date,
status, num_ordered_copies, num_booked_copies, amount)
VALUES (100005, 1, 5, 21, 03, 2020, "2020-04-15", "accepted", 10, 0, 560 );
Query OK, 1 row affected (0.01 sec)

MariaDB [sskangle]> SELECT * FROM OrdersBooks where order_id = 10;
+----------+--------+---------+---------+-----------+-------------+------------+-------------+-----------+--------------------+------
-------------+--------+
| order_id | isbn   | edition | dist_id | order_day | order_month | order_year | exp_del_date | status    |
num_ordered_copies | num_booked_copies | amount |
+----------+--------+---------+---------+-----------+-------------+------------+-------------+-----------+--------------------+------
-------------+--------+
|       10 | 100005 |       1 |       5 |        21 |           3 |       2020 | 2020-04-15   | accepted  |                 10 |
0 |    560 |
+----------+--------+---------+---------+-----------+-------------+------------+-------------+-----------+--------------------+------
-------------+--------+
1 row in set (0.00 sec)
```

- **Input orders from distributors, for an issue of a publication per distributor, for a certain date**
  Assumption: The OrdersPerPub table is already created. Entries for corresponding distributor and issue, publication must be present.

```
INSERT INTO OrdersPerPub (isbn, issue_wm, issue_year, dist_id, order_day, order_month, order_year,
exp_del_date, status, num_ordered_copies, num_booked_copies, amount)
VALUES (100003, 3, 2020, 5, 21, 03, 2020, "2020-04-15", "accepted", 20, 0, 210);
Query OK, 1 row affected (0.01 sec)

MariaDB [sskangle]> SELECT * FROM OrdersPerPub;
+----------+--------+----------+------------+---------+-----------+-------------+------------+-------------+-----------+-------------
-------+--------------------+--------+
| order_id | isbn   | issue_wm | issue_year | dist_id | order_day | order_month | order_year |
exp_del_date | status    | num_ordered_copies | num_booked_copies | amount |
+----------+--------+----------+------------+---------+-----------+-------------+------------+-------------+-----------+-------------
-------+--------------------+--------+
|      510 | 100003 |        3 |       2020 |       5 |        21 |           3 |       2020 | 2020-04-15  | accepted  |
20 |                  0 |    210 |
+----------+--------+----------+------------+---------+-----------+-------------+------------+-------------+-----------+-------------
-------+--------------------+--------+
1 rows in set (0.00 sec)
```

- **Bill distributor for an order**

```
UPDATE Distributors
SET balance_due = balance_due + 560
where dist_id = 7;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [sskangle]> SELECT * FROM Distributors;
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------+-------------+
| dist_id | dist_name          | dist_type  | location     | city        | street_addr     | phone    | person_of_contact | balance_due |
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------+-------------+
|       7 | Grand Central Distributor | wholesaler | Jersey City  | New Jersey  | Hudson Street   | 9191567779 | Chris     |     560 |
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------+-------------+
1 row in set (0.01 sec)
```

- **Change outstanding balance of a distributor on receipt of a payment**

```
UPDATE Distributors
SET balance_due = balance_due - 560
where dist_id = 7;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [sskangle]> SELECT * FROM Distributors;
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------+-------------+
| dist_id | dist_name          | dist_type  | location     | city        | street_addr     | phone    | person_of_contact | balance_due |
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------+-------------+
|       7 | Grand Central Distributor | wholesaler | Jersey City  | New Jersey  | Hudson Street   | 9191567779 | Chris     |       0 |
+---------+--------------------------+------------+--------------+-------------+-----------------+------------+------------------+-------------+
1 row in set (0.01 sec)
```

4. Report Generation

a. Generate Monthly Reports:

- **Number and total price of copies of each publication bought per distributor per month**

```
SELECT dist_id,isbn,order_month,order_year,sum(num_ordered_copies) AS copies,sum(amount) AS price
from OrdersBooks GROUP BY dist_id,isbn,order_month,order_year
UNION ALL
SELECT dist_id,isbn,order_month,order_year,sum(num_ordered_copies) AS copies,sum(amount) AS price
from OrdersPerPub GROUP BY dist_id,isbn,order_month,order_year;
```

```
+---------+--------+-------------+------------+--------+--------------------+
| dist_id | isbn   | order_month | order_year | copies | price              |
+---------+--------+-------------+------------+--------+--------------------+
|       1 | 100001 |           2 |       2020 | 100    |               3040 |
|       2 | 100001 |          12 |       2019 | 5      |                152 |
|       2 | 100002 |           1 |       2020 | 20     | 424.79998779296875 |
|       4 | 100006 |          12 |       2019 | 15     |                147 |
|       5 | 100007 |           3 |       2020 | 70     |                735 |
|       5 | 100007 |          12 |       2019 | 30     |                315 |
|       6 | 100002 |           3 |       2020 | 35     |  743.4000244140625 |
|       6 | 100005 |           2 |       2020 | 25     |               1415 |
|       6 | 100005 |           3 |       2020 | 40     |               2264 |
|       2 | 100004 |           3 |       2020 | 100    |               3050 |
|       2 | 100008 |           2 |       2020 | 40     |  541.5999755859375 |
|       3 | 100008 |           3 |       2020 | 6      |  81.23999786376953 |
|       3 | 100010 |           3 |       2020 | 15     |  114.5999984741211 |
|       4 | 100009 |           3 |       2020 | 32     |  640.9600219726562 |
|       4 | 100010 |           1 |       2020 | 50     |                382 |
|       5 | 100003 |           3 |       2020 | 35     |              367.5 |
|       5 | 100009 |           3 |       2020 | 6      | 120.18000030517578 |
|       6 | 100004 |           3 |       2020 | 50     |                382 |
+---------+--------+-------------+------------+--------+--------------------+
18 rows in set (0.01 sec)
```

- **Total revenue of the publishing house**

  Assumptions:
  - The total revenue takes only the income of Payment received from Distributors into consideration.
  - The description for which is “Payment Received” in the Transactions table.

```
SELECT SUM(amount) AS Total_Revenue
FROM Transactions
WHERE description= “Payment Received” AND trans_month=12 AND trans_year=2019;
```

```
+---------------+
| Total_Revenue |
+---------------+
|           467 |
+---------------+
1 row in set (0.00 sec)
```

- **Total expenses (shipping costs and salaries)**

  Assumptions:
  - The total revenue takes into consideration shipping cost and salary payment.
  - The description for which is “Shipping Cost” or “Salary Paid” in the Transactions table.

```
SELECT ABS(SUM(amount)) AS Total_Expenses
FROM Transactions
WHERE trans_month=02 AND trans_year=2019 AND description IN (“Shipping Cost”,”Salary Paid”);
```

```
+-------------------+
| Total_Expenses    |
+-------------------+
| 54.15999984741211 |
+-------------------+
1 row in set (0.00 sec)
```

b. Calculate the total current number of distributors

Assumptions:
- This query combines output from two tables for orders of Books and PeriodicPublications.
- The distributors are considered “current” if they have at least one order in either of the two tables with status “accepted” or “processing”.

```
SELECT COUNT(dist_id) AS Total_Current_Distributors
FROM (
SELECT DISTINCT dist_id FROM OrdersBooks  WHERE status IN("accepted","processing") UNION
SELECT DISTINCT dist_id FROM OrdersPerPub  WHERE status IN("accepted","processing")
)UNION_DIST;
```

```
+---------------------------+
| Total_Current_Distributors |
+---------------------------+
|                         6 |
+---------------------------+
1 row in set (0.00 sec)
```

c. Calculate total revenue (since inception) per city

Assumptions:
- The total revenue takes only the income from Payment of Distributors into consideration.
- The description for which is “Payment Received” in the Transactions table.

```
SELECT d.city,SUM(amount) AS Total_Revenue
FROM Transactions t
INNER JOIN
Distributors d
ON t.client_id = d.dist_id AND description="Payment Received"
GROUP BY d.city;
```

```
+--------------+-------------------+
| city         | Total_Revenue     |
+--------------+-------------------+
| College Park | 693.5999755859375 |
| Raleigh      | 2699.180000305176 |
+--------------+-------------------+
2 rows in set (0.01 sec)
```

d. Calculate total revenue (since inception) per distributor

Assumptions:

- The total revenue takes only the income from Payment of Distributors into consideration.
- The description for which is “Payment Received” in the Transactions table.

```
SELECT client_id,SUM(amount) AS Total_Revenue
FROM Transactions
WHERE description="Payment Received"
GROUP BY client_id;
```

```
+-----------+-------------------+
| client_id | Total_Revenue     |
+-----------+-------------------+
|         2 | 693.5999755859375 |
|         5 | 435.1800003051758 |
|         6 |      2264 |
+-----------+-------------------+
3 rows in set (0.00 sec)
```

e. Calculate total revenue (since inception) per location

Assumptions:

- The total revenue takes only the income from Payment of Distributors into consideration.
- The description for which is “Payment Received” in the Transactions table.

```
SELECT d.location,SUM(amount) AS Total_Revenue
FROM Transactions t
INNER JOIN
Distributors d
ON t.client_id = d.dist_id AND description="Payment Received"
GROUP BY d.location;
```

```
+---------------+-------------------+
| location      | Total_Revenue     |
+---------------+-------------------+
| Ashville      | 2699.180000305176 |
| Prince George | 693.5999755859375 |
+---------------+-------------------+
2 rows in set (0.00 sec)
```

f. Calculate total payments to the editors and authors per work type (book authorship, article authorship, or editorial work)

Assumptions: The total payment to editors and authors is recorded in the Transactions table with the description “Salary Paid”.

```
SELECT ABS(SUM(amount)) AS Total_Payment, "Authors" AS Work_Type
FROM Transactions
WHERE description="Salary Paid" AND client_id IN(SELECT emp_id FROM Authors)
UNION ALL
SELECT ABS(SUM(amount)) AS Total_Payment, "Journalists" AS Work_Type
FROM Transactions
WHERE description="Salary Paid" AND client_id IN(SELECT emp_id FROM Journalists)
UNION ALL
SELECT ABS(SUM(amount)) AS Total_Payment, "Editors" AS Work_Type
FROM Transactions
WHERE client_id IN(SELECT distinct(emp_id) FROM EditsChapters UNION SELECT distinct(emp_id) FROM
EditsArticles);
```

```
+---------------+-------------+
| Total_Payment | Work_Type   |
+---------------+-------------+
|         775 | Authors     |
|         415 | Journalists |
|         360 | Editors     |
+---------------+-------------+
3 rows in set (0.00 sec)
```

g. Calculate total payments to the editors and authors per time period

Assumptions: The transactions for salary payment of authors/editors/journalists are those employees whose emp_type is staff/invited in the Employees table.

```
SELECT ABS(SUM(amount)) AS Total_Payment,trans_month AS Month,trans_year AS Year
FROM Transactions
WHERE
client_id IN(SELECT emp_id FROM Employees WHERE emp_type IN("staff","invited"))
GROUP BY trans_month,trans_year;
```

```
+--------------------+-------+------+
| Total_Payment      | Month | Year |
+--------------------+-------+------+
|             115 | 1 | 2020 |
|  487.4399757385254 |     2 | 2019 |
|            1140 | 2 | 2020 |
| 2145.7620067596436 |     3 | 2020 |
|             295 | 11 | 2019 |
| 420.30000019073486 |    12 | 2019 |
+--------------------+-------+------+
6 rows in set (0.00 sec)
```

EXPLAIN directive in MariaDB

Explain directive displays the query plan that the optimizer chooses.

Query 1:

a. Query

```
MariaDB [sskangle]> explain SELECT d.location,SUM(amount) AS Total_Revenue
-> FROM Transactions t
-> INNER JOIN
-> Distributors d
-> ON t.client_id = d.dist_id AND description="Payment Received"
-> GROUP BY d.location;
```

b. EXPLAIN Query Output

```
+------+-------------+-------+------+--------------+-------------+---------+-------------------+------+--------------------------------+
| id   | select_type | table | type | possible_keys | key         | key_len | ref | rows | Extra           |
+------+-------------+-------+------+--------------+-------------+---------+-------------------+------+--------------------------------+
|    1 | SIMPLE      | d | ALL | PRIMARY       | NULL | NULL | NULL              | 7 | Using temporary; Using filesort |
|    1 | SIMPLE      | t | ref | trans_index   | trans_index | 4 | sskangle.d.dist_id |    1 | Using where |
+------+-------------+-------+------+--------------+-------------+---------+-------------------+------+--------------------------------+
2 rows in set (0.00 sec)
```

c. Create index on location

```
MariaDB [sskangle]> create index loc_index on Distributors(location);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

d. EXPLAIN after creating index

```
MariaDB [sskangle]> explain SELECT d.location,SUM(amount) AS Total_Revenue  FROM Transactions t INNER JOIN
Distributors d ON t.client_id = d.dist_id AND description="Payment Received"  GROUP BY d.location;
+------+-------------+-------+-------+--------------+-------------+---------+-------------------+------+-------------+
| id   | select_type | table | type  | possible_keys | key | key_len | ref               | rows | Extra |
+------+-------------+-------+-------+--------------+-------------+---------+-------------------+------+-------------+
|    1 | SIMPLE      | d | index | PRIMARY       | loc_index | 32 | NULL            | 7 | Using index |
|    1 | SIMPLE      | t | ref | trans_index   | trans_index | 4 | sskangle.d.dist_id |    1 | Using where |
+------+-------------+-------+-------+--------------+-------------+---------+-------------------+------+-------------+
2 rows in set (0.01 sec)
```

Query 2:

a. Query

SELECT client_id,SUM(amount) AS Total_Revenue FROM Transactions WHERE description="Payment Received" GROUP BY client_id;

b. EXPLAIN Query Output

MariaDB [sskangle]> explain SELECT client_id,SUM(amount) AS Total_Revenue FROM Transactions WHERE description="Payment Received" GROUP BY client_id;

```
+------+-------------+--------------+------+---------------+------+---------+------+------+-----------------------------------------------+
| id   | select_type | table        | type | possible_keys | key  | key_len | ref  | rows | Extra                                         |
+------+-------------+--------------+------+---------------+------+---------+------+------+-----------------------------------------------+
|    1 | SIMPLE      | Transactions | ALL  | NULL          | NULL | NULL    | NULL |   24 | Using where; Using temporary; Using filesort |
+------+-------------+--------------+------+---------------+------+---------+------+------+-----------------------------------------------+
1 row in set (0.00 sec)
```

c. Create Index on client_id

MariaDB [sskangle]> create index trans_index on Transactions(client_id);
Query OK, 0 rows affected (0.03 sec)
Records: 0 Duplicates: 0 Warnings: 0

d. EXPLAIN after creating index

MariaDB [sskangle]> explain SELECT client_id,SUM(amount) AS Total_Revenue FROM Transactions WHERE description="Payment Received" GROUP BY client_id;

```
+------+-------------+--------------+-------+---------------+-------------+---------+------+------+-------------+
| id   | select_type | table        | type  | possible_keys | key         | key_len | ref  | rows | Extra       |
+------+-------------+--------------+-------+---------------+-------------+---------+------+------+-------------+
|    1 | SIMPLE      | Transactions | index | NULL          | trans_index | 4       | NULL |   24 | Using where |
+------+-------------+--------------+-------+---------------+-------------+---------+------+------+-------------+
```

For both the above queries, we can observe that when the query is written without an index created, the type field of EXPLAIN output shows “ALL” denoting full scan of the table needs to be performed to obtain the final output. On the contrary, after index creation, the same query execution plan (EXPLAIN) shows the type field to be index, denoting that not the full table is scanned, but instead indices facilitate faster querying of the table.

Relational Algebra

Query 1: Let each editor view the information on the publications he/she is responsible for

```
SELECT Publications.isbn, title, genre, EditsChapters.edition, EditsChapters.chapter_id FROM Publications
INNER JOIN
EditsChapters
WHERE emp_id = 9 AND
EditsChapters.isbn = Publications.isbn;
```

Relational Algebra:

$$\prod_{\text{Publications.isbn,title,genre,EditsChapters.edition,EditsChpaters.chapter\_id}} \left(\text{Publications} \bowtie_{\text{emp\_id =9 AND EditsChapters.isbn = Publications.isbn}} \text{EditsChapters} \right)$$

Correctness Proof:

- Suppose we consider a tuple from Publications relation and a tuple from EditsChapters relation such that the value of isbn is same in both tuples and the value of emp_id in the tuple of EditsChapters relation is for a specific editor.
- Each such combination of tuples will give the information on the publications i.e. isbn,title,edition,chapter_id for the given editor which is what is expected from the above mentioned specification.

Query 2: Calculate total revenue (since inception) per city

```
SELECT d.city,SUM(amount) AS Total_Revenue
FROM Transactions t
INNER JOIN
Distributors d
ON t.client_id = d.dist_id AND description="Payment Received"
GROUP BY d.city;
```

Relational Algebra:

$$\prod_{\text{d.city,Total\_Revenue}} \left(\gamma_{\text{d.city, sum(amount)} \rightarrow \text{Total\_Revenue}} \left(\rho_d(\text{Distributors}) \bowtie_{\substack{\text{description = “Payment Received” AND} \\ \text{t.client\_id=d.dist\_id}}} \rho_t(\text{Transactions}) \right) \right)$$

Correctness Proof:

- Suppose we consider a tuple t from the Transactions relation and a tuple d from the Distributors relation such that the value of t.client_id and d.dist_id is equal and value of description in tuple d is “Payment Received”.
- Each combination of such tuple (revenue over all distributors in that city.
- The combination of the city and the total revenue obtained will give us the revenue/payment received from the distributors.

- When we aggregate the resultant combination by grouping them by the city in the d tuples, the query gives the city and the sum of revenue from that city is what is expected from the above mentioned specification.

Explanation of previous erroneous solution:

```
SELECT SUM(amount) AS Total_Revenue
FROM Transactions t
INNER JOIN
Distributors d
ON t.client_id = d.dist_id AND description="Payment Received"  GROUP BY d.city;
```

Output:

```
+------------------+
| Total_Revenue    |
+------------------+
| 693.5999755859375 |
| 2699.180000305176 |
+------------------+
2 rows in set (0.01 sec)
```

- Suppose we consider a tuple t from the Transactions relation and a tuple d from the Distributors relation such that the value of t.client_id and d.dist_id is equal and value of description in tuple d is “Payment Received”.
- Each combination of such tuple(t,d) will give us the revenue/payment received from the distributors.
- When we aggregate the resultant combination by grouping them by the city in the d tuples, the query gives the sum of revenue over all distributors in that city.
- Since this query selects only the total revenue per city in the result, it does not display the name of the city for which the revenue has been obtained.
- This erroneous solution has been corrected above to obtain the results according to the specification.