Project report 1 Feedback and corrections :

5. (out of 80 points) 50
(note: better to organize them into the same four categories)
-20 missing APIs for “Bill distributor for an order; change outstanding balance of a distributor on receipt of a payment.”
-10 for total payment within a time period, it makes more sense to use timeFrom, timeTo as input

9. (out of 80 points) 40
-10 all primary key should be underlined
-30 too many unmatched conversions: in management view, finance_report_ID in CollectReport is not shown anywhere in the diagram. In distributor view, five extra schemas, missing two(see pdf). In editor view, need to keep the name of entity and attribute the same as shown in the diagram(see pdf). (note: the conversion must be exact, the errors I listed on the pdf is not exhaustive. Double check by yourself)

Page Number	Changes
7	missing APIs for “Bill distributor for an order; change outstanding balance of a distributor on receipt of a payment.”
8	total payment within a time period, it makes more sense to use timeFrom, timeTo as input
14,15	all primary key should be underlined
9, 11	Too many unmatched conversions

WolfPubDb Management System

For WolfCity publishing house

CSC 540 Database Management
Systems Project 1

Shakshi Singhai, Parth Thosani, Nisarg Shah, Prashan Sengo, February 14th, 2022

Assumptions:

1. We assume that the publishing house is the owner and regulates various activities within the system because it's the entity under which books are published.
2. We are assuming that the author and editor are the same entity.
3. Publication house has all the books readily available
4. We are assuming that books don't have articles.
5. We assume that Magazines and Journals are a part of the type of Periodic publication
6. Books are written in a specific format that consists of chapters.
7. There are multiple publishers working with the Wolfpub publishing house.

1. Problem Statement:

The publishing house leverages all the information within the business to create a database management system WolfPubDb. The system has 4 important tasks which are performed by different entities in the system:

- Editing and publishing: Editors should be able to perform any basic action to a publication (update/view publications).
- Production of a book edition: Add, update, or delete, books and any properties of a book (article, author of book, payment).
- Distribution: Add, update, delete distributors along with dealing with any transaction related to the distributors.
- Reports: Generate any numeric reports on transactions and books.

In the database, the data is stored in a structured format as compared to a traditional file system. A simple set of files may not be used for concurrent updates, but a database management system makes it possible to get and update the information efficiently with possible higher storage and retrieval capacity. DBMS has an effective querying system along with greater data protection and backup system. Data redundancy is greatly reduced in DBMS and processes are flexible and data dealt is always consistent. Private authorization to users is possible in DBMS which secures the data. Any update with respect to publishing/editing information would be visible to all participants of the database users. Storing reports and monetary transactions between distributors and publishing houses would be efficient with the help of DBMS as compared to file systems

2. Intended Users:

- **Distributor:** Distributors procure different editions of books from different publishing houses which might be written by various authors. They pay a sum for the orders to the publishing houses**.** They interact with publishing houses to get the orders.

- **Editor /authors:** They are responsible for adding, updating, and deleting various versions of the books which cater to different types of audiences based on the content of the

periodic publication.

- **Management:** The management is responsible for storing and analyzing the reports of both the entities of production and distribution of various types of books. They have an additional responsibility to store the information of the payments of various editors and authors based on their type. They generate various reports which summarize monthly reports based on the income from production houses and distributors. They calculate the total expenditure and revenue generated by the publishing house and their interaction with editors and distributors.

- **Admin/Publishing House:**
  The publishing house is the medium through which the editors/authors publish their books, monthly journals, and weekly magazines. They sell their books to the audience via the distributors. Store all information related to all the current and past orders from each distributor, bills and payments periodically. Store all information related to editors/authors and the various editions of the books which are published under the name of the publishing house.

3. Five Main Entities:

1. publication: title, editor(s), type (book, journal, magazine), periodicity for periodic publication (weekly, monthly, etc), typical topics (e.g., general, sports, technology, etc);

2. edition (1st, 2nd, etc), ISBN number, and publication date for a book.

3. issue of a periodic publication: title, date of issue, articles in the issue.

4. for a book or article: author(s), title, date of creation, the text of the article.

5. distributor: name, type (wholesale distributor, bookstore, library), street address, city, phone number, contact person, balance (how much is owed to the publishing house);

6. order (this is a request for a certain number of copies of a book or of an issue of a periodic publication to be produced by a certain date to meet a distributor's demand): Order Number, distributor, the title of the book, or issue, number of copies, date, price, shipping cost, payment completed (Yes, No),

7. Payment Information: Type (credit, debit), Name (Staff authors, Staff editor, Invited author, Invited editor, distributor), Cheque No., amount, Transaction date,

4. Tasks and Operations- Realistic Situations:

- Situation 1 (Distribution):
  A distributor named Parth goes to the publishing house and pays his pending dues.
  He then places a new order for 100 books to be delivered by 20 Feb 2022 to a

publishing house and owes an amount of $1000 which is updated in his account balance.

- Situation 2 (Publishing a book):
  Author named Shakshi has sent a request to the publishing house for a new edition of the book. This request is forwarded to the publishing house for approval and to distribute this book to various publishers.

5. Application Program Interfaces:

Editing and Publishing:

- addPublication(PublicationID,Editors,Types,Periodicity,Title,Topic,Date of Publication) -> return confirmation
- updatePublication(PublicationID,Editors,Types,Periodicity,Title,Topic,Date of Publication) -> return confirmation
- deletePublication(PublicationID,Editors,Types,Periodicity,Title,Topic,Date of Publication) -> return confirmation
- assignEditorToPublication(Publication ID, EID) -> return confirmation
- viewPublicationInformationByEditor(EID, Publication ID) -> return publication
- addArticles(ArticleID, Description, Text, PublicationID) -> return confirmation
- updateArticles(ArticleID, Description, Text, PublicationID) -> return confirmation
- deleteArticles(ArticleID, Description, Text, PublicationID) -> return confirmation
- addChapters(ChapterID, Number of Pages) -> return confirmation
- updateChapters(ChapterID, Number of Pages) -> return confirmation
- deleteChapters(ChapterID, Number of Pages) -> return confirmation
- viewPaymentInformation(Date, Amount, EID) -> return payment details
- assignPaymentToEditor(EID, Balance, Account Number, Date) -> return confirmation

Production:

- addBooks(ISBN, Title, Edition) -> return confirmation
- updateBooks(ISBN, Title, Edition) -> return confirmation
- deleteBooks(ISBN, Title, Edition) -> return confirmation
- addIssueOfPublication(Type, Periodic Length, Publication ID, Issue Date) -> return confirmation
- updateIssueOfPublication(Type, Periodic Length, Publication ID, Issue Date) -> return confirmation
- deleteIssueOfPublication(Type, Periodic Length, Publication ID, Issue Date) -> return confirmation
- addArticles(ArticleID, Description, Text, PublicationID) -> return confirmation

- updateArticles(ArticleID, Description, Text, PublicationID) -> return confirmation
- deleteArticles(ArticleID, Description, Text, PublicationID) -> return confirmation
- addChapters(ChapterID, Number of Pages) -> return confirmation
- updateChapters(ChapterID, Number of Pages) -> return confirmation
- deleteChapters(ChapterID, Number of Pages) -> return confirmation
- updateArticleText (ArticleID, PublicationID, new Text) -> return confirmation and updated article
- findBooksByTopic(Publication ID, ISBN, Edition, Topic) -> return books
- findBooksByDate(Publication ID, ISBN, Edition, Topic, Date) -> return books
- findBooksByAuthorName(Publication ID, ISBN, Edition, Topic, Author Name) -> return books
- findArticles(Article ID, Text, Description) -> return article
- addPayment(EID, Amount, Date) -> return confirmation
- updatePayment(EID, Amount, Date) -> return confirmation
- deletePayment(EID, Amount, Date) -> return confirmation
- trackRecordofPaymentClaimed(EID) -> return Date

Distributor:

- addDistributor(DistributorID, Type, Address, Balance, Name, Phone Number) -> return confirmation
- updateDistributor(DistributorID, Type, Address, Balance, Name, Phone Number) -> return confirmation
- deleteDistributor(DistributorID, Type, Address, Balance, Name, Phone Number) -> return confirmation
- getOrdersfromDistributorsPerEdition(Distributor ID, PublicationID , Edition, No of copies, date)) -> return ordersID
- billDistributor(DistributorID, OrderID) -> total cost
- updateOutstandingBalance(DistributorID, currentBalance, PaymentAmount) -> return confirmation
- viewAccountDetailsOfDistributor(Past Orders, Date, Balance, Account Number) -> return account details
- viewTransactionDetails (Order ID, Delivery Time, Delivery date) -> return transaction details
- generateBillForDistributorOfOrder(Distributor ID, Order ID, Amount, Balance, Number of Copies, Placement, Date) -> return bill

Reports

- getNumberOfCopiesBoughtByDistributorPerMonth(Publication ID,Distributor ID, Date) -> return number of books

- getRevenuePublishingHouse(PID) -> return revenue up to current date
- getTotalExpenseOfPublishingHouse(PID) -> return expense
- getNumberOfDistributors(Distributor ID) -> return distributor
- getRevenuePerCity(City) -> return revenue
- getRevenuePerLocation(Address) -> return revenue
- getRevenuePerDistributor(Distributor ID) -> return revenue
- getPaymentofEditorsPerTimePeriodperWork(EID,Amount,timeFrom,timeTo) ->return payment
- viewDistributorReports(Distributor ID, Total Shipping Cost, Date, Total Price, Number of Copies, Publication ID, City) -> return reports
- generateReportsForPublication(PID) -> return reports
- generateReportsForDistrbutor(Distributor ID) -> return reports
- viewEditorReports(Editor ID, Payment, Work type, Date) -> return reports
- viewInceptionFinanceReports(Total Distributor, Total shipping cost, Total revenue, Total Salary, End Date) -> return reports

6. Description of Views:

Distributor: The distributors can view publication information, information on orders they have made along with information on their balance. The distributor needs general information on the merchandise they are distributing along with any financial information associated with the merchandise(order costs, balance).

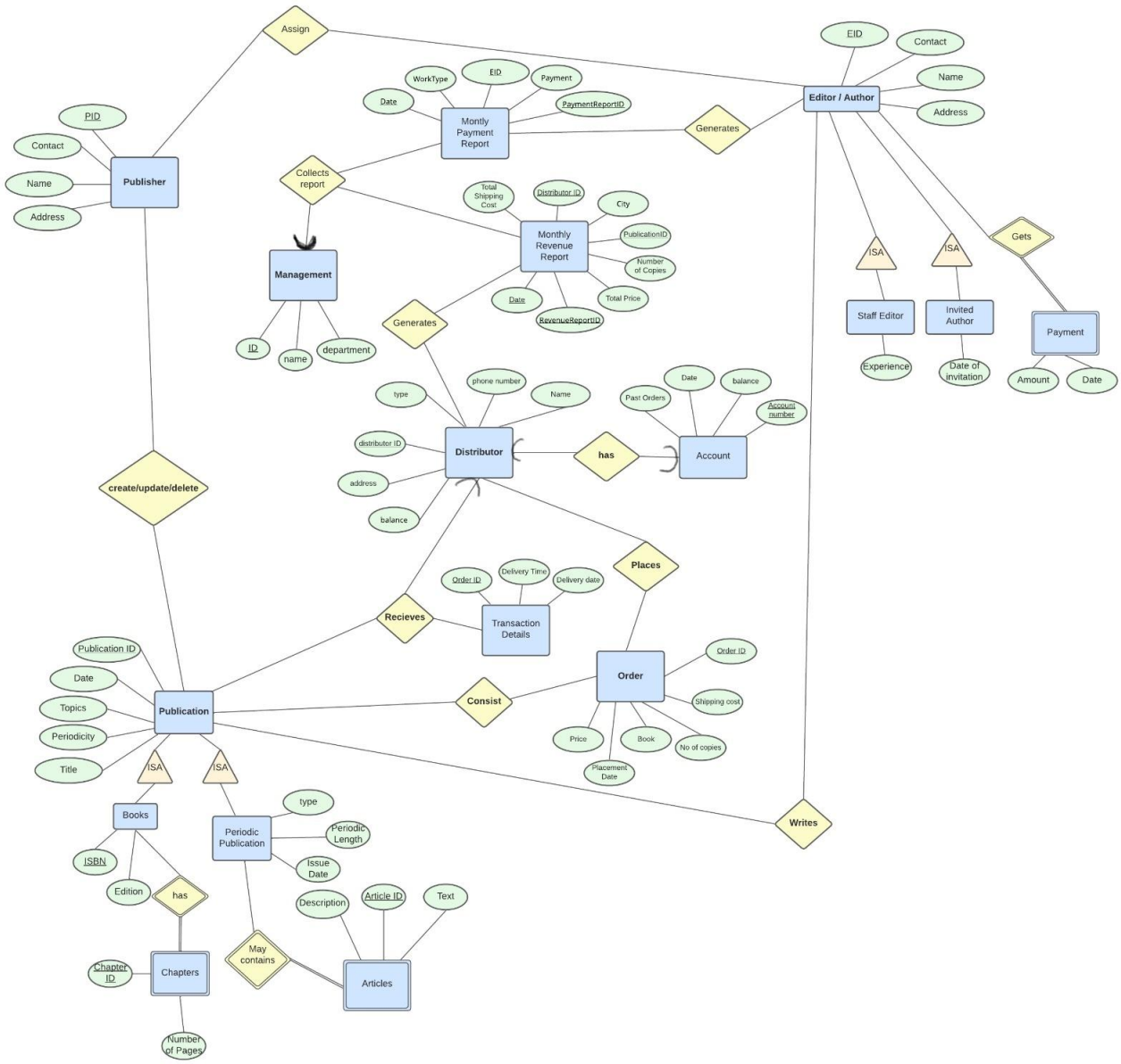
Editor /authors: The editors and authors need any book information along with information about other editors and authors. They also need information on the finances like how much they get paid. The editors and authors need information on the books they are working on along with any finances that they get from the book.

Management: The management is responsible for knowing all the monetary and report aspects of the WolfCity publishing house. The management should have access to the amount of books issued along with the amount distributors paid. They should also have information on the payment of the authors/editors.

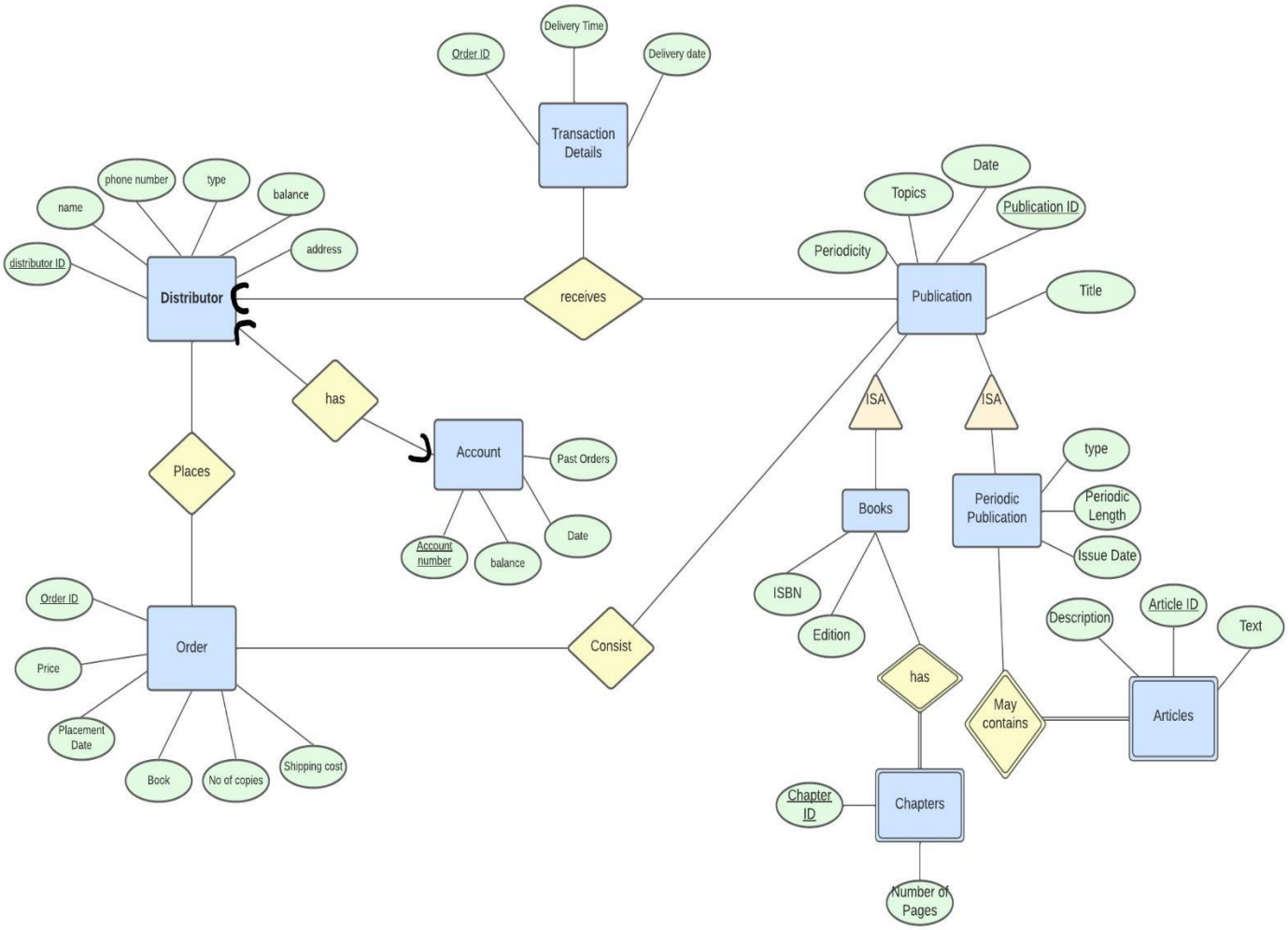
Publication House/Admin: The publication house is responsible for publishing books authors/editors send them along with knowing the distributors those books go to. As such the publication house should know information about the authors/editors that publish the books along with any information about the book itself. The publication house should also know who the distributors of the books are and the order details distributors make

7. Local E/R Diagrams:

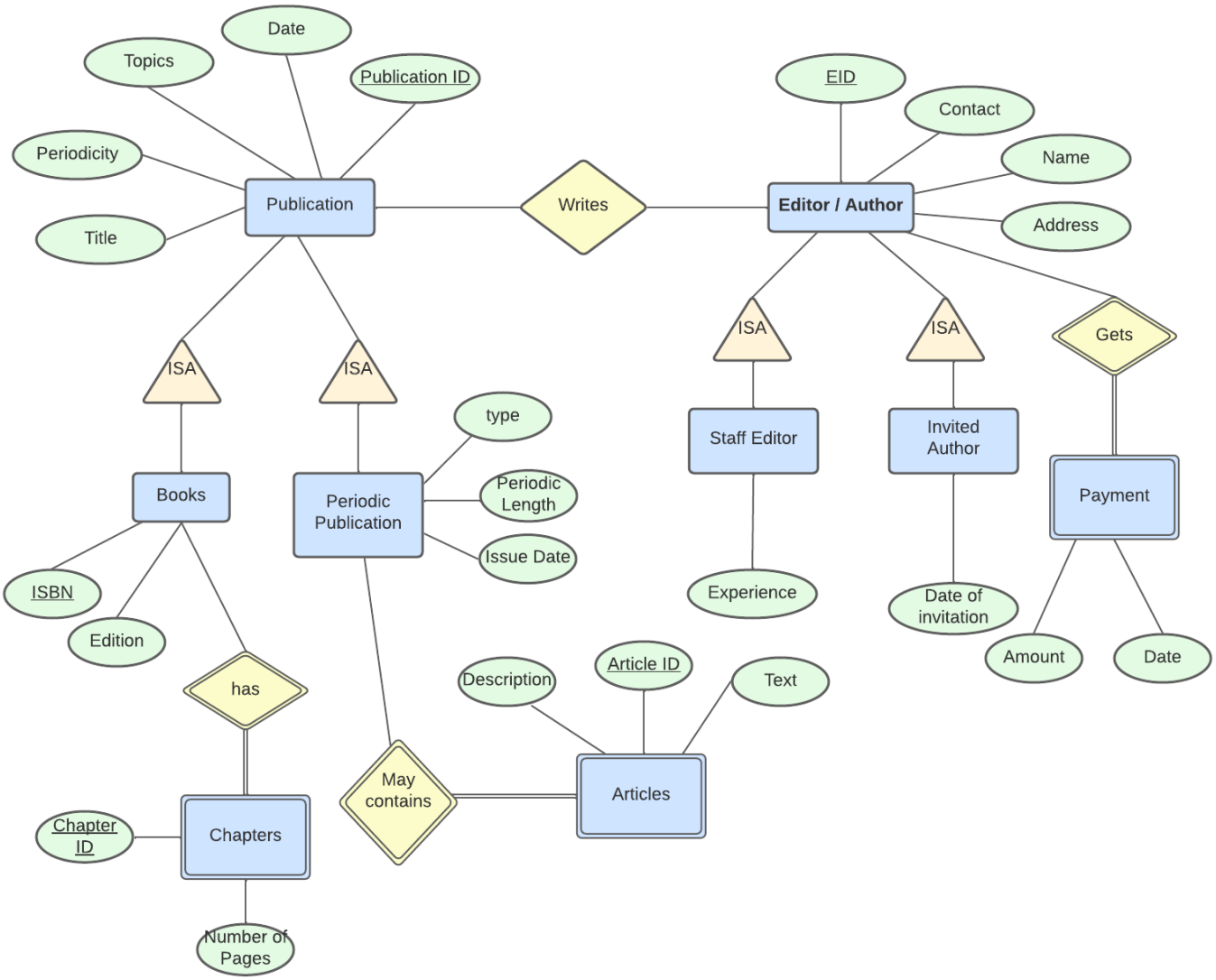
Admin's View:



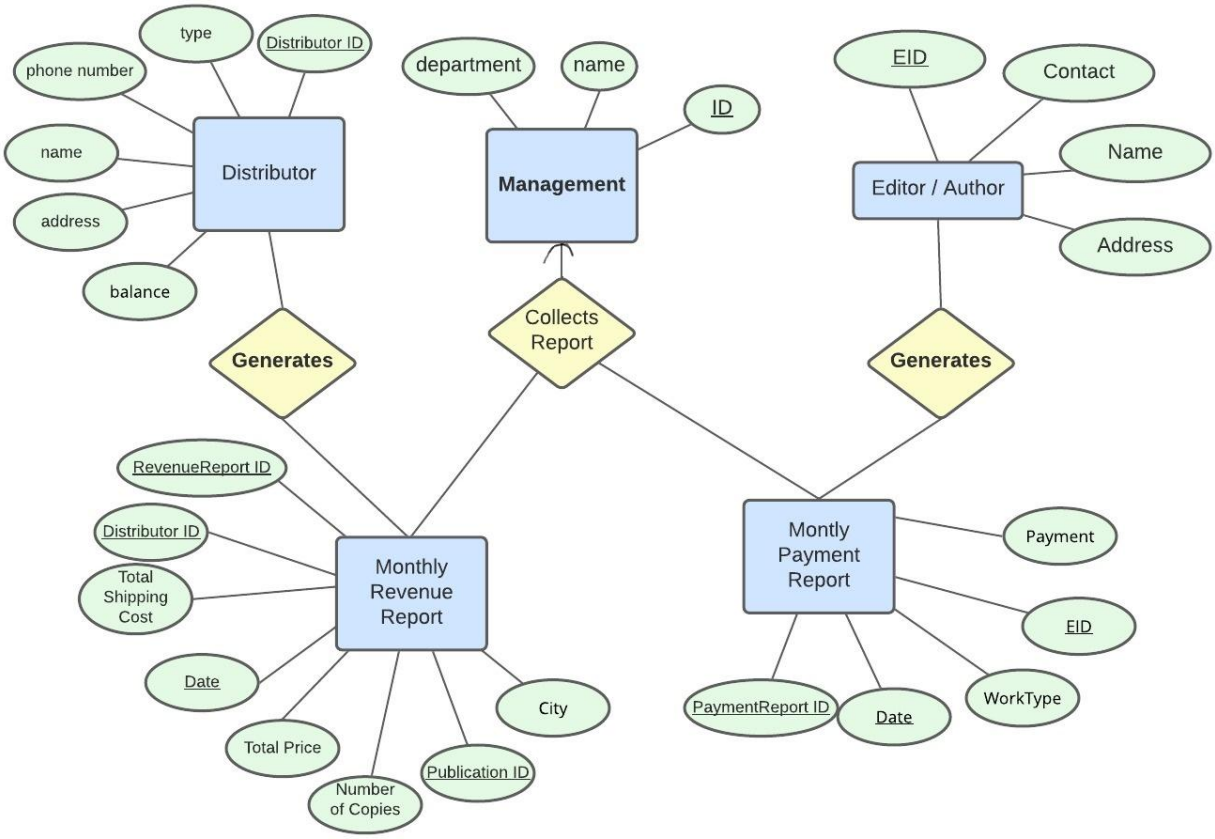
Distributor's View



Editors View



Management's View



8. Description of Local E/R diagrams:

Distributor: Distributor has a unique distributor ID as a key since no two distributors have the same ID. Many orders can be placed by many distributors. Distributors also generate monthly revenue reports.

Publishers: Publishers create, update, and delete publications and assign editors. The publisher also gives reports to the management

Publication: send orders to multiple distributors and have a unique PID. Publishers publish books and issue periodic publications. Many authors can collaborate and write for multiple publications

Orders: Orders have an associated order ID as a key. Publications take orders from distributors.

Editor/Author: Editor/Author has a unique key, EID. There can be a staff editor and invited author who writes for publications and gets payment which is a weak entity.

Books: Books are uniquely identified by ISBN and can have many chapters within it distinguished by chapter ID and ISBN.

Periodic Publication: Periodic publication may have articles with it making it a weak entity.

Account: Each distributor has an account consisting of an account number as a unique attribute along with details of balance and past orders.

Management: Management collects reports from the editor regarding the payment of regular and invited editors and authors. It also contains information about the distributor and publisher.

Monthly Revenue and Payment Report: The distributor produces a monthly revenue report(Gives the general information of how much books sold and money made per month in distributions) for the management while the editor/authors produce a monthly payment report(All the payment that need to be made to given author/editor for given month). As such both could have used the distributor ID and editor/author ID as part of their keys and become a weak entity set however we chose not to do this and opted to use a unique report ID for the key as there is a third relationship that connects both reports and using a special monthly revenue report id and monthly payment report id makes more sense than to use the distributor id and author id for the relationship.

9. Local Relational Schemas:

Management View:

- Management(ID, name, department)
- Distributor(distributor_ID, type, phone_number, name, address, balance)
- Editor/Author(EID, contact, name, address)
- MonthlyRevenueReport(revenue_report_ID, Distributor_ID,total_shipping_cost, date, total_price, number_copies, publication_ID, city)
- MonthlyPaymentReport(payment_report_ID, EID payment, workType, date)
- GenerateRevenueReport(distributor_ID, revenue_report_ID)
- GeneratePaymentReport(EID, payment_report_ID)
- CollectReport(ID, revenue_report_ID, payment_report_ID)

Distributor View:

- Distributor (distributor_ID, name, type, address, phone number,balance)
- PlacesOrder(distributor_ID, Order_ID)
- Order(Order_ID, price, Shipping cost, No of copies, Book, placement date)
- Transaction_details(Order_ID, delivery_time, delivery_date)
- Account(Account number, date, balance, past_orders)
- hasAccount(distributor_ID, Account number)
- consist(Order_ID, Publication_ID)
- Publication(Publication_ID, title, date, topics, periodicity)
- Books(Publication_ID, ISBN, edition)
- Chapters(Publication_ID, ChapterID, number_of_pages)
- Periodic_publication(Publication_ID, type, periodic_length, issue_date)
- Articles(Publication_ID, ArticleID, Description, text)

Editor View:

- Publication(Publication_ID, title, date, topics, periodicity)
- Writes(Publication_ID, EID)
- Editor(EID, name, address, contact)
- Staff editor(EID, experience)
- Invited author(EID, date_of_invitation)
- Payment(EID, amount, date)
- Books(Publication_ID, ISBN, edition)
- Chapters(Publication_ID, ChapterID, number_of_pages)
- Periodic_publication(Publication_ID, type, periodic_length, issue date)
- Articles(Publication_ID, ArticleID, Description, text)

Admin View:

- Editor/ Author(EID, name, address, contact)
- Staff editor(EID, experience)
- Invited author(EID, date_of_invitation)
- Payment(EID, amount, date)
- Publication(PublicationID, title, date, topics, periodicity)
- Publisher(PID,Contact,Name,Address)

- Books(PublicationID, ISBN, edition)
- Periodic_publication(PublicationID, type, periodic_length, issue date)
- Chapters(PublicationID, ChapterID, number_of_pages)
- Articles(PublicationID, ArticleID, Description, text)
- Management(ID, name, department)
- MonthlyRevenueReport(revenue_report_ID, Distributor_ID,total_shipping_cost, date, total_price, number_copies, publication_ID, city)
- MonthlyPaymentReport(payment_report_ID, EID, payment, workType, date)
- GenerateRevenueReport(distributor_ID, revenue_report_ID)
- GeneratePaymentReport(EID, payment_report_ID)
- CollectReport(ID, revenue_report_ID, payment_report_ID)
- Distributor (distributor ID, name, type, address, phone number, balance)
- PlacesOrder(distributor_id, Order_id)
- Order(Order ID, price, Shipping cost, No of copies, Book, placement date)
- Transaction_details(order_id, delivery_time, delivery_date)
- Account(Account_number, date, balance, past_orders)
- hasAccount(distributor_id, Account_number)
- consist(Order_id, publication_ID)
- assignsAuthors(PID, EID)
- CreateUpdateDeletePublication(PID, PublicationID)
- writesPublication(PublicationID, EID)

10. Local Schema Documentation:

Entity Sets to Relations:

- The following entities were converted into the relational schema using the same attributes: Publication, Publisher, Management, Distributors, Editors/Authors, MonthlyRevenueReport, MonthlyPaymentReport, TransactionDetails, Account, Order,
- All the relationships that were connected to the given entity sets above were transformed to a relation by taking the two unique keys from the entity sets the relationship was connecting as attributes.

Combining Multiway relationships to relations:

- The multiway relationship ‘CollectsReports' was converted to a relational schema with the uniqueID of the management and the unique ID of the monthly payment report and the monthly revenue report. This was done to reduce redundancy and to collect the reports by date.
- The multiway relationship “Receives” was also converted to a relational schema with the unique ID of the publication, Distributor, and transaction detail entity sets.

iSA to Relations:

- The entity sets that were subsets of another entity set were made into relations using the E/R viewpoint to reduce redundancy of information.
- The entity sets that were subsets of another entity are the following: StaffEditor, Invited Author, Books, and Periodic Publication.

Weak entity to Relations:

- Articles, Chapters, and Payment entity set were converted to a relation by taking the

unique key attributes of its own entity set and the supporting entity sets key attributes and placing them as its own attribute.

- None of the relationships connecting to the weak entity sets were turned into relations as there was no need to and because it would introduce redundancy.

Project report 2 Feedback and corrections :

1.(out of 80 points) 70
-10 some issues: insufficient 3NF justifications, why some potential FDs do not hold.

2.(out of 120 points) 115
-5 minor issue: Contradicting integrity constraints

3. (out of 80 points) 60
-20 many issues: multiple missing/mismatching integrity constraints and/or foreign references and/or primary keys as indicated in section 2.

Page Number	Changes
4,5,6,7	insufficient 3NF justifications
10,11,12,13	Contradicting integrity constraints
13, 14,15, 16, 17, 18, 19, 20	multiple missing/mismatching integrity constraints and/or foreign references and/or primary keys as indicated in section 2.

WolfPubDb Management System

For WolfCity publishing house

CSC 540 Database Management Systems
Project Report 2
Shakshi Singhai, Parth Thosani,
Nisarg Shah, Prashan Sengo,
March 10th, 2022

Assumptions:

1. Article length should be of max length- 1000
2. We assume that the mentions of Location is the city in the database.
3. One editor can only get one payment at a particular date.

1. Global Relational Database Schema:

- **Distributor (distributorID, Name, Type, Address, Phone, ContactPerson, Balance)**

  distributorID → distributorID, Name, Type, Address, Phone, ContactPerson, Balance

  distributorID is a unique primary id assigned to each and every distributor. This is also true for all the other attributes of the Distributor.As we have only one unique identifier and it's a superkey, we can derive all the attributes from the following distributorID and hence it's in BCNF. As it's in BCNF, it is also in 3NF.

  Name -> Name, Type, Address, Phone, ContactPerson, Balance cannot hold true as we can have multiple people with the same name and hence, it's not uniquely used to identify other values as there might be two people named "John" with different distributorID, different phone number and hence if we make name as unique identifier, it will lead to inconsistency when getting other values. Similar is with Type -> Type, name, address, phone, contactPerson, Balance. As we have 3 types of distributors, if we keep a unique identifier it will not correctly get other values as there are multiple rows for the same types.

- **Editor(EID, Name, Address, Contact)**

  EID → EID, Name, Address, Contact

  Every Editor is assigned a separate EID through which he/she is identified in the publishing house. Since EID is a superkey and none of its subset is, it is the primary key and the relation is in 3NF.

  Name, Address, Contact cannot be a unique identifier and we cannot get Name -> Address as for same name there can be multiple addresses as multiple people might have same name but different address. Also, with Name, Address, Contact cannot be the superkey as it cannot give EID.

- **StaffEditor(EID, Experience)**

  EID → EID, Experience

  This holds since this relation only has two attributes and EID is a superkey so it is in BCNF and therefore 3NF. As we are considering the E-R Approach, and staff editor is in IS-A relation with Editor/Author, we get only EID from the parent Editor/Author.

  Experience -> EID, Experience is not possible as we know that particular years of experience cannot give the EID for the particular user and hence it cannot be a superkey.

- **InvitedAuthor(EID, Date_of_invitation)**

  EID → EID, Date_of_invitation

It is a subclass of Editor and as we consider E-R approach, we take the EID from parent class along with date_of_invitation and we can get all the attributes and it is in BCNF because any relation with 2 attributes is in BCNF and hence it's in 3NF.

Date_of_invitation -> EID, Date_of_invitation is not possible as we know that particular date cannot give the EID for the particular user and hence it cannot be a superkey.

- **Payment(EID, Amount, Date)**

  EID→ EID, Amount, Date

  As each editor be it any type will have a common function of getting the payment and hence for every EID, it will have when they receive the payment along with the payment and hence this table will have all the attributes and it's unique based on unique EID and hence it's a superkey. If we don't have EID, we cannot specify for which editor the particular amount and when did they receive the payment. Also, the entity for payment is a weak entity.

  Date->Amount is Not possible and it fails as for a particular date there can be multiple transactions of payment and hence can be multiple amounts so we cannot derive it correctly.

- **Publication(PublicationID, Title, Date, Topics, Periodicity)**

  PublicationID → PublicationID, Title, Date, Topics, Periodicity

  PublicationID can be used to uniquely identify all the attributes. Since it is the minimal superkey, this relation is in BCNF.

  We need PublicationID as a unique identifier as we know that if we consider Date,Topics, Periodicity or title we cannot get the entire information about the publication of particular content.

- **Publisher (PID, Name, Contact, Address)**

  PID → PID, Name, Contact, Address

  PID can be used to uniquely identify all the attributes.

  Name, Address, Contact cannot be a unique identifier and we cannot get Name -> Address as for same name there can be multiple addresses as multiple people might have same name but different address. Also, with Name, Address, Contact cannot be the superkey as it cannot give PID.

- **Books (PublicationID, ISBN, Edition)**

  PublicationID → PublicationID, ISBN, Edition

Every book can be identified by the publication_ID. Moreover every edition can also be identified by the ISBN but we have assumed the publication_ID to be the primary key. Hence this relation is also in 3NF. Also, Edition -> ISBN does not hold true as we know that for a same edition number, it cannot have the same ISBN number. For example, Book A with ISBN X will have edition number 2 and Book B with ISBN Y will also have edition number 2 and hence it gives wrong information. Also Edition -> PublicationID is not possible as various books may have the same number of editions and hence it cannot be put into one tuple.

- **PeriodicPublication (PublicationID, Type, Periodic_length, Issue_date)**

  PublicationID → PublicationID, Type, Periodic_length, Issue_date

  We could get the details of periodic publication of content using publicationID. We know that each periodic type of any publication will have a different publicationID. For example, if a magazine has many issues, it will have the same type name as magazines but it will have different publication ID as the length and issue_date will be different. Hence, by publicationID we can get other attributes and it is a superkey and hence it's in 3NF and BCNF.

  The values type, periodic_length, and issue_date cannot be individually used as the superkey for a functional dependency as 2 publications can have the same values for the type, periodic_length, and issue_date as such there are no other non trivial FD beyond the one listed.

- **Chapters (PublicationID, ChapterID, Number_of_pages)**

  PublicationID, ChapterID → PublicationID, ChapterID, Number_of_pages

  Here both publicationID and ChapterID are used to uniquely identify the number of pages. It is possible that 2 different publications might have the same number of pages for the same chapterID or 2 chapters of the same publication to have the same number of pages. Thus a single attribute is not enough and we are bound to take a composite key consisting of PublicationId and ChapterID. Hence, it is necessary that the particular schema should be in 3NF and BCNF as both the attributes are superkeys.

- **Articles(PublicationID, ArticleID, Description, Text)**

  PublicationID, ArticleID → PublicationID, ArticleID, Description, Text

  Here an article is a weak entity and in order to identify an article uniquely we will need publication primary key(PublicationID) along with article ID and this will give us information about article description and text. Because the left hand side of the functional dependencies is a superkey, this relation is in BCNF and therefore 3NF. There is no other FD because no combination of the other attributes is sufficient to determine an Article uniquely. For an article description can be Null and can't be used to identify an article

- **Management(ManagementID, Name, Department)**

ManagementID → ManagementID, Name, Department

Management_ID successfully gives the name and department. We cannot have a name -> department or department -> name as we can have the same name but different departments and also for the same department we can have different names. Hence, they do not hold true. The only distinguishing attribute is management_ID which will denote uniquely the tuple even if the name and department happen to be the same.

- **MonthlyRevenueReport(<u>ReportID</u>, TotalShippingCost , Date, TotalPrice, NumberCopies, City)**

  ReportID → ReportID, TotalShippingCost, Date, TotalPrice, NumberCopies, City

  Corresponding to every monthly revenue report there is a unique ReportID that can be used to identify the distributor's report information like total_shipping_cost, date, total_price, number_copies, city. However if we consider any other attribute we can uniquely identify the other information relating to the report. Since the left-hand side is a superkey, this relation is in BCNF and therefore in 3NF.

- **MonthlyPaymentReport(<u>ReportID</u>, Payment, Date, WorkType)**

  ReportID → ReportID, Payment, Date, WorkType

  We can easily identify details of the report like date, worktype, etc. using the ReportID. This payment report corresponds to the payments made to the editors and staff of the publishing house. Since Report ID is the primary key and uniquely identifies. We can't use date as primary key as on same date we can have multiple reports likewise for a worktype we can have multiple reports

- **GenerateRevenueReport(<u>ReportID</u> , <u>DistributorID</u>)**

  ReportID , DistributorID → ReportID, DistributorID is in 3NF because it is in BCNF, as there are 2 attributes corresponding to the primary keys of CollectRevenueReport and Distributor.

- **GeneratePaymentReport (<u>ReportID,</u> <u>EID</u>)**
  ReportID , EID→ ReportID, EIDis in 3NF because it is in BCNF, as there are 2 attributes in this functional dependency, and both the attributes are keys.

- **CollectRevenueReport(<u>ManagementID</u>, <u>ReportID</u>)**

  ManagementID, ReportID→ ManagementID, ReportID

This relation involves collecting unique keys from both the entities i.e. Monthly revenue report and management. Since it has 2 attributes, this relation is in BCNF.

- **CollectPaymentReport(<u>ManagementID</u>, <u>ReportID</u>)**

  ManagementID, ReportID → ManagementID, ReportID

  It is a relationship between Management and Monthly Payment Report. It has 2 attributes and hence is in BCNF.

- **PlacesOrder(<u>DistributorID</u>, <u>OrderID</u>)**

  DistributorID, OrderID → DistributorID, OrderID

  Is in BCNF and therefore 3NF because it only contains two attributes, which are in the superkey.

- **Order(<u>OrderID</u>, Price, ShippingCost, NumCopies, Book, Date)**

  OrderID → OrderID , Price, ShippingCost, NumCopies, Book, Date

  This holds because an Order_ID is used to uniquely identify all the details regarding an order like price, Shipping cost, No of copies, book, date. No other attributes can be used in combination to do this as multiple orders can have the same cost and same number of copies or same book and can be ordered on the same date. Because the left hand side is a super key, it is in BCNF (and thereby 3NF).

- **TransactionDetails(<u>OrderID</u>, DeliveryTime, DeliveryDate)**

  OrderID → OrderID , DeliveryTime, DeliveryDate

  This holds true as an Order_ID can uniquely identify all the details related to a transaction that is delivery_time and delivery_date. Any other attributes can't be used to uniquely identify as there can be multiple transactions in a day at a time. Since the left hand side of the FD is a superkey, then this FD holds in 3NF

- **Account(<u>AccountNum</u>, Date, Balance, PastOrders)**

  AccountNum → AccountNum, Date, Balance, PastOrders

  This holds because through an account number we can get other details of the distributor's account like bank balance of any particular day or number of past orders. But vice versa is not true through date, or balance we can't get the account number. Since all left-hand sides are superkeys, this relation is in BCNF and therefore in 3NF

- **hasAccount(<u>AccountNum</u> , <u>DistributorID</u>)**

AccountNum , DistributorID → AccountNum , DistributorID

It is in 3NF because it is in BCNF, as there are two attributes in this functional dependency, and those are superkeys.

- **Consist(<u>OrderID</u>, <u>PublicationID</u>)**

  OrderID, PublicationID→ OrderID, PublicationID is in BCNF and therefore 3NF because it only contains two attributes, which are in the superkey.

- **assignsAuthors(<u>PID</u>, <u>EID</u>)**

  PID, EID → PID, EID

  Is in BCNF and therefore 3NF because it only contains two attributes, which are in the superkey. Publisher assigns an editor and therefore this entity should have the unique key of both Publisher and editor.

- **CreateUpdateDeletePublication(<u>PID</u>, <u>PublicationID</u>)**

  PID, PublicationID → PID, PublicationID

  Is in BCNF and therefore 3NF because it only contains two attributes, which are in the superkey.

- **writesPublication(<u>PublicationID</u>, <u>EID</u>)**

  PublicationID, EID → PublicationID, EID

  Is in BCNF and therefore 3NF because it only contains two attributes, which are in the superkey.

2. Design for Global Schema:

<u>Design decision for global schema:</u>

We have considered the broad sections (Editors, Management, Publication, Publisher, Distributor) for translating the diagrams and making sure that there's appropriate mapping of data for all the entities and relations and there's no extra storage of redundant data. We have considered the admin's view primarily as it covers the majority of the functionalities and views of other stakeholders as well.

We have decomposed the CollectReport relation which management takes in from Editor and distributor and have divided it into CollectRevenueReport, CollectPaymentReport as we know that they both don't have anything in common apart from management_ID and hence if we consider only one table for both, we will have null values in the table.

For example, if we consider the Revenue report, the column for Payment report will have null and vice versa and this will increase the storage costs and will reduce the speed of query recovery. Various other relations take in the primary key of the connecting entities as it is defined by them. We have mentioned a unique identifier (primary key) in the majority of the schemas but some also have composite keys. We ensured that all the relations follow strict normalization which would help the system retrieve information in a fast and correct way.

- **Distributor (distributorID, Name, Type, Address, Phone, ContactPerson, Balance)**
  - distributorID is the primary key and it cannot be null
  - Other attributes like, Name, Type, Address, Phone are not allowed to be null.
  - Balance can be a null value. The null value means here that a distributor has 0 balance in their account.
  - ContactPerson can be null value as even without a contact person we can reach the distributor through the Name, Address and Phone number. Therefore, a null Contact person signifies a distributor which has no alternative contact person.

- **Editor(EID, Name, Address, Contact)**
  - EID is the primary key and it cannot be null
  - Name, Address and Contact can't be null.

- **StaffEditor(EID, Experience)**
  - EID is the primary key, foreign key from Editor and it cannot be null
  - Experience can't be null

- **InvitedAuthor(EID, Date_of_invitation)**
  - EID is the primary key, foreign key from Editor and it cannot be null
  - Date_of_invitation can't be null

- **Payment(EID, Amount, Date)**
  - EID and Date are the primary key.
  - EID is the foreign key from Editor and it cannot be null.
  - Amount and Date can't be null as we will record the Amount and the Date when an Editor gets the payment.

- **Publication(PublicationID, Title, Date, Topics, Periodicity)**
  - PublicationID is the primary key and it cannot be null
  - Title and Date can't be null.
  - Periodicity can be null as the publication can be published only at that time.
  - Topics can be null as it can mean the publication is too general to be classified into a specific genre.

- **Publisher (PID, Name, Contact, Address)**
  - PID is the primary key and it cannot be null

    - Name, Address and Contact can't be null

- **Books (PublicationID, ISBN, Edition)**
    - PublicationID is the primary key, foreign key from Publication and it cannot be null
    - ISBN and edition are not allowed to be null.

- **PeriodicPublication (PublicationID, Type, Periodic_length, Issue_date)**
    - PublicationID is the primary key, foreign key from Publication and it cannot be null
    - Type, periodic_length, and issue_date are not allowed to be null

- **Chapters (PublicationID, ChapterID, Number_of_pages)**
    - PublicationID and ChapterID forms the primary key as this is a weak entity set and it cannot be null
    - PublicationID is a foreign key from the publications table
    - Number_of_pages are not allowed to be null

- **Articles(PublicationID, ArticleID, Description, Text)**
    - PublicationID and ArticleID forms the primary key as this is a weak entity set and it cannot be null
    - Text is not allowed to be null
    - PublicationID is a foreign key from the publications table
    - Description can be null as it's not necessary for an article to have a description about it. We can have articles that have text alone and not description. Therefore, a null description means an article doesn't have any description and just text.

- **Management(ManagementID, Name, Department)**
    - ManagementID is the primary key and it cannot be null
    - Name and Department are not allowed to be null.

- **MonthlyRevenueReport(RevenueReportID, TotalShippingCost , Date, TotalPrice, NumberCopies, City)**
    - RevenueReportID is the primary key and it cannot be null
    - TotalShippingCost , TotalPrice, NumberCopies, Date and City cannot be null as the report needs to be from a particular month/year combo and from a particular place. Along with that the cost and price can be at minimum 0.

- **MonthlyPaymentReport(PaymentReportID, Payment, Date, WorkType)**
    - PaymentReportID is the primary key and it cannot be null
    - WorkType,Date and Payment can't be null as the report belongs to a particular month, year combo and for a given worktype.

- **GenerateRevenueReport(RevenueReportID , DistributorID)**
    - RevenueReportID and DistributorID forms primary keys and it cannot be null

    - DistributorID is a foreign key from the distributors table and RevenueReportID is the foreign key from MonthlyRevenueReport

- **GeneratePaymentReport (PaymentReportID, EID)**
    - PaymentReportID and EID forms primary keys and it cannot be null
    - EID is a foreign key from the editors table and PaymentReportID is the foreign key from MonthlyPaymentReport

- **CollectRevenueReport(ManagementID, RevenueReportID)**
    - ManagementID and Revenue ReportID forms the primary key and it cannot be null
    - ManagementID is a foreign key from the management table and ReportID is the foreign key from MonthlyRevenueReport

- **CollectPaymentReport(ManagementID, PaymentReportID)**
    - ManagementID and PaymentReportID forms the primary key and it cannot be null
    - ManagementID is a foreign key from the management table and PaymentReportID is the foreign key from MonthlyPaymentReport

- **PlacesOrder(DistributorID, OrderID)**
    - DistributorID and OrderID forms the primary key and it cannot be null
    - DistributorID is a foreign key from the distributor table and OrderID is a foreign key from the order table

- **Order(OrderID, Price, ShippingCost, NumCopies, Book, Date)**
    - Order ID is the primary key and it cannot be null
    - Price, ShippingCost, NumCopies, Book and Date can't be null

- **TransactionDetails(OrderID, DeliveryTime, DeliveryDate)**
    - Order ID is the primary key and it cannot be null.
    - DeliveryTime , DeliveryDate cannot be null
    - OrderID is a foreign key from the order table

- **Account(AccountNum, Date, Balance, PastOrders)**
    - AccountNum is the primary key and it cannot be null
    - Date, PastOrders and Balance can't be null

- **hasAccount(AccountNum , DistributorID)**
    - Distributor ID and AccountNum forms the primary key and it cannot be null
    - Referential integrity exists between distributor and account number as each distributor can have only 1 account and each account only belongs to one distributor
    - distributorID is a foreign key from the distributor table and AccountNum is the foreign key from Account.

- **Consist(OrderID, PublicationID)**
  - Order ID and PublicationID forms the primary key and it cannot be null
  - OrderID is a foreign key from the order table and PublicationID is a foreign key from the publications table

- **assignsAuthors(PID, EID)**
  - PID and EID forms the primary key and it cannot be null
  - EID is a foreign key from the editors table and PID is a foregin key from the Publisher table.

- **CreateUpdateDeletePublication(PID, PublicationID)**
  - PID and PublicationID forms the primary key and it cannot be null
  - PublicationID is a foreign key from the publication table and PID is a foregin key from the Publisher table.

- **writesPublication(PublicationID, EID)**
  - PublicationID and EID forms the primary key and it cannot be null
  - EID is a foreign key from the Editor table and PublicationID is the foreign key from the Publication table.

3. Base Relations:

CREATE TABLES:

```
CREATE TABLE IF NOT EXISTS Distributor(
   distributorID INT AUTO_INCREMENT,
   Name VARCHAR(128) NOT NULL,
   Type VARCHAR(128) NOT NULL,
   Address VARCHAR(256) NOT NULL,
   City VARCHAR(128),
   Phone VARCHAR(16) NOT NULL,
   ContactPerson VARCHAR(128),
   Balance DECIMAL(9,2) NULL,
   PRIMARY KEY(distributorID)
);

CREATE TABLE IF NOT EXISTS Editor(
   EID INT AUTO_INCREMENT,
   Name VARCHAR(128) NOT NULL,
   Address VARCHAR(256) NOT NULL,
   Contact VARCHAR(16) NOT NULL,
```

```
    PRIMARY KEY(EID)
);

CREATE TABLE IF NOT EXISTS StaffEditor(
    EID INT,
    Experience VARCHAR(128) NOT NULL,
    PRIMARY KEY(EID),
    FOREIGN KEY(EID) REFERENCES Editor(EID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS InvitedAuthor(
    EID INT,
    Date_of_invitation DATE NOT NULL,
    PRIMARY KEY(EID),
    FOREIGN KEY(EID) REFERENCES Editor(EID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS Payment(
    EID INT,
    Amount DECIMAL(9,2) NOT NULL,
    Date DATE NOT NULL,
    PRIMARY KEY(EID, Date),
    FOREIGN KEY(EID) REFERENCES Editor(EID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS Publication(
    PublicationID INT AUTO_INCREMENT,
    Title VARCHAR(128) NOT NULL,
    Date DATE NOT NULL,
    Topics VARCHAR(128),
    Periodicity VARCHAR(128),
    Price DECIMAL(9,2) NOT NULL,
    EID INT,
    PRIMARY KEY(PublicationID),
    FOREIGN KEY(EID) REFERENCES Editor(EID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
```

```
);

CREATE TABLE IF NOT EXISTS Publisher(
   PID INT AUTO_INCREMENT,
   Name VARCHAR(128) NOT NULL,
   Contact VARCHAR(16) NOT NULL,
   Address VARCHAR(256) NOT NULL,
   PRIMARY KEY(PID)
);

CREATE TABLE IF NOT EXISTS Books(
   PublicationID INT AUTO_INCREMENT,
   ISBN VARCHAR(128) NOT NULL,
   Edition VARCHAR(128) NOT NULL,
   PRIMARY KEY(PublicationID),
   FOREIGN KEY(PublicationID) REFERENCES Publication(PublicationID)
   ON UPDATE CASCADE
   ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS PeriodicPublication(
   PublicationID INT,
   Type VARCHAR(128) NOT NULL,
   Periodic_length INT NOT NULL,
   Issue_date DATE NOT NULL,
   PRIMARY KEY(PublicationID),
   FOREIGN KEY(PublicationID) REFERENCES Publication(PublicationID)
   ON UPDATE CASCADE
   ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS Chapters(
   PublicationID INT,
   ChapterID INT,
   Number_of_pages INT NOT NULL,
   PRIMARY KEY(PublicationID, ChapterID),
   FOREIGN KEY(PublicationID) REFERENCES Publication(PublicationID)
   ON UPDATE CASCADE
   ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS Articles(
   PublicationID INT,
   ArticleID INT,
```

```
    Description VARCHAR(128),
    Text VARCHAR(1000) NOT NULL,
    PRIMARY KEY(PublicationID, ArticleID),
    FOREIGN KEY(PublicationID) REFERENCES Publication(PublicationID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS Management(
    ManagementID INT AUTO_INCREMENT,
    Name VARCHAR(128) NOT NULL,
    Department VARCHAR(128) NOT NULL,
    PRIMARY KEY(ManagementID)
);

CREATE TABLE IF NOT EXISTS MonthlyRevenueReport(
    RevenueReportID INT,
    TotalShippingCost DECIMAL(9,2) NOT NULL,
    Date Date NOT NULL,
    TotalPrice DECIMAL(9,2) NOT NULL,
    NumberCopies INT NOT NULL,
    City VARCHAR(128) NOT NULL,
    PRIMARY KEY(RevenueReportID )
);

CREATE TABLE IF NOT EXISTS MonthlyPaymentReport(
    PaymentReportID INT,
    Payment DECIMAL(9,2) NOT NULL,
    Date Date NOT NULL,
    WorkType VARCHAR(128) NOT NULL,
    PRIMARY KEY(PaymentReportID )
);

CREATE TABLE IF NOT EXISTS GenerateRevenueReport(
    RevenueReportID INT,
    DistributorID INT,
    PRIMARY KEY(RevenueReportID,DistributorID),
    FOREIGN KEY(RevenueReportID) REFERENCES
MonthlyRevenueReport(RevenueReportID)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    FOREIGN KEY(DistributorID) REFERENCES Distributor(DistributorID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
```

```
);

CREATE TABLE IF NOT EXISTS GeneratePaymentReport(
   PaymentReportID INT,
   EID INT,
   PRIMARY KEY(PaymentReportID,EID),
   FOREIGN KEY(PaymentReportID) REFERENCES
MonthlyPaymentReport(PaymentReportID)
   ON UPDATE CASCADE
   ON DELETE CASCADE,
   FOREIGN KEY(EID) REFERENCES Editor(EID)
   ON UPDATE CASCADE
   ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS CollectRevenueReport(
   ManagementID INT,
   RevenueReportID INT,
   PRIMARY KEY(RevenueReportID,ManagementID),
   FOREIGN KEY(ManagementID) REFERENCES Management(ManagementID)
   ON UPDATE CASCADE
   ON DELETE CASCADE,
   FOREIGN KEY(RevenueReportID) REFERENCES
MonthlyRevenueReport(RevenueReportID)
   ON UPDATE CASCADE
   ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS CollectPaymentReport(
   ManagementID INT,
   PaymentReportID INT,
   PRIMARY KEY(PaymentReportID,ManagementID),
   FOREIGN KEY(ManagementID) REFERENCES Management(ManagementID)
   ON UPDATE CASCADE
   ON DELETE CASCADE,
   FOREIGN KEY(PaymentReportID) REFERENCES
MonthlyPaymentReport(PaymentReportID)
   ON UPDATE CASCADE
   ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS Orders(
   OrderID INT AUTO_INCREMENT,
   Price Decimal(9,2) NOT NULL,
```

```
    ShippingCost Decimal(9,2) NOT NULL,
    NumCopies INT NOT NULL,
    PublicationID INT,
    Date DATE NOT NULL,
    PRIMARY KEY(OrderID),
    FOREIGN KEY(PublicationID) REFERENCES Publication(PublicationID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS AddOrUpdateOrder(
    DistributorID INT,
    OrderID INT,
    Amount Decimal(9,2) Not NULL,
    Status INT,
    PRIMARY KEY(DistributorID, OrderID),
    FOREIGN KEY(DistributorID) REFERENCES Distributor(DistributorID)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    FOREIGN KEY(OrderID) REFERENCES Orders(OrderID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS TransactionDetails(
    OrderID INT,
    DeliveryDate DATE NOT NULL,
    PRIMARY KEY(OrderID),
    FOREIGN KEY(OrderID) REFERENCES Orders(OrderID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS Account(
    AccountNum INT,
    Date DATE NOT NULL,
    Balance DECIMAL(9,2) NOT NULL,
    PastOrders INT NOT NULL,
    PRIMARY KEY(AccountNum)
);

CREATE TABLE IF NOT EXISTS hasAccount(
    AccountNum INT,
    DistributorID INT NOT NULL,
```

```
    PRIMARY KEY(AccountNum, DistributorID),
    FOREIGN KEY(AccountNum) REFERENCES Account(AccountNum)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    FOREIGN KEY(DistributorID) REFERENCES Distributor(DistributorID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS Consist(
    OrderID INT,
    PublicationID INT,
    PRIMARY KEY(OrderID,PublicationID),
    FOREIGN KEY(OrderID) REFERENCES Orders(OrderID)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    FOREIGN KEY(PublicationID) REFERENCES Publication(PublicationID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS assignsAuthors(
    PID INT,
    EID INT,
    PRIMARY KEY(PID,EID),
    FOREIGN KEY(PID) REFERENCES Publisher(PID)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    FOREIGN KEY(EID) REFERENCES Editor(EID)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE IF NOT EXISTS CreateUpdateDeletePublication(
    PID INT,
    PublicationID INT,
    PRIMARY KEY(PID,PublicationID),
    FOREIGN KEY(PID) REFERENCES Publisher(PID)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    FOREIGN KEY(PublicationID) REFERENCES Publication(PublicationID)
    on UPDATE CASCADE
    ON DELETE CASCADE
);
```

```
CREATE TABLE IF NOT EXISTS writesPublication(
   PublicationID INT,
   EID INT,
   PRIMARY KEY(EID,PublicationID),
   FOREIGN KEY(PublicationID) REFERENCES Publication(PublicationID)
   ON UPDATE CASCADE
   ON DELETE CASCADE,
   FOREIGN KEY(EID) REFERENCES Editor(EID)
   ON UPDATE CASCADE
   ON DELETE CASCADE
);
```

```
+--------------------------------------------------------------------------------+
|                                                                                |
+--------------------------------------------------------------------------------+
| ----------------------------CREATING TABLES.....---------------------------- |
+--------------------------------------------------------------------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)
```

SELECT QUERIES:

SELECT * FROM `Distributor`;

distributorID	Name	Type	Address	Phone	ContactPerson	Balance
1	john	Wholesale	2518 Avent Ferry Road	+19848883402	Mary	400
2	Kevin	Wholesale	2512 Avent Ferry Road	+19842239405	Jake	200
3	Mark	Bookstore	2520 Avent Ferry Road	+191041239420	Lily	64
4	Steve	Library	2514 Avent Ferry Road	+19843483402	Mary	400
5	Mike	Library	2517 Avent Ferry Road	+19863245402	Lucy	772

5 rows in set (0.00 sec)

SELECT * FROM `Editor`;

EID	Name	Address	Contact
1	Johnny	2512 Gorman Street Road Raleigh NC	9848883402
2	Kevin	2122 Gorman Street Road	+19842239405
3	Mark	2520 Gorman Street Road	+191041239420
4	Steve	2514 Gorman Street Road	+19841298765
5	Mike	2517 Gorman Street Road	+19863245402

5 rows in set (0.00 sec)

SELECT * FROM `StaffEditor`;

EID	Experience
1	2 Years
2	3 Years
3	4 Years
4	1.5 Years
5	6 Years

5 rows in set (0.00 sec)

SELECT * FROM `InvitedAuthor`;

EID	Date_of_invitation
1	2022-01-07
2	0000-00-00
3	2022-03-02
4	2021-01-04
5	2021-01-06

5 rows in set (0.00 sec)

SELECT * FROM `Payment`;

EID	Amount	Date
1	102	2022-01-21
2	942	2022-02-04
3	1200	2022-03-19
4	512	2021-01-13
5	765	2021-01-26

5 rows in set (0.01 sec)

SELECT * FROM `Publication`;

PublicationID	Title	Date	Topics	Periodicity
1	Introduction to Computing	2020-01-21	Science	Yearly
2	Intro to Database Management System	2020-01-04	Technology	Yearly
3	Autobiography of Sachin Tendulkar	2019-03-19	Sports	Yearly
4	Mathematics Edition 1	2021-02-13	Mathematics	Monthly
5	Marvel Comics	2021-01-26	Fictional	Weekly

5 rows in set (0.00 sec)

SELECT * FROM `Publisher`;

PID	Name	Contact	Address
1	Angel	+19848883402	2512 Kensington Park Raleigh NC
2	Paul	+19842239405	2122 Kensington Park Road
3	Suzane	+191041239420	2620 Kensington Park Road
4	Maria	+19841298765	2524 Kensington Park Road
5	Michael	+19863245402	2519 Kensington Park Road

5 rows in set (0.00 sec)

SELECT * FROM `Books`;

PublicationID	ISBN	Edition
1	978-3-16-148410-0	1st Edition
2	978-3-16-148410-1	2nd Edition
3	978-3-16-148410-2	4th Edition
4	978-3-16-148410-3	1st Edition
5	978-3-16-148410-4	2nd Edition

5 rows in set (0.00 sec)

SELECT * FROM `Chapters`;

PublicationID	ChapterID	Number_of_pages
1	1	400
2	2	500
3	3	600
4	4	700
5	5	800

5 rows in set (0.00 sec)

SELECT * from `PeriodicPublication`;

PublicationID	Type	Periodic_length	Issue_date
1	Journal	2	2020-10-05
2	Magazine	5	2020-06-21
3	Magazine	1	2020-09-13
4	Journal	8	2020-03-03

4 rows in set (0.00 sec)

SELECT * FROM `Articles`;

PublicationID	ArticleID	Description	Text
1	1	AGN as potential factories for eccentric black hole mergers	There is some weak evidence that the black hole merger named GW190521 had a non-zero eccentricity. In addition, the masses of the component black holes exceeded the limit predicted by stellar evolution. The large masses can be explained by successive mergers which may be efficient in gas disks surrounding active galactic nuclei, but it is difficult to maintain an eccentric orbit all the way to the merger, as basic physics would argue for circularization6. Here we show that active galactic nuclei disk environments can lead to an excess of eccentric mergers, if the interactions between single and binary black holes are frequent5 and occur with mutual inclinations of less than a few degrees. We further illustrate that this eccentric population has a different distribution of the inclination between the spin vectors of the black holes and their orbital angular momentum at merger referred to as the spin–orbit tilt, compared with the remaining circular mergers.
2	2	Electron-catalysed molecular recognition	Molecular recognition and supramolecular assembly cover a broad spectrum of non-covalently orchestrated phenomena between molecules. Catalysis of such processes, however, unlike that for the formation of covalent bonds, is limited to approaches that rely on sophisticated catalyst design. Here we establish a simple and versatile strategy to facilitate molecular recognition by extending electron catalysis which is widely applied in synthetic covalent chemistry, into the realm of supramolecular non-covalent chemistry.
3	3	Weak cubic CaSiO3 perovskite in the Earth's mantle	Cubic CaSiO3 perovskite is a major phase in subducted oceanic crust, where it forms at a depth of about 550 kilometres from majoritic garnet. However, its rheological properties at temperatures and pressures typical of the lower mantle are poorly known. Here we measured the plastic strength of cubic CaSiO3 perovskite at pressure and temperature conditions typical for a subducting slab up to a depth of about 1200 kilometres. In contrast to tetragonal CaSiO3, previously investigated at room temperature we find that cubic CaSiO3 perovskite is a comparably weak phase at the temperatures of the lower mantle.
4	4	A non-canonical tricarboxylic acid cycle underlies cellular identity	The tricarboxylic acid (TCA) cycle is a central hub of cellular metabolism, oxidizing nutrients to generate reducing equivalents for energy production and critical metabolites for biosynthetic reactions. Despite the importance of the products of the TCA cycle for cell viability and proliferation, mammalian cells display diversity in TCA-cycle activity1. How this diversity is achieved, and whether it is critical for establishing cell fate, remains poorly understood. Here we identify a non-canonical TCA cycle that is required for changes in cell state. Genetic co-essentiality mapping revealed a cluster of genes that is sufficient to compose a biochemical alternative to the canonical TCA cycle, wherein mitochondrially derived citrate exported to the cytoplasm is metabolized by ATP citrate lyase, ultimately regenerating mitochondrial oxaloacetate to complete this non-canonical TCA cycle.
5	5	Ribosome collisions induce mRNA cleavage and ribosome rescue in bacteria	Ribosome rescue pathways recycle stalled ribosomes and target problematic mRNAs and aborted proteins for degradation. In bacteria, it remains unclear how rescue pathways distinguish ribosomes stalled in the middle of a transcript from actively translating ribosomes3. Here, using a genetic screen in Escherichia coli, we discovered a new rescue factor that has endonuclease activity. SmrB cleaves mRNAs upstream of stalled ribosomes, allowing the ribosome rescue factor tmRNA (which acts on truncated mRNAs3) to rescue upstream ribosomes.

5 rows in set (0.00 sec)

SELECT * FROM `Management`;

ManagementID	Name	Department
1	John	Financial
2	Steve	Distribution
3	Mike	Financial
4	Jake	Distribution

4 rows in set (0.00 sec)

SELECT * FROM `MonthlyRevenueReport`;

```
+----------+---------+------------+--------------------+
| ReportID | Payment | Date       | WorkType           |
+----------+---------+------------+--------------------+
|        1 | 1203.41 | 2021-03-20 | Book Authorship    |
|        2 |  921.41 | 2021-03-20 | Article Authorship |
|        3 | 1203.41 | 2021-03-05 | Editorial Work     |
|        4 | 1203.41 | 2021-03-20 | Article Authorship |
+----------+---------+------------+--------------------+
4 rows in set (0.00 sec)
```

SELECT * FROM `MonthlyPaymentReport`;

```
+----------+------------------+------------+------------+-------------+------------+
| ReportID | TotalShippingCost | Date       | TotalPrice | NumberCopies | City       |
+----------+------------------+------------+------------+-------------+------------+
|        1 |           231.40 | 1989-05-14 |      20.00 |          40 | Cary       |
|        2 |          2541.40 | 1989-05-14 |      65.00 |          30 | Durham     |
|        3 |          2674.40 | 1989-05-14 |      48.00 |          70 | Charlette  |
|        4 |          2021.40 | 1989-05-14 |      70.00 |          40 | Greensboro |
+----------+------------------+------------+------------+-------------+------------+
4 rows in set (0.00 sec)
```

SELECT * FROM `GenerateRevenueReport`;

```
+----------+---------------+
| ReportID | DistributorID |
+----------+---------------+
|        1 |             1 |
|        1 |             2 |
|        2 |             1 |
|        2 |             3 |
+----------+---------------+
4 rows in set (0.00 sec)
```

SELECT * FROM `GeneratePaymentReport`;

ReportID	EID
1	1
1	2
2	1
2	3

4 rows in set (0.00 sec)

SELECT * FROM `Orders`;

OrderID	Price	ShippingCost	NumCopies	Book	Date
1	100.00	100.00	1	The Train	2022-01-21
2	200.00	200.00	2	Science Test	2022-01-22
3	300.00	300.00	3	Maths Book	2022-01-23
4	400.00	400.00	4	English Book	2022-01-24
5	500.00	500.00	5	Social Science	2022-01-25

5 rows in set (0.01 sec)

SELECT * FROM `TransactionDetails`;

OrderID	DeliveryTime	DeliveryDate
1	2022-01-21 15:35:07	2022-01-21
2	2022-01-22 15:35:07	2022-01-22
3	2022-01-23 15:35:07	2022-01-23
4	2022-01-24 15:35:07	2022-01-24
5	2022-01-25 15:35:07	2022-01-25

5 rows in set (0.00 sec)

SELECT * FROM `Account`;

AccountNum	Date	Balance	PastOrders
1	2022-01-21	100.00	23
2	2022-01-22	200.00	45
3	2022-01-23	300.00	32
4	2022-01-24	400.00	63
5	2022-01-25	500.00	79

5 rows in set (0.00 sec)

SELECT * FROM `hasAccount`;

AccountNum	DistributorID
1	1
2	2
3	3
4	4
5	5

5 rows in set (0.00 sec)

SELECT * FROM `Consist`;

OrderID	PublicationID
1	1
2	2
3	3
4	4
5	5

5 rows in set (0.00 sec)

SELECT * FROM `assignsAuthors`;

PID	EID
1	1
2	2
3	3
4	4
5	5

5 rows in set (0.00 sec)

SELECT * FROM `CreateUpdateDeletePublication`;

PID	PublicationID
1	1
2	2
3	3
4	4
5	5

5 rows in set (0.00 sec)

SELECT * FROM `writesPublication`;

PublicationID	EID
1	1
2	2
3	3
4	4
5	5

5 rows in set (0.00 sec)

UPDATE QUERIES:

UPDATE Distributor SET Phone='+19195555405' WHERE distributorID=1;
UPDATE Editor SET Address='2120 Parkwood Village Raleigh NC' WHERE EID=4;
UPDATE StaffEditor SET Experience='1 Year' WHERE EID=1;
UPDATE InvitedAuthor SET Date_of_invitation='2022-02-27' WHERE EID=1;
UPDATE Payment SET Amount=230 WHERE EID=1;
UPDATE Publication SET Periodicity='Monthly' WHERE PublicationID=1;
UPDATE Publisher SET Address='2200 Avent Ferry Road' WHERE PID=1;

```
UPDATE Books SET ISBN='978-3-16-148410-9' WHERE PublicationID=1;
UPDATE Chapters SET Number_of_pages = 900 WHERE PublicationID=1 AND
ChapterID=1;
UPDATE PeriodicPublication set Type='Magazine' where PublicationID=4;
UPDATE Articles SET Text = 'The evolution, evolvability and engineering of gene
regulatory DNA' WHERE PublicationID=1 AND ArticleID=1;
UPDATE Management SET Name ='MAX' WHERE ManagementID=1;
UPDATE MonthlyRevenueReport SET NumberCopies=2 WHERE ReportID=1;
UPDATE MonthlyPaymentReport set WorkType='Editorial Work' where ReportID=4;
UPDATE Orders set Date='2022-01-29'  where OrderID=5;
UPDATE TransactionDetails  set DeliveryDate='2022-01-29'  where OrderID=5;
UPDATE Account set Balance=5000  where AccountNum=5;
```

DELETE QUERIES:

```
DELETE FROM Distributor WHERE distributorID=4;
DELETE FROM StaffEditor WHERE EID=5;
DELETE FROM InvitedAuthor WHERE EID=5;
DELETE FROM Payment WHERE EID=5;
DELETE FROM Editor WHERE EID=5;
DELETE FROM Chapters WHERE PublicationID=5 AND ChapterID=5;
DELETE FROM Articles WHERE ArticleID=5;
DELETE FROM Books WHERE PublicationID=5;
DELETE from PeriodicPublication where PublicationID=4;
DELETE FROM Publication WHERE PublicationID=5;
DELETE FROM Publisher WHERE PID=5;
DELETE FROM Management WHERE ManagementID=1;
DELETE FROM MonthlyRevenueReport WHERE ReportID=1;
DELETE from MonthlyPaymentReport where ReportID=4;
DELETE FROM Orders  WHERE OrderID=3;
DELETE FROM TransactionDetails  WHERE OrderID=3;
DELETE FROM Account  WHERE AccountNum=3;
```

4. SQL Queries:

4.1 Information Processing:

Editing and publishing.

- **Update Basic Information on Publication:**

SQL Query: UPDATE Publication SET Periodicity='Monthly' WHERE PublicationID=1;

```
+---------------+--------------------------------------+------------+-------------+-------------+
| PublicationID | Title                                | Date       | Topics      | Periodicity |
+---------------+--------------------------------------+------------+-------------+-------------+
|             1 | Introduction to Computing            | 2020-01-21 | Science     | Monthly     |
|             2 | Intro to Database Management System  | 2020-01-04 | Technology  | Yearly      |
|             3 | Autobiography of Sachin Tendulkar    | 2019-03-19 | Sports      | Yearly      |
|             4 | Mathematics Edition 1                | 2021-02-13 | Mathematics | Monthly     |
|             5 | Marvel Comics                        | 2021-01-26 | Fictional   | Weekly      |
+---------------+--------------------------------------+------------+-------------+-------------+
5 rows in set (0.00 sec)
```

- **Enter Basic Information on New Publication:**

SQL Query: INSERT INTO Publication VALUES(11,'Introduction to quantum computing','2021-01-21','Research','Yearly');

```
MariaDB [ssingha2]> INSERT INTO Publication VALUES(11,'Introduction to quantum computing','2021-01-21','Research','Yearly');
Query OK, 1 row affected (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM Publication;
+---------------+--------------------------------------+------------+-------------+-------------+
| PublicationID | Title                                | Date       | Topics      | Periodicity |
+---------------+--------------------------------------+------------+-------------+-------------+
|             1 | Introduction to Computing            | 2020-01-21 | Science     | Monthly     |
|             2 | Intro to Database Management System  | 2020-01-04 | Technology  | Yearly      |
|             3 | Autobiography of Sachin Tendulkar    | 2019-03-19 | Sports      | Yearly      |
|             6 | Introduction to Computing            | 2020-01-21 | Science     | Yearly      |
|             7 | Intro to Database Management System  | 2020-01-04 | Technology  | Yearly      |
|             8 | Autobiography of Sachin Tendulkar    | 2019-03-19 | Sports      | Yearly      |
|             9 | Mathematics Edition 1                | 2021-02-13 | Mathematics | Monthly     |
|            10 | Marvel Comics                        | 2021-01-26 | Fictional   | Weekly      |
|            11 | Introduction to quantum computing    | 2021-01-21 | Research    | Yearly      |
+---------------+--------------------------------------+------------+-------------+-------------+
9 rows in set (0.00 sec)
```

- **Assign Editor to Publication:**

```
MariaDB [ssingha2]> insert into writesPublication  values(6,4);
Query OK, 1 row affected (0.01 sec)

MariaDB [ssingha2]> SELECT * FROM writesPublication;
+---------------+------+
| PublicationID | EID  |
+---------------+------+
|             1 |    1 |
|             2 |    2 |
|             3 |    3 |
|             4 |    4 |
|             6 |    4 |
+---------------+------+
5 rows in set (0.00 sec)
```

- **Each editor view the information on the publications he/she is responsible for:**
  SQL QUERY: SELECT wp.EID, p.* FROM Publication p JOIN writesPublication wp on p.PublicationID=wp.PublicationID;

```
MariaDB [ssingha2]> SELECT wp.EID, p.* FROM Publication p JOIN writesPublication wp on p.PublicationID=wp.PublicationID;
+------+---------------+-----------------------------------+------------+------------+-------------+
| EID  | PublicationID | Title                             | Date       | Topics     | Periodicity |
+------+---------------+-----------------------------------+------------+------------+-------------+
|    1 |             1 | Introduction to Computing         | 2020-01-21 | Science    | Monthly     |
|    2 |             2 | Intro to Database Management System | 2020-01-04 | Technology | Yearly      |
|    3 |             3 | Autobiography of Sachin Tendulkar | 2019-03-19 | Sports     | Yearly      |
|    4 |             6 | Introduction to Computing         | 2020-01-21 | Science    | Yearly      |
+------+---------------+-----------------------------------+------------+------------+-------------+
4 rows in set (0.00 sec)
```

- **Adding Articles (for periodic publication) or chapters/sections (for books):**
  SQL QUERY: insert into PeriodicPublication values(11,'Magazine', 8, '2020-03-03');

```
MariaDB [ssingha2]> SELECT * FROM PeriodicPublication;
+---------------+----------+-----------------+------------+
| PublicationID | Type     | Periodic_length | Issue_date |
+---------------+----------+-----------------+------------+
|             7 | Magazine |               5 | 2020-06-21 |
|             8 | Magazine |               1 | 2020-09-13 |
|             9 | Journal  |               8 | 2020-03-03 |
|            10 | Journal  |               8 | 2020-03-03 |
|             6 | Journal  |               8 | 2020-03-03 |
+---------------+----------+-----------------+------------+
5 rows in set (0.00 sec)

MariaDB [ssingha2]> insert into PeriodicPublication values(11,'Magazine', 8, '2020-03-03');
Query OK, 1 row affected (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM PeriodicPublication;
+---------------+----------+-----------------+------------+
| PublicationID | Type     | Periodic_length | Issue_date |
+---------------+----------+-----------------+------------+
|             7 | Magazine |               5 | 2020-06-21 |
|             8 | Magazine |               1 | 2020-09-13 |
|             9 | Journal  |               8 | 2020-03-03 |
|            10 | Journal  |               8 | 2020-03-03 |
|             6 | Journal  |               8 | 2020-03-03 |
|            11 | Magazine |               8 | 2020-03-03 |
+---------------+----------+-----------------+------------+
6 rows in set (0.00 sec)
```

- **Delete Articles (for periodic publication) or chapters/sections (for books):**
  SQL QUERY : DELETE FROM Publication WHERE PublicationID=4;

```
MariaDB [ssingha2]> SELECT * FROM Chapters;
+---------------+-----------+-----------------+
| PublicationID | ChapterID | Number_of_pages |
+---------------+-----------+-----------------+
|             1 |         1 |             400 |
|             2 |         2 |             500 |
|             3 |         3 |             600 |
+---------------+-----------+-----------------+
3 rows in set (0.00 sec)

MariaDB [ssingha2]> DELETE FROM Publication where PublicationID =1;
Query OK, 1 row affected (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM Chapters;
+---------------+-----------+-----------------+
| PublicationID | ChapterID | Number_of_pages |
+---------------+-----------+-----------------+
|             2 |         2 |             500 |
|             3 |         3 |             600 |
+---------------+-----------+-----------------+
2 rows in set (0.00 sec)
```

SQL QUERY: DELETE FROM Publication where PublicationID =6;

```
MariaDB [ssingha2]> SELECT * FROM Articles where PublicationID=6;
+---------------+-----------+------------------------------------------------------+------------------------------------------------------------------------------------+
| PublicationID | ArticleID | Description                                          | Text                                                                               |
+---------------+-----------+------------------------------------------------------+------------------------------------------------------------------------------------+
|             6 |         1 | AGN as potential factories for eccentric black hole mergers | There is some weak evidence that the black hole merger named GW190521 had a non-zero eccentricity. |
+---------------+-----------+------------------------------------------------------+------------------------------------------------------------------------------------+
1 row in set (0.00 sec)

MariaDB [ssingha2]> DELETE FROM Publication where PublicationID =6;
Query OK, 1 row affected (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM Articles where PublicationID=6;
Empty set (0.00 sec)
```

Production of a book edition or of an issue of a publication.

- **Enter a new book edition:**
  SQL QUERY : insert into Books values(7,'9710-3-16-148410-0','6st Edition');

```
MariaDB [ssingha2]> insert into Books values(7,'9710-3-16-148410-0','6st Edition');


MariaDB [ssingha2]> SELECT * FROM Books;
+---------------+--------------------+-------------+
| PublicationID | ISBN               | Edition     |
+---------------+--------------------+-------------+
|             2 | 978-3-16-148410-1  | 2nd Edition |
|             3 | 978-3-16-148410-2  | 4th Edition |
|             7 | 9710-3-16-148410-0 | 6st Edition |
+---------------+--------------------+-------------+
3 rows in set (0.00 sec)
```

- **Enter a new issue of a publication:**

SQL QUERY : insert into PeriodicPublication values(3,'Journal', 3, '2021-10-05');

```
MariaDB [ssingha2]> insert into PeriodicPublication values(3,'Journal', 3, '2021-10-05');
Query OK, 1 row affected (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM PeriodicPublication ;
+--------------+----------+-----------------+------------+
| PublicationID | Type     | Periodic_length | Issue_date |
+--------------+----------+-----------------+------------+
|            7 | Magazine |               5 | 2020-06-21 |
|            8 | Magazine |               1 | 2020-09-13 |
|            9 | Journal  |               8 | 2020-03-03 |
|           10 | Journal  |               8 | 2020-03-03 |
|           11 | Magazine |               8 | 2020-03-03 |
|            3 | Journal  |               3 | 2021-10-05 |
+--------------+----------+-----------------+------------+
6 rows in set (0.00 sec)
```

- **Update a book edition or publication issue.**

SQL QUERY : UPDATE Books SET Edition="3rd Edition" WHERE PublicationID=3;

```
MariaDB [ssingha2]> SELECT * FROM Books;
+--------------+--------------------+-------------+
| PublicationID | ISBN               | Edition     |
+--------------+--------------------+-------------+
|            2 | 978-3-16-148410-1  | 2nd Edition |
|            3 | 978-3-16-148410-2  | 4th Edition |
|            7 | 9710-3-16-148410-0 | 6st Edition |
+--------------+--------------------+-------------+
3 rows in set (0.00 sec)

MariaDB [ssingha2]> UPDATE Books SET Edition="3rd Edition" WHERE PublicationID=3;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [ssingha2]> SELECT * FROM Books;
+--------------+--------------------+-------------+
| PublicationID | ISBN               | Edition     |
+--------------+--------------------+-------------+
|            2 | 978-3-16-148410-1  | 2nd Edition |
|            3 | 978-3-16-148410-2  | 3rd Edition |
|            7 | 9710-3-16-148410-0 | 6st Edition |
+--------------+--------------------+-------------+
3 rows in set (0.00 sec)
```

SQL QUERY : Update PeriodicPublication set Type='Journal' where PublicationID=7;

```
MariaDB [ssingha2]> SELECT * FROM PeriodicPublication;
+---------------+----------+-----------------+------------+
| PublicationID | Type     | Periodic_length | Issue_date |
+---------------+----------+-----------------+------------+
|             7 | Magazine |               5 | 2020-06-21 |
|             8 | Magazine |               1 | 2020-09-13 |
|             9 | Journal  |               8 | 2020-03-03 |
|            10 | Journal  |               8 | 2020-03-03 |
|            11 | Magazine |               8 | 2020-03-03 |
|             3 | Journal  |               3 | 2021-10-05 |
+---------------+----------+-----------------+------------+
6 rows in set (0.00 sec)

MariaDB [ssingha2]> Update PeriodicPublication set Type='Journal'  where PublicationID=7;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [ssingha2]> SELECT * FROM PeriodicPublication;
+---------------+----------+-----------------+------------+
| PublicationID | Type     | Periodic_length | Issue_date |
+---------------+----------+-----------------+------------+
|             7 | Journal  |               5 | 2020-06-21 |
|             8 | Magazine |               1 | 2020-09-13 |
|             9 | Journal  |               8 | 2020-03-03 |
|            10 | Journal  |               8 | 2020-03-03 |
|            11 | Magazine |               8 | 2020-03-03 |
|             3 | Journal  |               3 | 2021-10-05 |
+---------------+----------+-----------------+------------+
6 rows in set (0.00 sec)
```

- **Delete a book edition or publication issue.**
  SQL QUERY : DELETE FROM Publication WHERE PublicationID=2;

```
MariaDB [ssingha2]> SELECT * FROM Books;
+---------------+--------------------+-------------+
| PublicationID | ISBN               | Edition     |
+---------------+--------------------+-------------+
|             3 | 978-3-16-148410-2  | 3rd Edition |
|             7 | 9710-3-16-148410-0 | 6st Edition |
|             2 | 978-3-16-148410-1  | 2nd Edition |
+---------------+--------------------+-------------+
3 rows in set (0.00 sec)

MariaDB [ssingha2]> DELETE FROM Publication WHERE PublicationID=2;
Query OK, 1 row affected (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM Books;
+---------------+--------------------+-------------+
| PublicationID | ISBN               | Edition     |
+---------------+--------------------+-------------+
|             3 | 978-3-16-148410-2  | 3rd Edition |
|             7 | 9710-3-16-148410-0 | 6st Edition |
+---------------+--------------------+-------------+
2 rows in set (0.00 sec)
```

SQL QUERY: DELETE FROM Publication WHERE PublicationID=8;

```
MariaDB [ssingha2]> SELECT * FROM PeriodicPublication;
+---------------+----------+-----------------+------------+
| PublicationID | Type     | Periodic_length | Issue_date |
+---------------+----------+-----------------+------------+
|             7 | Journal  |               5 | 2020-06-21 |
|             8 | Magazine |               1 | 2020-09-13 |
|             9 | Journal  |               8 | 2020-03-03 |
|            10 | Journal  |               8 | 2020-03-03 |
|            11 | Magazine |               8 | 2020-03-03 |
|             3 | Journal  |               3 | 2021-10-05 |
+---------------+----------+-----------------+------------+
6 rows in set (0.00 sec)

MariaDB [ssingha2]> DELETE FROM Publication WHERE PublicationID=8;
Query OK, 1 row affected (0.01 sec)

MariaDB [ssingha2]> SELECT * FROM PeriodicPublication;
+---------------+----------+-----------------+------------+
| PublicationID | Type     | Periodic_length | Issue_date |
+---------------+----------+-----------------+------------+
|             7 | Journal  |               5 | 2020-06-21 |
|             9 | Journal  |               8 | 2020-03-03 |
|            10 | Journal  |               8 | 2020-03-03 |
|            11 | Magazine |               8 | 2020-03-03 |
|             3 | Journal  |               3 | 2021-10-05 |
+---------------+----------+-----------------+------------+
5 rows in set (0.00 sec)
```

- **Update Article title, topic, or date:**
  SQL QUERY: Update Publication set Title='new Title9' ,Topics = 'new Topic9', Date = '2022-01-07' where publicationID = (SELECT PublicationID From Articles Where ArticleID = 4);

```
MariaDB [ssingha2]> SELECT * FROM Publication;
+---------------+-----------------------------------+------------+------------+-------------+
| PublicationID | Title                             | Date       | Topics     | Periodicity |
+---------------+-----------------------------------+------------+------------+-------------+
|             1 | Introduction to Computing         | 2020-01-21 | Science    | Yearly      |
|             2 | Intro to Database Management System | 2020-01-04 | Technology | Yearly      |
|             3 | Autobiography of Sachin Tendulkar | 2019-03-19 | Sports     | Yearly      |
|             7 | Intro to Database Management System | 2020-01-04 | Technology | Yearly      |
|             9 | Mathematics Edition 1             | 2021-02-13 | Mathematics | Monthly     |
|            10 | Marvel Comics                     | 2021-01-26 | Fictional  | Weekly      |
|            11 | new Title                         | 2022-01-07 | new Topic  | Yearly      |
+---------------+-----------------------------------+------------+------------+-------------+
7 rows in set (0.00 sec)

MariaDB [ssingha2]> Update Publication set Title='new Title9' ,Topics = 'new Topic9', Date = '2022-01-07'  where publicationID = (SELECT PublicationID From Articles Where ArticleID = 4);
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [ssingha2]> SELECT * FROM Publication;
+---------------+-----------------------------------+------------+------------+-------------+
| PublicationID | Title                             | Date       | Topics     | Periodicity |
+---------------+-----------------------------------+------------+------------+-------------+
|             1 | Introduction to Computing         | 2020-01-21 | Science    | Yearly      |
|             2 | Intro to Database Management System | 2020-01-04 | Technology | Yearly      |
|             3 | Autobiography of Sachin Tendulkar | 2019-03-19 | Sports     | Yearly      |
|             7 | Intro to Database Management System | 2020-01-04 | Technology | Yearly      |
|             9 | new Title9                        | 2022-01-07 | new Topic9 | Monthly     |
|            10 | Marvel Comics                     | 2021-01-26 | Fictional  | Weekly      |
|            11 | new Title                         | 2022-01-07 | new Topic  | Yearly      |
+---------------+-----------------------------------+------------+------------+-------------+
7 rows in set (0.00 sec)
```

- **Enter text of an article**
  SQL QUERY: INSERT INTO Articles values(11,11,'AGN as potential factories for eccentric black hole mergers','There is some weak evidence that the black hole merger named GW190521 had a non-zero eccentricity.');

```
MariaDB [ssingha2]> insert into Articles values(11,11,'AGN as potential factories for eccentric black hole mergers','There is some weak evidence that the black hole merger named GW190521 had a non-zero eccentricity.');
Query OK, 1 row affected (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM Articles where PublicationID=11;
+---------------+-----------+-----------------------------------------------------+-----------------------------------------------------------------------------------------------------+
| PublicationID | ArticleID | Description                                         | Text                                                                                                |
+---------------+-----------+-----------------------------------------------------+-----------------------------------------------------------------------------------------------------+
|            11 |        11 | AGN as potential factories for eccentric black hole mergers | There is some weak evidence that the black hole merger named GW190521 had a non-zero eccentricity. |
+---------------+-----------+-----------------------------------------------------+-----------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

- **Update text of an article**
  SQL QUERY: UPDATE Articles SET Text = 'The evolution, evolvability and engineering of gene regulatory DNA' WHERE PublicationID=7;

```
MariaDB [ssingha2]> UPDATE Articles SET Text = 'The evolution, evolvability and engineering of gene regulatory DNA' WHERE PublicationID=7;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [ssingha2]> SELECT * FROM Articles where PublicationID=7;
+---------------+-----------+-----------------------------------------+-----------------------------------------------------------------------+
| PublicationID | ArticleID | Description                             | Text                                                                  |
+---------------+-----------+-----------------------------------------+-----------------------------------------------------------------------+
|             7 |         2 | Electron-catalysed molecular recognition | The evolution, evolvability and engineering of gene regulatory DNA |
+---------------+-----------+-----------------------------------------+-----------------------------------------------------------------------+
1 row in set (0.00 sec)
```

- **Find books by date**
  SQL QUERY: Select * From Books Where PublicationID IN (SELECT PublicationID From Publication Where Date = '2020-01-04');

```
MariaDB [ssingha2]> Select * From Books Where PublicationID IN (SELECT PublicationID From Publication Where Date = '2020-01-04');
+---------------+--------------------+-------------+
| PublicationID | ISBN               | Edition     |
+---------------+--------------------+-------------+
|             7 | 9710-3-16-148410-0 | 6st Edition |
+---------------+--------------------+-------------+
1 row in set (0.00 sec)
```

- **Find books by topic**
  SQL QUERY: Select * From Books Where PublicationID IN (SELECT PublicationID From Publication Where Topics = 'Sports');

```
MariaDB [ssingha2]> Select * From Books Where PublicationID IN (SELECT PublicationID From Publication Where Topics = 'Sports');
+---------------+--------------------+-------------+
| PublicationID | ISBN               | Edition     |
+---------------+--------------------+-------------+
|             3 | 978-3-16-148410-2 | 3rd Edition |
+---------------+--------------------+-------------+
1 row in set (0.00 sec)
```

- **Find books by author's name**
  SQL QUERY: SELECT * From Books Where PublicationID IN (SELECT PublicationID From writesPublication Where EID = (SELECT EID From Editor Where Name = 'Mark'));

```
MariaDB [ssingha2]> SELECT * From Books Where PublicationID IN (SELECT PublicationID From writesPublication Where EID = (SELECT EID From Editor Where Name = 'Mark'));
+---------------+--------------------+-------------+
| PublicationID | ISBN               | Edition     |
+---------------+--------------------+-------------+
|             3 | 978-3-16-148410-2 | 3rd Edition |
+---------------+--------------------+-------------+
1 row in set (0.01 sec)
```

- **Find articles by topic**
  SQL QUERY: Select * From Articles Where PublicationID IN (SELECT PublicationID From Publication Where Topics = 'Technology');

```
MariaDB [ssingha2]> Select * From Articles Where PublicationID IN (SELECT PublicationID From Publication Where Topics = 'Technology');
+---------------+-----------+---------------------------------------+-----------------------------------------------------------------------+
| PublicationID | ArticleID | Description                           | Text                                                                  |
+---------------+-----------+---------------------------------------+-----------------------------------------------------------------------+
|             7 |         2 | Electron-catalysed molecular recognition | The evolution, evolvability and engineering of gene regulatory DNA |
+---------------+-----------+---------------------------------------+-----------------------------------------------------------------------+
1 row in set (0.00 sec)
```

- **Find articles by date**
  SQL QUERY: Select * From Articles Where PublicationID IN (SELECT PublicationID From Publication Where Date = '2020-01-04');

```
MariaDB [ssingha2]> Select * From Articles Where PublicationID IN (SELECT PublicationID From Publication Where Date = '2020-01-04');
+---------------+-----------+---------------------------------------+-----------------------------------------------------------------------+
| PublicationID | ArticleID | Description                           | Text                                                                  |
+---------------+-----------+---------------------------------------+-----------------------------------------------------------------------+
|             7 |         2 | Electron-catalysed molecular recognition | The evolution, evolvability and engineering of gene regulatory DNA |
+---------------+-----------+---------------------------------------+-----------------------------------------------------------------------+
1 row in set (0.00 sec)
```

- **Find articles by author's name**
  SQL QUERY: SELECT * From Articles Where PublicationID IN (SELECT PublicationID From writesPublication Where EID = (SELECT EID From Editor Where Name = 'Chris123'));

```
MariaDB [ssingha2]> SELECT * From Articles Where PublicationID IN (SELECT PublicationID From writesPublication Where EID = (SELECT EID From Editor Where Name = 'Chris123'));
+---------------+-----------+---------------------------------------+-----------------------------------------------------------------------+
| PublicationID | ArticleID | Description                           | Text                                                                  |
+---------------+-----------+---------------------------------------+-----------------------------------------------------------------------+
|             7 |         2 | Electron-catalysed molecular recognition | The evolution, evolvability and engineering of gene regulatory DNA |
+---------------+-----------+---------------------------------------+-----------------------------------------------------------------------+
1 row in set (0.00 sec)
```

- **Enter payment for author or editor, and keep track of when each payment was claimed by its addressee.**

```
MariaDB [ssingha2]> SELECT * FROM Payment;
+-----+--------+------------+
| EID | Amount | Date       |
+-----+--------+------------+
|   1 |    230 | 2022-01-21 |
|   2 |    942 | 2022-02-04 |
|   3 |   1200 | 2022-03-19 |
|   4 |    512 | 2021-01-13 |
+-----+--------+------------+
4 rows in set (0.01 sec)

MariaDB [ssingha2]> insert into Editor values(5,'Mike','2517 Gorman Street Road','+19863245402');
Query OK, 1 row affected (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM Editor;
+-----+----------+-----------------------------------+---------------+
| EID | Name     | Address                           | Contact       |
+-----+----------+-----------------------------------+---------------+
|   1 | Chris123 | 2512 Gorman Street Road Raleigh NC | 9848883402    |
|   2 | Kevin    | 2122 Gorman Street Road           | +19842239405  |
|   3 | Mark     | 2520 Gorman Street Road           | +191041239420 |
|   4 | Chris    | 2120 Parkwood Village Raleigh NC  | +19841298765  |
|   5 | Mike     | 2517 Gorman Street Road           | +19863245402  |
+-----+----------+-----------------------------------+---------------+
5 rows in set (0.00 sec)

MariaDB [ssingha2]> Insert INTO Payment VALUES(5,20000.00, '2020-01-05');
Query OK, 1 row affected (0.01 sec)

MariaDB [ssingha2]> SELECT * FROM Payment;
+-----+--------+------------+
| EID | Amount | Date       |
+-----+--------+------------+
|   1 |    230 | 2022-01-21 |
|   2 |    942 | 2022-02-04 |
|   3 |   1200 | 2022-03-19 |
|   4 |    512 | 2021-01-13 |
|   5 |  20000 | 2020-01-05 |
+-----+--------+------------+
5 rows in set (0.01 sec)
```

Distribution

- **Enter a new distributor;**

```
MariaDB [nsshah5]> INSERT INTO Distributor VALUES(6,'Mike','Library','2517 Avent Ferry Road','+19863245402','Gwen',772);
Query OK, 1 row affected (0.01 sec)
```

- **update distributor information;**

```
MariaDB [nsshah5]> Update Distributor set Address='2516 Avent Ferry Road' where DistributorID=6;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

- **delete a distributor.**

```
MariaDB [nsshah5]> delete from Distributor where DistributorID=6;
Query OK, 1 row affected (0.01 sec)
```

- **Input Order**

  SQL QUERY: insert into Orders values(3, 100, 100, 1,'The Train', '2022-01-21');

```
MariaDB [nsshah5]> insert into Orders values(3, 100, 100, 1,'The Train', '2022-01-21');
Query OK, 1 row affected (0.00 sec)
```

SQL QUERY : insert into PlacesOrder values(1,3);

```
MariaDB [nsshah5]> insert into PlacesOrder values(1,3);
Query OK, 1 row affected (0.00 sec)
```

SQL QUERY: insert into Consist values(3,2);

```
MariaDB [nsshah5]> insert into Consist values(3,2);
Query OK, 1 row affected (0.00 sec)
```

SQL QUERY: insert into TransactionDetails values(3,'2022-01-21 15:35:07','2022-01-21');

```
MariaDB [nsshah5]> insert into TransactionDetails values(3,'2022-01-21 15:35:07','2022-01-21');
Query OK, 1 row affected (0.00 sec)
```

- **Bill distributor for an order;**

SQL QUERY: insert into Orders values(3, 100, 100, 1,'The Train', '2022-01-21');

```
MariaDB [ssingha2]> SELECT * FROM Orders;
+---------+--------+--------------+-----------+----------------+------------+
| OrderID | Price  | ShippingCost | NumCopies | Book           | Date       |
+---------+--------+--------------+-----------+----------------+------------+
|       1 | 100.00 |       100.00 |         1 | The Train      | 2022-01-21 |
|       2 | 200.00 |       200.00 |         2 | Science Test   | 2022-01-22 |
|       4 | 400.00 |       400.00 |         4 | English Book   | 2022-01-24 |
|       5 | 500.00 |       500.00 |         5 | Social Science | 2022-01-29 |
+---------+--------+--------------+-----------+----------------+------------+
4 rows in set (0.00 sec)

MariaDB [ssingha2]> insert into Orders values(3, 100, 100, 1,'The Train', '2022-01-21');
Query OK, 1 row affected (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM Orders;
+---------+--------+--------------+-----------+----------------+------------+
| OrderID | Price  | ShippingCost | NumCopies | Book           | Date       |
+---------+--------+--------------+-----------+----------------+------------+
|       1 | 100.00 |       100.00 |         1 | The Train      | 2022-01-21 |
|       2 | 200.00 |       200.00 |         2 | Science Test   | 2022-01-22 |
|       3 | 100.00 |       100.00 |         1 | The Train      | 2022-01-21 |
|       4 | 400.00 |       400.00 |         4 | English Book   | 2022-01-24 |
|       5 | 500.00 |       500.00 |         5 | Social Science | 2022-01-29 |
+---------+--------+--------------+-----------+----------------+------------+
5 rows in set (0.00 sec)
```

SQL QUERY: insert into PlacesOrder values(1,3);

```
MariaDB [ssingha2]> insert into PlacesOrder values(1,3);
Query OK, 1 row affected (0.01 sec)

MariaDB [ssingha2]> SELECT * FROM PlacesOrder ;
+---------------+---------+
| DistributorID | OrderID |
+---------------+---------+
|             1 |       3 |
+---------------+---------+
1 row in set (0.01 sec)
```

SQL QUERY: insert into Consist values(3,2);

```
MariaDB [ssingha2]> insert into Consist values(3,2);
Query OK, 1 row affected (0.01 sec)

MariaDB [ssingha2]> SELECT * FROM Consist;
+---------+---------------+
| OrderID | PublicationID |
+---------+---------------+
|       3 |             2 |
+---------+---------------+
1 row in set (0.00 sec)
```

SQL QUERY: insert into TransactionDetails values(3,'2022-01-21 15:35:07','2022-01-21');

```
MariaDB [ssingha2]> insert into TransactionDetails values(3,'2022-01-21 15:35:07','2022-01-21');
Query OK, 1 row affected (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM TransactionDetails;
+---------+---------------------+--------------+
| OrderID | DeliveryTime        | DeliveryDate |
+---------+---------------------+--------------+
|       1 | 2022-01-21 15:35:07 | 2022-01-21   |
|       2 | 2022-01-22 15:35:07 | 2022-01-22   |
|       4 | 2022-01-24 15:35:07 | 2022-01-24   |
|       5 | 2022-03-10 21:49:55 | 2022-01-29   |
|       3 | 2022-01-21 15:35:07 | 2022-01-21   |
+---------+---------------------+--------------+
5 rows in set (0.00 sec)
```

- **change outstanding balance of a distributor on receipt of a payment.**
  SQL QUERY: Update Account set Balance=Account.Balance + (Select price From Orders where OrderID = 3) where AccountNum in (SELECT AccountNum From hasAccount Where DistributorID = 1);

```
MariaDB [ssingha2]> Update Account set Balance=Account.Balance + (Select price From Orders where OrderID = 3)  where AccountNum in (SELECT AccountNum From hasAccount Where DistributorID = 1);
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [ssingha2]> SELECT * FROM TransactionDetails;
+---------+---------------------+--------------+
| OrderID | DeliveryTime        | DeliveryDate |
+---------+---------------------+--------------+
|       1 | 2022-01-21 15:35:07 | 2022-01-21   |
|       2 | 2022-01-22 15:35:07 | 2022-01-22   |
|       4 | 2022-01-24 15:35:07 | 2022-01-24   |
|       5 | 2022-03-10 21:49:55 | 2022-01-29   |
|       3 | 2022-01-21 15:35:07 | 2022-01-21   |
+---------+---------------------+--------------+
5 rows in set (0.00 sec)

MariaDB [ssingha2]> SELECT * FROM Account;
+------------+------------+---------+------------+
| AccountNum | Date       | Balance | PastOrders |
+------------+------------+---------+------------+
|          1 | 2022-01-21 |  200.00 |         23 |
|          2 | 2022-01-22 |  200.00 |         45 |
|          4 | 2022-01-24 |  400.00 |         63 |
|          5 | 2022-01-25 | 5000.00 |         79 |
+------------+------------+---------+------------+
4 rows in set (0.00 sec)
```

Reports

Generate monthly reports:

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

  SQL QUERY: SELECT c.PublicationID, p.DistributorID, Month(o.Date), SUM(o.numCopies), SUM(o.Price *o.NumCopies) FROM Orders o JOIN PlacesOrder p ON o.OrderID = p.OrderID JOIN Consist c ON c.OrderID = o.OrderID GROUP BY c.PublicationID, p.DistributorID, Month(o.Date);

```
+---------------+---------------+---------------+-----------------+---------------------------+
| PublicationID | DistributorID | Month(o.Date) | SUM(o.numCopies) | SUM(o.Price *o.NumCopies) |
+---------------+---------------+---------------+-----------------+---------------------------+
|             1 |             1 |             1 |              10 |                   1000.00 |
|             2 |             1 |             1 |              20 |                   4000.00 |
|             2 |             3 |             1 |              20 |                   4000.00 |
|             4 |             5 |             1 |              40 |                  16000.00 |
+---------------+---------------+---------------+-----------------+---------------------------+
4 rows in set (0.00 sec)
```

- **Number of each publication bought per distributor per month;**

  SQL QUERY: SELECT c.PublicationID, p.DistributorID, Month(o.Date), SUM(o.numCopies) FROM Orders o JOIN PlacesOrder p ON o.OrderID = p.OrderID JOIN Consist c ON c.OrderID = o.OrderID GROUP BY c.PublicationID, p.DistributorID, Month(o.Date);

```
+---------------+---------------+---------------+-----------------+
| PublicationID | DistributorID | Month(o.Date) | SUM(o.numCopies) |
+---------------+---------------+---------------+-----------------+
|             1 |             1 |             1 |               8 |
|             2 |             1 |             1 |              16 |
|             2 |             3 |             1 |              16 |
|             4 |             5 |             1 |              32 |
+---------------+---------------+---------------+-----------------+
4 rows in set (0.00 sec)
```

- **Total price of copies of each publication bought per distributor per month;**

  SQL QUERY: SELECT c.PublicationID, p.DistributorID, SUM(o.Price), Month(o.Date) FROM Orders o JOIN PlacesOrder p ON o.OrderID = p.OrderID JOIN Consist c ON c.OrderID = o.OrderID GROUP BY c.PublicationID, p.DistributorID, Month(o.Date);

```
+---------------+---------------+---------------+---------------+
| PublicationID | DistributorID | SUM(o.Price) | Month(o.Date) |
+---------------+---------------+---------------+---------------+
|             1 |             1 |        800.00 |             1 |
|             2 |             1 |       1600.00 |             1 |
|             2 |             3 |       1600.00 |             1 |
|             4 |             5 |       3200.00 |             1 |
+---------------+---------------+---------------+---------------+
4 rows in set (0.00 sec)
```

- **total revenue of the publishing house;**

SQL QUERY: SELECT SUM(TotalPrice) from MonthlyRevenueReport;

```
+------------------+
| SUM(TotalPrice) |
+------------------+
|           183.00 |
+------------------+
1 row in set (0.01 sec)
```

- **total expenses (i.e., shipping costs and salaries).**

  SQL QUERY: SELECT SUM(un.TotalExpenses) from ((SELECT SUM(Payment) as TotalExpenses from MonthlyPaymentReport mpr) UNION (SELECT SUM(TotalShippingCost) as TotalExpenses from MonthlyRevenueReport mrr)) un;

```
+-------------------------+
| SUM(un.TotalExpenses) |
+-------------------------+
|                10565.43 |
+-------------------------+
1 row in set (0.00 sec)
```

- **Calculate the total current number of distributors;**

  SQL QUERY: SELECT Count(*) from Distributor;

```
+----------+
| Count(*) |
+----------+
|        4 |
+----------+
1 row in set (0.00 sec)
```

- **calculate total revenue (since inception) per city**

  SQL QUERY: SELECT city,SUM(TotalPrice) from MonthlyRevenueReport GROUP BY City;

```
+------------+------------------+
| city       | SUM(TotalPrice)  |
+------------+------------------+
| Charlette  |            48.00 |
| Durham     |            65.00 |
| Greensboro |            70.00 |
+------------+------------------+
3 rows in set (0.01 sec)
```

- **calculate total revenue (since inception) per distributor**
  SQL QUERY: SELECT g.distributorID,SUM(m.TotalPrice) from MonthlyRevenueReport m,GenerateRevenueReport g where m.ReportID = g.ReportID;

```
+---------------+--------------------+
| distributorID | SUM(m.TotalPrice)  |
+---------------+--------------------+
|             1 |             130.00 |
+---------------+--------------------+
1 row in set (0.00 sec)
```

- **calculate total revenue (since inception) per location**
  SQL QUERY: SELECT city,SUM(TotalPrice) from MonthlyRevenueReport GROUP BY City;

```
+------------+------------------+
| city       | SUM(TotalPrice)  |
+------------+------------------+
| Charlette  |            48.00 |
| Durham     |            65.00 |
| Greensboro |            70.00 |
+------------+------------------+
3 rows in set (0.01 sec)
```

- **Calculate total payments to the editors and authors, per time period**
  SQL QUERY: SELECT Date,SUM(Amount) from Payment GROUP BY Date;

Date	SUM(Amount)
2021-01-13	512
2022-01-21	230
2022-02-04	942
2022-03-19	1200

4 rows in set (0.00 sec)

- **Calculate total payments to the editors and authors, per editor time period**
  SQL QUERY: SELECT Date,EID,SUM(Amount) from Payments GROUP BY Date, EID;

Date	EID	SUM(Amount)
2021-01-13	4	512
2022-01-21	1	230
2022-02-04	2	942
2022-03-19	3	1200

4 rows in set (0.01 sec)

- **Calculate total payments to the editors and authors, per time period per work type (book authorship, article authorship, or editorial work).**
  SQL QUERY: SELECT Date,WorkType,SUM(Payment) from MonthlyPaymentReport GROUP BY Date, WorkType;

Date	WorkType	SUM(Payment)
2021-03-05	Editorial Work	1203.41
2021-03-20	Article Authorship	921.41
2021-03-20	Book Authorship	1203.41

3 rows in set (0.00 sec)

4.2 EXPLAIN directive in MariaDB

Query 1: Find Books by Topics

1. **SQL ->** EXPLAIN Select * From Books Where PublicationID = (SELECT PublicationID From Publication Where Topics = 'Science');
2. Screenshot for EXPLAIN:

```
MariaDB [nsshah5]> EXPLAIN Select * From Books Where PublicationID = (SELECT PublicationID From Publication Where Topics = 'Science');
+------+-------------+-------------+------+---------------+---------------+---------+-------+------+-------------+
| id   | select_type | table       | type | possible_keys | key           | key_len | ref   | rows | Extra       |
+------+-------------+-------------+------+---------------+---------------+---------+-------+------+-------------+
|    1 | PRIMARY     | Books       | ref  | PublicationID | PublicationID | 5       | const |    1 | Using where |
|    2 | SUBQUERY    | Publication | ALL  | NULL          | NULL          | NULL    | NULL  |    4 | Using where |
+------+-------------+-------------+------+---------------+---------------+---------+-------+------+-------------+
2 rows in set (0.00 sec)
```

3. CREATE INDEX topic_index USING HASH ON Publication(Topics);
4. Screenshot for EXPLAIN after CREATE INDEX:

```
MariaDB [nsshah5]> CREATE INDEX topic_index
    -> USING HASH
    -> ON Publication(Topics);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [nsshah5]> EXPLAIN Select * From Books Where PublicationID = (SELECT PublicationID From Publication Where Topics = 'Science');
+------+-------------+-------------+------+---------------+---------------+---------+-------+------+--------------------------+
| id   | select_type | table       | type | possible_keys | key           | key_len | ref   | rows | Extra                    |
+------+-------------+-------------+------+---------------+---------------+---------+-------+------+--------------------------+
|    1 | PRIMARY     | Books       | ref  | PublicationID | PublicationID | 5       | const |    1 | Using where              |
|    2 | SUBQUERY    | Publication | ref  | topic_index   | topic_index   | 131     | const |    1 | Using where; Using index |
+------+-------------+-------------+------+---------------+---------------+---------+-------+------+--------------------------+
2 rows in set (0.00 sec)
```

Query 2: Find books by author's name

1. **SQL ->** SELECT * From Books Where PublicationID = (SELECT PublicationID From writesPublication Where EID = (SELECT EID From Editor Where Name = 'John'));
2. Screenshot for EXPLAIN:

```
MariaDB [nsshah5]> explain SELECT * From Articles Where PublicationID = (SELECT PublicationID From writesPublication Where EID = (SELECT EID From Editor Where Name = 'John'));
+------+-------------+--------+------+---------------+------+---------+------+------+-------------------------------------------------+
| id   | select_type | table  | type | possible_keys | key  | key_len | ref  | rows | Extra                                           |
+------+-------------+--------+------+---------------+------+---------+------+------+-------------------------------------------------+
|    1 | PRIMARY     | NULL   | NULL | NULL          | NULL | NULL    | NULL | NULL | Impossible WHERE noticed after reading const tables |
|    2 | SUBQUERY    | NULL   | NULL | NULL          | NULL | NULL    | NULL | NULL | no matching row in const table                  |
|    3 | SUBQUERY    | Editor | ALL  | NULL          | NULL | NULL    | NULL |    4 | Using where                                     |
+------+-------------+--------+------+---------------+------+---------+------+------+-------------------------------------------------+
3 rows in set (0.00 sec)
```

3. CREATE INDEX name_index USING HASH ON Editor(name);
4. Screenshot for EXPLAIN after CREATE INDEX:

```
MariaDB [nsshah5]> CREATE INDEX name_index
    -> USING HASH
    -> ON Editor(name);
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [nsshah5]> explain SELECT * From Articles Where PublicationID = (SELECT PublicationID From writesPublication Where EID = (SELECT EID From Editor Where Name = 'John'));
+------+-------------+--------+------+---------------+------------+---------+-------+------+-------------------------------------------------+
| id   | select_type | table  | type | possible_keys | key        | key_len | ref   | rows | Extra                                           |
+------+-------------+--------+------+---------------+------------+---------+-------+------+-------------------------------------------------+
|    1 | PRIMARY     | NULL   | NULL | NULL          | NULL       | NULL    | NULL  | NULL | Impossible WHERE noticed after reading const tables |
|    2 | SUBQUERY    | NULL   | NULL | NULL          | NULL       | NULL    | NULL  | NULL | no matching row in const table                  |
|    3 | SUBQUERY    | Editor | ref  | name_index    | name_index | 130     | const |    1 | Using where; Using index                        |
+------+-------------+--------+------+---------------+------------+---------+-------+------+-------------------------------------------------+
3 rows in set (0.00 sec)
```

4.3 Query Correctness

1.Number of each publication bought per distributor per month;

SELECT c.PublicationID, p.DistributorID, Month(o.Date), SUM(o.numCopies) FROM Orders o JOIN PlacesOrder p ON o.OrderID = p.OrderID JOIN Consist c ON c.OrderID = o.OrderID GROUP BY c.PublicationID, p.DistributorID, Month(o.Date);

$\pi_{\text{PublicationID, DistributorID, Month, SumCopies}}(\gamma_{\text{PublicationID, DistributorID,Month(Date)}\rightarrow\text{Month, SUM(numCopies)}\rightarrow\text{SumCopies}}((\text{PlacesOrder} \bowtie_{\text{PlacesOrder.OrderID = Orders.OrderID}} \text{Orders}) \bowtie_{\text{Orders.OrderID = Consists.OrderID}} \text{Consists})))$

Suppose p is any tuple in the placesOrder relation, o is any tuple in the Orders relation and c is any tuple in the Consists relation such that any value of p.OrderId, o.OrderID, and c.OrderId is the same. The combination of the tuples p, o, and c will give all the order details a distributor places for any given publication. This combination will return the DistributorID, PublicationID, OrderID, price, shippingCost, NumCopies, Book, date. We then group and aggregate the tuples in each group according to the Id of the publication, the Id of the distributor, the month order was sent out, and finally the amount of copies sold. We take the resulting table and only pull out the Publication ID, DistributorID, Month of Order and the Sum of Copies for each group in the grouping. The end result gives a table filled with tuples that give the total number of copies for each book(publicationID) that was ordered by each distributor for each month which is what we are expected to find.

2. Total price of copies of each publication bought per distributor per month;

SELECT c.PublicationID, p.DistributorID, SUM(o.Price), Month(o.Date) FROM Orders o JOIN PlacesOrder p ON o.OrderID = p.OrderID JOIN Consist c ON c.OrderID = o.OrderID GROUP BY c.PublicationID, p.DistributorID, Month(o.Date);

$\pi_{\text{PublicationID, DistributorID, SumPrice, Month}}(\gamma_{\text{PublicationID, DistributorID, SUM(price)}\rightarrow\text{SumPrice, Month(Date)}\rightarrow\text{Month}}((\text{PlacesOrder} \bowtie_{\text{PlacesOrder.OrderID = Orders.OrderID}} \text{Orders}) \bowtie_{\text{Orders.OrderID = Consists.OrderID}} \text{Consists})))$

Suppose p is any tuple in the placesOrder relation, o is any tuple in the Orders relation and c is any tuple in the Consists relation such that any value of p.OrderId, o.OrderID, and c.OrderId is the same. The combination of the tuples p, o, and c will give all the order details a distributor places for any given publication. This combination will return the DistributorID, PublicationID, OrderID, price, shippingCost, NumCopies, Book, date. We

then group and aggregate the tuples in each group according to the Id of the publication, the Id of the distributor, the sum of the price for orders, and finally the month of the orders. We take the resulting table and only pull out the Publication ID, DistributorID,the Sum of price of orders, and Month of Order for each group in the grouping. The end result gives a table filled with tuples that give the total price of copies for each book(publicationID) that was ordered by each distributor for each month which is what we are expected to find.

WolfPubDb Management System

For WolfCity publishing house

CSC 540 Database Management
Systems Project Report 3

Team AA
Shakshi Singhai, Parth Thosani, Nisarg Shah, Prashan Sengo,
April 14th, 2022

Assumption:

- Date in Publication and in periodic publication can be different depending upon user input
- All the dates are inputted by the user in the exact format specified(yyyy-mm-dd).
- Delivery date of the order is 14 days from order date irrespective of the quantity and availability of publications.
- Shipping cost is fixed to 10% of the total cost of the order.
- User is not going to update the ID field in the tables.

1. **Submit revised versions of the previous reports - you will get credit for the improvements, scaled by 50%. You will need to submit both the relevant numbered pages of the original reports and your revisions; the revisions should (1) mention the item and page numbers in the original report and (2) have the improved parts highlighted.**

   Submitted with this file.

2. **Document the program logic of the transactions in the applications. Submit, as parts of your project report 3 papers.**
   a. **the parts of the code that contain the transactions (points will be taken off by the grader if these parts of the code are not easy to locate), and**
   b. **your documentation. Make sure that your transactions use the COMMIT and ROLLBACK statements.**

Transaction 1: Inserting Distributor Information

In main we are calling a function *newDist* which will return True if the function doesn't give any error. If there is some error in inserting it will return False.
Based on these values we are applying transactions in main. If the function returns True we are committing the data using "con.commit()". If the function returns False we are applying a rollback transaction to revert to the previous stage.

Code:

```
public static void main(String[] args) throws Exception {
       //connect to the database
        Class.forName("org.mariadb.jdbc.Driver");
        Connection con =
DriverManager.getConnection("jdbc:mariadb://classdb2.csc.ncsu.edu:3306/" +
user,user,password);

     System.out.println("\t\t [25] - Enter new distributor");
     case "25":Savepoint distInsert=con.setSavepoint("beforeDistInsert");
           if(Distribution.newDist(con,inputReader))
                con.commit(); //Commit the transaction
           else
```

```
                con.rollback(distInsert); //Roll the transaction back to
the beginning
                break;
```

```
    String[] columns={"distributorID", "Name", "Type", "Address",
"Phone", "ContactPerson", "Balance"};
     public static boolean newDist(Connection conn, Scanner inputReader){
        System.out.println("Enter Distributor name:");
        String name=inputReader.next();
        System.out.println("Enter Distributor type:");
        String type=inputReader.next();
        System.out.println("Enter Distributor Address:");
        String add=inputReader.next();
        System.out.println("Enter Distributor Phone:");
        String phone=inputReader.next();
        System.out.println("Enter Distributor Contact person:");
        String cp=inputReader.next();
        System.out.println("Enter balance:");
        int bal=inputReader.nextInt();
     String query = "INSERT INTO Distributor (Name, Type, Address, Phone,
ContactPerson, Balance) VALUES(?,?,?,?,?,?);";
        try{
           PreparedStatement stinsert=conn.prepareStatement(query,
Statement.RETURN_GENERATED_KEYS);
           stinsert.setString(1, name);
           stinsert.setString(2, type);
           stinsert.setString(3, add);
           stinsert.setString(4, phone);
           stinsert.setString(5, cp);
           stinsert.setInt(6, bal);
           stinsert.executeQuery();
           System.out.println("1 Row inserted!");
        } catch(Exception e){
           e.printStackTrace();
           return false; // If there is error inserting the distributor
it will return False in main}
        return true; // If no error it will return True.
}
```

Transaction 2: Deleting Distributor Information

In main we are calling a function *deleteDist* which will return True if the function doesn't give any error. If there is some error in deleting it will return False.
Based on these values we are applying transactions in main. If the function returns True we are committing the data using "con.commit()" and data will be deleted. If the function returns False we are applying a rollback transaction to revert to the previous stage.

Code:

```
public static void main(String[] args) throws Exception {
        Class.forName("org.mariadb.jdbc.Driver");
        Connection con =
DriverManager.getConnection("jdbc:mariadb://classdb2.csc.ncsu.edu:3306/" +
user,user,password);
      System.out.println("\t\t [27] - Delete a distributor. ");
      case "27" : Savepoint deldist=con.setSavepoint("beforeDistInsert");
      if(Distribution.deleteDist(con,inputReader))
            con.commit(); //Commit the transaction
      else
            con.rollback(deldist); //Roll the transaction back to the
beginning
            break;
```

```
    public static boolean deleteDist(Connection conn, Scanner
inputreader){
        List<String> delcolnames=new ArrayList();
        List<Object> delcolvals=new ArrayList();
        System.out.println("enter number of conditions:");
        int n=inputreader.nextInt();
        for(int i=0;i<n;i++){
        System.out.println("Enter col name:");
        delcolnames.add(inputreader.next());
        System.out.println("Enter col value:");
        delcolvals.add(inputreader.next()); }
        try{
            StringBuilder query=new StringBuilder("DELETE FROM Distributor
where ");
            for(int i=0; i<delcolnames.size();i++){
                query.append(delcolnames.get(i) + "=? and ");
```

```
            }
            query.replace(query.length()-5,query.length()-1,";");
            PreparedStatement
stdelete=conn.prepareStatement(query.toString());

            for(int i=0;i<delcolvals.size();i++)
                stdelete.setObject(i+1, delcolvals.get(i));
            stdelete.executeQuery();
            System.out.println("Rows Deleted");
        } catch(Exception e){
            e.printStackTrace();
            return false; // If there is error deleting the distributor it
will return False in main
        }
        return true; // If no error it will return True.
      }
```

3. **High-level design decisions, which are any choices/decisions that you had to make when designing your database and applications.**

Design Decisions:

The program starts by displaying all the tasks and operations the user can do along with the numbers that the user has to enter to execute the given task and operation. When the user wants to exit the application they can type “0” to terminate the program along with closing the database connections.

When it comes to the internal structure of the application we see that we have 6 files. The App.java is responsible for starting the application and routing the users requests for a particular task/operation to the particular method that is responsible for handling the task/operation found in the other files. The Initilizer file is responsible for setting up and destroying the database and has methods to create, delete, and add dummy values to all the tables in the database. The last four files (Distribution.java, Edit_Publish.java, Production.java, and Report.java) are responsible for holding the methods that deal with all the tasks and operations in the respective groups from the project narrative. The methods in the last few files query the user for input on the task/operation, call/update/delete/insert values into the database, and return output to the command line for the given task/operation.

4. **As part of the documentation submitted as part of your project 3 report paper, explain who in your team played what functional role (e.g., software engineer, database designer/administrator, etc, see above under Organization) in each part (1 through 3) of the project. Submit the documentation.**

Functional Roles:

Part 1: Editing and publishing.

Software Engineer: Shakshi (Prime), Parth (Backup)
Database Designer/Administrator: Parth (Prime), Prashan (Backup)
Application Programmer: Prashan (Prime), Nisarg (Backup)
Test Plan Engineer: Nisarg (Prime), Shakshi (Backup)

Part 2: Production of a book edition or of an issue of a publication.

Software Engineer: Parth (Prime), Prashan(Backup)
Database Designer/Administrator: Shakshi(Prime), Parth(Backup)
Application Programmer: Nisarg (Prime), Shakshi(Backup)
Test Plan Engineer: Prashan (Prime), Nisarg(Backup)

Part 3: Distribution

Software Engineer: Nisarg (Prime), Prashan (Backup)
Database Designer/Administrator: Nisarg (Prime), Parth(Backup)
Application Programmer: Parth (Prime), Shakshi(Backup)
Test Plan Engineer: Shakshi(Prime), Prashan(Backup)

Part 4: Reports

Software Engineer: Prashan (Prime), Shakshi (Backup)
Database Designer/Administrator: Parth (Prime), Nisarg(Backup)
Application Programmer: Nisarg (Prime), Shakshi(Backup)
Test Plan Engineer: Prashan (Prime), Parth (Backup)