SAINT MARY'S UNIVERSITY SINCE 1802

One University. One World. Yours.

Master of Science

Computing and Data Analytics

MCDA 5540: Managing and Programming Databases

Project Report

Submitted by

Shubham Chumber(A00433094)

Nishant Malhotra(A00430215)

Sachit Jain(A00432721)

Submitted to

Trishla Shah

Table of contents:

1. Executive summary…………………………………………………… 3
2. About Data ………………………………………………………... 3
3. EER Diagram ……………………………………………………... 5
4. Relational Schema ………………………………………………. 6
5. Assumptions …………………………………………………….. 7
6. Normalization ……...……………………………………………… 9
7. Conclusion ……………………………………………………….. 10

1. Executive Summary

In this project we aim to tackle the problem of rectifying a Library Management system that stores information regarding the Library resources and its monthly expenditure. Our goal is to come up with an effective Database design to query, store and access resources and information. A good database schema is essential as it helps in maintaining data integrity, consistency and removes the possibility of redundant data and ambiguity. In order to complete the undertaken tasks, we assume the role of a Database Administrator to come up with an effective Database architecture. To achieve this task, we have used the RDBMS concepts such as Normalization, ER and EER model.

2. About Data

The Current information system consists of all the data pertaining to Authors and their books and Journals. However, we need to optimize this structure as it is not optimized for Journal and their implicit articles. This is because each scientific journal has a large multitude of articles ranging to many hundreds. Further, each Scientific Journal can have multiples editions published in a year. So we need to come up with an effective way to keep track of all articles information within each journal disseminated over years.

Along with extensive Journal information, we are also required to maintain constraints on the data that is kept in the tables such as any magazine must have a name assigned i.e. a tuple cannot have Null values in its field. Further we need to ensure that articles published within the same volume have exact same Publication year and that every magazine must maintain a Unique Volume field such that no two volumes can have same volume number.

We are also supposed to maintain information of the loyal Customer of the Halifax Science Library. Since the terms of loyal customer are ambiguous, we have made the assumption that it is those customers which have been visiting the Library for the past five years, as we are supposed to maintain records of Customers over a period of five years. We need keep record of various attributes pertaining to a customer.

As a part of the customer information we need to maintain, in addition to personal information, we need to keep track of the customer’s transactions. Each customer transaction needs to store the items purchased along with the price of each item. Every

customer can have multiple transactions hence we need to reference the Customer Id as a foreign key to Transactions entity.
The total price computed for the transaction is a function of discount code field mentioned in the customer table. The discount code in itself is a function of the total business (amount spent).

In addition to keeping track of the resources, HSL also wants keep a check of its monthly expenditures. The rent overhead is fixed for all the months in a year. Further, the employee cost also needs to be calculated which is the aggregation of the Salaries (hourly rate*hours worked).

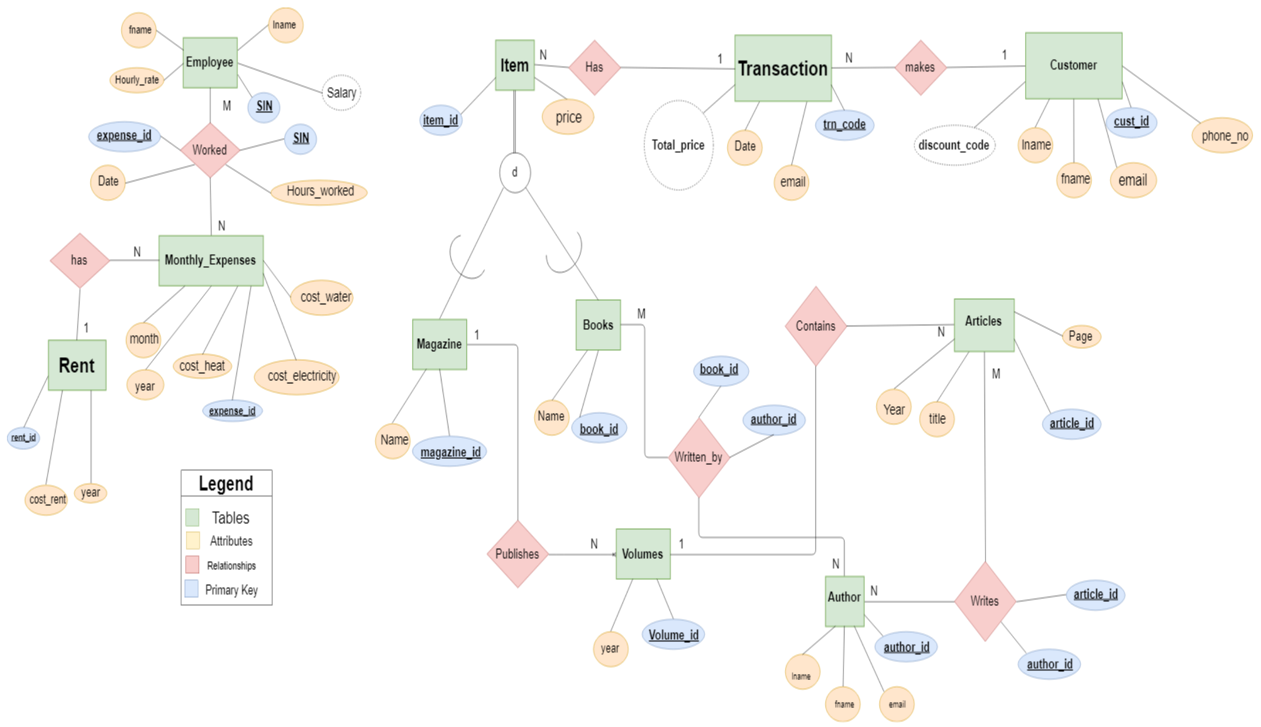
As part of our solution, we came up with an entity - relationship Diagram in order to diagrammatically depict the various entities within the Database, the data they will hold(fields) and their connections and relationships among each other.

The ER has 11(eleven) entities namely customer, transaction, book, author, magazine, volume, articles, library items, monthly expense, rent and employee.

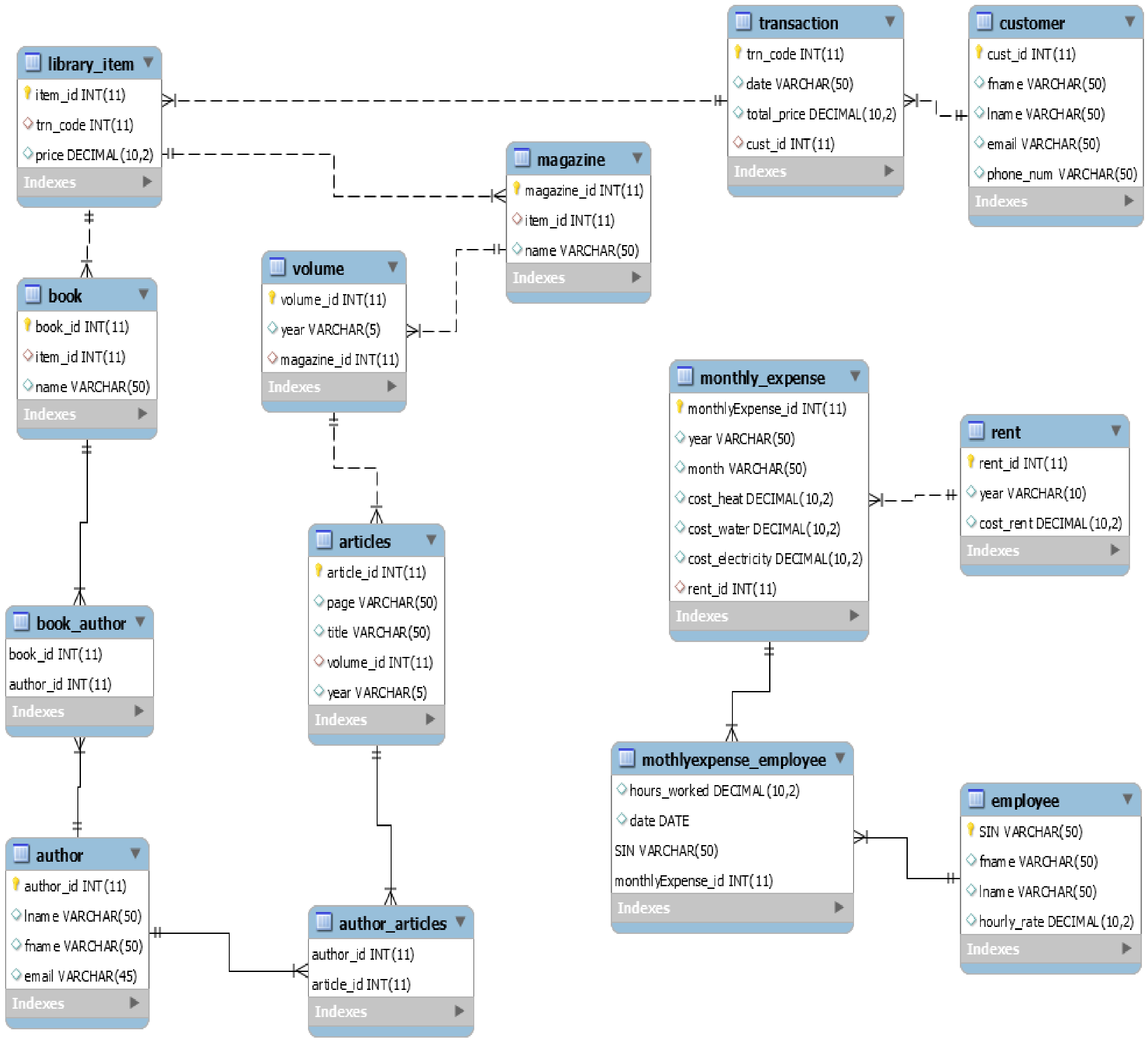
The customer table stores all the information pertaining to a single customer such as the customer id, name, email address and many more customer specifics. This entity is related to a Transactions entity with stores billing specifics such as the items bought, price and we have a customer id field to keep track if a particular transaction belongs to which customer. This Transaction entity is also linked to the Library item entity. This ways, transaction can refer to Library items and get detailed product information. The library entity is the super class and has a specialization of ‘Book’ and ‘Magazine’. The specialized classes have their own fields. In addition, both the entities are linked with another entity ‘Author’. In this entity we store author specific information. However, we will be using author id as the foreign key in Book and Magazine entities to relate the Library item products with the entire data for Author, as and when needed.

We have another set of identities, however they are not linked directly to the above set of entities. This section of ER is focused on calculating, estimating and querying the expenses. We have a ‘monthly expense’ entity which has fields for storing the electric, heat and water bills for a particular month of a year. Employees are assets to company’s effective business and all of them are entitled to a monthly remuneration. So we came up with another entity ‘Employee’ that stores the employees SIN, name and other personal information and hourly rate. When we relate this entity to expense table, on the relation we defined two additional attributes that allow us to store the numbers of hours an employee worked and on which date. We have another entity Rent which stores the year and the associated monthly rent(rent for each month in a year is same).

3. EER - Diagram



4. Relational Schema



Explanation:

Library_item (**item_id**, price, trn_code)
FOREIGN KEY Library_item (trn_code)
REFERENCES Transaction(trn_code)

Book (**book_id**, item_id , name)
FOREIGN KEY Book (item_id)
REFERENCES Library_item(item_id)

Book_Author (**book_id**, **author_id**)
FOREIGN KEY Book_Author (book_id)
REFERENCES Book(book_id)
FOREIGN KEY Book_Author (author_id)
REFERENCES Author(author_id)

Magazine (**magazine_id**, item_id, name)
FOREIGN KEY Magazine (item_id)
REFERENCES Library_item(item_id)

Volume (**volume_id**, year, **magazine_id**)
FOREIGN KEY Volume (magazine_id)
REFERENCES Magazine(magazine_id)

Articles (**article_id** , page, title , volume_id, year)
FOREIGN KEY Articles (volume_id)
REFERENCES Volume(volume_id)

Author_articles (**author_id**, **article_id**)
FOREIGN KEY Author_articles (article_id)
REFERENCES Articles(article_id)
FOREIGN KEY Author_articles (author_id)
REFERENCES Author(author_id)

Author (**author_id**, name, fname, lname)

Transaction (**trn_code**, date, total_price, item_name, cust_id)
FOREIGN KEY Transaction (cust_id)
REFERENCES Customer(cust_id)

Customer (**cust_id**, fname, lname, email, phone_num)

Monthly_expense (**monthlyExpense_id**, year, month, cost_heat, cost_water, cost_electricity, rent_id)
FOREIGN KEY Monthly_expense (rent_id)

REFERENCES Rent(rent_id)

MothlyExpense_employee (hours_worked, date, **SIN**, **monthlyExpense_id**)
FOREIGN KEY MothlyExpense_employee (monthlyExpense_id)
REFERENCES Monthly_expense(monthlyExpense_id)
FOREIGN KEY MothlyExpense_employee (SIN)
REFERENCES Employee(SIN)

Employee (**SIN**, fname, lname, hourly_rate)

Rent (**rent_id**, year, cost_rent)

5. Assumptions

- **Transaction: Items** → **1: N** because multiple items can be found in one transaction however, multiple transaction cannot be mapped to same items as every transaction has a unique ID.
- **Customer: Transactions** → **1: N** because multiple transactions can be mapped to single user however a single transaction cannot belong to multiple customers.
- **Magazine: Volumes** → **1: N** because one magazine has multiple volumes in a single year however a unique volume cannot belong to multiple Magazines.
- **Volumes: Articles** → **1: N** because one Volume can have a multitude articles however one article cannot be in multiple volumes.
- **Books: Author** & **Author: Articles** → **M: N** relationships, as multiple authors can contribute to multiple items (book or article) and one author can write multiple article or books.
- **Monthly_expense**: **Employee** → **M: N** because one monthly expense consists of multiple employee (salaries) and one employee will contribute to multiple monthly expense reports.
- **Rent: Monthly_expense** → **1: N** as property expenditure of multiple months will depend on one Rent entry for that year.
- We have taken the Date field as Varchar instead of Date datatype in tables transaction and relation worked as it allows us to simplify parsing data between multiple applications.

6. Normalization

Our relational schema is Normalized to **3NF** highest normal form.

Monthly Expenses - We broke this entity into three tables -

1) Employee - We have added information pertaining to an individual employee. We broke employee table from monthly expense because employee can have varying working hours but fixed hour rate, further they can be part-time, full-time or contractual, so the hours worked per month vary. Hence, accommodating this info in monthly expense table would add redundancy.
2) Monthly Expense- This table has fixed expenses which do not change(Assumption)
3) Rent - We separated data into Rent table as for each month in a particular year, the rent is fixed. The Expense table contains year which we can be used to calculate Rent for all 12 months of that year from the corresponding year.

Magazine - the table was normalized as follows -

1) Magazine - Simple stores the Name of Magazine and its corresponding ID.
2) Volumes - This table stores all information for every volume of a particular Magazine. This allowed us to reduce redundancy in Magazine table.
3) Articles - This allowed us to reduce redundancy in volume table as all articles in a particular magazine can be referenced by Volume_Id foreign key thus help us achieve redundancy without leading to any loss of Data.

7. Conclusion

In our Report, we have optimized the Library's' information storage structure by coming up with an optimized Database structure that not only stores all the business information but also achieved consistent data standards and removes any scope of potential redundancies. We normalized our Schema to the highest Normal form and successfully merged the existing Data with the new Database tables.