SAINT MARY'S UNIVERSITY SINCE 1802

One University. One World. Yours.

COMPUTING AND DATA ANALYTICS 5540

-- TEAM PROJECT

FINAL REPORT

Submitted by

Team Members:

NAME	A#	Email
Manoj Bandaru	A00433174	Manoj.bandaru@smu.ca
Goutham Bommu	A00432541	Goutham.bommu@smu.ca
Manoj Morishetty	A00432854	Manoj.morishetty@smu.ca
Chaitanya Varma	A00432282	Chaitanya.varma.mudundi@smu.ca

Submitted to

Trishla Shah

Contents

Figures

Executive Summary:

Introduction:

The Halifax Science Library (HSL) maintains an SQL database containing tables about various publications. HSL now wants to implement the following requirements:

- To sell library Items Store transaction of sales of items in library.
- Maintain records of all magazines, volumes and articles.
- Record monthly expenses of HSL.

Approach:

A Data model has been designed for HSL. Information about magazines, articles, customers, transactions and monthly expenses are stored. We started from creating ER/EER diagrams and continued to relational schema. All tables are normalized till 3NF as per the requirements. A file named articles. json has been extracted, transformation has been done using python scripts which are triggered by bash scripts. Thus, loading all information from source json file to MySQL tables. MVC application has been created from this data model through which we can see total number of tables available, create customer, create transaction, cancel transaction and much more.

Data Requirements:

Existing Tables:

HSL personnel has already extracted and recorded in the SQL database some names and contact information of authors from available magazines. The Existing tables which we were provided are:

- Author
- Magazine
- Item

New Tables:

We created new tables keeping in mind of the requirement's and making a note of Discount code and transaction sum. After continuous brainstorming we arrived at some tables which are better for HSL to maintain. The new tables are:

- Article
- Author_list
- Customer
- Employee_monthly_expenses
- Employees
- Monthly_expenses
- Master
- Trans_cost
- Trans_item
- Transactions
- Volume

Design: [4]

For any organization designing database is crucial. We started with ER/EER diagrams after analyzing the business requirements. The ER/EER diagrams are transformed into tables which are in 3NF.

ER diagrams:

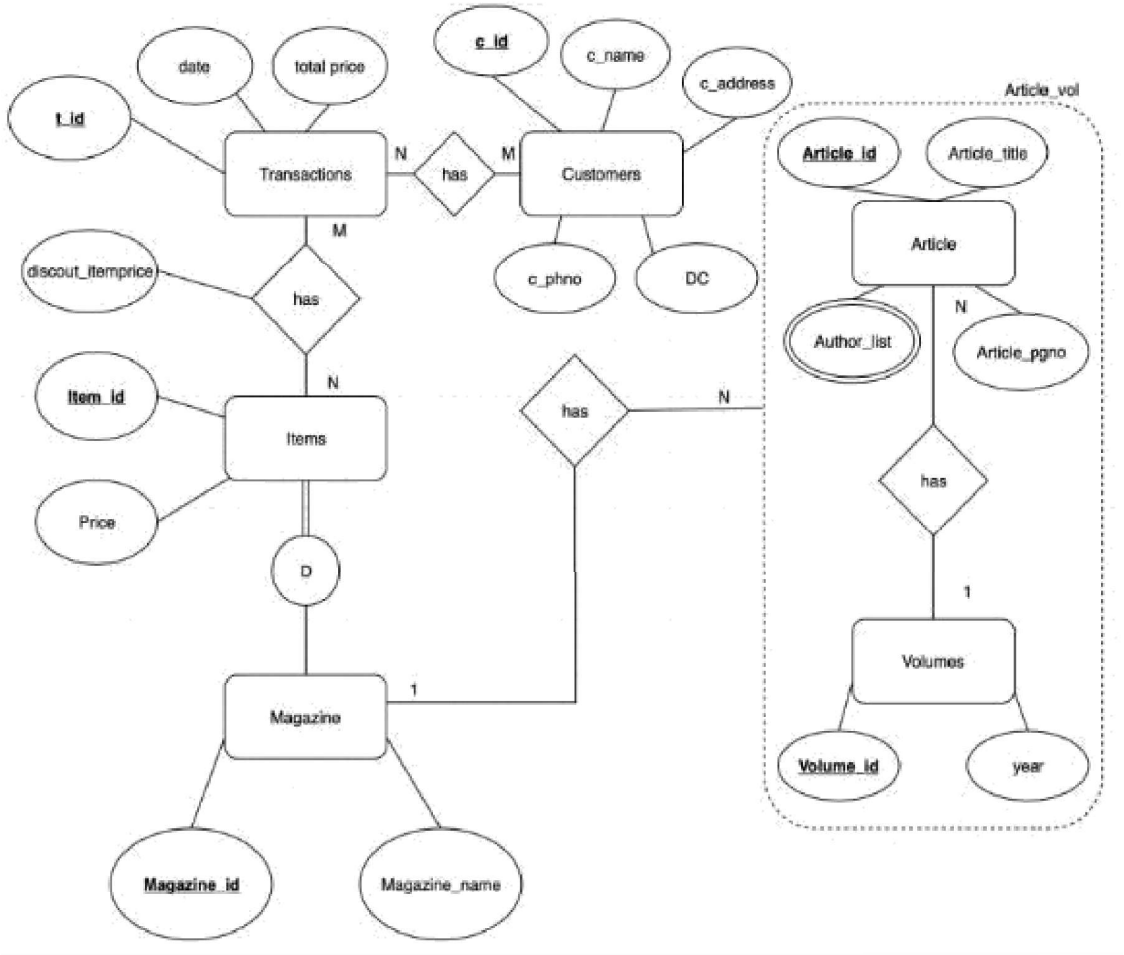


Figure 1- EER Diagram

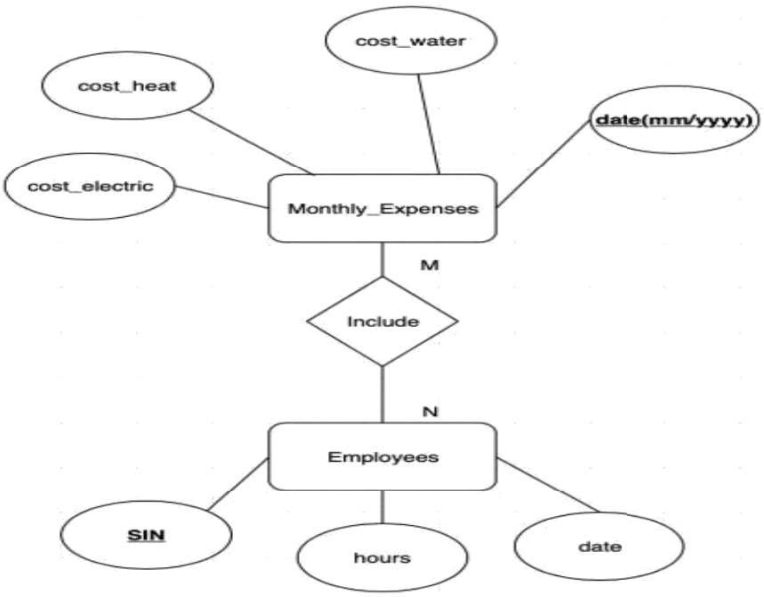
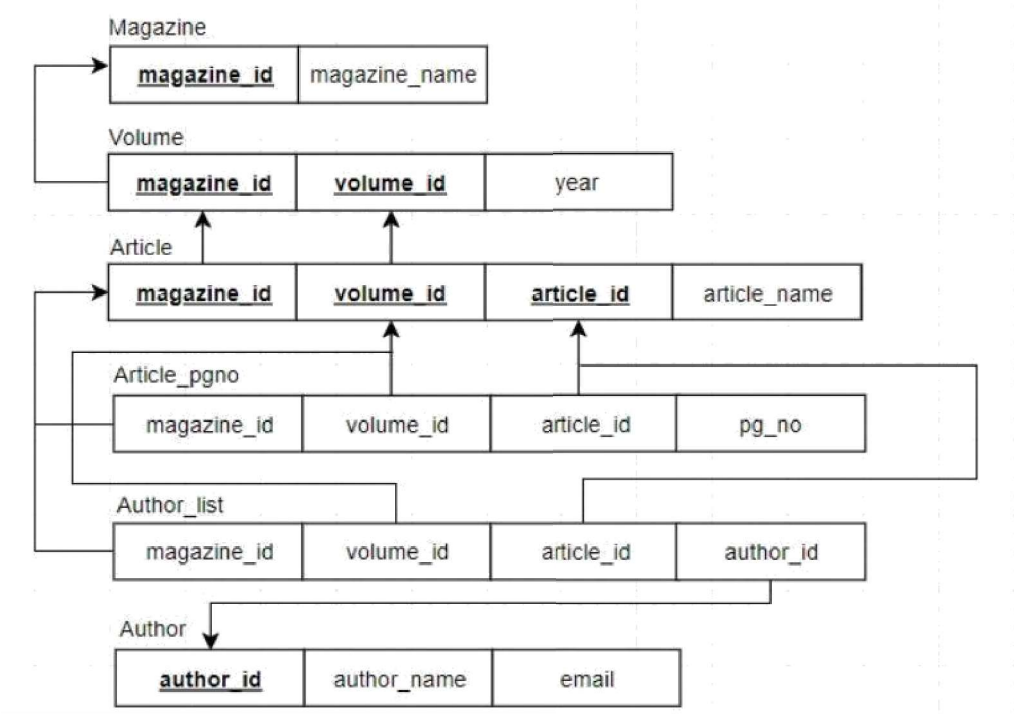


Figure 2-ER Diagram

Relational Schema:

Customer & Transactions table:



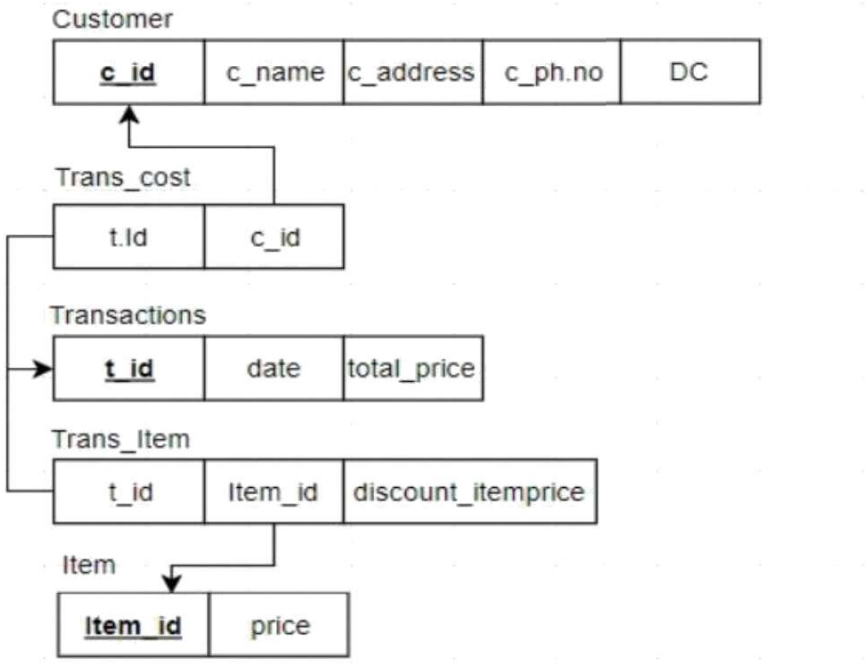


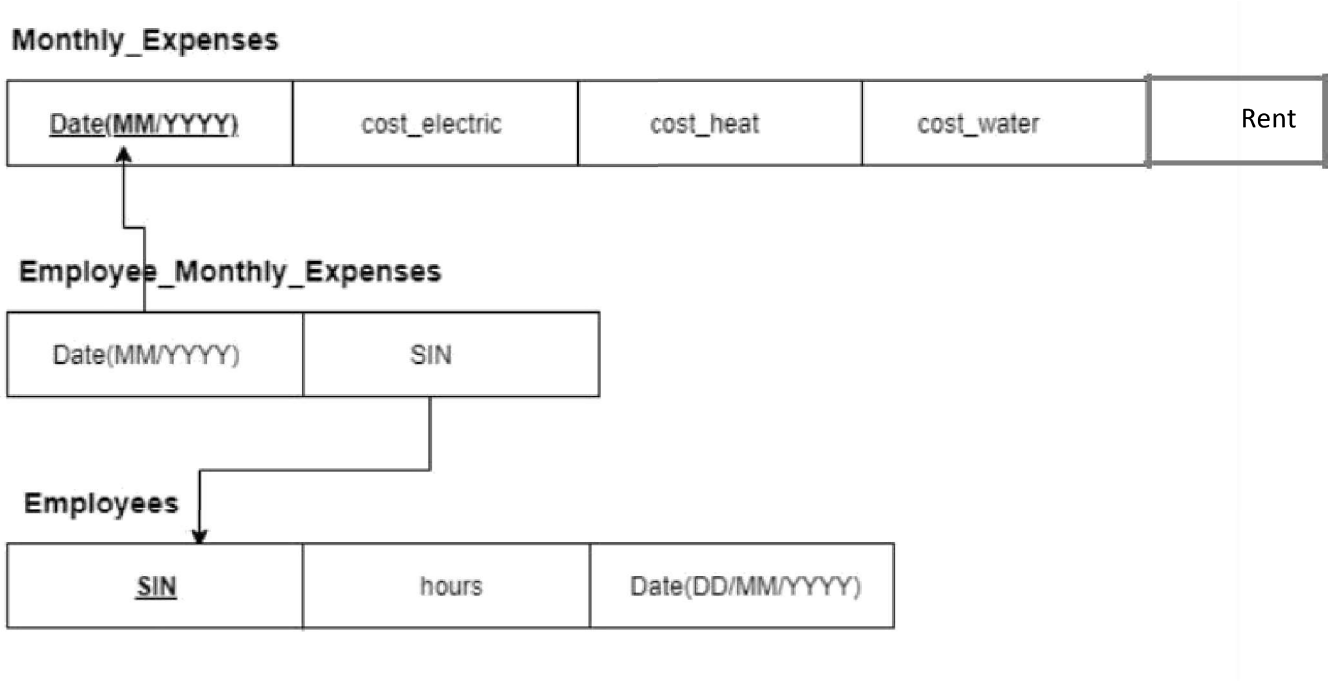
Figure 3- Relational Schema

Description:

- The Magazine entity has an attributes of magazine name (magazine_name) and magazine Id (magazine_id), where magazine id as primary key.
- As Article entity and Volume entity depend on magazine Id, also article entity has a relation with volume, we represented those two entities in aggregation. The cardinality between them is represented as 1-N.
- The Volume Entity has attributes as volume id and Year of publication.
- The Article Entity has attributes as article_id as primary key, article_title, volume_id, magazine_id where author_list is multi valued.
- As author_list is multi-valued, a new table Author_list has been framed which has attributes as author_id, article_id, magazine_id, volume_id.
- Customers Entity have attributes of Customer Id(c_id), Customer name(c_name), customer address(c_address), customer phone number(c_ph.no) and Discount code (DC).
- We have transactions entity which has an attribute of Transaction Id (t_id), date of transaction, and Total price (total_price) for that transaction.
- As we have M-N cardinality between these two attributes which we named the table as Trans_cost and has attributes of transaction Id and customer id as foreign key from respective entities.
- We have Item Entity which has an attribute of Item_Id and Price of each item.

- As we have M-N cardinality between Transactions and Item entities, which lead to relation table of Trans_Item where that entity has transaction Id, Item Id as foreign key references from transactions and item entities.
- The Trans_Item has an attribute of discount_itemprice and as per our design we will calculate discounted price for each Item in transaction using DC.

Monthly Expenses Table:



Description:

- Monthly Expenses entity has electric, heat, water, rent and date with format MM/YYYY as primary key.
- The employee's entity will have an attributes of SIN as primary key, Hours worked and date of the year on which they worked.
- As the cardinality between two entities is M-N, we have relation entity, which is employee_monthly_expenses that has attributes of Date and SIN as foreign key references from two respective entities.

Normalization:

Normalization is the process of organizing data in a database. This includes creating tables and establishing relationships between those tables according to rules designed both to perfect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency.

- All our tables are normalized till 3NF to eliminate redundancy and inconsistent dependency by following normalization rules.

Assumptions while making design:

For designing ER/EER we made some assumptions which lead to mapping of tables and feasible for querying the tables. The assumptions we made are:

- Every magazine has a unique magazine id and it's each volume are also identified with unique ID's.
- As we have ITEM table which is existing table from HSL personnel, we assumed that magazine is one of the items so that it would be easy to query.

Data to MongoDB:

Description:

The main objective of this task is to get relevant information from the source "articles.json" file that suits our MySQL tables. The file "articles.json" has the data regarding articles, authors and some useless data which must be cleaned to make this process easy. So, task is to write a BASH script which will trigger a python script that will clean up the "articles.json" file.

Data2mongo BASH Script:

Algorithm:

New Tables Creation:

Initially the tables of the database are created and "new_tables.sql" file is executed by this BASH Script, which will connect to MySQL and executes our script "new_tables.sql" by using the database "project".

Data Cleaning and creating Mongo collection:

To get the relevant information from file "articles.json", cleaning of the file must be done and as mentioned earlier BASH script triggers python script and cleans the "articles.json" file and creates the file "article.json" file, further this "article.json" has been imported as collection to mongo DB.

[2][3][6]

Mongo to SQL:

Description:

The objective of this task is to load the data which we retrieved from 1st BASH Script and loading the data into MySQL tables at required attributes. For this task we thought of two approaches where we can read the collections from mongo DB and import the data into MySQL tables at required constraints.

Approach 1:

Initial approach is to read the collections from mongo DB, export the collections to intermediate result set called “Article.csv”. Then this result set is imported to MySQL tables using BASH script. This result set is imported to master table in database which is done by mongo2db BASH script. As trigger exists on the master table, for importing each record from csv file into the master table it gets segregated into specific tables as per set constraints.

Approach 2:

Alternate approach for this exporting data into MySQL is to write a python script which will read both collections Author and Article, then exporting them to required tables in MySQL as per fields. This approach is solely depending on Python file.

Conclusion:

If we use the second approach, while inserting each record form mongo collection to MYSQL tables, many validations must be done to make the record unique which creates a load on MYSQL and as well as for developer to validate all scenarios.

Instead in approach 1, we created a master table in database and a trigger was written on it. So that all validations get done at the database side which makes job simple.

As database is created with good constrained tables, it is easy to scaffold a table to views in ASP.NET MVC architecture and handle the data. So, we went on with MVC web application.

So, to conclude on the approaches and we proceeded with approach 1 for the web application as we considered that approach is more feasible for the application which eliminates redundancy and maintains referential integrity.

[6]

MVC web Application:

Description:

The Database has most significant use when combined with an application. The final part of the task is to provide an interface for the users to perform specific function for the end user. An application can be self-contained or group of programs. The program is a set of operations that runs the application for the user. To perform specific functionalities for the end user we choose to create a web application with ASP.NET framework MVC. The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. We The functionalities which are attained through this application are:

- Show Tables
- Add new article
- Add new customer
- Add new transaction
- Cancel transaction

Home Page:

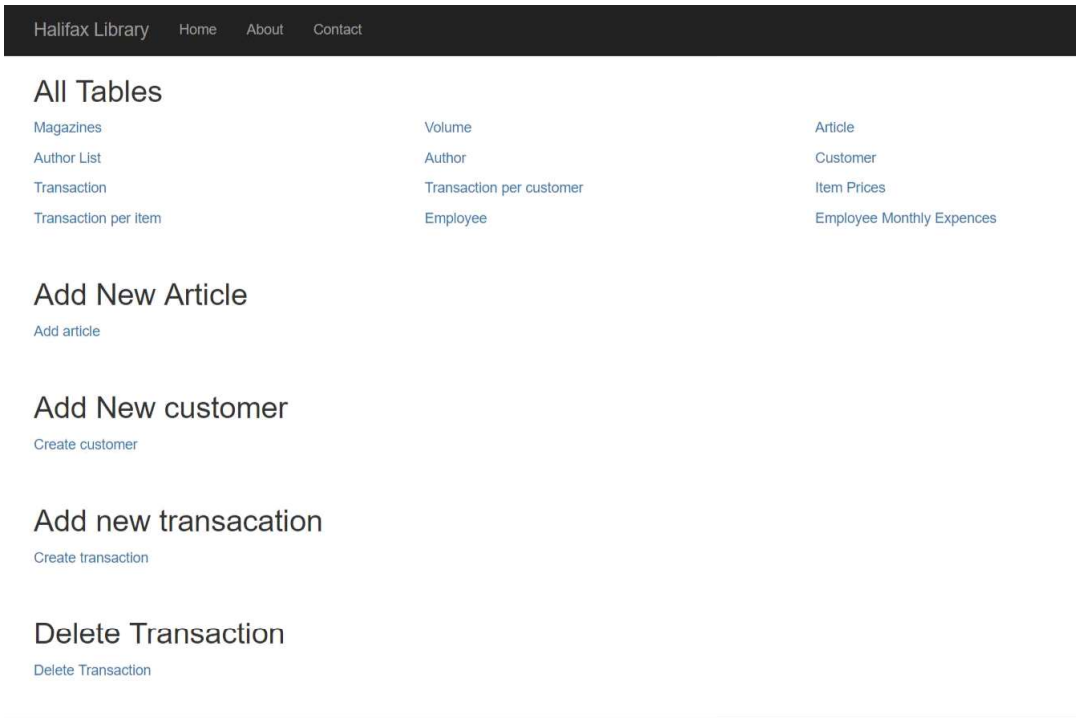


Figure 4-Home Page

Show Tables:

The Home page has a module of all tables which will display all tables names and clicking on the table will give you the data in each table.

Each section will display the data according to the business requirements.

Add new article:

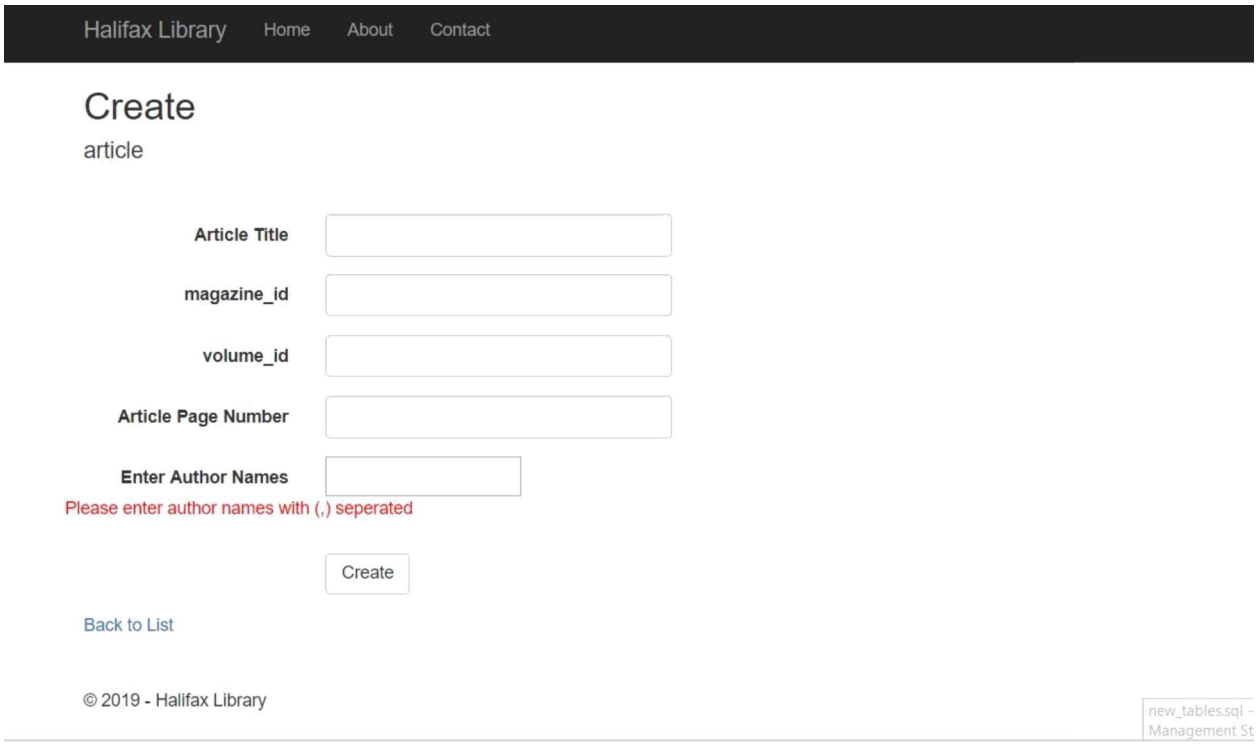


Figure 5-Add new article

Description:

- We will use this module to enter information about article into database.
- The module "Add new article" gives you the prompt shown above where admin will add the attributes of the article which are required.
- In the author names section, if article is written by multiple author's admin should enter the author names with (,) separated.
- Clicking on the create button will insert a new article into database.

Add new Customer:

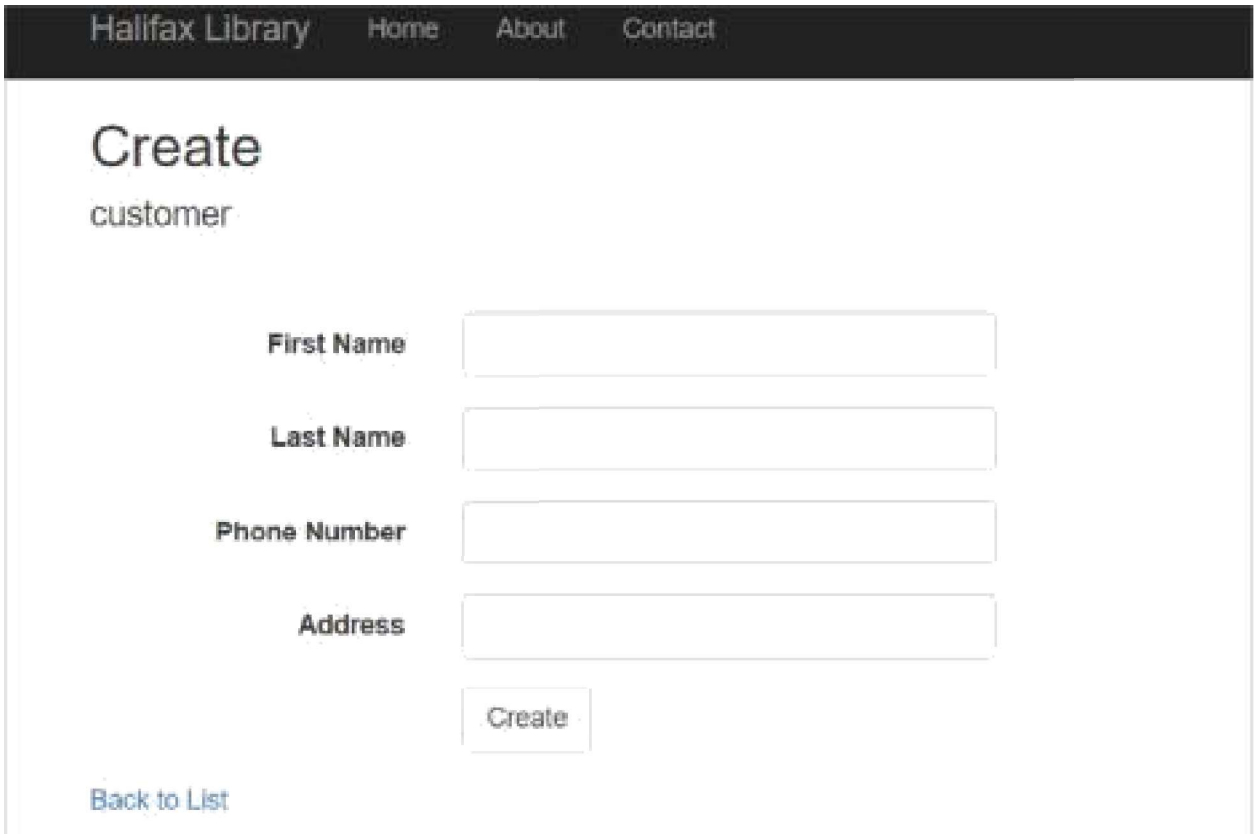


Figure 6-Add new Customer

Description:

- We will use this module to enter information about customer into database.
- The module “Add new customer” gives you the prompt shown above where admin will add the attributes of the customer which are required.
- If a customer already exists in the database, we are prompting that customer already exists at the top of field section.
- Clicking on the create button will insert a new customer into database.

Add new transaction:

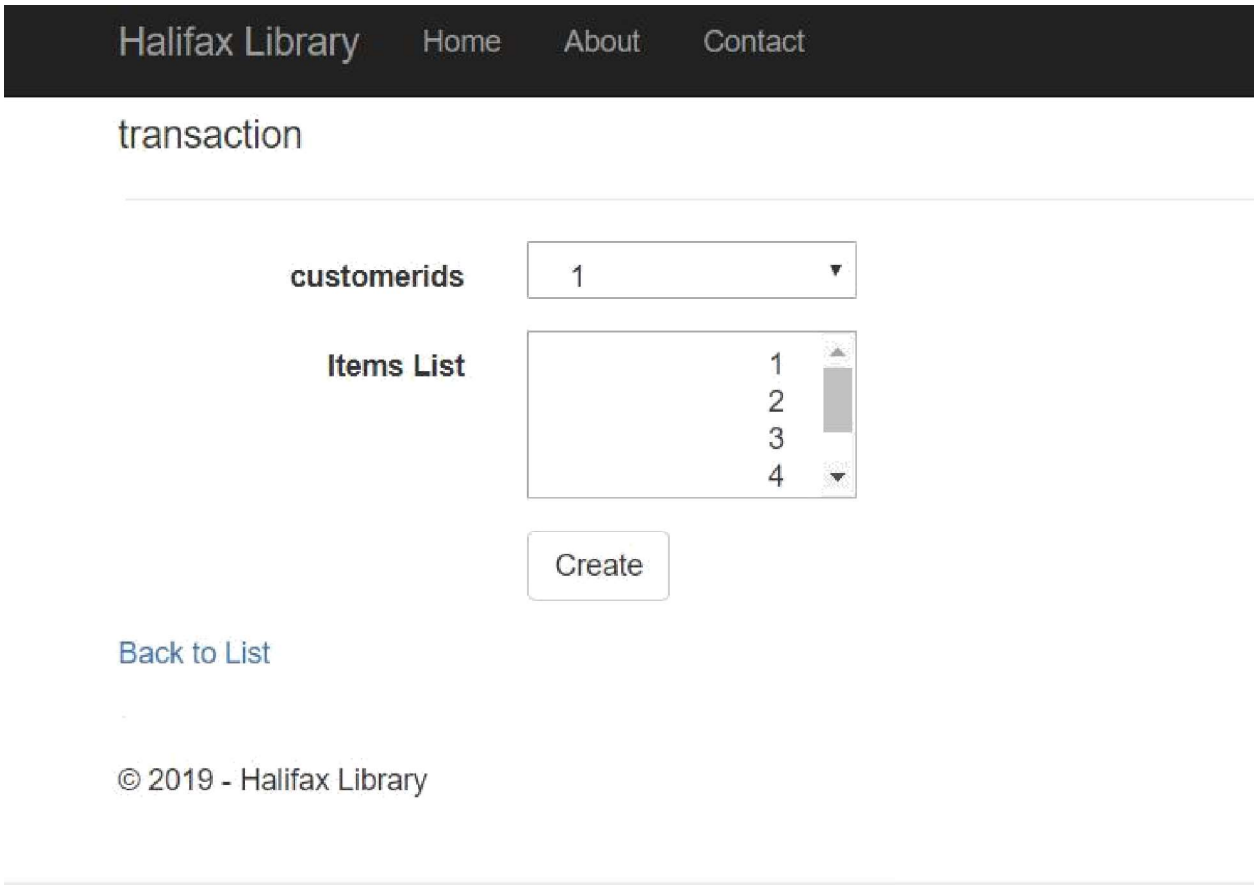


Figure 7-Add new Transaction

Description:

- In this module, admin can make a transaction pertaining to customer by selecting customer id and his required items in the list box.
- This triggers the index view and inserts the data into trans_cost and trans_item tables.
- Now, admin can view the transaction details from transaction table.
- The Trans_cost table now has t_id, item_id and discounted_item price.
- The Trans_item table now has t_id and customer ID.

Cancel Transaction:

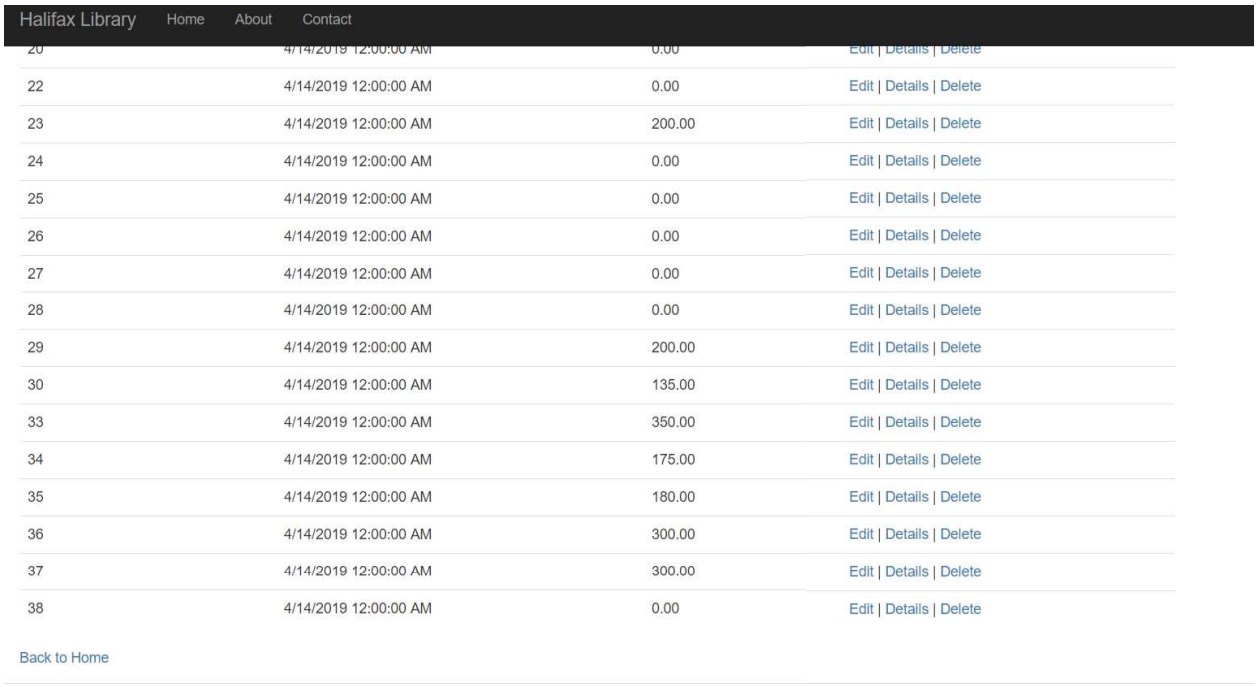


Figure 8-Cancel Transaction

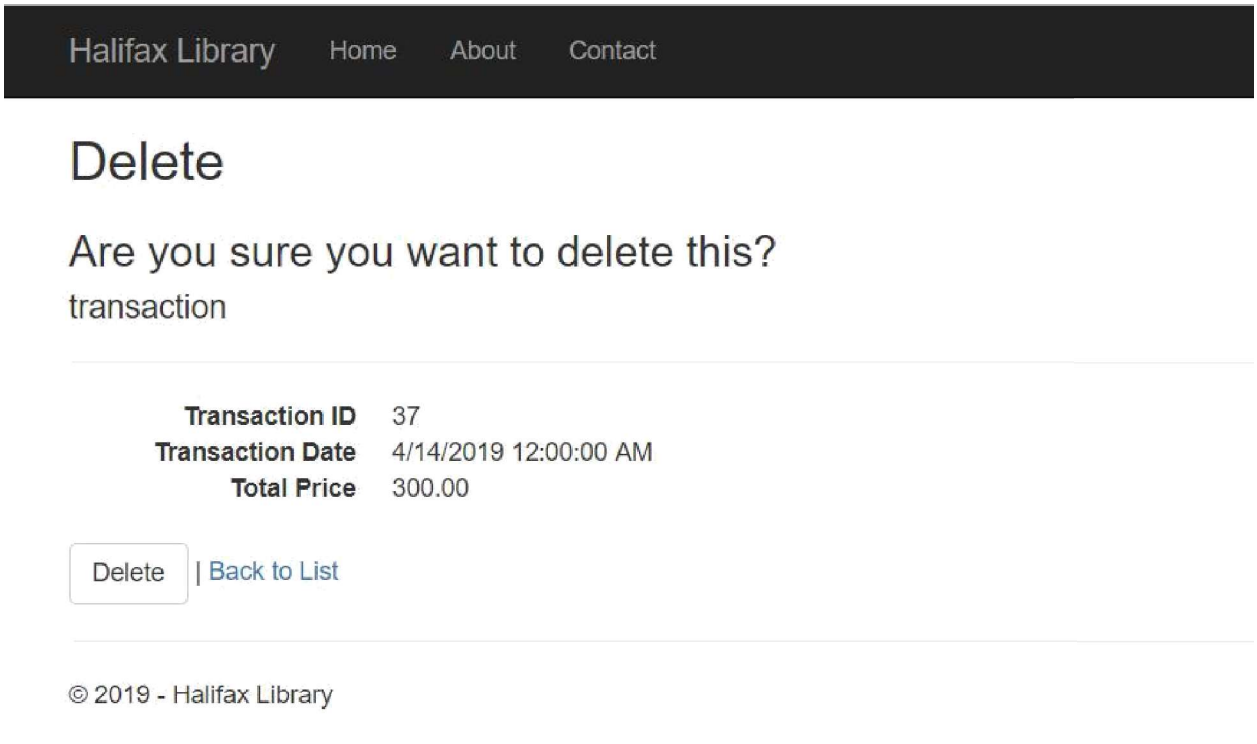


Figure 9-Delete Transaction Prompt

Description:

The module cancel transaction, first has all the details of all transactions based on transaction ID admin can cancel the transaction (Delete Transaction).

Then if user deletes the transaction ID, our application prompts to clarify the deletion and finally deletes the transaction on confirming the deletion.

Conclusion:

As per the problem stated by HSL and by analyzing business requirements of the Halifax Science Library, Design has been made fulfilling all constraints and required attributes. Then looking up to Existing tables, new tables are invented to map them for easy querying.

The source “articles.json” has required data which fits into MySQL newly created tables, before further proceedings as the source has useless data and must be transformed to an efficient manner, we used Python for the transformation and took the cleaned output as “Article.json”.

Before loading into MySQL tables, the task is to load the cleaned Json file into mongo DB and to create collection in mongo DB. Further, another BASH script will get triggered in this BASH which is required to do as per the task and this script exports collection into .csv file.

The .csv file will be imported into master table in the same database and with a trigger on the master table sparks all data into individual tables as per required attributes. All tables in MySQL are populated with data.

Finally, an application has been built for specific functionalities in ASP.NET framework MVC, which will make HSL personnel to see all tables, add new article, add new customer, add new transaction, Cancel transaction.

Appendix-A(References):

1. https://smu.brightspace.com/d2l/le/content/45923/Home (used these for complete reference)
2. https://stackoverflow.com/questions/33796151/mysql-error-1265-data-truncated-for-column-name-at-row
3. https://stackoverflow.com/questions/642154/how-to-convert-strings-into-integers-in-python
4. https://www.slideshare.net/brindamary/mapping-er-and-eer-model
5. https://smu.brightspace.com/d2l/le/content/45923/viewContent/475492/View (BASH SCRIPT)

Appendix-B(Triggers):

```
DELIMITER $$
CREATE
    TRIGGER database_update AFTER INSERT
    ON master
    FOR EACH ROW BEGIN
        DECLARE x INT;
        DECLARE mag_Id INT;
        DECLARE auth_Id INT;

        SET x = (SELECT COUNT(*) FROM magazine WHERE magazine_name = NEW.magazine_name);
        IF (x = 0) THEN
            INSERT INTO magazine(magazine_name) VALUES(NEW.magazine_name);
        END IF;

        SET mag_Id = (SELECT _id FROM magazine WHERE magazine_name = NEW.magazine_name);
        SET x = (select COUNT(*) from volume where magazine_id = mag_Id && volume_id = NEW.volume_id);
        IF (x = 0) THEN
            INSERT INTO volume(magazine_id, volume_id, year_published) VALUES(mag_Id, NEW.volume_id, NEW.year_published);
        END IF;

        SET x = (select COUNT(*) from article where magazine_id = mag_Id && volume_id = NEW.volume_id && article_id = NEW.article_id);
        IF (x = 0) THEN
            INSERT INTO article(magazine_id, volume_id, article_id, article_name, pg_no) VALUES(mag_Id, NEW.volume_id, NEW.article_id, NEW.article_name, NEW.pg_no);
        END IF;

        SET auth_Id = (SELECT COUNT(*) FROM author WHERE fname != NEW.fname);
        SET x = (select COUNT(*) from author where author_id = auth_Id);
        IF (x = 0) THEN
            INSERT INTO author(fname, lname) VALUES(NEW.fname, NEW.lname);
        END IF;


        SET x = (select COUNT(*) from author_list where magazine_id = mag_Id && volume_id = NEW.volume_id && article_id = NEW.article_id && author_id = auth_id );
        IF (x = 0) THEN
            INSERT INTO author_list(magazine_id, volume_id, article_id, author_id) VALUES(mag_Id, NEW.volume_id, NEW.article_id, auth_Id );
        END IF;

    END $$
DELIMITER ;
```

Figure 10-Trigger

Appendix-C (BASH Scripts):

data2mongo.sh

mongo2sql.sh

Appendix-D (Python Scripts):

cleanJson.py

mongo_to_sql.py

Appendix-E (SQL Script):