



**One University. One World. Yours.**

# **Managing & Programming Database**

## **MCDA5540**

**Master of Science in Computing and Data Analytics  
Team Project Halifax Science Library (HSL)**

**Submitted by:**

Caner Adil Irfanoglu (A00425840)  
Sunil Padikar (A00428089)  
Vinay Govindan (A00429120)  
Gaganpreet Singh (A00429660)

**Submitted to:**

TRISHLA SHAH

## Contents

EER Diagram:	3
Schema :	4
Current Status of tables :	6
Data Load Scripting workflow	8
Data2mongo.sh:	8
Make_coll_ordparts.js:	9
mongo2sql.sh:	9
Make_coll_orderparts1.js:	10
Php Application workflow	11
main.php:	12
add_article.php:	13
add_customer.php:	15
Create_new_transaction.php:	17
Cancel_transaction.php:	19
show_tables.php:	22
print_tables.php:	23
References:	25

## EER Diagram:

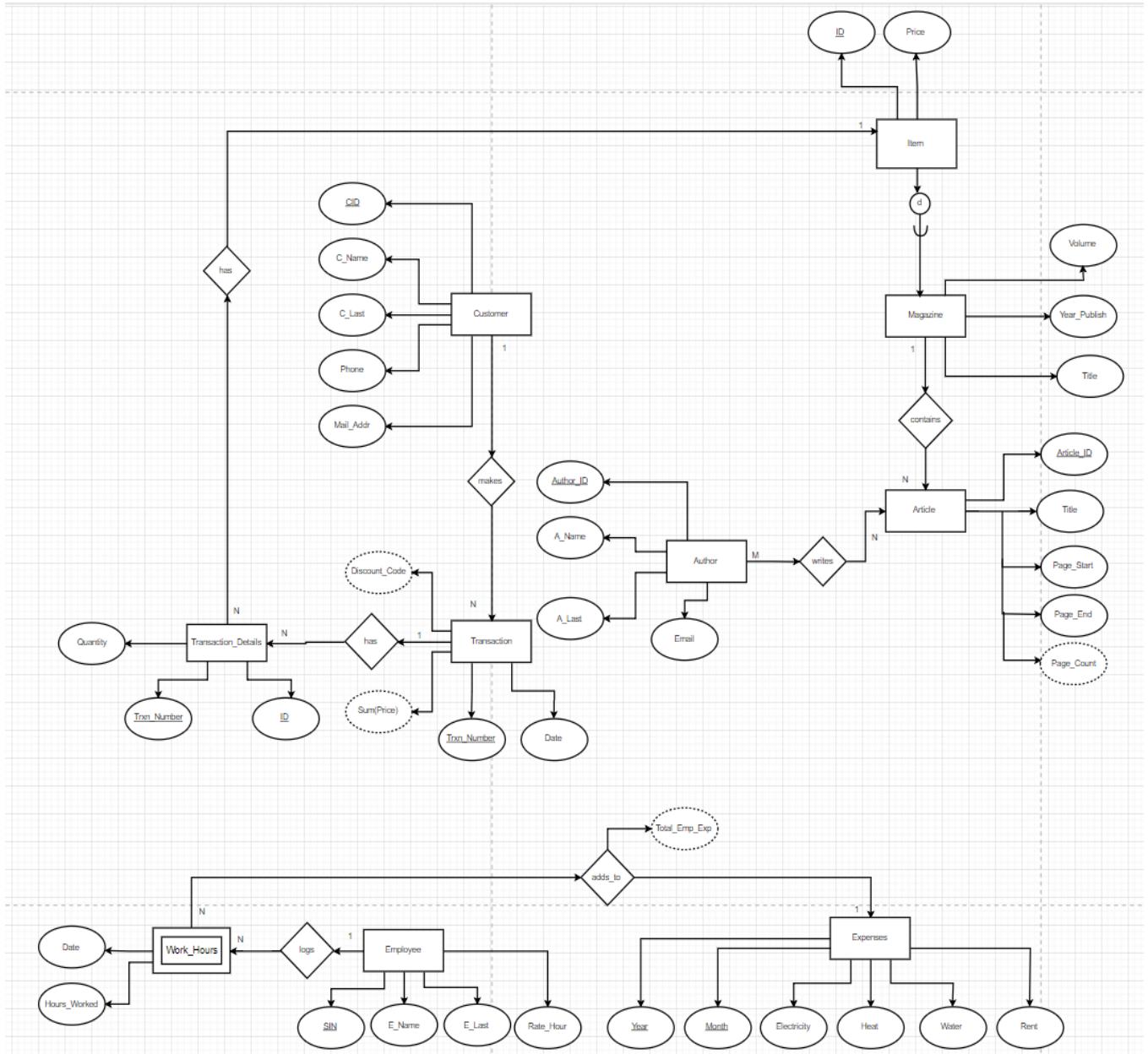


Figure 1 EER Diagram

## Schema :

All the tables below are in 1NF as each field has only one datatype, has only a single value per row and each row can be uniquely identified.

### **ITEM(ID, Price)**

ITEM tables has no partial or transitive dependency as price is uniquely identified by the primary key ID and price cannot determined ID uniquely hence the relation is in BCNF.

Note: ID is unique for each vol. magazine and it is foreign key in MAGAZINE table as Magazine\_ID

### **MAGAZINE(Maganize\_ID unique NOT NULL , Title, Year\_Publish, Volume)**

#### **Foreign Key MAGAZINE (Maganize\_ID) references ITEM (ID)**

Magazine is in 2NF and has no partial dependency as all the fields depends on both Title and Volume which put together forms the Candidate Key. It is in 3NF, BCNF as Maganize\_ID and Year\_Publish depends only on Title+ Volume & not the other way around hence there is no transitive dependency.

### **ARTICLE(Article\_ID, Title, Page\_Start, Page\_End, Magazine\_ID NOT NULL)**

#### **Foreign Key ARTICLE (Magazine\_ID) references MAGAZINE (Magazine\_ID)**

The table Article is already in 3NF, BCNF as there is No partial and transitive dependencies since all the fields depend on Article\_ID & not the other way around.

### **AUTHOR(Author\_ID, A\_Name, A\_Last, Email)**

The Author table is in 3NF, no partial and transitive dependencies as all the fields depend on Article\_ID.

### **WRITES(Author\_ID, Article\_ID)**

#### **Foreign Key WRITES (Article\_ID) references ARTICLE (Article\_ID)**

#### **Foreign Key WRITES (Author\_ID) references AUTHOR (Author\_ID)**

Writes table satisfies BCNF because both Author\_ID and Article\_ID form the candidate key and there is no partial or transitive dependency.

### **CUSTOMER(CID, C\_Name, C\_Last, Phone, Mail\_Addr)**

The Customer table is already in 3NF, BCNF as there is no partial and transitive dependencies since all the fields depend on CID and not the other way around.

### **TRANSACTION (Trxn\_Number, Date, CID, Discount\_Code)**

**Foreign Key TRANSACTION(CID) references CUSTOMER(CID)**

The Transaction table is in 3NF, BCNF since there is no partial or transitive dependency and when a customer levels up their discount code will change even though CID remains the same. So, Discount Code is only determined by Trxn\_Number and so are the other fields.

**TRANSACTION\_DETAILS (Trxn\_Number, ID, Quantity)**

**Foreign Key TRANSACTION\_DETAILS(Trxn\_Number) references TRANSACTION(Trxn\_Number)**

**Foreign Key TRANSACTION\_DETAILS(ID) references ITEM(ID)**

The Transaction\_Details table is in 3NF, BCNF since there is no partial or transitive dependency and since a customer can purchase same items on different transactions Trxn\_Number is required alongside with ID. Also, Trxn\_Number is not sufficient to determine quantity, since the transaction can include same quantities of different items.

So, Quantity is only determined by using Trxn\_Number and ID and not the other way around.

**EMPLOYEE (SIN, E\_Name, E\_Last, Rate\_Hour)**

The Employee table is already is in 3NF, BCNF where there is no partial and transitive dependencies as all the fields depend on SIN and not the other way around.

**WORK\_HOURS (Date, SIN, Hours\_Worked)**

**Foreign Key WORK\_HOURS (SIN) references EMPLOYEE (SIN)**

The WORK\_HOURS table is already is in 3NF, BCNF where there is no partial and transitive dependencies as all the fields depend on Date and SIN and not the other way around.

**EXPENSES (Year, Month, Electricity, Heat, Water, Rent, Total\_Emp\_Exp)**

The Expenses table is already is in 3NF, BCNF as there is no partial and transitive dependencies since all the fields depend on Year and Month and not the other way around.

## Current Status of tables :

Here we are just considering a sample of two tables “author” and “item” table to show its current state of query execution plan.

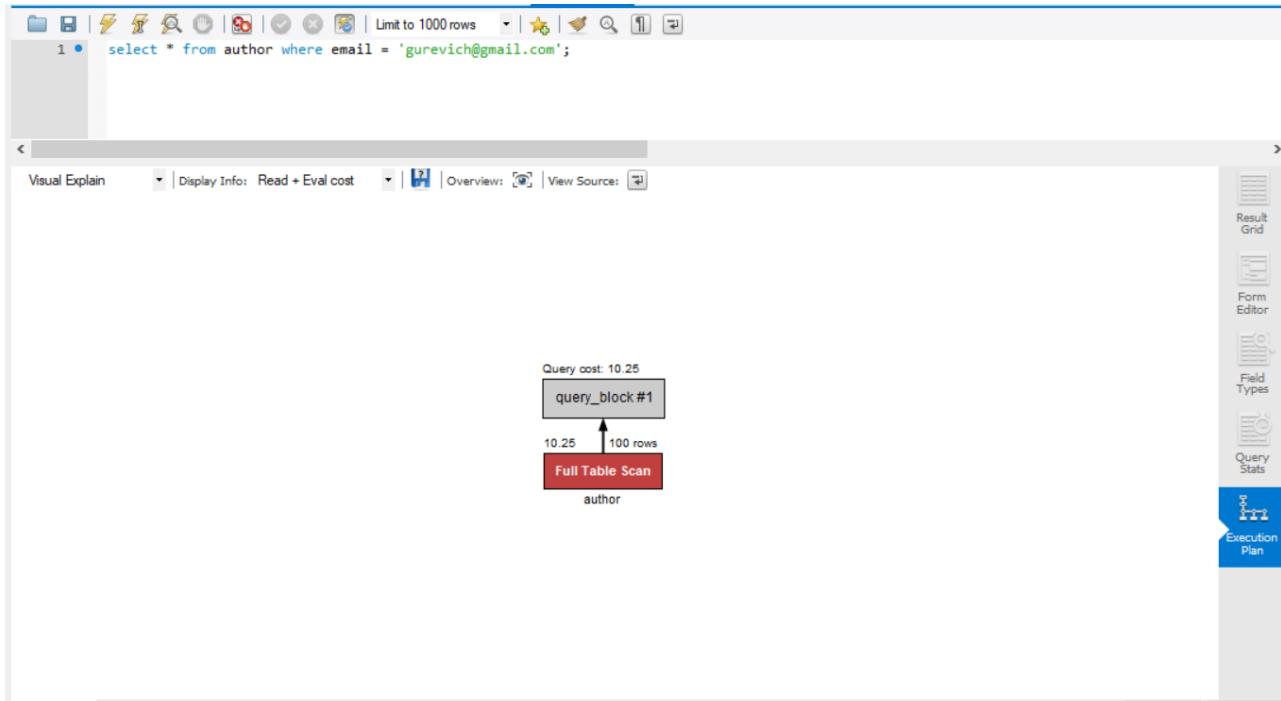


Figure 2 Query without optimization

The author table now upon querying for a author based on the email even thought it is logically unique to every author scans the entire table. Indexing the author table based on the email would reduce the number of scans.

The screenshot shows a database management system's query optimizer interface. At the top, a SQL query is displayed: `1 • select * from item where price = 10;`. Below the query, there are several tabs: "Visual Explain" (selected), "Display Info", "Read + Eval cost", "Overview", and "View Source". On the right side, there is a vertical toolbar with icons for "Result Grid", "Form Editor", "Field Types", "Query Stats", and "Execution Plan" (which is highlighted in blue). The main area displays a visual execution plan for a "Full Table Scan" on the "item" table. The plan shows a red box labeled "Full Table Scan" pointing to the "item" table. Above it, a box labeled "query\_block #1" contains the text "Query cost: 1.05" and "8 rows". A small arrow points from the "Full Table Scan" box to the "query\_block #1" box. At the bottom left, there is a tab labeled "item 7" with an "X" button. At the bottom right, there are "Apply" and "Revert" buttons.

Figure 3 Query after Optimization

Similarly the item table can also be indexed based on the price which would reduce the number of rows being scanned.

## Data Load Scripting workflow

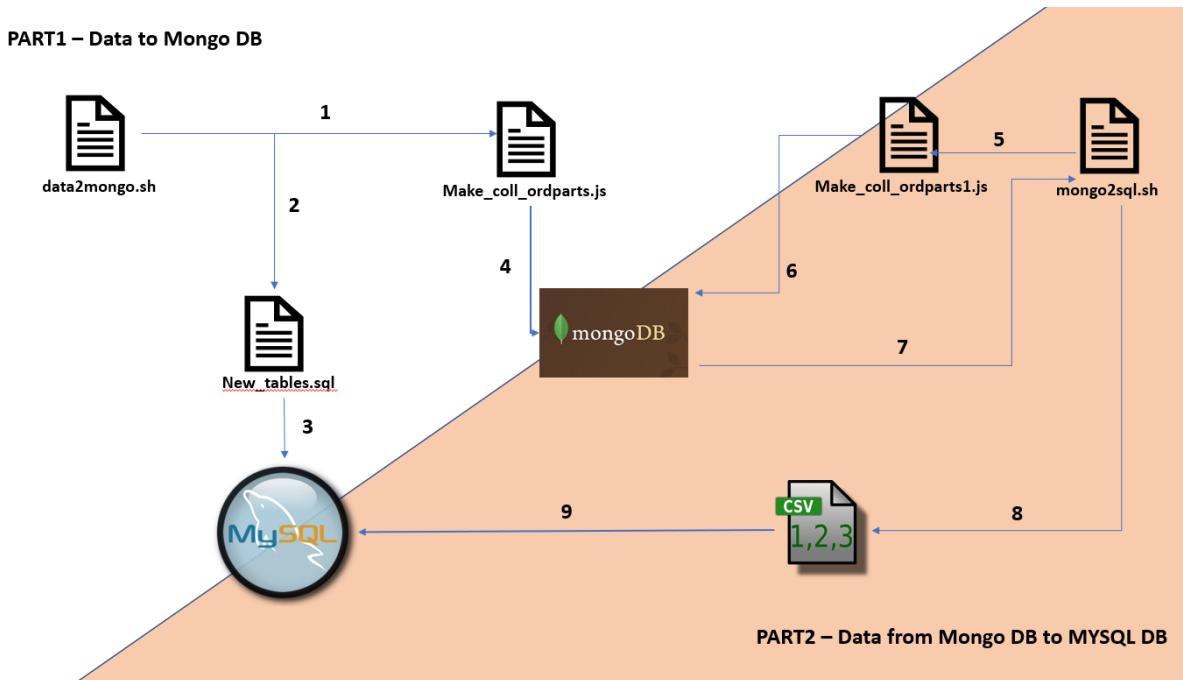


Figure 4 Data Flow Diagram

### Data2mongo.sh:

Executes `new_tables.sql` which contains new schema of our database and loads data from `article.json` file to `ARTICLE_PRE` collection. It also executes `Make_coll_ordparts.js` to create `WRITES` and `AUTHOR` collections in the mongoDB.

```
v_govindan@dev:~/project$ ./data2mongo.sh
Step 1: MySQL:
MySQL Username: v_govindan 1-> MySQL Username
MySQL password:
MySQL Database name: v_govindan 2-> MySQL DB Name
The SQL file to be processed (without the .sql extension): new_tables 3-> Name of the SQL file that you have to run without the .sql extension
Executing the SQL file new_tables.sql
mysql: [Warning] Using a password on the command line interface can be insecure.
Done Execution of file new_tables

Moving to Mongo DB
Step 2: MongoDB:
Mongo Username: v_govindan 1-> Mongo Username
Mongodh password: 2-> Mongo Database Name
Mongo Database name: v_govindan
The JSON file to be processed (without the .json extension): test 3-> Name of the JSON file without the json extension
Entering Mongo DB
-
2018-11-26T12:21:38.516-0400    connected to: localhost
2018-11-26T12:21:38.653-0400    imported 312 documents
MongoDB shell version: 3.2.16
connecting to: v_govindan
true
MongoDB shell version: 3.2.16
connecting to: v_govindan
true
MongoDB shell version: 3.2.16
connecting to: v_govindan
false
MongoDB shell version: 3.2.16
connecting to: v_govindan
true
MongoDB shell version: 3.2.16
connecting to: v_govindan
true
All done!

v_govindan@dev:~/project$
```

Figure 5 Data2mongo script working

Follow the below steps while running:

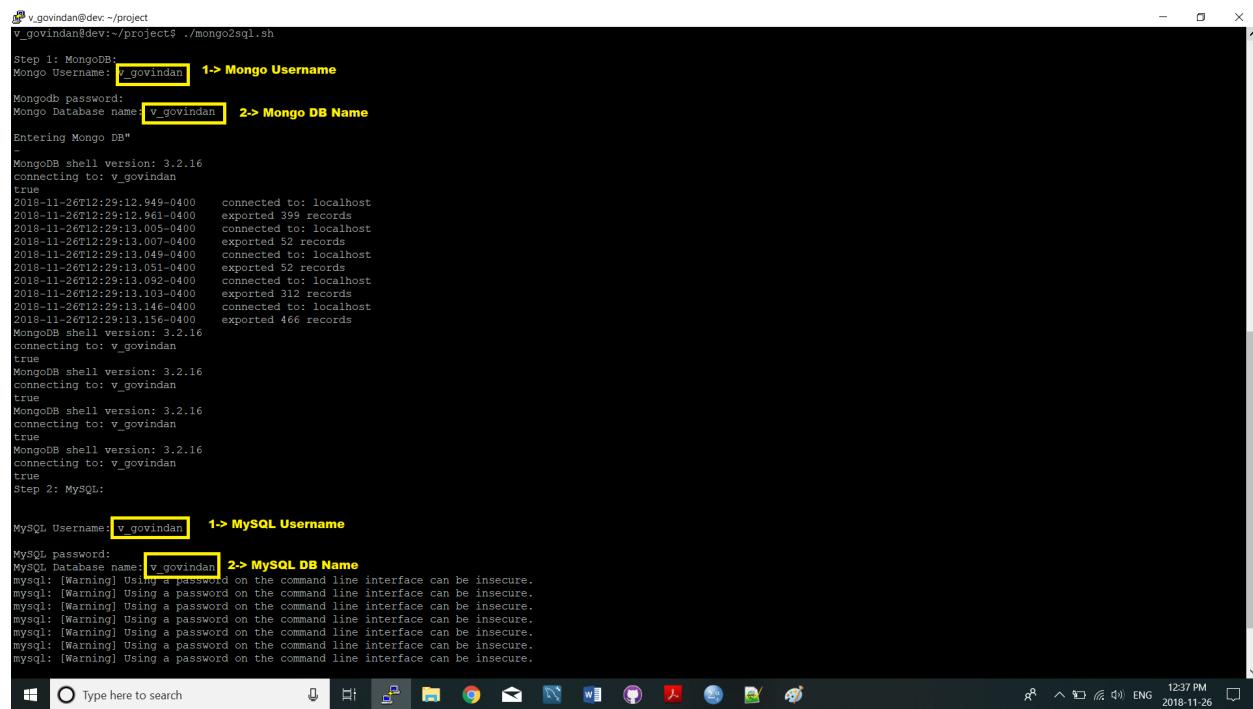
1. In the location ./data2mongo.sh to run the script
2. Enter your mysql credentials when prompted.
3. Enter the name of the database.
4. Specify the sql file to be run without .sql extension in this case “new\_tables”
5. Enter your mongo credentials when prompted.
6. Enter the name of the mongo database required.
7. Specify the json file to be run without .json extension in this case “test”(which is a snippet of the articles.json).

### Make\_coll\_ordparts.js:

Reads data from ARTICLE\_PRE collection and parses it and loads into AUTHOR, WRITES collections.

### mongo2sql.sh:

Executes Make\_coll\_orderparts1.js to create collections according to our schema (ARTICLE, AUTHOR, ITEM, WRITES) in the mongoDB. It also exports the data in new collection to respective csv files. Then, imported data from csv to sql tables and Creates indexes according to search criteria in our php application.



```
v_govindan@dev:~/project$ ./mongo2sql.sh

Step 1: MongoDB:
Mongo Username: v_govindan 1-> Mongo Username
MongoDB password:
Mongo Database name: v_govindan 2-> Mongo DB Name

Entering Mongo DB"
MongoDB shell version: 3.2.16
connecting to: v_govindan
true
2018-11-26T12:29:12.949-0400 connected to: localhost
2018-11-26T12:29:12.961-0400 exported 399 records
2018-11-26T12:29:13.005-0400 connected to: localhost
2018-11-26T12:29:13.007-0400 exported 52 records
2018-11-26T12:29:13.009-0400 connected to: localhost
2018-11-26T12:29:13.054-0400 exported 100 records
2018-11-26T12:29:13.092-0400 connected to: localhost
2018-11-26T12:29:13.103-0400 exported 319 records
2018-11-26T12:29:13.146-0400 connected to: localhost
2018-11-26T12:29:13.156-0400 exported 466 records
MongoDB shell version: 3.2.16
connecting to: v_govindan
true
true
Step 2: MySQL:

MySQL Username: v_govindan 1-> MySQL Username
MySQL password:
MySQL Database name: v_govindan 2-> MySQL DB Name
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.

12:27 PM 2018-11-26
```

Figure 6 mongo2sql script workflow

```
v.govindan@dev:~/project
Mongo Database name: v_govindan
Entering Mongo DB*
MongoDB shell version: 3.2.16
connecting to: v_govindan
true
2018-11-26T12:29:12.949-0400 connected to: localhost
2018-11-26T12:29:12.961-0400 exported 399 records
2018-11-26T12:29:13.005-0400 connected to: localhost
2018-11-26T12:29:13.007-0400 exported 52 records
2018-11-26T12:29:13.049-0400 connected to: localhost
2018-11-26T12:29:13.051-0400 exported 52 records
2018-11-26T12:29:13.092-0400 connected to: localhost
2018-11-26T12:29:13.103-0400 exported 312 records
2018-11-26T12:29:13.146-0400 connected to: localhost
2018-11-26T12:29:13.156-0400 exported 466 records
MongoDB shell version: 3.2.16
connecting to: v_govindan
true
Step 2: MySQL:
MySQL Username: v_govindan
MySQL password:
MySQL Database name: v_govindan
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
All done!
v_govindan@dev:~/project$
```

Figure 7 mongo2sql script workflow Continue

Follow the below steps while running:

1. In the location ./mongo2sql.sh to run the script
2. Enter your mongo credentials when prompted.
3. Enter the name of the mongo database required.
4. Enter your mysql credentials when prompted.
5. Enter the name of the database.

#### Make\_coll\_orderparts1.js:

Reads data from ARTICLE collection and creates ARTI, MAGAZINE, ITEM according to our schema.

## Php Application workflow

### Part 3 – PHP Flow

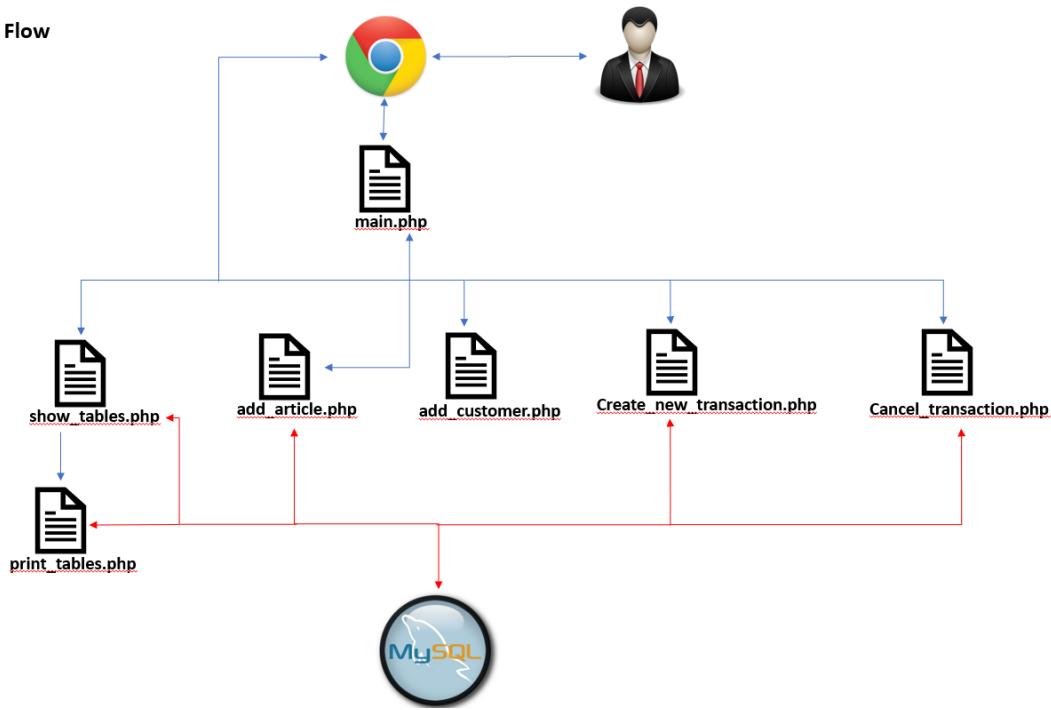


Figure 8 Php Application workflow

main.php:

Main page can be accessed from the below link:

[http://dev.cs.smu.ca/~v\\_govindan/proj/main.php](http://dev.cs.smu.ca/~v_govindan/proj/main.php)

It's our home page and has navigation links to all other pages like Show tables, Add Article, Add Customer, Add Transaction, Cancel Transaction.

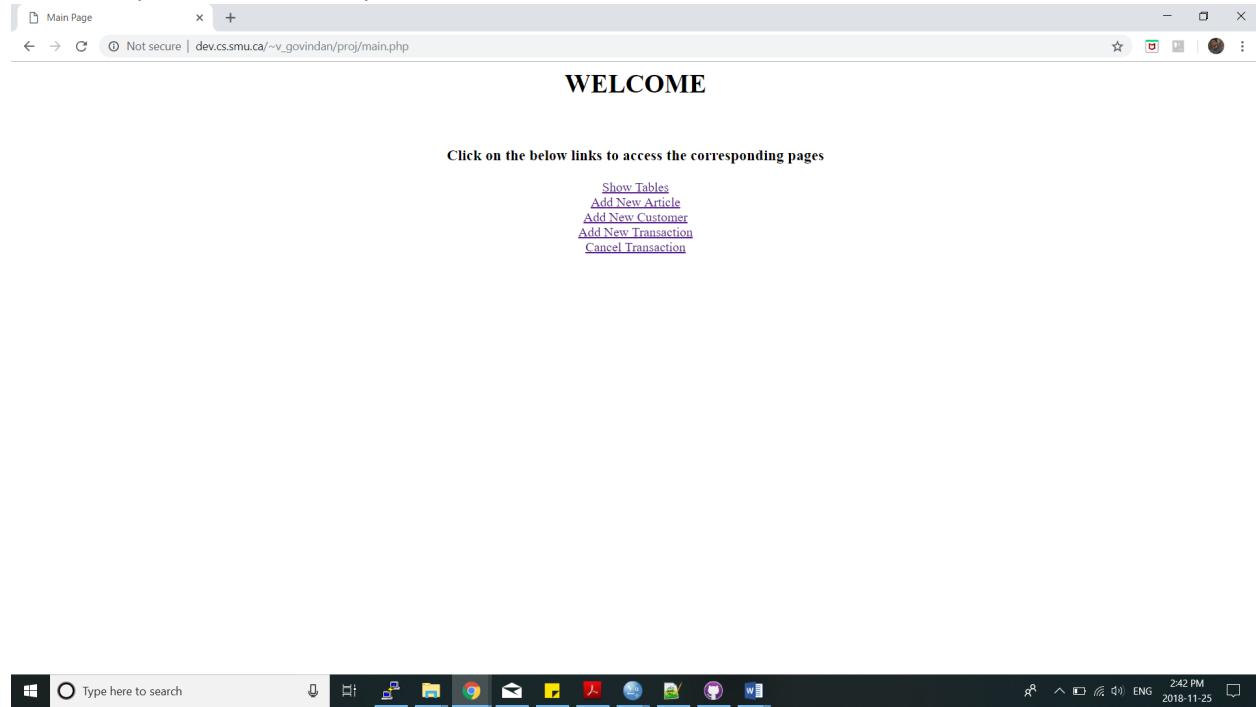


Figure 9 Main page

## add\_article.php:

Article Title, Page Start, Page End, Magazine, Authors are the inputs from the users. On submit it successfully adds new article to the database.

The screenshot shows a web browser window titled "Add Article". The page contains a form with the following fields:

- Title :
- Page Start :
- Page End :
- Magazine :
- Author :

Below the form are two buttons: "submit" and "reset". At the bottom of the page is a link: "back to MAIN menu".



Figure 10 Add Article page

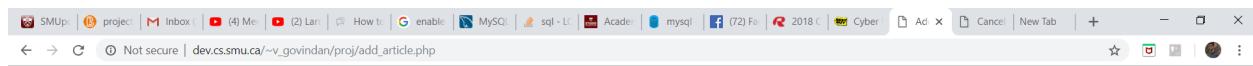
The screenshot shows the same "Add Article" page as Figure 10, but with some fields populated:

- Title :
- Page Start :
- Page End :
- Magazine :
- Author :

Below the form, a dropdown menu for "Magazine" is open, showing a list of options starting with "Acta Inf. Vol.41". A "submit" button is visible at the bottom of the dropdown menu. At the bottom of the page is a link: "back to MAIN menu".



Figure 11 Add Article Magazine selection



## Add Article

please add Articles filling up the below form

Title :	<input type="text" value="ProjectProtoType"/>
Page Start :	<input type="text" value="56"/>
Page End :	<input type="text" value="69"/>
Magazine :	Acta Inf. Vol.41 ▾
Author :	<ul style="list-style-type: none"><li>Nathan Goodman</li><li>Oded Shmueli</li><li>Chua-Huang Huang</li><li>Christian Lengauer</li></ul>

[back to MAIN menu](#)

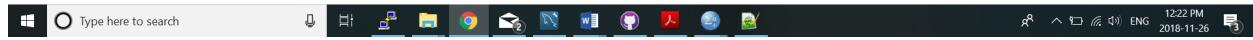
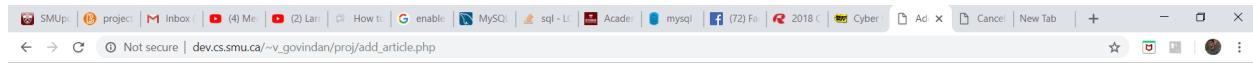


Figure 12 Add Article Author selection



## Add Article

please add Articles filling up the below form

Title :	<input type="text"/>
Page Start :	<input type="text"/>
Page End :	<input type="text"/>
Magazine :	Acta Inf. Vol.41 ▾
Author :	<ul style="list-style-type: none"><li>Nathan Goodman</li><li>Oded Shmueli</li><li>Chua-Huang Huang</li><li>Christian Lengauer</li></ul>

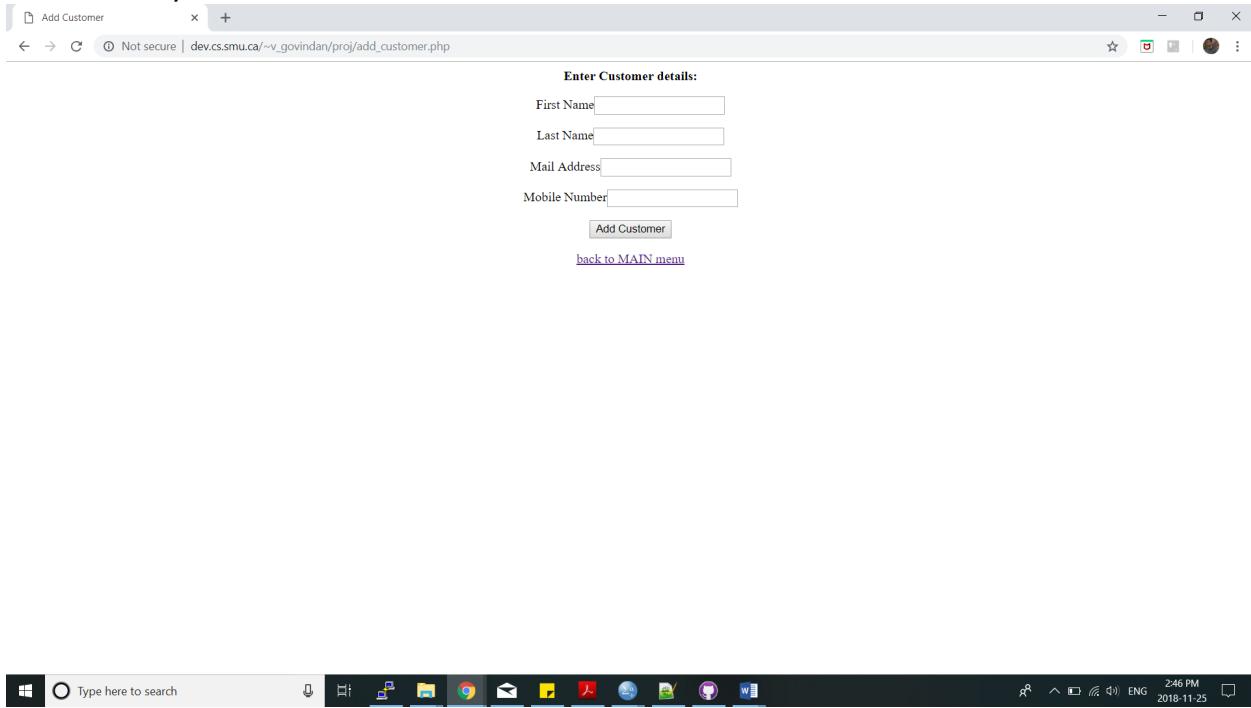
[back to MAIN menu](#)



Figure 13 Add Article Successful

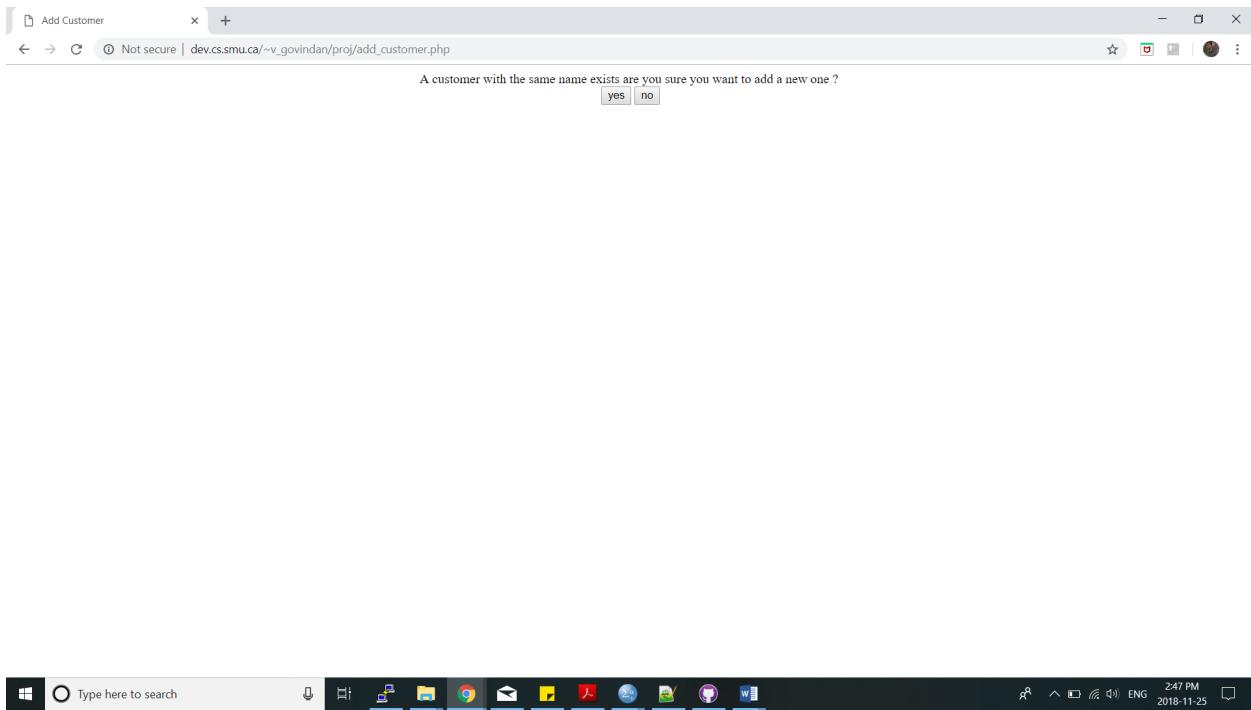
## add\_customer.php:

First Name, Last Name, Mail Address, Mobile Number are the inputs from the users. On submit it successfully adds new Customer to the database.



The screenshot shows a Windows desktop environment. At the top is the taskbar with icons for File Explorer, Task View, File, Print, Mail, Photos, OneDrive, Google Chrome, Task Scheduler, Task Manager, and Word. The system tray shows battery level at 88%, a signal strength icon, ENG, and the date 2018-11-25. Below the taskbar is a window titled "Add Customer". The address bar shows "Not secure | dev.cs.smu.ca/~v\_govindan/proj/add\_customer.php". The main content area is titled "Enter Customer details:" and contains four input fields: "First Name" (with placeholder "First Name"), "Last Name" (with placeholder "Last Name"), "Mail Address" (with placeholder "Mail Address"), and "Mobile Number" (with placeholder "Mobile Number"). Below these fields is a "Add Customer" button and a link "back to MAIN menu".

Figure 14 Add new Customer



The screenshot shows a Windows desktop environment. The taskbar and system tray are identical to Figure 14. The browser window titled "Add Customer" now displays a message: "A customer with the same name exists are you sure you want to add a new one ?" with "yes" and "no" buttons. The address bar remains the same: "Not secure | dev.cs.smu.ca/~v\_govindan/proj/add\_customer.php".

Figure 15 Add new Customer duplicate entry Confirmation

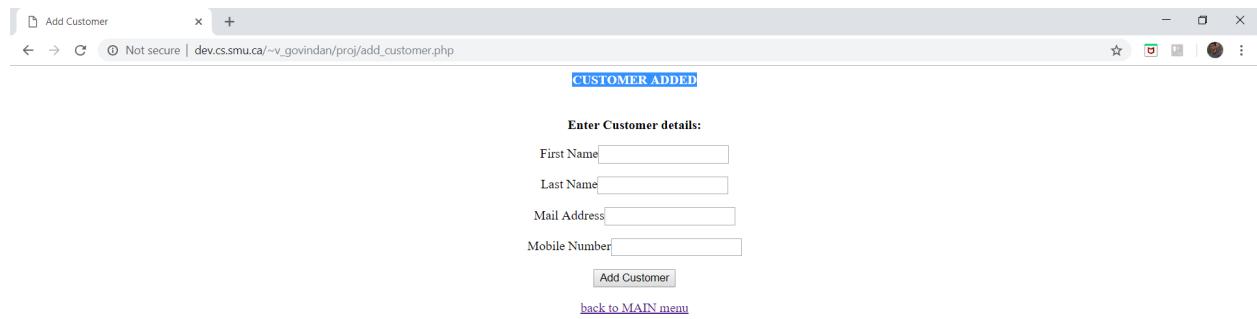


Figure 16 Add new Customer Successful

### Create\_new\_transaction.php:

Customer Number, MagazineID are the inputs from the users. On submit it successfully adds new Transaction and Transaction details to the database. This page also calculates Discount code dynamically while creating new transaction.

New Transaction

Customer Number :  Magazine :

Vinay  
gagan  
Vinay  
Vinay

Acta Inf - vol 41 - price 6  
Acta Inf - vol 37 - price 5  
Acta Inf - vol 27 - price 3

Submit

[click to go to main page](#)

Figure 17 Add new Transaction

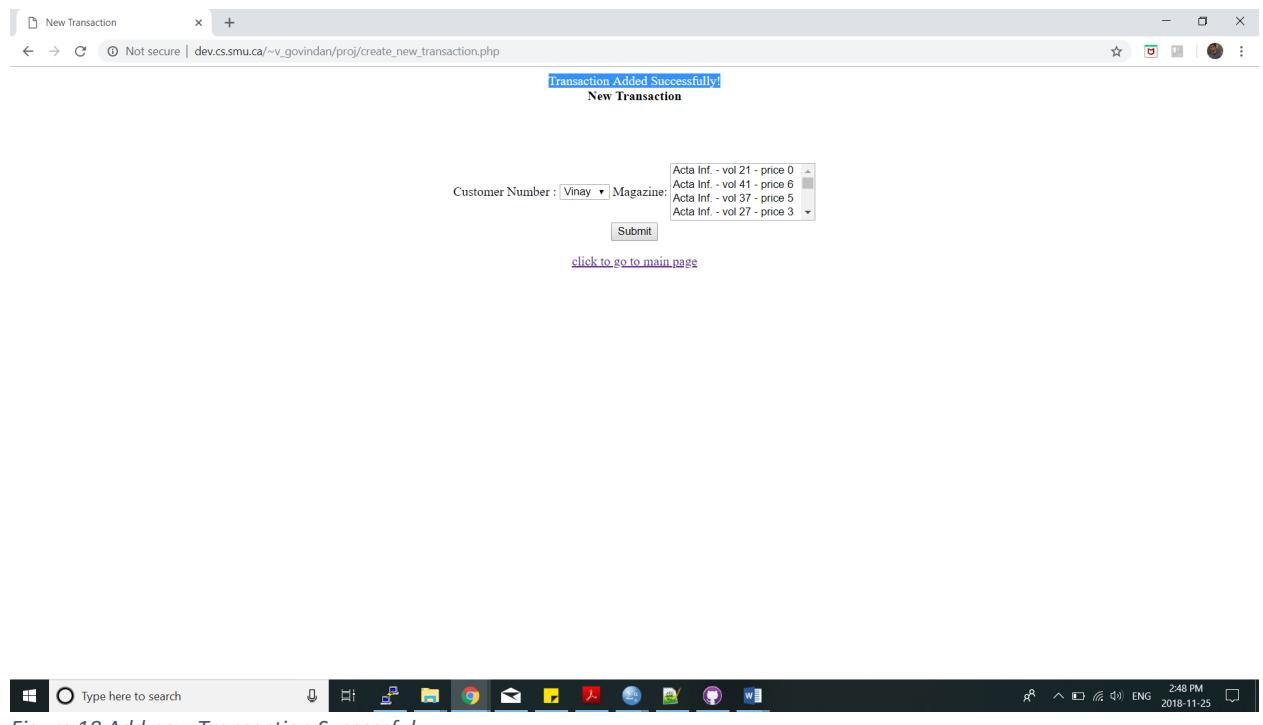


Figure 18 Add new Transaction Successful

### Cancel\_transaction.php:

Transaction Number is the input from the users. On submit it removes entries for given Transaction Number from Transaction and Transaction Details tables. If given transaction is older than 30 days, transaction cannot be cancelled.

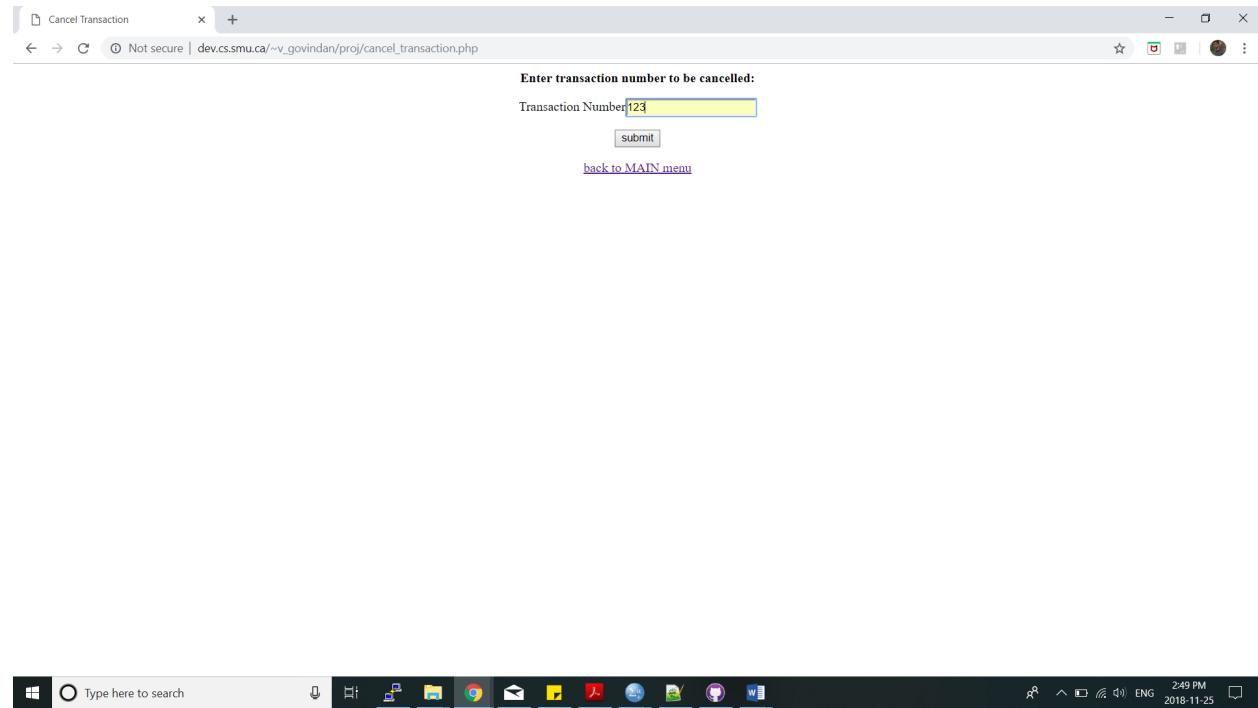


Figure 19 Cancel Transaction

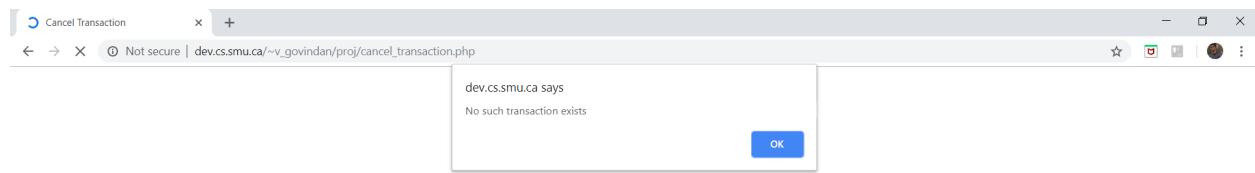


Figure 20 Cancel Transaction Invalid ID Warning

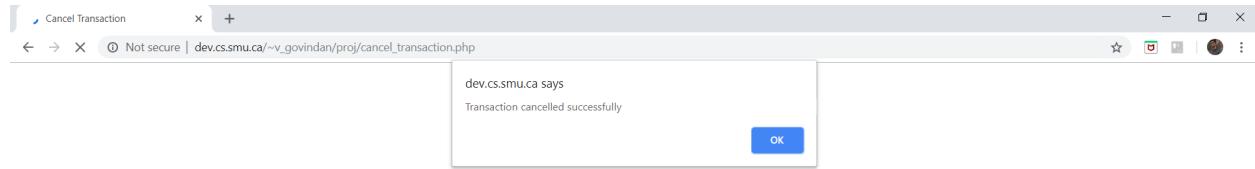


Figure 21 Cancel Transaction Successful

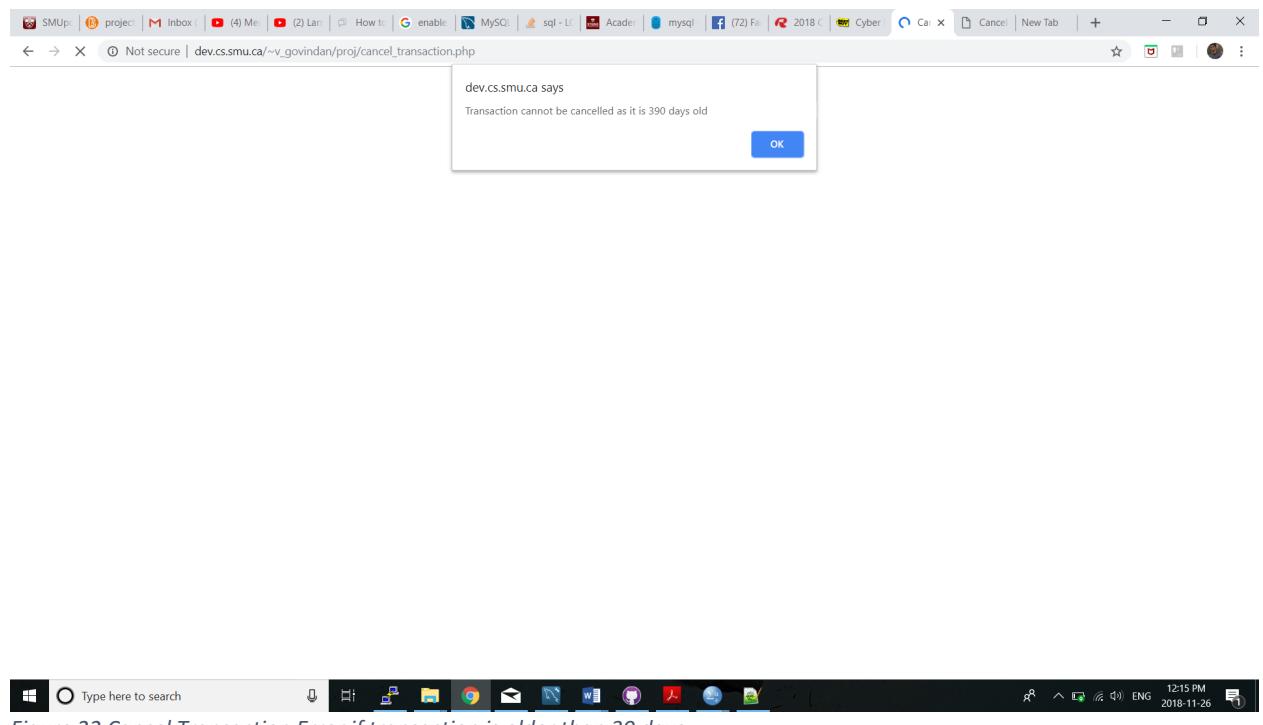


Figure 22 Cancel Transaction Error if transaction is older than 30 days

## show\_tables.php:

Displays all the table names in a database. Search for table names. For the Transaction table total amount displayed is not a field of Transaction table but, is result of join query.



**Enter the name of the table you want to access:**

There are 11 tables

submit

[back to MAIN menu](#)



Figure 23 Show Tables

## print\_tables.php:

Displays data of the selected tables.

The screenshot shows a Microsoft Edge browser window with the URL [http://dev.cs.smu.ca/~v\\_govindan/proj/print\\_table.php](http://dev.cs.smu.ca/~v_govindan/proj/print_table.php). The page title is "print\_table.php". The content of the page includes the following text:  
Successfully connected to server  
Successfully selected database "v\_govindan"  
There are 14 rows in the table  
A table with 14 rows and 5 columns is displayed:

Trxn_Number	Date	CID	Discount_Code	Total_Amount
1	2018-11-25	1	0	23
2	2017-11-01	1	0	0
3	2018-11-25	1	0	7
5	2018-11-25	2	0	0
7	2018-11-26	2	0	32
11	2018-11-26	1	0	16
13	2018-11-26	1	0	162
15	2018-11-26	1	2	8.95
17	2018-11-26	3	0	213
19	2018-11-26	4	0	81
23	2018-11-26	4	1	8.975
27	2018-11-26	6	1	17.95
29	2018-11-26	6	0	23

At the bottom of the page are two links: [back](#) and [back to MAIN menu](#).

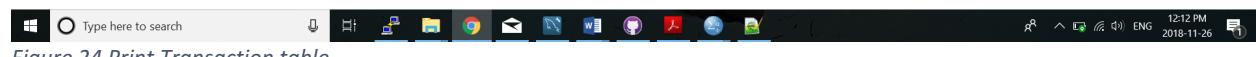


Figure 24 Print Transaction table

The screenshot shows a Microsoft Edge browser window with the URL [http://dev.cs.smu.ca/~v\\_govindan/proj/print\\_table.php](http://dev.cs.smu.ca/~v_govindan/proj/print_table.php). The page title is "print\_table.php". The content of the page includes the following text:  
Successfully connected to server  
Successfully selected database "v\_govindan"  
There are 399 rows in the table  
A table with 399 rows and 4 columns is displayed:

Author_ID	A_Name	A_Last	Email
1	Nathan	Goodman	NathanGoodman@gmail.com
2	Oded	Shmueli	OdedShmueli@gmail.com
3	Chua-Huang	Huang	Chua-HuangHuang@gmail.com
4	Christian	Lengauer	ChristianLengauer@gmail.com
5	Alain	Finkel	AlainFinkel@gmail.com
6	Annie	Choquet	AnnieChoquet@gmail.com
7	Symeon	Bozapalidis	SymeonBozapalidis@gmail.com
8	Zoltán	Fülip0001	ZoltanFilip0001@gmail.com
9	George	Rahonis	GeorgeRahonis@gmail.com
10	Victor	Khomenko	VictorKhomenko@gmail.com
11	Alex	Kondratyev	AlexKondratyev@gmail.com
12	Maciej	Koutny	MaciejKoutny@gmail.com
13	Walter	Vogler	WalterVogler@gmail.com
14	Maria	Calzarossa	MariaCalzarossa@gmail.com
15	M.	Italiani	M.Italiani@gmail.com
16	Giuseppe	Serazzi	GiuseppeSerazzi@gmail.com
17	Christian	Stahl	ChristianStahl@gmail.com
18	Richard	Müller0001	RichardMüller0001@gmail.com
19	T.	C.Hu	T.C.Hu@gmail.com
20	K.	C.Tan	K.C.Tan@gmail.com
21	Carol	Critchlow	CarolCritchlow@gmail.com
22	Prakash	Panangaden	PrakashPanangaden@gmail.com
23	Sergei	Gorlatch	SergeiGorlatch@gmail.com
24	Yijie	Han	YijieHan@gmail.com
25	Yoshihide	Igarashi	YoshihideIgarashi@gmail.com
26	X.	J.Chen	X.J.Chen@gmail.com

At the bottom of the page are two links: [back](#) and [back to MAIN menu](#).

Figure 25 Print Author Table

print\_table.php

Not secure | dev.cs.smu.ca/~v\_govindan/pro/print\_table.php

back back to MAIN menu

ID	Author Name	Email	Author Name
372	Gigliola Vaglini	GigliolaVaglini@gmail.com	
373	M. Kempf	M.Kempf@gmail.com	
374	Rudolf Bayer	RudolfBayer@gmail.com	
375	Ulrich Gintzer	UlrichGintzer@gmail.com	
376	Elias Soisalon-Soininen	EliasSoisalon-Soininen@gmail.com	
377	Yennnu Huang	YennnuHuang@gmail.com	
378	Pankaj Jalote	PankajJalote@gmail.com	
379	Yiwei Jiang	YiweiJiang@gmail.com	
380	Yong He	YongHe@gmail.com	
381	Foto N.Afrati	FotoN.Afrati@gmail.com	
382	Rada Chirkova	RadaChirkova@gmail.com	
383	Manolis Gergatsoulis	ManolisGergatsoulis@gmail.com	
384	Vassia Pavlaki	VassiaPavlaki@gmail.com	
385	Dirk Hauschildt	DirkHauschildt@gmail.com	
386	Matthias Jantzen	MatthiasJantzen@gmail.com	
387	Arturo Carpi	ArturoCarpi@gmail.com	
388	Flavio D'Alessandro	FlavioD'Alessandro@gmail.com	
389	Gadi Taubenfeld	GadiTaubenfeld@gmail.com	
390	Shlomo Moran	ShlomoMoran@gmail.com	
391	Hans-Dieter Ehrich	Hans-DieterEhrich@gmail.com	
392	Carlos Caleiro	CarlosCaleiro@gmail.com	
393	Mike Paterson	MikePaterson@gmail.com	
394	Ingo Wegener	IngoWegener@gmail.com	
395	Shmuel Sagiv	ShmuelSagiv@gmail.com	
396	Michael Rodeh	MichaelRodeh@gmail.com	
397	Gaston H.Gonnet	GastonH.Gonnet@gmail.com	
398	Lawrence D.Rogers	LawrenceD.Rogers@gmail.com	
399	J. AlanGeorge	J.AlanGeorge@gmail.com	

Figure 26 Print Author Table Continue

## References:

1. [w3schools](#)
2. [stackoverflow](#)
3. For PHP files tutorial 6 material.
4. Computing and Data Analytics 5540 -- Team Project.pdf
5. JavaScript, Shell script have referenced tutorial 8 material