Simon Fraser University
CMPT 354
Summer 2022

Group Project - Implementation of a Relational Database

Project Title:	Swift Shipping Ltd.
Project Milestone:	Milestone 4(a): Implementation

#	Student Name	Student Number	Email Address
1	Peeyush Sharma	301540329	psa139@sfu.ca
2	Ketan Dhingra	301439331	kda59@sfu.ca
3	Harveer Singh Virk	301451685	hsv@sfu.ca

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above.

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by Simon Fraser University

Swift Logistics

About Project

Our project includes a shipping company called Swift logistics, operating its services within Canada. The company offers standard and expedited parcel shipping throughout various provinces. Our group developed the company's database and the website using MySql, PHP, HTML, JavaScript, and CSS. The Database handles customer details, parcel information including tracking updates, and company information including its employees, vehicles, and locations.

Website Functionality

- Customers can:
    - sign-up by making a new account on the website.
    - Log in to their existing account.
    - Customer can update his personal information.
    - Request a parcel shipment within Canada.
    - Can put the delivery address of the parcel.
    - Can input parcel details such as height, weight, and length
    - Track their parcel online.

- Employees can:
    - Log in to their existing account provided by IT/backend.
    - All employees can see the delivery status of the shipment
    - Combine different tables to get information
    - Filter out shipments with specific dimensions
    - Employee can see the company’s reports-
        - Average order price by province
        - Order with an amount greater than province average order price
        - Get the lists of customers filtered by order types such as standard or expedited
- Manager can:
    - Manager can do everything that an employee can do.
    - See employees' information.
    - Delete Orders from the database.

Placing an Order

1. The customer has to enter the address where they wish to send the parcel.
2. The customer then enters the weight and dimensions of the parcel.
3. A couple of shipping options (i.e. standard or expedited) are then provided to the customers.
4. Lastly, the customer completes the payment, by placing the order.

Changes in Schema

1. Customer Table
    a. password domain changed from char(30) to char(100)
    Reason: In order to hash the actual password to a system generated one

    b. phone data type changed from int(14) to char(14)
    Reason: To store the phone number in the correct format eg: (123) 456-7890

2. Shipment Table
    a. Ref_ID data type changed from int(30) to Ref_ID char(30)
    Reason: To use alphabets in Ref_ID

    b. Order_ID data type changed from int(30) to Order_ID char(30)
    Reason:To use alphabets in Order_ID

    c. Field added first_name char(20) NOT NULL
    Reason: To store the parcel recipient's first name

    d. Field added last_name char(20)
    Reason: To store the parcel recipient's last name

    e. Order_ID foreign key from Order table changed from ON DELETE NO ACTION to ON DELETE CASCADE
    Reason: Delete function implemented. Now when order is deleted, the shipments are also deleted.

3. Delivery status Table
    a. Ref_ID data type changed from int(30) to Ref_ID char(30)
    Reason: To use alphabets in Ref_ID

    b. Ref_ID foreign key from Shipment table changed from ON DELETE NO ACTION to ON DELETE CASCADE
    Reason: Delete function implemented. Now when order is deleted, the delivery status is also deleted.

4. Employee table
    a. Password domain changed from char(30) to char(100)
    Reason: In order to hash the actual password to a system generated one

    b. Phone data type changed from int(14) to char(14) and UNIQUE constraint added
    Reason: To store the phone number in correct format eg: (123) 456-7890 and to make sure every employee has a unique phone number.

Queries and Snapshots

1. Selection Query

```
$sql="Select * FROM ".$_POST["Table"]." Where
".$_POST["column"].$_POST["operator"]."?"
```

The user is given the option to filter the shipments based on its dimensions or type of shipment. For eg, actual data in the shipment table is:

Ref_ID	Order_ID	Emp_ID	First_Name	Last_Name	Weight	Length	Width	Height	Price	Type	Street_Address	Postal_Code
101	1	1001	Jack	Nelson	5.36	4.5	4.4	3.6	19.85	Express	2925 Virtual Way	V5M-4X5
102	1	1001	Jack	Nelson	8.5	3.3	5.7	4.3	20	Express	2925 Virtual Way	V5M-4X5
1123	1	1001	Jack	Nelson	5.36	4.5	4.4	3.6	19.85	Express	2925 Virtual Way	V5M-4X5
201	2	1354	Anile	Bhardwaj	2.3	6.3	5.12	9.3	25.1	Regular	10 Kincora Heights NW	T3R-1N3
301	3	1532	Rajesh	Kakkar	3.7	3.12	4.8	7.3	20	Regular	201 - 7500 120 St	V3W-3N1
401	4	1354	Betty	Baker	4	6.8	4.1	5.9	21.15	Regular	64 Hidden Spring Close NW	T3A-5K2
501	5	3921	Himesh	Sharma	9.1	15.3	15.3	39	22.5	Regular	81 Ravine Edge Dr	L4E-4J3
SID62e451757c06d3.48954001	OID62e4517576b4d5.07381111	1		*NULL*	12	32	34	23	1231.13	Standard	9823 bsdkdsvdk	G3R-3R3

A query stating: “Get from “Shipment” table where height > 5” is applied as shown below in GUI:

Resulting Table:

Shipments

Ref_ID	Height	Width	Weight	Length	Order_ID
201	9.3	5.12	2.3	6.3	2
301	7.3	4.8	3.7	3.12	3
401	5.9	4.1	4	6.8	4
501	39	15.3	9.1	15.3	5
SID62e451757c06d3.48954001	23	34	12	32	OID62e4517576b4d5.07381111

2. Projection Query

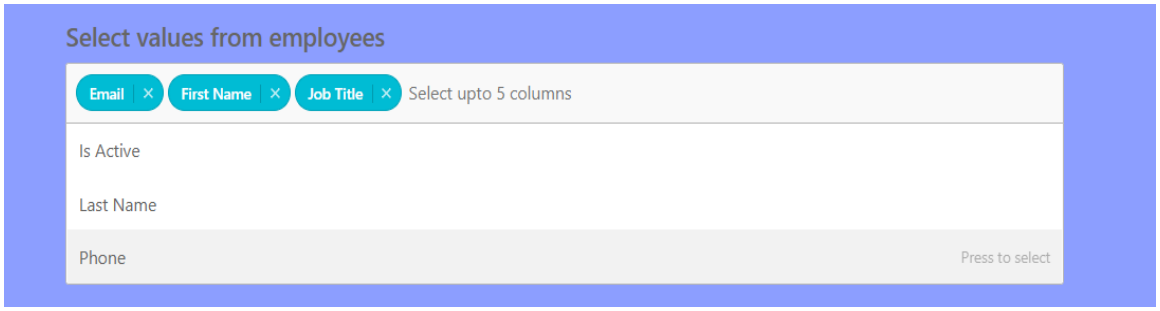
```
$sql="Select emp_id";
                foreach($columns as $col)
        {
            $sql=$sql.' ,'.$col;
        }
        $sql=$sql." from employee"
```

This dynamic query allows the company managers to access their employees' information by selecting specific columns they want.

Actual columns in the Employee table:

Emp_ID	Location_ID	First_name	Last_Name	Email	Password	Job_Title	Is_Active	Phone
1	NULL	Online	User	OnlineUser@swiftlogistics.com	12345	NULL	1	2345678765
1001	36	Mark	Robert	mrobert12@hotmail.com	799ILoveThisCompany@@	Driver	1	(604) 897-1254
1298	12	Usman	Ali	usmanahmed52@gmail.com	AlialiUsman1234!	Driver	1	(413) 568-1200
1354	45	Jack	Neil	NeilJack3@gmail.com	IamJack34!	Driver	1	(236) 496-1486
1532	22	Harpreet	Sidhu	hpsingh@yahoo.com	CalgaryCanada2019$$	Driver	1	(204) 396-2890
3921	45	Rick	White	whiter604@gmail.com	6042584563	Driver	1	(604) 258-4563
9632	45	Olivia	Watson	oliviawatson678@gmail.com	OliviaCool3434	Warehouse Worker	0	(204) 696-1598
9685	12	Steven	Garden	stgr@hotmail.com	SwiftPassword4321	Manager	1	(672) 458-4547

The manager can select specific columns they want to see through UI:



The resulting table will only have those values:

Employees

Email	first_name	job_title
OnlineUser@swiftlogistics.com	Online	
mrobert12@hotmail.com	Mark	Driver
usmanahmed52@gmail.com	Usman	Driver
NeilJack3@gmail.com	Jack	Driver
hpsingh@yahoo.com	Harpreet	Driver
whiter604@gmail.com	Rick	Driver
oliviawatson678@gmail.com	Olivia	Warehouse Worker
stgr@hotmail.com	Steven	Manager

3. Join Query

```
if($_POST["Tablej"]=="Shipment")
      {
        $columns = ["Ref_ID","Stage_Id","Status"];
        $sql="Select Stage_Id,s.Ref_ID,Status ";
      }
      else if($_POST["Tablej"]=="Employee")
      {
        $columns = ["first_name","last_name","Address"];
        $sql="Select first_name,last_name,Address ";
      }
      else if($_POST["Tablej"]=="DeliveryPersonnel")
      {
        $columns = ["first_name","last_name","phone"];
        $sql="Select first_name,last_name,phone ";
      }

        $sql=$sql."FROM ".$_POST["Tablej"]." s";
        if($_POST["columnj"]=="different delivery stages")
        $sql=$sql." INNER JOIN DeliveryStatus k ON s.ref_id =
k.ref_id order by ref_id,stage_id";
        else if($_POST["columnj"]=="working address location")
        $sql=$sql." INNER JOIN Location k ON s.location_id =
k.location_id";
        else if($_POST["columnj"]=="Phone numbers")
        $sql=$sql." INNER JOIN employee k ON s.emp_id = k.emp_id";
```

This dynamic query can display:

a. Shipments with their delivery stages by joining the shipment table and delivery status tables.
b. Employee names with their work location address by joining the employee and location tables.
c. Delivery driver's name and phone number by joining the employee and delivery personnel tables.

The query joins the table according to the customer's input in the UI. For e.g., the following are the actual employee and location tables in the database:

Employee

Emp_ID	Location_ID	First_name	Last_Name	Email	Password	Job_Title	Is_Active	Phone
1	*NULL*	Online	User	OnlineUser@swiftlogistics.com	12345	*NULL*	1	2345678765
1001	36	Mark	Robert	mrobert12@hotmail.com	799ILoveThisCompany@@	Driver	1	(604) 897-1254
1298	12	Usman	Ali	usmanahmed52@gmail.com	AlialiUsman1234!	Driver	1	(413) 568-1200
1354	45	Jack	Neil	NeilJack3@gmail.com	IamJack34!	Driver	1	(236) 496-1486
1532	22	Harpreet	Sidhu	hpsingh@yahoo.com	CalgaryCanada2019$$	Driver	1	(204) 396-2890
3921	45	Rick	White	whiter604@gmail.com	6042584563	Driver	1	(604) 258-4563
9632	45	Olivia	Watson	oliviawatson678@gmail.com	OliviaCool3434	Warehouse Worker	0	(204) 696-1598
9685	12	Steven	Garden	stgr@hotmail.com	SwiftPassword4321	Manager	1	(672) 458-4547

Location

Location_ID	Name	Address
12	Calgary Warehouse	23 Kincora Drive NE, Calgary, AB, T2M-9B5
22	Montreal Loc	325 Old Road, Montreal, QB, H1L-2K1
23	Winnipeg Loc	456 Roxwood Place, Winnipeg, MB, R2P-1K9
36	Toronto Warehouse	5698 Tims Street, Toronto, ON, M4C-1B2
45	Vancouver Warehouse	2151 Canada Drive, Vancouver, BC, V5L-0A1

The user can select the tables to join from UI:

Result:

first_name	last_name	Address
Mark	Robert	5698 Tims Street, Toronto, ON, M4C-1B2
Usman	Ali	23 Kincora Drive NE, Calgary, AB, T2M-9B5
Jack	Neil	2151 Canada Drive, Vancouver, BC, V5L-0A1
Harpreet	Sidhu	325 Old Road, Montreal, QB, H1L-2K1
Rick	White	2151 Canada Drive, Vancouver, BC, V5L-0A1
Olivia	Watson	2151 Canada Drive, Vancouver, BC, V5L-0A1
Steven	Garden	23 Kincora Drive NE, Calgary, AB, T2M-9B5

4. Aggregation query

```
SELECT ca.Province,AVG(p.Amount) As Average FROM `order` o
INNER JOIN payment p
ON o.Payment_ID=p.Payment_ID
INNER JOIN  customer c
ON c.Email=o.Email
INNER JOIN customer_address ca
ON ca.Postal_Code=c.Postal_Code
GROUP BY ca.Province
```

The query displays the Average order price Province. It joins the payment, order, and customer_address table and finds the order average grouped by province.
UI Snapshot:

Avg. Order price by Province (Aggregation)

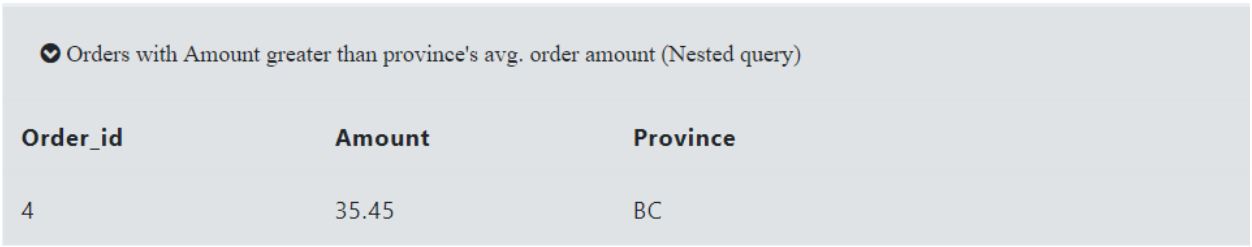
Province	Average
AB	563.22998046875
BC	25.983333587646
MB	25.10000038147
ON	39.849998474121

5. Nested aggregation query with group by:

```
SELECT  order_id,p.Amount,ca.Province from `order` o
INNER JOIN customer c
ON c.Email=o.Email
INNER JOIN customer_address ca
ON ca.Postal_Code=c.Postal_Code
INNER JOIN payment p
ON o.Payment_ID=p.Payment_ID
INNER JOIN
(SELECT ca.Province,AVG(p.Amount) As Average FROM `order` o
INNER JOIN payment p
ON o.Payment_ID=p.Payment_ID
INNER JOIN  customer c
ON c.Email=o.Email
INNER JOIN customer_address ca
ON ca.Postal_Code=c.Postal_Code
 GROUP BY ca.Province) nested
ON ca.Province=nested.province
 where p.Amount > average
```

This query displays the list of orders with the order amount greater than the province's average amount.

UI Snapshot:



Orders with Amount greater than province's avg. order amount (Nested query)

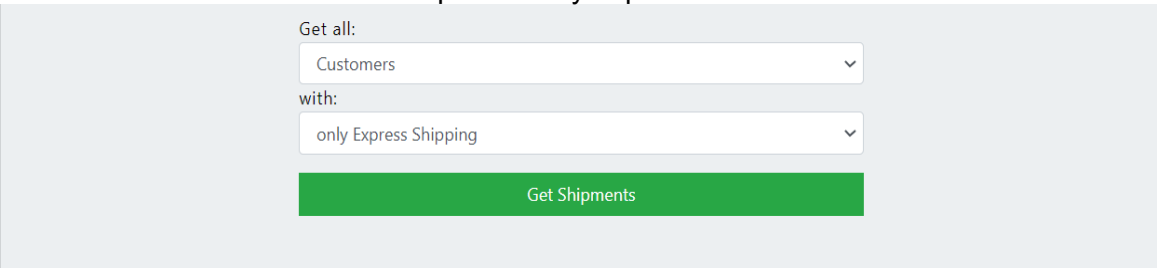
Order_id	Amount	Province
4	35.45	BC

6. Division Query

```
if($_POST["Table"]=="Customers")
{
$columns = ["first_name","last_name","phone"];
$sql="Select first_name,last_name,phone FROM CUSTOMER c";
}
if($_POST["column"]=="No Orders")
$sql=$sql." WHERE NOT EXISTS (Select o.email from `Order` o Where
o.email=c.email)";
else
$sql=$sql." WHERE EXISTS (Select o.email,Order_ID from `Order` o
WHERE  o.email=c.email AND NOT EXISTS
(Select type FROM shipment s where type='Regular' AND
o.order_id=s.order_id ))";
```

This query allows the user to see only specific types of customers based on their type of orders placed. For eg:

a. The customers who have placed only express orders:



Shipments

first_name	last_name	phone
Ketan	Dhingra	(651) 623-5681
Raj	Sharma	(647) 897-4545

Additional Queries

Employee Login Query

```
        // Prepare a select statement
        $sql = "SELECT email, password,er.Is_Manager FROM employee e
        INNER JOIN employee_role er ON e.Job_Title=er.Job_Title
        WHERE email = ?";
```

This query is used on the employee login page to select the correct combination of email and password from the existing employee table.

Customer Login Query

```
// Validate credentials
   if(empty($email_err) && empty($password_err)){
        // Prepare a select statement
        $sql = "SELECT email, password FROM customer WHERE email = ?";
```

This query is used on the customer login page to select the correct combination of email and password from the existing customer table.

Customer Sign-up Query

```
$sql = "SELECT email FROM customer WHERE email = ?";
$sql = "SELECT postal_code FROM customer_address WHERE
postal_code = ?";
$sql = "INSERT INTO customer_address (postal_code, city,
province, country) VALUES (?, ?, ?, ?)";
 $sql = "INSERT INTO customer (email, first_name, last_name,
password, phone, street_address, postal_code) VALUES (?, ?, ?,
?, ?, ?, ?)";
```

This query is implemented on the customer signup page to collect new customer's data and enter in the respective table.

Customer Profile Query

```
sql = "SELECT
c.first_name,c.last_name,c.email,c.street_Address,ca.City,ca.P
rovince,c.Postal_Code,phone FROM customer c
INNER JOIN customer_address ca ON c.Postal_Code=ca.Postal_Code
WHERE c.email  = ?";
```

This query is used to display the customer's information on the profile page from different tables in the database.

Customer Profile Update Query

```
if($_SERVER["REQUEST_METHOD"] == "POST"){
                    $sql="Update customer SET";
                     // Check if email is empty
                     $count=0;
                    if($fname!=trim($_POST["fname"]))
                    {
                        $sql=$sql."
first_name='".trim($_POST["fname"])."'";
                        $fname=trim($_POST["fname"]);
                        $count++;
                    }
                    if($lname!=trim($_POST["lname"]))
                    {
                        if($count!=0)
                        $sql=$sql.",";
                        $sql=$sql."
last_name='".trim($_POST["lname"])."'";
                        $lname=trim($_POST["lname"]);
                        $count++;
                    }

                    if($phone!=trim($_POST["phone"]))
                    {
                        if($count!=0)
                        $sql=$sql.",";
                        $sql=$sql."
phone='".trim($_POST["phone"])."'";
                        $phone=trim($_POST["phone"]);
                        $count++;
                    }
                    if($add!=trim($_POST["add"]))
                    {
                        if($count!=0)
                        $sql=$sql.",";

                        $sql=$sql."
street_Address='".trim($_POST["add"])."'";
                        $add=trim($_POST["add"]);
                        $count++;
                    }
```

```
                    if($pcode!=trim($_POST["pcode"]))
                    {
                        if($count!=0)
                        $sql=$sql.",";
                        $sql=$sql."
postal_code='".trim($_POST["pcode"])."'";
                        $pcode=trim($_POST["pcode"]);
                        $count++;
                    }
                    $sql=$sql." Where email=?";
```

This query is used to update the customer's information in the database if requested through UI.

Placing an Order

```
 $sql = "INSERT INTO Payment(payment_id, Amount, Type) VALUES
(?, ?, ?)";
$sql = "INSERT INTO `Order`(order_id, email, payment_id,
qty,order_date) VALUES (?, ?, ?, ?,?)";
$sql = "SELECT postal_code FROM shipment_address WHERE
postal_code = ?";
$sql = "INSERT INTO shipment_address (postal_code, city,
province, country) VALUES (?, ?, ?, ?)";
$sql = "INSERT INTO shipment(ref_id, order_id, emp_id,
weight,length,width,height,price,type,street_address,postal_co
de) VALUES (?, ?, ?, ?,?,?, ?, ?, ?,?,?)";
```

These queries are used to collect information of the new order - placed by the customer using UI - and to add it to the respective tables.

Track Order Query

```
$sql = "SELECT
s.Ref_ID,s.Order_ID,c.Street_Address,ca.City,ca.Province,c.Pos
tal_Code,s.Street_Address,sa.City,sa.Province,s.Postal_Code,c.
First_Name,c.Last_Name FROM shipment s
INNER JOIN `order` o ON s.Order_ID=o.Order_ID
INNER JOIN customer c ON c.Email=o.Email
INNER JOIN customer_address ca ON c.Postal_Code=ca.Postal_Code
INNER JOIN shipment_address sa ON s.Postal_Code=sa.Postal_Code
WHERE Ref_ID  = ?";
$sql="Select s.ref_id,Stage_Id,Location_id,Status,Date FROM
shipment s INNER JOIN DeliveryStatus k ON s.ref_id = k.ref_id
WHERE s.ref_id=? order by ref_id,stage_id ";
```

This query is used to print the tracking information of the customer's order on the UI.

Order Deletion Query

```
$sql = "SELECT * FROM `order` WHERE order_id = ?";
$sql="DELETE FROM `Order` Where order_id=? ";
```

This query is used to delete the order records from the database. The customer enters the order ID in the UI and if the order exists with the same order ID, the order is deleted. Customer is notified with a confirmation message or error message if the order ID does not match with any order.

BONUS ATTEMPT

To gain bonus points:

1. Our team added a trigger in the sql query, which automatically inserts the delivery status of a newly placed order as “Order is Placed” in the delivery status table.
2. Worked hard on making an enhanced multi-functioning UI.