# CPSC 304 Project Cover Page

Milestone #: 4

Date: April 1, 2021

Group Number: 45

| Name | Student Number | CS Alias (Userid) | Preferred E-mail Address |
|---|---|---|---|
| Nazish Tazeem | 45548682 | h5a2b | nazish@student.ubc.ca |
| Austin Lee | 82785106 | h2s8 | wjaustinlee@gmail.com |
| Jeffrey Kwok | 32713125 | g6m8 | kwokjeff@outlook.com |

**Repository link**:
https://github.students.cs.ubc.ca/CPSC304-2020W-T2/CPSC304Project_project_g6m8_h2s8_h5a2b.git

**SQL script**: See <project.sql> in the top level of the repository.

**README**: See <README.txt> in the top level of the repository.

## Project Description

### a. A short description of the final project, and what it accomplishes.

The domain of our project is a shopping rewards program like Drop Rewards, where members earn points by shopping through our rewards program and taking surveys. Points can then be redeemed for various rewards.

Our app simulates business analytics and administrator portal to be used by company staff. The app serves two purposes:

1) perform administrative actions such as adding new accounts, modifying member data, and deleting rewards.
2) organize and display member-generated data for analysis, such as the kinds of transactions made by members and the rewards redeemed by accounts.

The first purpose facilitates the day-to-day operations of the program. Users are able to add new accounts for new signups of the product, and delete rewards for products that have sold out or where merchants no longer wish to participate as a redemption partner. To extend our functionality, our intent in the future is to allow accounts to be removed, rewards to be added, and implement other addition and removal features to facilitate basic usage of the product.

The second purpose allows user data to be collected, analyzed, and sold to other businesses (e.g. for marketing purposes), which is the reward program's primary source of revenue. Our app enables staff to use the company's databases without explicit knowledge of databases or SQL commands through an easy-to-navigate UI.

Due to time constraints, our project is of small scale, yet it illustrates the value an app like ours can provide to a rewards program company that wants to transform its members' data into informed business decisions.

The queries we created in our app are meaningful and provide a breadth of valuable information that can be used to make informed business decisions. For example, our nested aggregation query displays the average balance of non-zero points accounts located within the US and Canada. The intent of this query is to identify accounts from our targeted regions (in this case, we assume that our most profitable customers are from the US and Canada) while excluding inactive accounts. This gives us a snapshot of point balance differences between accounts in the two regions. In the future, we could create further queries which break down this information by province and state, giving us more granular information on account balances per region.

Features we would like to implement in the future include: UI for every table in our schema (e.g., survey manager, merchant manager), UI controls to create, delete, and edit all account/member-related fields, more powerful and versatile analytics tools for our Transactions page, and input sanitation to guard against malicious attacks.

b. A description of how your final schema differed from the schema you turned in.
   i. If the final schema differed, explain why? Not that turning in a final schema that's different from what you planned is fine, we just want to know what changed and why.

Two changes were made to the final schema:

1. The IsA relationship between Member, Primary Member, and Supplementary Member has been replaced with a single Member entity. The SupplementaryMember table from milestone 2 has been removed in our final SQL script.

In our original schema, we introduced an IsA relationship with a Member as supertype, and Primary Member and Supplementary Member as subtypes. Due to our particular schema setup, our PrimaryMember and Member were merged into one table while Supplementary Member had a primary key (memberID, accountID, primaryMemberID) referencing Member (**memberID,** accountID, **memberID**). However, Oracle/SQL would not allow having multiple columns in the (foreign) primary key reference the same parent attribute (memberID) twice.

We consulted with both a TA, Michael, during office hours as well as our project TA, Jeremy. After reviewing the situation, Jeremy allowed us to remove the IsA relation and continue with just a single Member entity instead for Milestone 4.

2. Added dateTime to Redeems' composite primary key

In our original schema, the primary key for the Redeems entity was (rewardID, accountID, memberID). In our final schema, we added dateTime to Redeems primary key (rewardID, accountID, memberID, dateTime). We did this because our original primary key allowed each member to redeem a particular reward at most once. Our original intent was to allow members to redeem rewards multiple times, to mirror how a rewards program would work in real life.

We already had dateTime as an attribute in Redeems for M2, but it was not part of the primary key. This change fixes Redeems so that we can record multiple redemptions of the same reward for each account, as originally intended.

## Queries

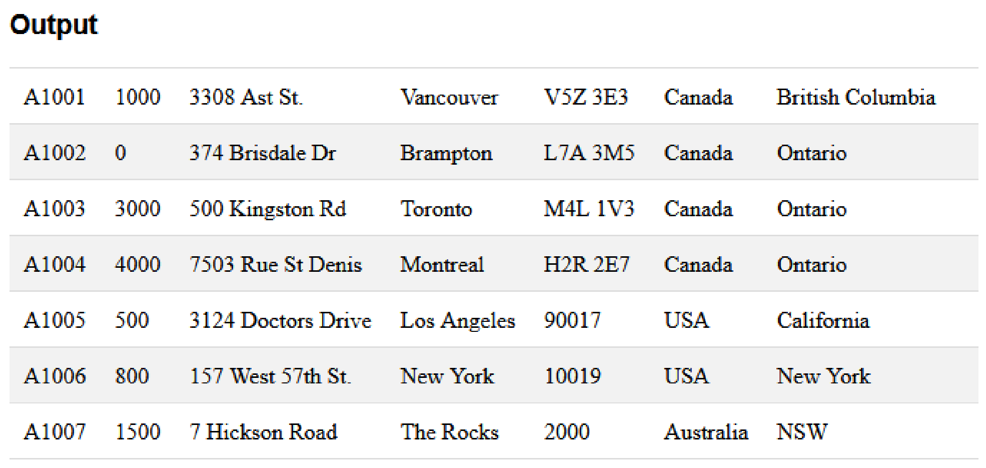
### Insert: Insert a new account

**Front end implementation**: account.php
**Back-end**: handleInsertRequest() in account-controller.php

**Sample Input:**
*AccountID*: A1008
*Starting Points Balance*: 0
*Street Address*: 123 ABC St.
*City*: Brooklyn
*Postal Code*: 11214
*Country*: USA
*State*: New York

**Query:**
INSERT INTO Account1
VALUES (‘A1008’, ‘0’, ‘123 ABC St.’, ‘Brooklyn’, ‘11214’, ‘USA’);
INSERT INTO Account2
VALUES (‘11214’, ‘USA’, ‘New York’);

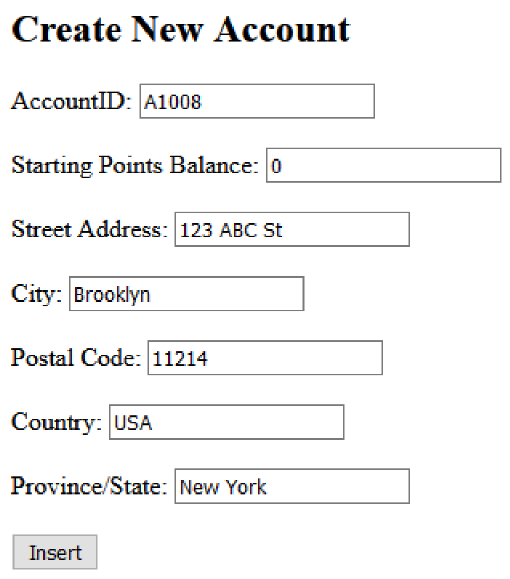
**Screenshot Prior to Execution of Query**



**Output**

| | | | | | | |
|---|---|---|---|---|---|---|
| A1001 | 1000 | 3308 Ast St. | Vancouver | V5Z 3E3 | Canada | British Columbia |
| A1002 | 0 | 374 Brisdale Dr | Brampton | L7A 3M5 | Canada | Ontario |
| A1003 | 3000 | 500 Kingston Rd | Toronto | M4L 1V3 | Canada | Ontario |
| A1004 | 4000 | 7503 Rue St Denis | Montreal | H2R 2E7 | Canada | Ontario |
| A1005 | 500 | 3124 Doctors Drive | Los Angeles | 90017 | USA | California |
| A1006 | 800 | 157 West 57th St. | New York | 10019 | USA | New York |
| A1007 | 1500 | 7 Hickson Road | The Rocks | 2000 | Australia | NSW |

## Query with Sample Input



### Create New Account

AccountID: A1008

Starting Points Balance: 0

Street Address: 123 ABC St

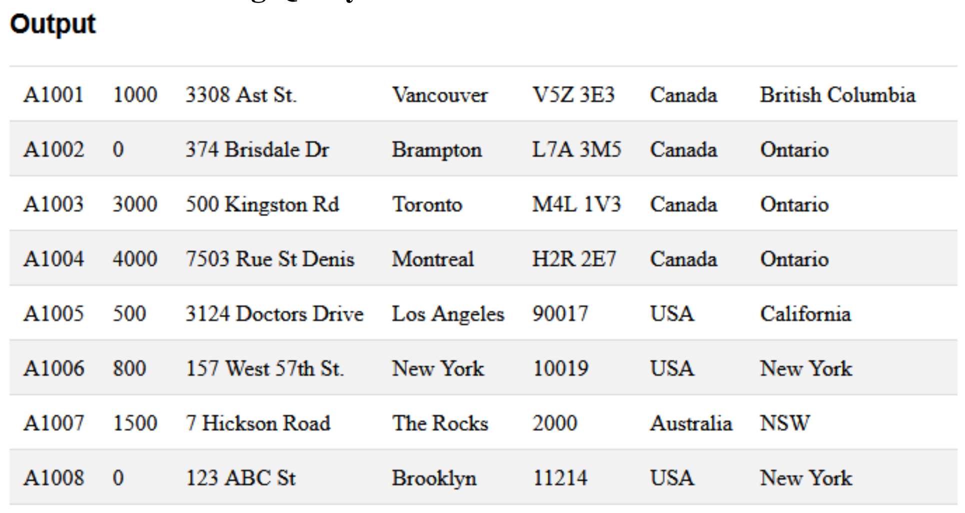
City: Brooklyn

Postal Code: 11214

Country: USA

Province/State: New York

## Result of Executing Query



**Output**

| | | | | | | |
|---|---|---|---|---|---|---|
| A1001 | 1000 | 3308 Ast St. | Vancouver | V5Z 3E3 | Canada | British Columbia |
| A1002 | 0 | 374 Brisdale Dr | Brampton | L7A 3M5 | Canada | Ontario |
| A1003 | 3000 | 500 Kingston Rd | Toronto | M4L 1V3 | Canada | Ontario |
| A1004 | 4000 | 7503 Rue St Denis | Montreal | H2R 2E7 | Canada | Ontario |
| A1005 | 500 | 3124 Doctors Drive | Los Angeles | 90017 | USA | California |
| A1006 | 800 | 157 West 57th St. | New York | 10019 | USA | New York |
| A1007 | 1500 | 7 Hickson Road | The Rocks | 2000 | Australia | NSW |
| A1008 | 0 | 123 ABC St | Brooklyn | 11214 | USA | New York |

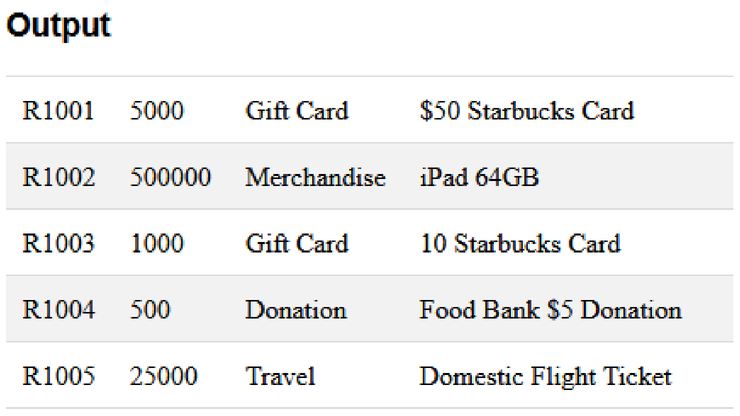
## Delete: Delete a reward

**Front end implementation**: reward.php
**Back-end**: deleteReward() in reward-controller.php

**Sample input**:
*Enter ID of the reward you wish to delete:* R1005

**Query**:
DELETE FROM Reward
WHERE rewardID = 'R1005';

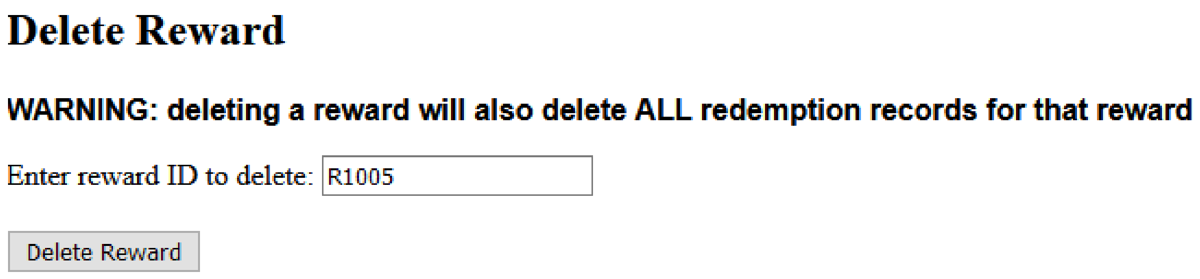
**Screenshot Prior to Execution of Query**



**Output**

| | | | |
|---|---|---|---|
| R1001 | 5000 | Gift Card | $50 Starbucks Card |
| R1002 | 500000 | Merchandise | iPad 64GB |
| R1003 | 1000 | Gift Card | 10 Starbucks Card |
| R1004 | 500 | Donation | Food Bank $5 Donation |
| R1005 | 25000 | Travel | Domestic Flight Ticket |

**Query with Sample Input**

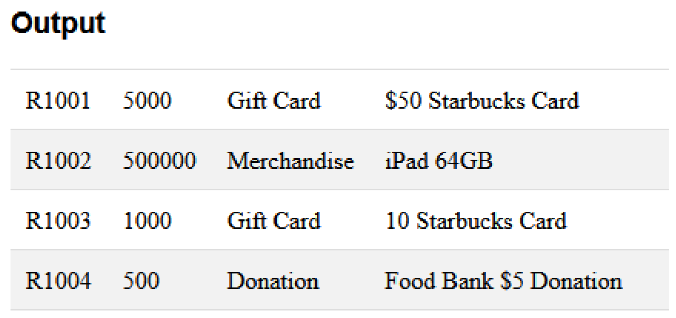


**Delete Reward**

**WARNING: deleting a reward will also delete ALL redemption records for that reward**

Enter reward ID to delete: R1005

**Result of Executing Query**



**Output**

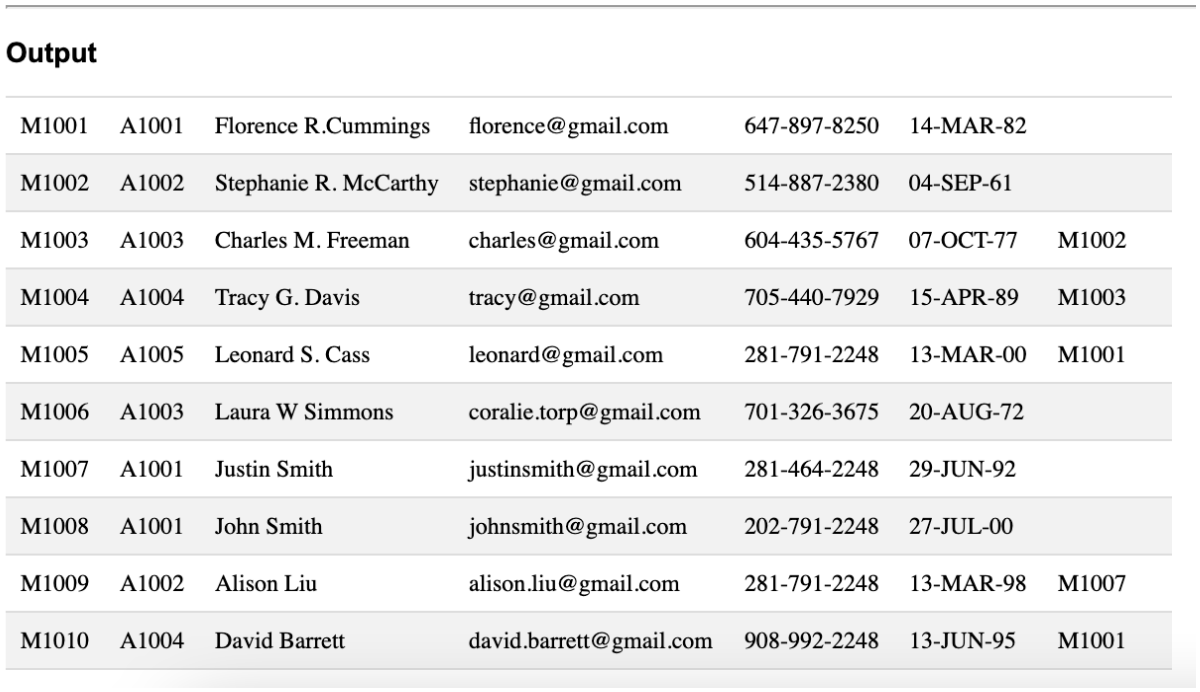
| | | | |
|---|---|---|---|
| R1001 | 5000 | Gift Card | $50 Starbucks Card |
| R1002 | 500000 | Merchandise | iPad 64GB |
| R1003 | 1000 | Gift Card | 10 Starbucks Card |
| R1004 | 500 | Donation | Food Bank $5 Donation |

## Update: update member email or phone

**Front end implementation**: member.php
**Back-end implementation**: handleUpdateMemberRequest() in member-controller.php

**Sample input**:
*Enter ID of the member you wish to update*: M1001
*New email*: foo@gmail.com
*New phone*: 111-111-1111

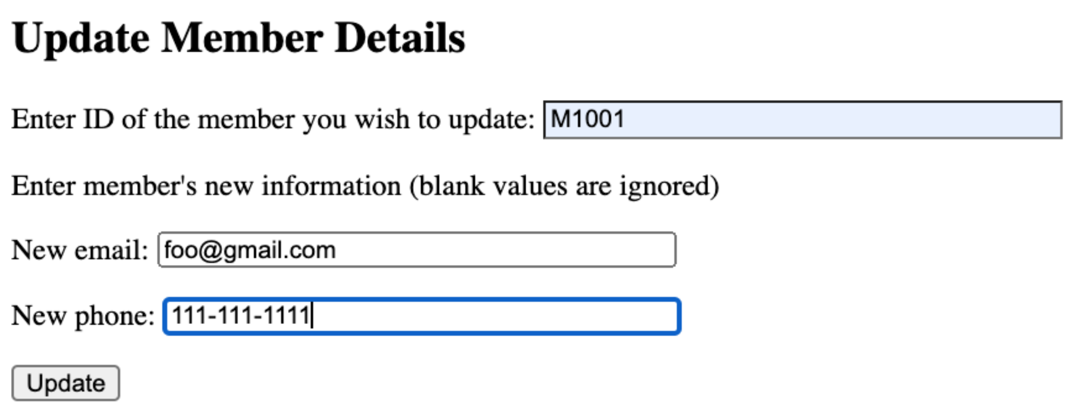
**Screenshot Prior to Execution of Query**:



**Output**

| | | | | | | |
|---|---|---|---|---|---|---|
| M1001 | A1001 | Florence R.Cummings | florence@gmail.com | 647-897-8250 | 14-MAR-82 | |
| M1002 | A1002 | Stephanie R. McCarthy | stephanie@gmail.com | 514-887-2380 | 04-SEP-61 | |
| M1003 | A1003 | Charles M. Freeman | charles@gmail.com | 604-435-5767 | 07-OCT-77 | M1002 |
| M1004 | A1004 | Tracy G. Davis | tracy@gmail.com | 705-440-7929 | 15-APR-89 | M1003 |
| M1005 | A1005 | Leonard S. Cass | leonard@gmail.com | 281-791-2248 | 13-MAR-00 | M1001 |
| M1006 | A1003 | Laura W Simmons | coralie.torp@gmail.com | 701-326-3675 | 20-AUG-72 | |
| M1007 | A1001 | Justin Smith | justinsmith@gmail.com | 281-464-2248 | 29-JUN-92 | |
| M1008 | A1001 | John Smith | johnsmith@gmail.com | 202-791-2248 | 27-JUL-00 | |
| M1009 | A1002 | Alison Liu | alison.liu@gmail.com | 281-791-2248 | 13-MAR-98 | M1007 |
| M1010 | A1004 | David Barrett | david.barrett@gmail.com | 908-992-2248 | 13-JUN-95 | M1001 |

**Query with Sample Input**



## Update Member Details

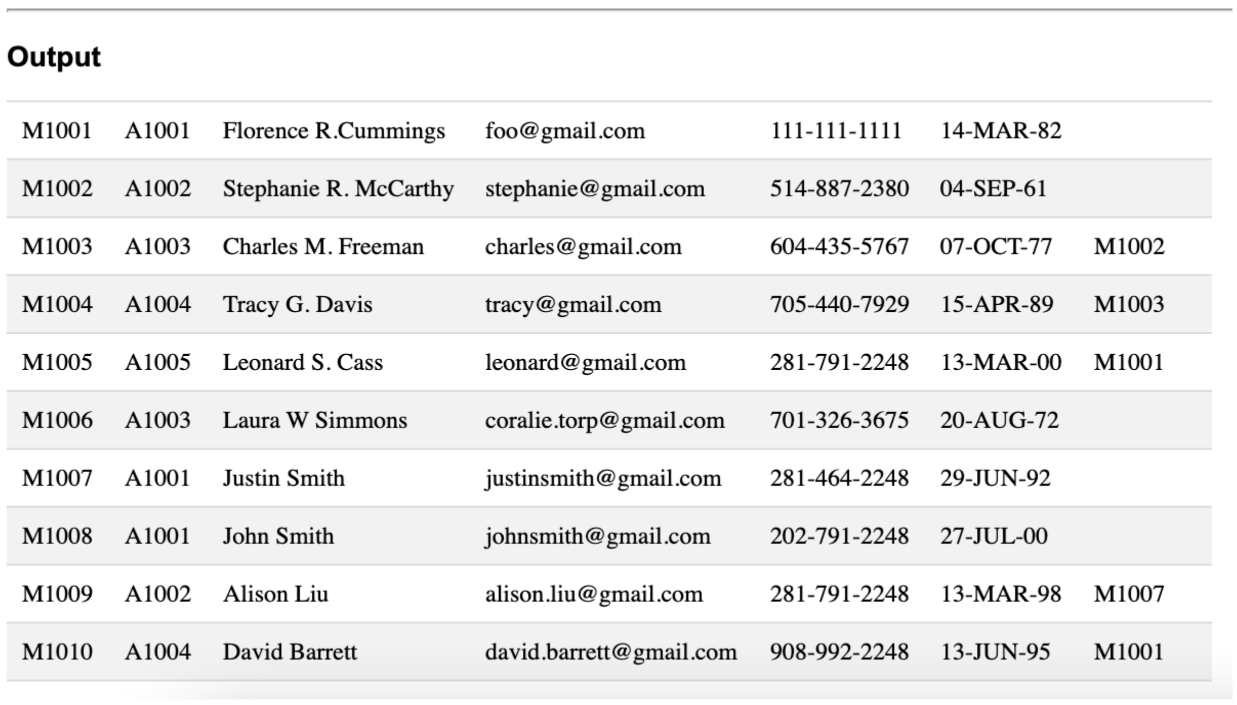
Enter ID of the member you wish to update: M1001

Enter member's new information (blank values are ignored)

New email: foo@gmail.com

New phone: 111-111-1111

**Result of Executing Query**



**Output**

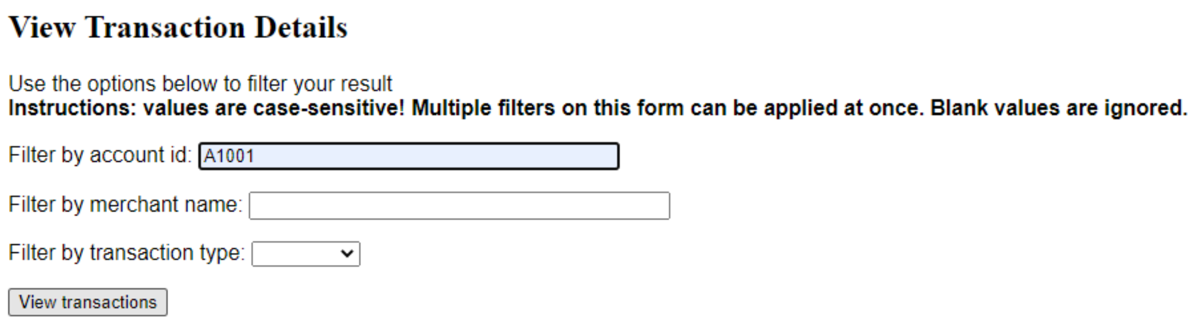
| | | | | | | |
|---|---|---|---|---|---|---|
| M1001 | A1001 | Florence R.Cummings | foo@gmail.com | 111-111-1111 | 14-MAR-82 | |
| M1002 | A1002 | Stephanie R. McCarthy | stephanie@gmail.com | 514-887-2380 | 04-SEP-61 | |
| M1003 | A1003 | Charles M. Freeman | charles@gmail.com | 604-435-5767 | 07-OCT-77 | M1002 |
| M1004 | A1004 | Tracy G. Davis | tracy@gmail.com | 705-440-7929 | 15-APR-89 | M1003 |
| M1005 | A1005 | Leonard S. Cass | leonard@gmail.com | 281-791-2248 | 13-MAR-00 | M1001 |
| M1006 | A1003 | Laura W Simmons | coralie.torp@gmail.com | 701-326-3675 | 20-AUG-72 | |
| M1007 | A1001 | Justin Smith | justinsmith@gmail.com | 281-464-2248 | 29-JUN-92 | |
| M1008 | A1001 | John Smith | johnsmith@gmail.com | 202-791-2248 | 27-JUL-00 | |
| M1009 | A1002 | Alison Liu | alison.liu@gmail.com | 281-791-2248 | 13-MAR-98 | M1007 |
| M1010 | A1004 | David Barrett | david.barrett@gmail.com | 908-992-2248 | 13-JUN-95 | M1001 |

## Selection: view transaction details

**Front end implementation**: transaction.php
**Back-end implementation**: handleDisplayTransactionRequest() in transaction-controller.php

**Sample input**:
*Filter by account id*: A1001
*Filter by merchant name*: Lululemon
*Filter by transaction type*: exchange
***note:** 0, 1, 2, or all filters can be applied at once

### Query with Sample Input



**View Transaction Details**

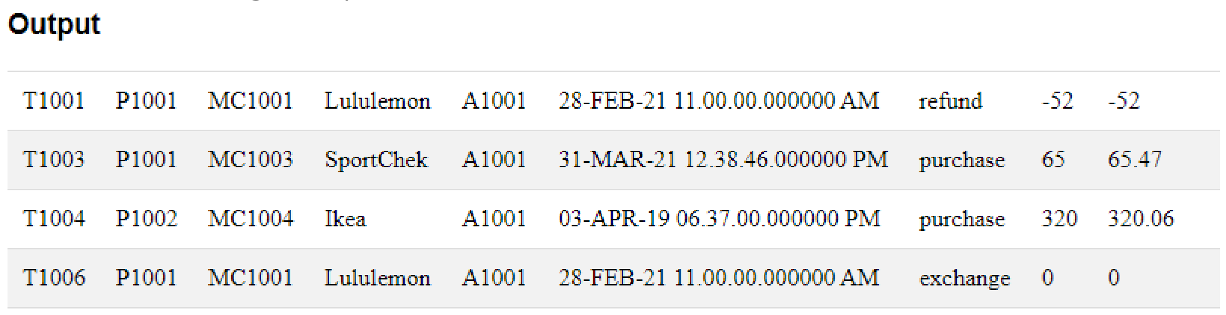
Use the options below to filter your result
**Instructions: values are case-sensitive! Multiple filters on this form can be applied at once. Blank values are ignored.**

Filter by account id: A1001

Filter by merchant name:

Filter by transaction type:

### Result of Executing Query



**Output**

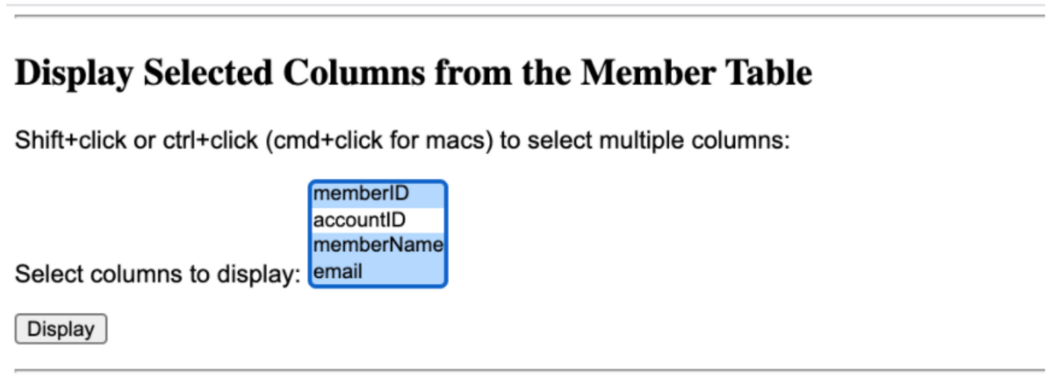
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| T1001 | P1001 | MC1001 | Lululemon | A1001 | 28-FEB-21 11.00.00.000000 AM | refund | -52 | -52 |
| T1003 | P1001 | MC1003 | SportChek | A1001 | 31-MAR-21 12.38.46.000000 PM | purchase | 65 | 65.47 |
| T1004 | P1002 | MC1004 | Ikea | A1001 | 03-APR-19 06.37.00.000000 PM | purchase | 320 | 320.06 |
| T1006 | P1001 | MC1001 | Lululemon | A1001 | 28-FEB-21 11.00.00.000000 AM | exchange | 0 | 0 |

Projection: display selected columns from the Member table.

**Front end implementation**: member.php
**Back-end implementation**: handleMemberProjectionRequest() in member-controller.php

**Sample input**:
*Select columns to display*: memberID, memberName, email, phone, birthDate.

**Query with Sample Input**

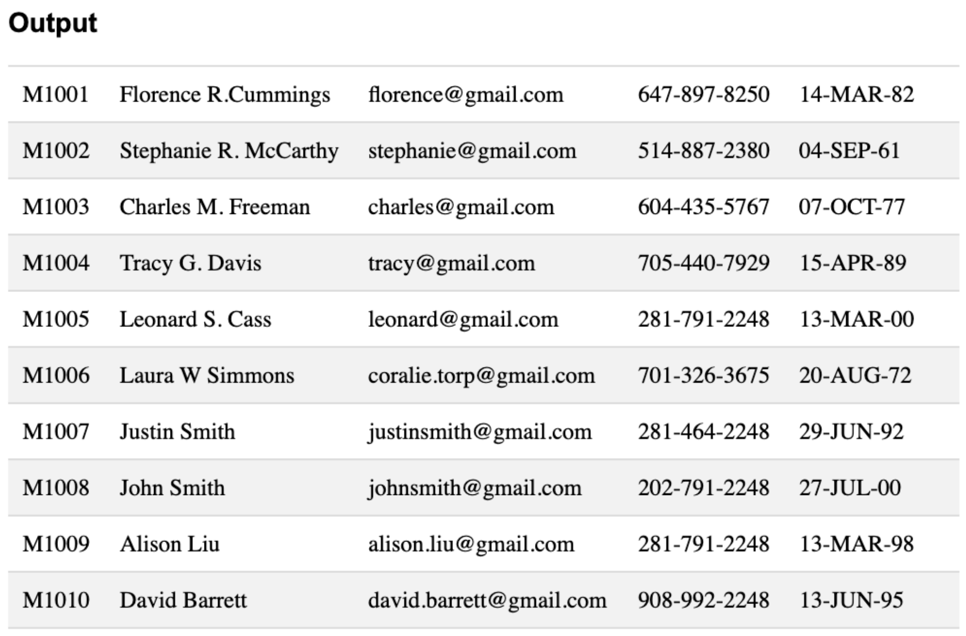


**Display Selected Columns from the Member Table**

Shift+click or ctrl+click (cmd+click for macs) to select multiple columns:

Select columns to display:

**Result of Executing Query**



**Output**

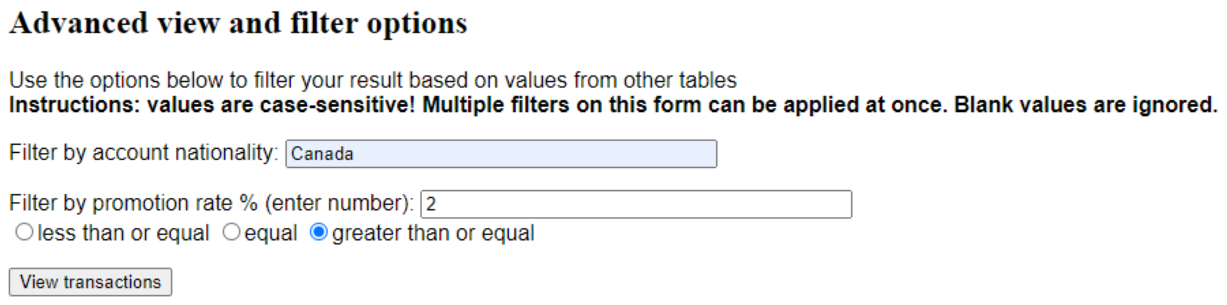
| | | | | |
|---|---|---|---|---|
| M1001 | Florence R.Cummings | florence@gmail.com | 647-897-8250 | 14-MAR-82 |
| M1002 | Stephanie R. McCarthy | stephanie@gmail.com | 514-887-2380 | 04-SEP-61 |
| M1003 | Charles M. Freeman | charles@gmail.com | 604-435-5767 | 07-OCT-77 |
| M1004 | Tracy G. Davis | tracy@gmail.com | 705-440-7929 | 15-APR-89 |
| M1005 | Leonard S. Cass | leonard@gmail.com | 281-791-2248 | 13-MAR-00 |
| M1006 | Laura W Simmons | coralie.torp@gmail.com | 701-326-3675 | 20-AUG-72 |
| M1007 | Justin Smith | justinsmith@gmail.com | 281-464-2248 | 29-JUN-92 |
| M1008 | John Smith | johnsmith@gmail.com | 202-791-2248 | 27-JUL-00 |
| M1009 | Alison Liu | alison.liu@gmail.com | 281-791-2248 | 13-MAR-98 |
| M1010 | David Barrett | david.barrett@gmail.com | 908-992-2248 | 13-JUN-95 |

## Join: join Transaction with Account or Promotion

**Front end implementation**: transaction.php
**Back-end implementation**: handleDisplayAdvancedTransactionRequest() in transaction-controller.php

**Sample input**:
*Filter by account nationality*: Canada
*Filter by promotion rate %*: 2, greater than or equal
***note:** 0, 1, or 2 filters can be applied at once

### Query with Sample Input



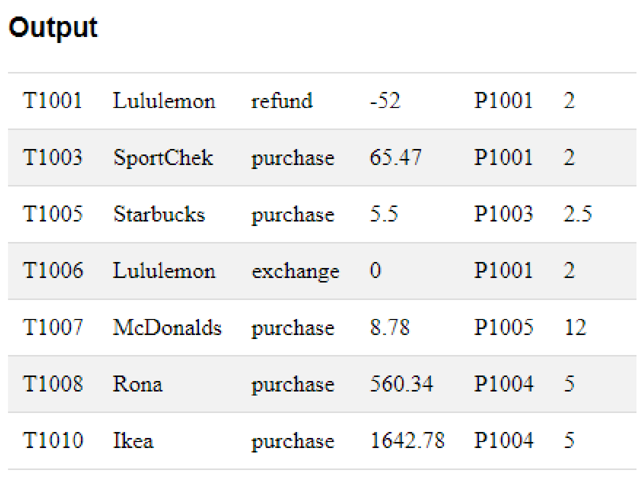
**Advanced view and filter options**

Use the options below to filter your result based on values from other tables
**Instructions: values are case-sensitive! Multiple filters on this form can be applied at once. Blank values are ignored.**

Filter by account nationality: Canada

Filter by promotion rate % (enter number): 2
less than or equal equal [✓ greater than or equal]

### Result of Executing Query



**Output**

| | | | | | |
|---|---|---|---|---|---|
| T1001 | Lululemon | refund | -52 | P1001 | 2 |
| T1003 | SportChek | purchase | 65.47 | P1001 | 2 |
| T1005 | Starbucks | purchase | 5.5 | P1003 | 2.5 |
| T1006 | Lululemon | exchange | 0 | P1001 | 2 |
| T1007 | McDonalds | purchase | 8.78 | P1005 | 12 |
| T1008 | Rona | purchase | 560.34 | P1004 | 5 |
| T1010 | Ikea | purchase | 1642.78 | P1004 | 5 |

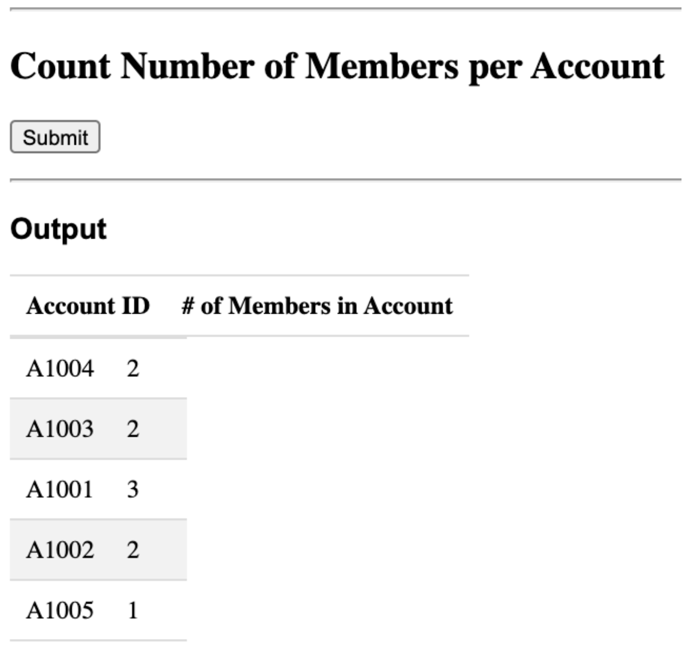
Aggregation with Group by: count number of members per account.

**Front end implementation**: member.php
**Back-end implementation**: handleCountMemberRequest() in member-controller.php

**Query**:

```
SELECT        accountID, COUNT(*)
FROM          Member
GROUP BY   accountID
```

**Result of Executing Query**



## Count Number of Members per Account

Submit

### Output

| Account ID | # of Members in Account |
|---|---|
| A1004 | 2 |
| A1003 | 2 |
| A1001 | 3 |
| A1002 | 2 |
| A1005 | 1 |

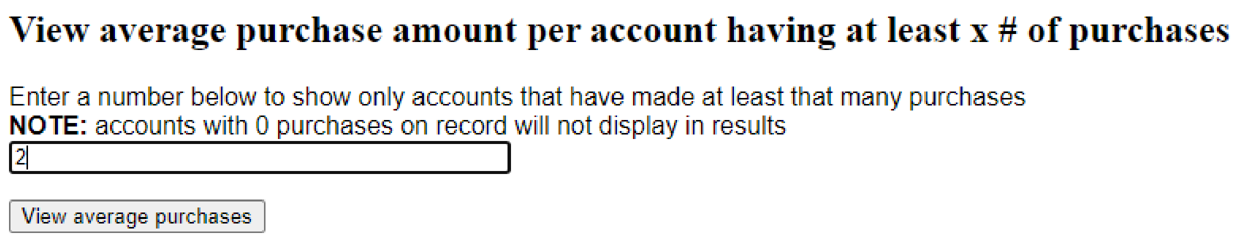
Aggregation with Having: average purchase amount for each account, having made at least x number of purchases

**Front end implementation**: transaction.php
**Back-end implementation**: handleDisplayAvgPurchaseByAccRequest() in transaction-controller.php

**Sample input**:
*Enter a number below (...)*: 1

**Query**:
```
SELECT accountID, AVG(transactionAmount)
FROM Transaction
WHERE type='purchase'
GROUP BY accountID
HAVING COUNT(*) >= $minPurchaseNum
```

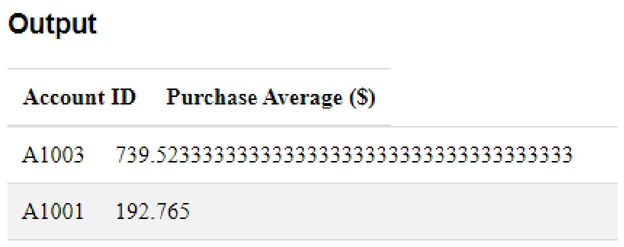
**Query with Sample Input**



**View average purchase amount per account having at least x # of purchases**

Enter a number below to show only accounts that have made at least that many purchases
**NOTE:** accounts with 0 purchases on record will not display in results
2

**Result of Executing Query**



**Output**

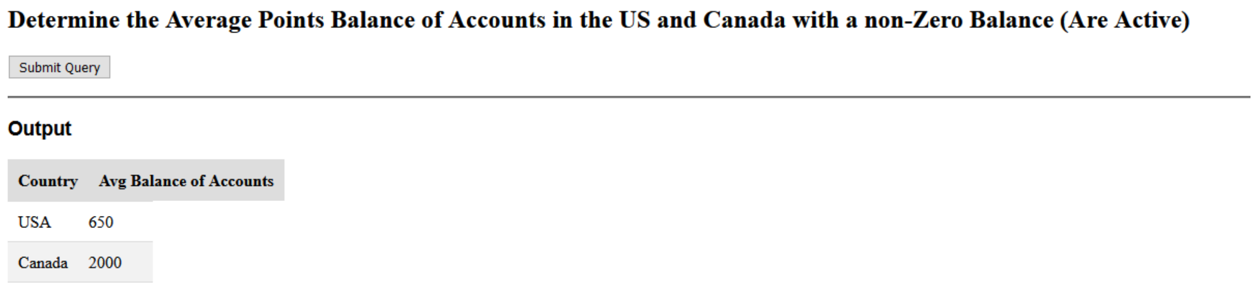
| Account ID | Purchase Average ($) |
|---|---|
| A1003 | 739.5233333333333333333333333333333333333 |
| A1001 | 192.765 |

**Nested Aggregation with Group By**: View the average points balance of accounts based in the US and Canada that have a non-zero balance, grouped by country.

**Front end implementation**: account.php
**Back-end implementation**: handleAvgAccBalanceRequest() in account-controller.php

**Query:**
```
SELECT country, AVG(pointBalance)
FROM (SELECT * FROM Account1 WHERE Account1.pointBalance > 0)
WHERE country = 'USA' OR country = 'Canada'
GROUP BY country;
```

**Result of Executing Query**



**Determine the Average Points Balance of Accounts in the US and Canada with a non-Zero Balance (Are Active)**

**Output**

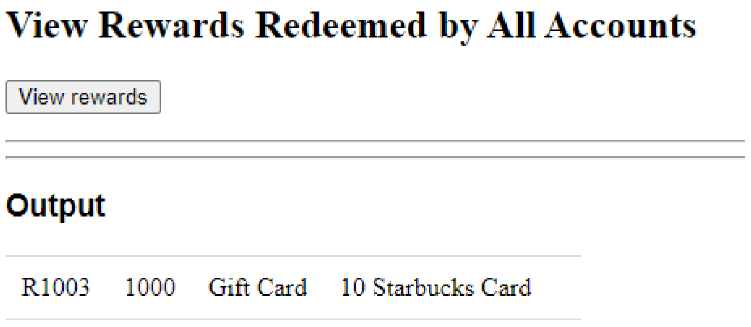
| Country | Avg Balance of Accounts |
|---|---|
| USA | 650 |
| Canada | 2000 |

## Division: view rewards redeemed by all accounts

**Front end implementation**: reward.php
**Back-end implementation**: handleDisplayRedeemedByAllRequest() in transaction-controller.php

**Query**:
```
SELECT * FROM Reward rwd
WHERE NOT EXISTS
     (SELECT a.accountID
     FROM Account1 a
     MINUS
          (SELECT rdm.accountID
          FROM Redeems rdm
          WHERE rdm.rewardID=rwd.rewardID))
```

**Result of Executing Query**



```
R1003    1000    Gift Card    10 Starbucks Card
```