# 



**SHAREBILLS**

A database project on simplified expense sharing among users

Subhash Daggubati

Tejaswi Lakkakula

BalaPavan Kommareddy

Sandeep Anuguthala

Anup Kumar Chittimalla

## **TABLE OF CONTENTS**

1. DATABASE PLANNING ……………………………………………………………………………………………………………..3
2. MISSION STATEMENT
3. MISSION OBJECTIVES
4. STANDARDS DEVELOPMENT
5. DATABASE DESIGN ………………………………………………………………………………………………………………….4
6. BUSINESS RULES
7. ERD DIAGRAM
8. DBMS FEATURES …………………………………………………………………………………………………………………….6
9. DATA DEFINITION LANGUAGE
10. DATA MANIPULATION LANGUAGE

DATABASE PLANNING

1. **MISSION STATEMENT**

Sharebill's mission is to make shared living and travel easier by providing neutral advice, fair judgement, and simplified expense sharing through Database management system.

1. **MISSION OBJECTIVES**

To maintain (INSERT, UPDATE, DELETE) DATA ON USERS

To maintain (INSERT, UPDATE, DELETE) DATA ON TRANSACTIONS

To maintain (INSERT, UPDATE, DELETE) DATA ON GROUPS

To maintain (INSERT, UPDATE, DELETE) DATA ON TRANSACTION\_DETAIL

To maintain (INSERT, UPDATE, DELETE) DATA ON GROUP\_MEMBERS

To maintain (INSERT, UPDATE, DELETE) DATA ON SETTLEUP\_REQUEST

To perform searches on USERS

To perform searches on TRANSACTIONS

To perform searches on GROUPS

To perform searches on TRANSACTION\_DETAIL

To perform searches on GROUP\_MEMBERS

To perform searches on SETTLEUP\_REQUEST

To track the status of settled transactions

To track status of transactions of Group members

To report the monthly transaction of a particular group.

1. **STANDARDS DEVELOPMENT**
2. The user should be a sharebills account holder
3. Transactions such as Group or non-group transaction is allowed
4. Any transaction can involve one or many users including himself
5. User (borrower) can send a Settle up request to another user (lender)
6. User can include the limited users from the group members
7. User can take note of any number of users in the non-group transaction
8. Settle up request should be initiated by borrower and settle up should be done by lender only.

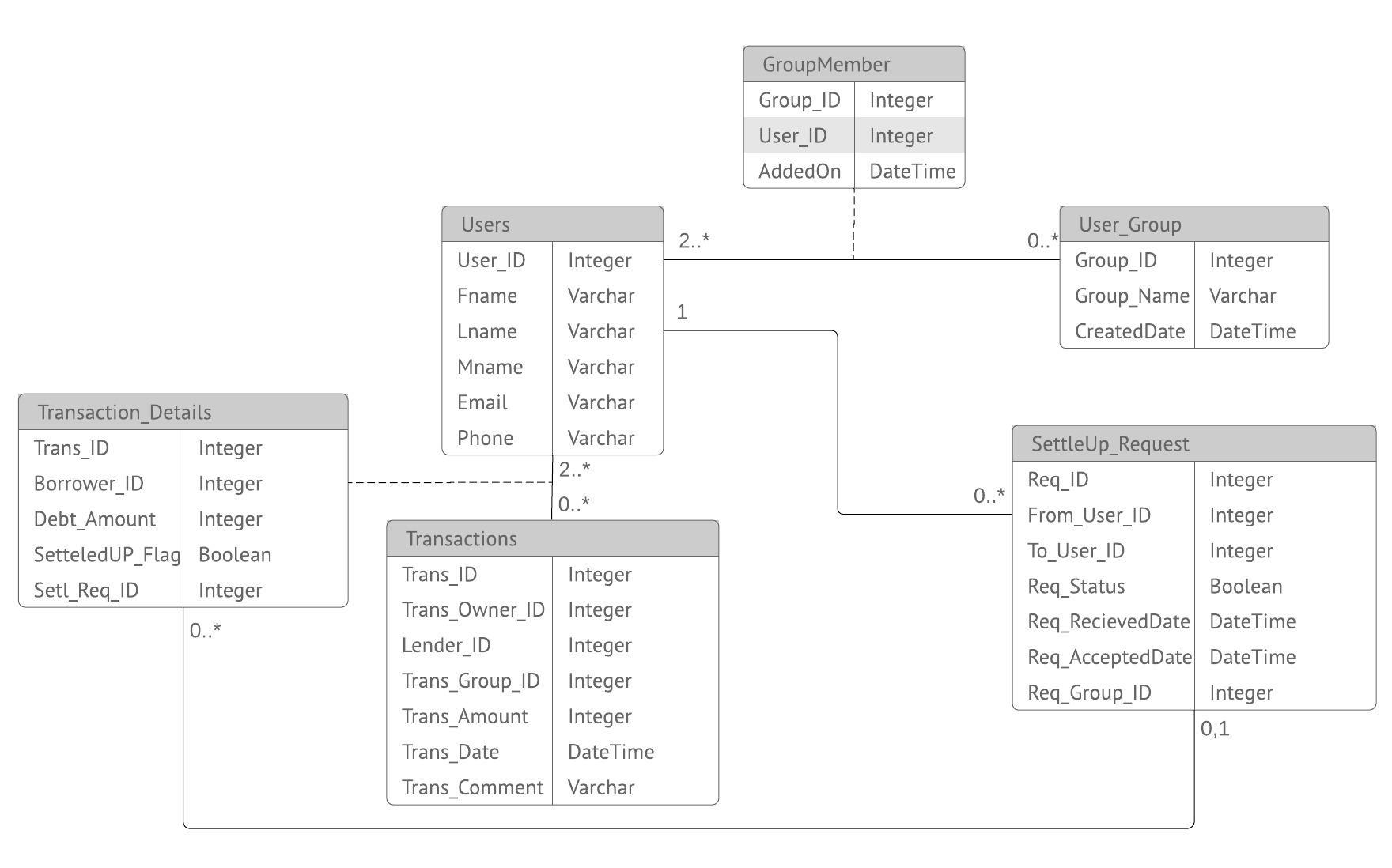
DATABASE DESIGN

1. **BUSINESS RULES**

ShareBills is a database where transactions of all shared expenses between users is stored.

* ShareBills can get transactions from individual users and users from different groups.
* Each user can be a part of none of the group or several groups.
* Each group can have two or more users.
* Every transaction involves two or more users. For example, Walmart expense is a transaction of common groceries with the amount of 200 USD that involved multiple users made on a particular day.
* Each user can have zero or many transactions.
* For each user there is zero settle up request or many settle up requests.
* Every user is identified by User ID. The first name, last name, middle name, email id, and phone number of all users are recorded in the system.
* A borrower can send a settle up request to lender once he repays the amount lent from borrower. Once the lender approves the settle up request, the rows in the Transaction detail entity will be updated.
* Transaction detail entity has information about transaction id, borrower id, debt amount, settled up flag, and settle up request id.
* For every group, group id, group name and group created date is maintained.
* For every single settle up request is known by Request id. In addition, request made by a user, request sent to a user with the request status, request received date, request accepted date, and request group id is stored in the system.
* A particular transaction is recognized by transaction id. Transaction owner id, Lender id along with the transaction group id, transaction date, transaction amount, and a transaction comment are also kept in the database.

1. **ERD DIAGRAM**



DBMS FEATURES

1. **DATA DEFINITION LANGUAGE**

* **Creating Tables**

CREATE TABLE **Users** (

   User\_ID INT not null auto\_increment,

   Fname VARCHAR(256),

   Lname VARCHAR(256),

   Mname VARCHAR(256),

   EmailID VARCHAR(256),

   phone VARCHAR(256),

   PRIMARY KEY (User\_ID)

);

CREATE TABLE **User\_Group** (

   Group\_ID INT not null auto\_increment,

   Group\_Name VARCHAR(256),

CreatedDate DateTime,

   PRIMARY KEY (Group\_ID)

);

CREATE TABLE **GroupMember** (

   Group\_ID INT,

   User\_ID INT,

   AddedOn DATETIME,

   FOREIGN KEY (Group\_ID)

       REFERENCES User\_Group (Group\_ID),

   FOREIGN KEY (User\_ID)

       REFERENCES Users (User\_ID),

   PRIMARY KEY (Group\_ID , User\_ID)

);

CREATE TABLE **Transactions** (

   Trans\_ID INT not null auto\_increment,

   trans\_Owner\_ID INT,

   Lender\_ID INT,

   Trans\_Group\_ID INT,

   Trans\_Amount INT,

   Trans\_Date DATETIME,

   Trans\_Comment VARCHAR(256),

   FOREIGN KEY (trans\_Owner\_ID)

       REFERENCES Users (User\_ID),

   FOREIGN KEY (Lender\_ID)

       REFERENCES Users (User\_ID),

   FOREIGN KEY (trans\_Group\_ID)

       REFERENCES User\_Group (Group\_ID),

   PRIMARY KEY (Trans\_ID)

);

CREATE TABLE **SettleUP\_Request** (

   Req\_ID INT not null auto\_increment,

   From\_User\_ID INT,

   To\_User\_ID INT,

   Req\_Status BOOLEAN,

   Req\_RecievedDate DATETIME,

   Req\_AcceptedDate DATETIME,

   Req\_Group\_ID INT,

   FOREIGN KEY (From\_User\_ID)

       REFERENCES Users (User\_ID),

   FOREIGN KEY (To\_User\_ID)

       REFERENCES Users (User\_ID),

   FOREIGN KEY (Req\_Group\_ID)

       REFERENCES User\_Group (Group\_ID),

   PRIMARY KEY (Req\_ID)

);

CREATE TABLE **Transaction\_Details** (

   Trans\_ID INT,

   Borrower\_ID INT,

   Debt\_Amount INT,

   SettledUp\_Flag BOOLEAN,

   setl\_Req\_ID INT,

   FOREIGN KEY (Trans\_ID)

       REFERENCES Transactions (Trans\_ID),

   FOREIGN KEY (Borrower\_ID)

       REFERENCES Users (User\_ID),

foreign key (setl\_Req\_ID)

references settleup\_request (Req\_ID),

   PRIMARY KEY (Trans\_ID , Borrower\_ID)

);

* **Inserting values into the table**

INSERT INTO USERS

( FNAME,LNAME,MNAME,EMAILID,PHONE)

VALUES('JAY','HARKER','NULL','jay@gmail.com','(405)780-2323'),

('CARL','LAYTON','NULL','carl@gmail.com','(405)780-2343'),

('MARC','LEVIN','NULL','marc@gmail.com','(405)780-2353'),

('AMY','DOMINGO','NULL','amy@gmail.com','(405)780-2363'),

('BOBBY','BEST','NULL','bobby@gmail.com','(405)780-2312');

INSERT INTO USER\_GROUP

(GROUP\_NAME,CREATEDDATE)

VALUES('APARTMENT EXPENSES','2018/12/1'),

('TRIP EXPENSES','2018/12/1');

INSERT INTO GROUPMEMBER

(GROUP\_ID,USER\_ID,ADDEDON)

VALUES(1,1,'2018/12/1'),

(1,2,'2018/12/1'),

(1,3,'2018/12/1'),

(1,4,'2018/12/1'),

(1,5,'2018/12/1'),

(2,2,'2018/12/1'),

(2,3,'2018/12/1'),

(2,4,'2018/12/1');

INSERT INTO TRANSACTIONS

(TRANS\_OWNER\_ID,LENDER\_ID,TRANS\_GROUP\_ID, TRANS\_AMOUNT,TRANS\_DATE,TRANS\_COMMENT)

VALUES(1,2,1,100,'2018/12/1','RENT FOR DECEMBER'),

(4,4,2,150,'2018/11/20','TRIP TO DALLAS'),

(2,2,NULL,60,'2018/12/1','RESTAURANT BILL'),

(3,5,NULL,75,'2018/11/20','MOVIE EXPENSES'),

(3,5,1,200,'2018/12/2','Wifi Bill'),

(2,2,2,300,'2018/12/3','Trip to OKC') ;

INSERT INTO TRANSACTION\_DETAILS

(TRANS\_ID,BORROWER\_ID,DEBT\_AMOUNT,SETTLEDUP\_FLAG,SETL\_REQ\_ID)

VALUES (1,1,20,0,NULL),

(1,3,20,0,NULL),

(1,4,20,0,NULL),

(1,5,20,0,NULL),

(2,2,50,0,NULL),

(2,3,50,0,NULL),

(3,4,20,0,NULL),

(3,5,20,0,NULL),

(5,1,40,0,NULL),

(5,2,40,0,NULL),

(5,3,40,0,NULL),

(5,4,40,0,NULL),

(6,3,100,0,NULL),

(6,4,100,0,NULL);

INSERT INTO SETTLEUP\_REQUEST

(FROM\_USER\_ID,TO\_USER\_ID,REQ\_STATUS,REQ\_RECIEVEDDATE,REQ\_ACCEPTEDDATE,REQ\_GROUP\_ID)

VALUES(1,2,0,'2018/12/1',NULL,1),

(3,4,0,'2018/12/1',NULL,2),

(3,5,0,'2018/12/1',NULL,NULL),

(5,2,0,'2018/12/1',NULL,NULL);

* **Stored Procedure**

CREATE PROCEDURE `Insert\_Transaction\_Details` (IN INPUT TEXT,

IN delimiter VARCHAR(10), IN debt\_amount Int, in trans\_id int)

BEGIN

DECLARE cur\_position INT DEFAULT 1 ;

DECLARE remainder TEXT;

DECLARE cur\_string VARCHAR(1000);

DECLARE delimiter\_length TINYINT UNSIGNED;

SET remainder = input;

SET delimiter\_length = CHAR\_LENGTH(delimiter);

WHILE CHAR\_LENGTH(remainder) > 0 AND cur\_position > 0 DO

SET cur\_position = INSTR(remainder, delimiter);

IF cur\_position = 0 THEN

SET cur\_string = remainder;

ELSE

SET cur\_string = LEFT(remainder, cur\_position - 1);

END IF;

IF TRIM(cur\_string) != '' THEN

INSERT INTO transaction\_details (Trans\_ID, Borrower\_ID, Debt\_Amount,SettledUp\_Flag) VALUES (trans\_ID,cast(cur\_string as unsigned), debt\_amount,0);

commit;

END IF;

SET remainder = SUBSTRING(remainder, cur\_position + delimiter\_length);

END WHILE;

END

* **Triggers**

/\*AFTER INSERT SETTLEUP\_REQUEST\*/

CREATE DEFINER=`root`@`localhost` TRIGGER `settleup\_request\_AFTER\_INSERT` AFTER INSERT ON `settleup\_request` FOR EACH ROW BEGIN

if (NEW.Req\_Group\_ID is not null) then

UPDATE Transaction\_Details

SET

   setl\_Req\_ID = NEW.Req\_ID

WHERE

   (borrower\_id = NEW.FROM\_USER\_ID OR borrower\_id = NEW.TO\_USER\_ID) and setl\_Req\_ID IS NULL AND

   Trans\_ID IN (SELECT

           td.trans\_ID

       FROM

           (select \* from transaction\_details) td join transactions t on t.trans\_id = td.trans\_id

where t.trans\_group\_id = NEW.Req\_Group\_ID and (t.lender\_id = NEW.From\_User\_ID or t.lender\_id = NEW.To\_User\_ID )

AND ( td.borrower\_id = NEW.To\_User\_ID or td.borrower\_id = NEW.From\_User\_ID));

else

UPDATE Transaction\_Details

SET

   setl\_Req\_ID = NEW.Req\_ID

WHERE

(borrower\_id = NEW.FROM\_USER\_ID OR borrower\_id = NEW.TO\_USER\_ID) and setl\_Req\_ID IS NULL AND

   Trans\_ID IN (SELECT

           td.trans\_ID

       FROM

           (select \* from transaction\_details) td join transactions t on t.trans\_id = td.trans\_id

where (t.lender\_id = NEW.From\_User\_ID or t.lender\_id = NEW.To\_User\_ID )

AND ( td.borrower\_id = NEW.To\_User\_ID or td.borrower\_id = NEW.From\_User\_ID));

End if;

END

/\*AFTER UPDATE SETTLEUP REQUEST\*/

CREATE DEFINER=`root`@`localhost` TRIGGER `settleup\_request\_AFTER\_UPDATE` AFTER UPDATE ON `settleup\_request` FOR EACH ROW BEGIN

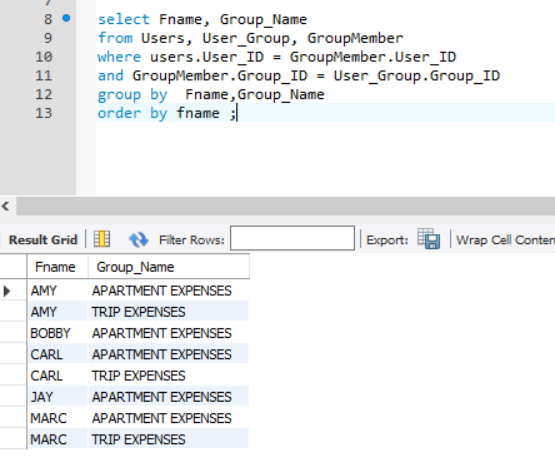
IF (OLD.Req\_Status =0 and NEW.Req\_Status=1) THEN

Update Transaction\_Details set Settledup\_Flag = 1 where Setl\_Req\_ID = OLD.Req\_ID;

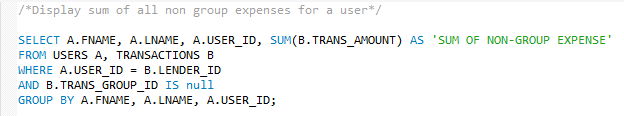
END IF;

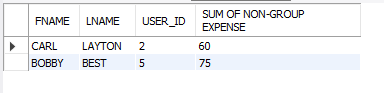
END

1. **DATA MANIPULATION LANGUAGE**
2. Display the names of all groups a user is part of.

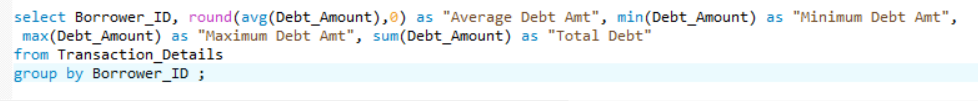


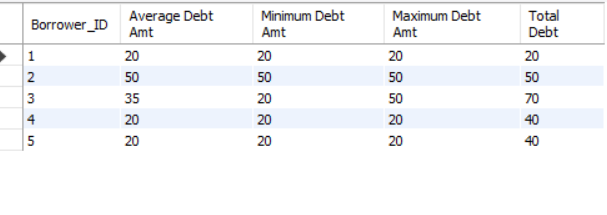
1. Display user name and sum of all non-group expenses for a user.



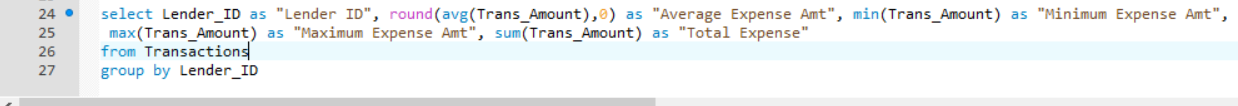


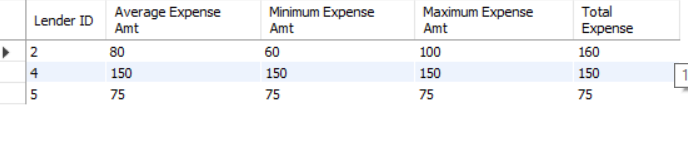
1. Display average, maximum, minimum and total of Debt amount by a user.

****

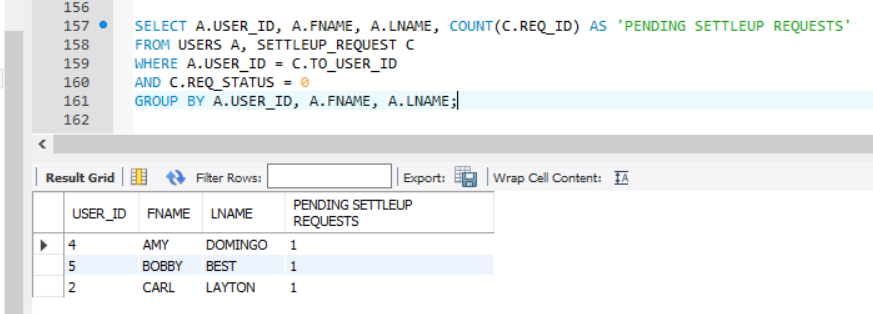
****

1. Display average, maximum, minimum and total expense by a user

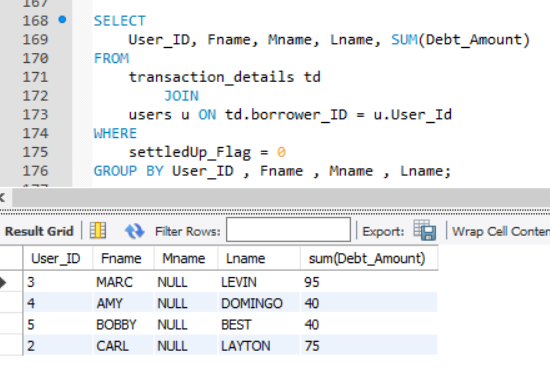
****

****

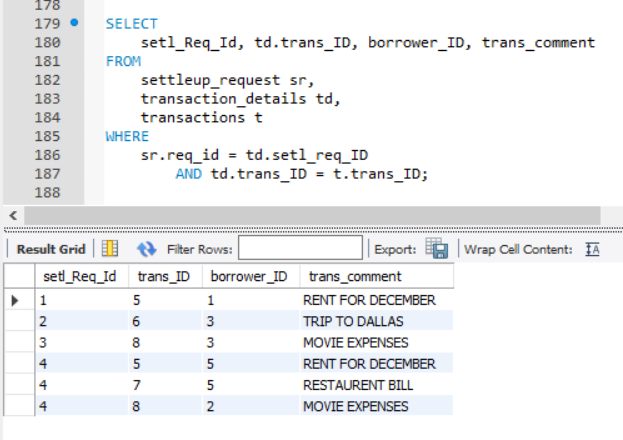
1. Display the no of pending settle up requests for every user.

****

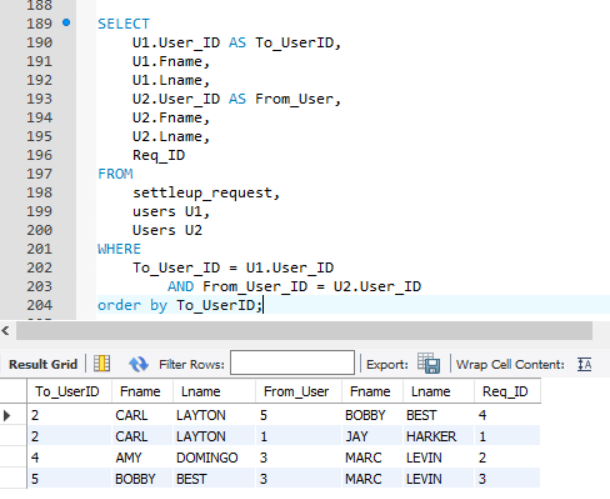
1. Display the amount borrowed by each user which is not repaid or settled.

****

1. Display the list of Transaction ID and Transaction comments that are associated with each Settleup request.

****

1. Display the list of pending settleup requests in each user queue from all users.

****