Stored Procedure #1

```
delimiter //
Create Procedure billSplit(receiptID int)
Begin
SELECT Contributes.UserId as UserId, SUM(Items.Price * Contributes.Percentage) as Paid
FROM Contributes NATURAL JOIN Items
Where Items.ReceiptID = receiptID
GROUP BY Contributes.UserId;
End//
```

Stored Procedure #2

```
Delimiter //
Create Procedure goodSpendingHabit(user int)
Begin
Select UserID
From Accounts Join AverageExpense on Accounts.Income Between
AverageExpense.MinIncome and AverageExpense.MaxIncome Natural Join
        (SELECT UserId, SUM(Items.Price * Contributes.Percentage) as totalPrice FROM
Contributes NATURAL JOIN Items GROUP BY UserId) Split
Where UserID = user and Split.totalPrice < avgExpense
Order By UserID;
End//
Delimiter ;
```

Trigger

```
DELIMITER //
CREATE TRIGGER update_budget AFTER INSERT ON Contributes
FOR EACH ROW
  BEGIN
    DECLARE c VARCHAR(255);
    DECLARE p FLOAT;
    SELECT Category, Price INTO c, p
    FROM Items
    WHERE ItemID = NEW.ItemID;
    IF c IS NOT NULL THEN
      UPDATE Budget SET Spent = Budget.Spent + (New.Percentage) * p
      WHERE Category = c AND UserID = NEW.UserID;
    END IF;
END//
DELIMITER ;
```

Transaction #1

```
DELIMITER //

CREATE PROCEDURE UpdateUserSpending(uid INT)
BEGIN
   DECLARE EXIT HANDLER FOR SQLEXCEPTION
   BEGIN
      ROLLBACK;
   END;

   SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

   START TRANSACTION;

   CREATE TEMPORARY TABLE IF NOT EXISTS UserSpending AS
   SELECT b.Category, SUM(i.Price * c.Percentage) AS TotalSpent
   FROM Accounts a
   JOIN Budget b ON a.UserID = b.UserID
   JOIN Items i ON i.Category = b.Category
   JOIN Receipts r ON i.ReceiptID = r.ReceiptID
   JOIN Contributes c ON c.ItemID = i.ItemID AND c.UserID = a.UserID
   WHERE a.UserID = uid
     AND MONTH(r.PurchaseDate) = MONTH(CURDATE())
     AND YEAR(r.PurchaseDate) = YEAR(CURDATE())
   GROUP BY b.Category;

   COMMIT;
   UPDATE Budget b
   JOIN UserSpending us ON b.Category = us.Category
   SET b.Spent = us.TotalSpent
   WHERE b.UserID = uid;
   COMMIT;
END//

DELIMITER ;
```

Transaction #2

```
DELIMITER //
CREATE PROCEDURE FindOverspending(user INT)
```

```
BEGIN
   DECLARE EXIT HANDLER FOR SQLEXCEPTION
   BEGIN
      ROLLBACK;
   END;
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;

START TRANSACTION;
SELECT B.Category, SUM(I.Price * C.Percentage) AS TotalSpentInCategory, B.Budget
FROM Budget B
JOIN Items I ON B.Category = I.Category
JOIN Receipts R ON I.ReceiptID = R.ReceiptID
JOIN Contributes C ON I.ItemID = C.ItemID
WHERE B.UserID = user
  AND R.UserID = user
  AND C.UserID = user
  AND (R.PurchaseDate BETWEEN DATE_FORMAT(CURRENT_DATE, '%Y-%m-01') AND
CURRENT_DATE)
GROUP BY B.Category, B.Budget
HAVING SUM(I.Price * C.Percentage) > B.Budget;

COMMIT;
END//
```