**DOCUMENTATION: EXPENSE TRACKER**

**Starting Variables:**

* store: (user input) store name, location of transaction
* expense\_cat: (user input) category of the expense – groceries, supplies, splurge; can self-define
* amount: (user input) transaction amount in CAD
* date: (user input) date of the transaction
* shared: (default: False) True if the expense is shared with other people, False if not
* shared\_number: (default: 0) number of people the expense is shared with
* percent\_shared: (default: 0) percent of the expense you spend
* recurring\_charge: (default: False) True if the expense is recurring, False if not
* name: (default: empty) the name(s) of the person/people the expense is shared with
* user\_id: (user input) stores the name of the user

**Library used:**

* mysql.connector
* datetime
* calendar
* PrettyTable: to print out the monthly expense in a table format

**FEATURES:**

**Class/Methods explanation:**

Class: Expense – stores the details of the expense, allows user to call a report based on the specified date, summing amount spent monthly, edit (date or amount) or delete the recurring charge

1. calc\_amount\_per\_person(): calculates the decimal value and therefore the amount shared with the person if the expense is shared
   1. No parameter
   2. Returns amount shared among people based on the decimal value calculated
2. get\_store\_id(): obtain store id from MySQL
   1. No parameter
   2. Returns the store id
3. get\_cat\_id(): obtain expense category id from MySQL
   1. No parameter
   2. Returns the expense category id
4. shared\_person(): stores the details about the expense with people, expense can be shared with more than one person (in MySQL the same expense is stored on different line if there’s more than one name input by the user)
   1. Parameters:
      1. amount\_shared: the amount shared per person
      2. store\_id: the store’s id
5. store\_data(): store user input data about the expense into the database
   1. Parameters:
      1. store: name of store
      2. expense\_cat: expense’s category define by user
      3. amount: full amount paid by user
      4. date: date of the expense
      5. user\_id: name of the user
      6. shared: (default = False) if the expense is shared with another person
      7. shared\_number: (default = 0) number of people that shared the expense
      8. percent\_shared: (default = 0) percent of the expense user spend
      9. recurring\_charge: (default = False) if the expense is a recurring charge
      10. name: a list of name or names of people the expense is shared with
   2. Stores user’s input data in the designated variables
   3. calls both get\_store\_id() and get\_cat\_id() to obtain the store id and category id, respectively
   4. If shared variable is true, calls calc\_amount\_per\_person() to get the amount shared per person and calls shared\_person() method to store the shared expense information. If the shared variable is false, the amount\_shared variable is set to 0.
   5. After obtaining all the information, the store\_data() method stores the expense in MySQL
   6. Returns “Data Stored” to indicate that the data has been stored
6. sum\_monthly(): allow user to see report of monthly expense
   1. Parameters:
      1. start\_range: start date from user’s input
      2. end\_range: end date from user’s input
      3. user\_id: user’s name
      4. leap\_year: (default = False) whether the year is a leap year
   2. New varibles:
      1. start\_date: date at the start of the expense from user’s input
      2. end\_date: date at the end of the expense from user’s input
   3. Calls the stored procedure in MySQL to get the monthly data dependent on range of date and user’s id
   4. Uses the data obtained from MySQL to print out the report:
      1. Loops through results stored in mycursor from MySQL and stores the results in a list (list name: list\_results) in the case of multiple rows
         1. Each item in the list is stored as a tuple
         2. Tuple contains: (date, store’s name, expense’s category, expense’s amount, if the expense is shared, percent shared per person, if the expense is a recurring charge, amount per person, name the expense is shared with)
      2. Set new temporary variables so these variables can be used to print out individual rows and weed out repeated rows (eg. Rows that corresponds to the shared\_expense with others)
         1. total\_amount: sums up the total amount for the time range called by user
         2. total\_shared\_amount: total amount that is shared with others
         3. name\_shared\_person: a list of name(s) of the person shared, used for printing out the total amount shared with each person at the end of report
         4. current/previous\_store: store’s name
         5. current/previous\_amount: expense’s amount
         6. current/previous\_date: expense’s date
         7. current/previous\_category: expense’s category
         8. shared: whether the expense is shared or not
      3. Loops through list\_results and set corresponding variable names with the correct data
         1. Set the “current” variables to the corresponding values
         2. Conditional statement: if the expense is shared, set “shared” variable to ‘Yes’. If not, then set to ‘No’.
         3. Conditional statement: checks if the row is a repeat of the previous row. If it is a repeat, print\_status is set to 0, meaning it won’t print. Else: current\_amount is added to the total\_amount, print\_status is set to 1, and if the expense is a shared amount, the amount is added to the total\_shared\_amount.
         4. Use the same set of values to set the “previous” variables
         5. Conditional statement: checks if the expense is a recurring charge, if True, recurring is set to ‘Yes’. If False, set to ‘No’.
         6. Conditional statement: checks the print\_status. If it is True, then the tuple of values are added to the print\_table.
      4. Prints out the final report
      5. Prints the total amount
      6. Calls the get\_sum\_by\_category with parameter start\_date and end\_date to get all the expense categories within the date range and summation by category. Summation by category is printed
      7. Prints the total amount of the expense shared with others
      8. Calls stored procedure in MySQL to get summation of expense shared with each person, which is then printed
7. get\_sum\_by\_category():
   1. Parameters:
      1. start\_range: start date from user’s input
      2. end\_range: end date from user’s input
   2. New variables:
      1. cat\_dic: stores the results from calling fetch\_expense\_categories()
      2. expense\_by\_cat: a list of tuples: (category id, sum of amount)
   3. Calls fetch\_expense\_categories to obtain all the expense’s categories present within the date range indicated by user
   4. Calls stored procedure in MySQL to get total amount of each category – stores in expense\_by\_cat
   5. Returns expense\_by\_cat
8. change\_amount\_recuring\_charge(): allows the user to edit the charge of a recurring charge on a particular date
   1. Parameters:
      1. date: date of the charge user want to edit
      2. store: name of the store of the recurring charge
      3. old\_amount: the original amount of the recurring charge
      4. new\_amount: the amount the user wants to change to
      5. category: the category of the charge
   2. Calls the stored procedure in MySQL
   3. Returns a status whether the charge was changed or the charge doesn’t exit
9. change\_date\_recurring\_charge(): allows the user to edit the date of the recurring charge for a particular date
   1. Parameters:
      1. old\_date: the original date of the charge
      2. new\_date: the new date the recurring charge would be changed to
      3. store: name of the store of the charge
      4. old\_amount: the original amount of the recurring charge
      5. category: category of the charge
   2. Calls stored procedure in MySQL to edit the date of the recurring charge
   3. Returns a status whether the charge was changed or the charge doesn’t exit
10. delete\_recurring\_charge(): allows user to delete a range of recurring charge
    1. Parameters:
       1. start\_date: start of the range the user want the recurring charge to be deleted
       2. end\_date: end of the range the user want to recurring charge to be deleted
       3. store: name of the store of the charge
       4. amount: amount of the charge
       5. category: category of the charge
    2. Calls stored procedure looking the rows of the charge in MySQL fetching the row id(s)
    3. Calls another stored procedure to delete the rows obtained from the previous query
    4. Returns status that the rows were deleted

MySQL tables:

**Expense**: primary key: ID (auto-generated) main expense table, stores all the expenses inputted by user

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Field** | **Table** | **Type** |
| 1 | ID | expense | INT |
| 2 | STORE\_ID | expense | INT |
| 3 | Amount | expense | FLOAT |
| 4 | Shared\_expense | expense | BOOL |
| 5 | Shared\_number | expense | INT |
| 6 | Percent\_shared | expense | FLOAT |
| 7 | Category\_id | expense | INT |
| 8 | Recurring\_charge | expense | BOOL |
| 9 | Shared\_amount | expense | FLOAT |
| 10 | Date | expense | DATE |
| 11 | USER\_ID | expense | VARCHAR(500) |

**Store**: primary key: Store\_id (auto-generated) stores name of the stores and their id

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Field** | **Table** | **Type** |
| 1 | Store\_name | store | VARCHAR(500) |
| 2 | Store\_id | store | INT |

**Expense\_categories**: primary key: Category\_id (auto-generated) stores the expense categories and their id

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Field** | **Table** | **Type** |
| 1 | Expense\_cat | expense\_categories | VARCHAR(500) |
| 2 | Category\_id | expense\_categories | INT |

Shared\_person: primary key: ID (auto-generated) stores the name

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Field** | **Table** | **Type** |
| 1 | ID | shared\_person | INT |
| 2 | Name | shared\_person | VARCHAR(500) |
| 3 | expense\_date | shared\_person | DATE |
| 4 | amount | shared\_person | FLOAT |
| 5 | store\_id | shared\_person | INT |