**Earmark Veraciously**

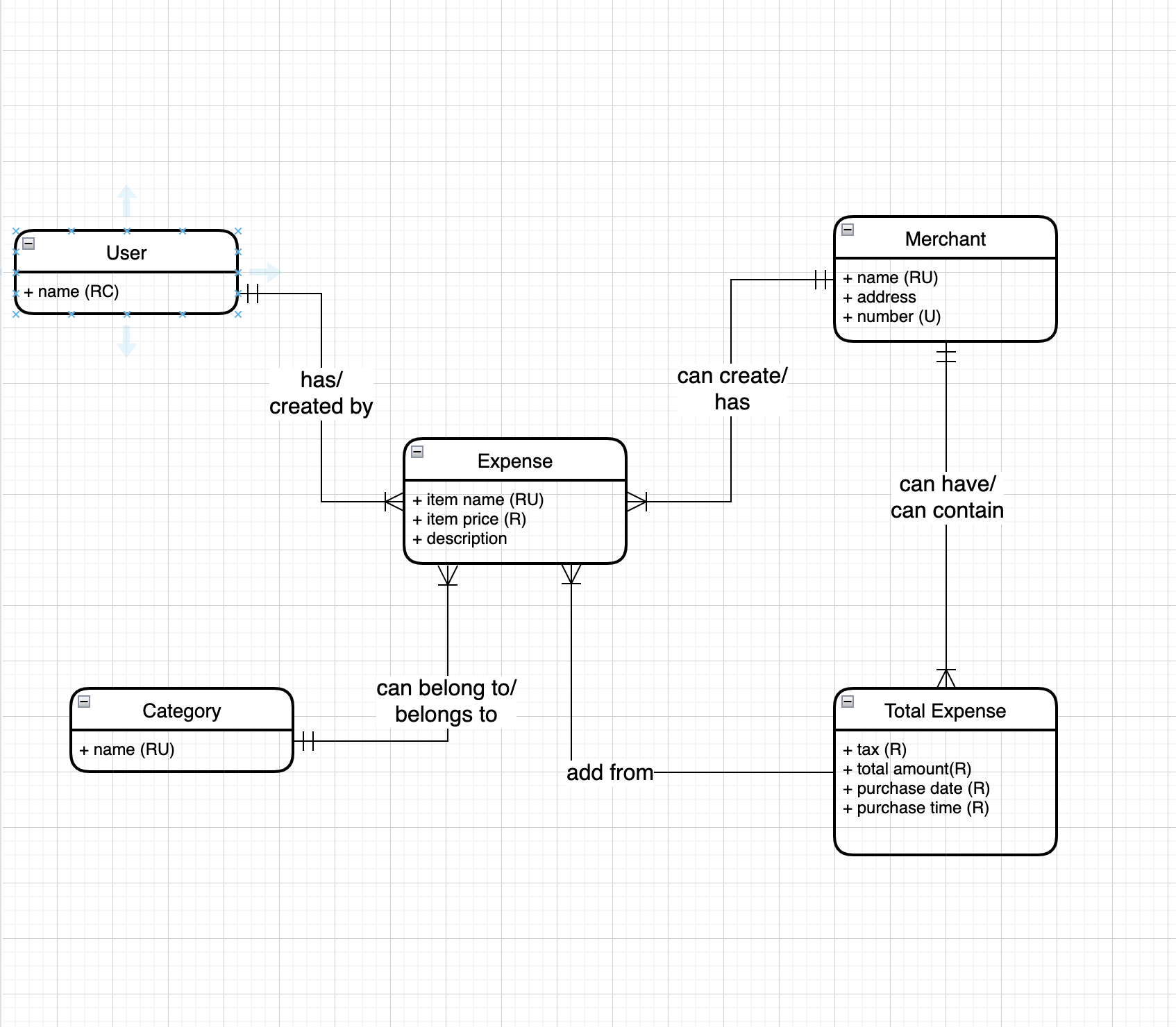
**Topic**

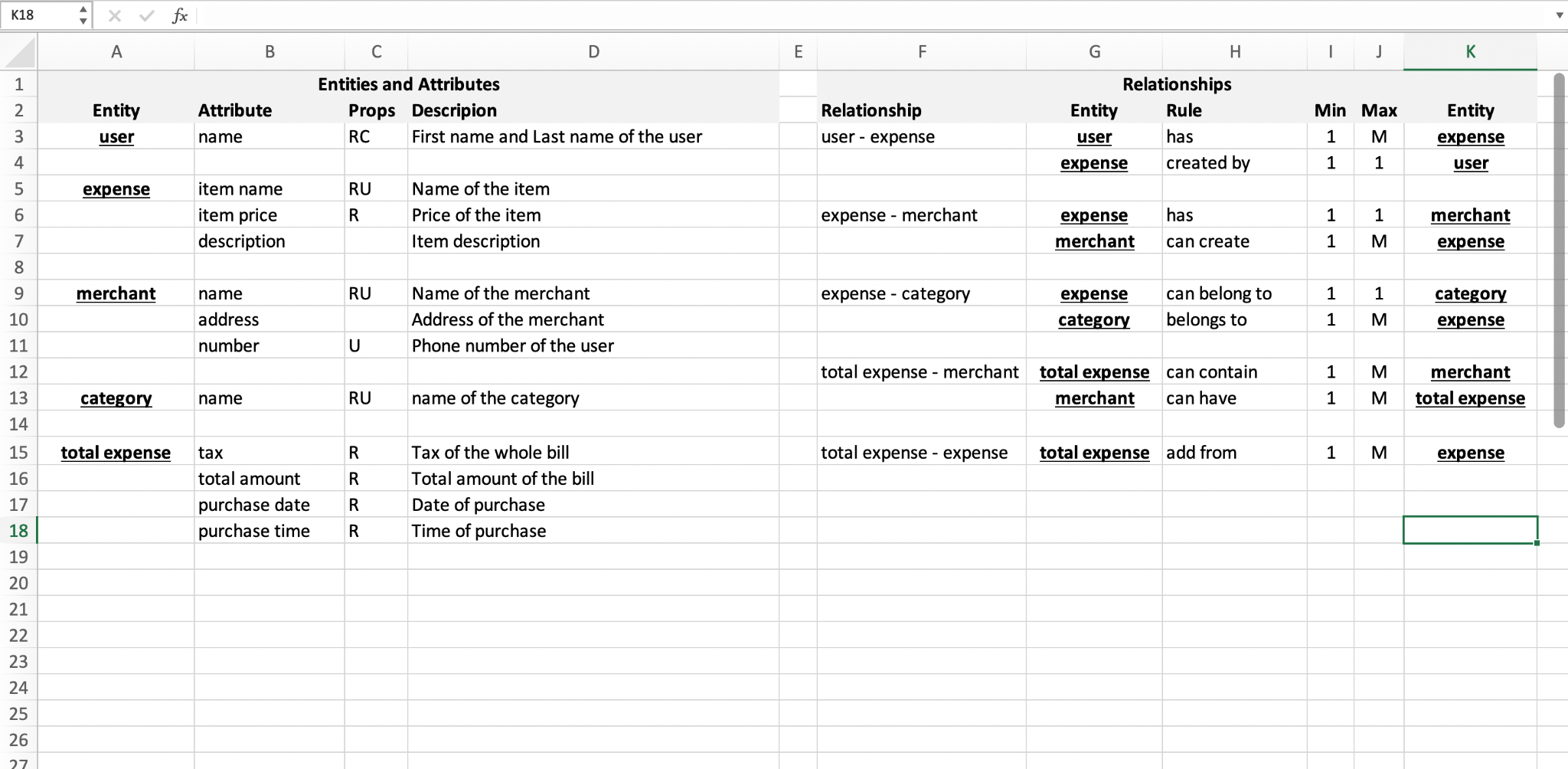
Our business problem deals with scanning receipts using the Azure form Recognizer API to extract the bill details using python, the details include item name, total expense, merchant name, etc. and using python we insert these details into Azure Data Studio and then use these details to help us understand how we can budget our expenses more effectively by using visualization using Power BI.

**Why?**

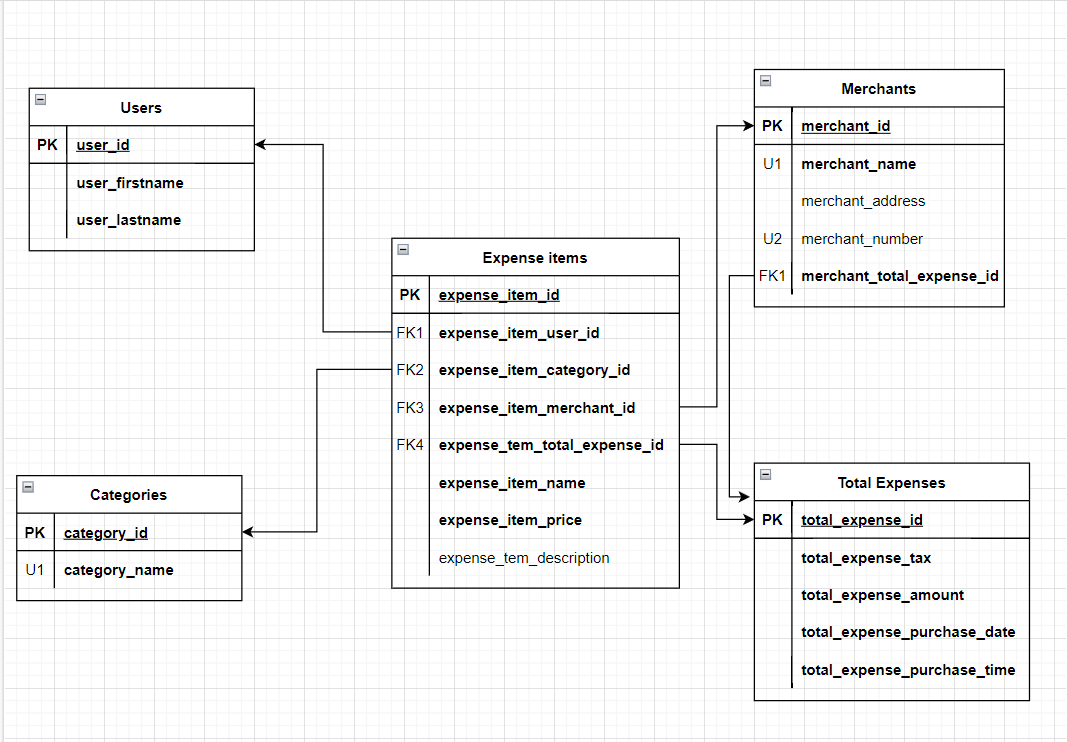
As a student living independently, grocery shopping is an essential requirement when living independently, this business problem addresses the issues that students face while managing multiple expenses in a month as calculating each individual expense becomes time consuming

**Conceptual Data Model Diagram**





**Logical Data Model**

****

**SQL UP/DOWN SCRIPT**

---- MAIN CODE

if not exists(select \* from sys.databases where name='budgets')

create database budgets

GO

use budgets

GO

-- DOWN

if exists(select \* from INFORMATION\_SCHEMA.TABLE\_CONSTRAINTS

where CONSTRAINT\_NAME='fk\_expenses\_exp\_user\_id')

alter table expenses drop constraint fk\_expenses\_exp\_user\_id

if exists(select \* from INFORMATION\_SCHEMA.TABLE\_CONSTRAINTS

where CONSTRAINT\_NAME='fk\_expenses\_exp\_merchant\_id')

alter table expenses drop constraint fk\_expenses\_exp\_merchant\_id

if exists(select \* from INFORMATION\_SCHEMA.TABLE\_CONSTRAINTS

where CONSTRAINT\_NAME='fk\_total\_expenses\_merchant\_id')

alter table total\_expenses drop constraint fk\_total\_expenses\_merchant\_id

if exists(select \* from INFORMATION\_SCHEMA.TABLE\_CONSTRAINTS

where CONSTRAINT\_NAME='fk\_expenses\_exp\_category\_id')

alter table expenses drop constraint fk\_expenses\_exp\_category\_id

drop table if exists expenses

drop table if exists users

drop table if exists merchants

drop table if exists total\_expenses

drop table if exists categories

drop view if exists v\_item\_wise\_expenditure

drop view if exists v\_person\_wise\_expenditure

GO

-- UP Metadata

create table expenses (

exp\_id int identity not null,

exp\_user\_id int not null,

exp\_item\_name VARCHAR(50) not null,

exp\_item\_price money not null,

exp\_merchant\_id int not null,

exp\_merchant\_name VARCHAR(50) not NULL,

exp\_purchase\_date DATE NOT NULL,

exp\_purchase\_time Time NOT NULL,

exp\_category\_id int not null,

exp\_description VARCHAR(50),

CONSTRAINT pk\_expenses\_exp\_id primary key(exp\_id)

)

CREATE TABLE users (

user\_id int identity not null,

user\_firstname varchar(50) not null,

user\_lastname varchar(50) not null,

constraint pk\_users\_user\_id primary key (user\_id)

)

alter table expenses

add constraint fk\_expenses\_exp\_user\_id foreign key (exp\_user\_id)

references users(user\_id)

CREATE table merchants (

merchant\_id int identity not null,

merchant\_name varchar(50) not null,

merchant\_address varchar(50),

merchant\_number varchar(50),

constraint pk\_merchants\_merchant\_id primary key (merchant\_id)

)

alter table expenses

add constraint fk\_expenses\_exp\_merchant\_id foreign key (exp\_merchant\_id)

references merchants(merchant\_id)

create table total\_expenses (

sr\_no int identity not null,

merchant\_id int not null,

tax money not null,

total\_amt money not NULL,

purchase\_date date not null,

purchase\_time time not null,

constraint pk\_total\_expenses\_sr\_no primary key (sr\_no)

)

alter table total\_expenses

add constraint fk\_total\_expenses\_merchant\_id foreign key (merchant\_id)

references merchants(merchant\_id)

CREATE TABLE categories (

category\_id int identity not null,

category\_name varchar(50) not NULL,

constraint pk\_categories\_category\_id primary key (category\_id)

)

alter table expenses

add constraint fk\_expenses\_exp\_category\_id FOREIGN key (exp\_category\_id)

references categories(category\_id)

GO

CREATE VIEW v\_item\_wise\_expenditure AS

with pivot\_source as (

select exp\_item\_name, sum(exp\_item\_price) as total , count(exp\_item\_name) as item\_count from expenses

group by exp\_item\_name

)

select distinct(C.exp\_item\_name), C.item\_count, C. total , category\_name from pivot\_source C join expenses A on C.exp\_item\_name=A.exp\_item\_name

join categories B on

A.exp\_category\_id=B.category\_id

GO

CREATE VIEW v\_person\_wise\_expenditure AS

with pivot\_source as (

select exp\_user\_id, sum(exp\_item\_price) as total from expenses

group by exp\_user\_id

)

select distinct(A.exp\_user\_id), A. total ,

(user\_firstname + ' ' + user\_lastname) as user\_name from

pivot\_source A join users B

on A.exp\_user\_id=B.user\_id

GO

-- UP Data

-- insert into state\_lookup (state\_code) values

-- ('NY'),('NJ'),('CT')

-- Inserted using python script

INSERT INTO users (user\_firstname, user\_lastname) VALUES ('Trishla', 'Jain');

INSERT INTO users (user\_firstname, user\_lastname) VALUES ('Nivedita', 'Ravi'),('Mikhail', 'Pinto');

INSERT INTO categories (category\_name) values ('Food'), ('Fun'), ('Rent'), ('Personal')

-- Verify

select \* from expenses

select \* from users

select \* from merchants

select \* from total\_expenses

select \* from categories

select \* from v\_item\_wise\_expenditure

select \* from v\_person\_wise\_expenditure

**Python Code**

import json

import time

from azure.core.exceptions import ResourceNotFoundError

from azure.core.credentials import AzureKeyCredential

from azure.ai.formrecognizer import FormRecognizerClient

import pyodbc

import datetime

import re

#DBMS connection

server = 'localhost'

database = 'budgets'

username = 'sa'

password = 'SU2orange!'

driver='{ODBC Driver 17 for SQL Server}'

driver = 'SQL Server'

# making global variables to set few values for insert statements

exp\_user\_id = 1

exp\_description = ""

merchant\_id = 0

merchant\_address = ""

merchant\_name = ""

merchant\_number = ""

tax = 0.0

total\_amt = 0.0

purchase\_date = datetime.datetime.now().date()

purchase\_time = datetime.datetime.now().time()

# purchase\_date = ""

# purchase\_time = ""

category\_name = 'Fun'

final\_merchant\_id = 0

category\_id = 0

product\_name = ""

product\_price = 0.0

credentials = json.load(open('C:/Users/trish/Desktop/syracuse/Sem 1/IST.659.M001.FALL21.Data Admin Concepts and Db Mgmt 17224.1221/credentials.json'))

API\_KEY = credentials['API\_KEY']

ENDPOINT = credentials['ENDPOINT']

form\_recognizer\_client = FormRecognizerClient(ENDPOINT, AzureKeyCredential(API\_KEY))

with open(r"C:\Users\trish\Desktop\syracuse\Sem 1\IST.659.M001.FALL21.Data Admin Concepts and Db Mgmt 17224.1221\images\TFun.jpeg", "rb") as f:

poller = form\_recognizer\_client.begin\_recognize\_receipts(f.read())

print(poller.status())

time.sleep(3)

if poller.status() == 'succeeded' or 'InProgress':

#if poller.status() == 'InProgress':

result = poller.result()

for receipt in result:

print(receipt.form\_type)

#print(receipt)

for name, field in receipt.fields.items():

if name == 'Items':

print('Purchase Item')

with open('JSONlog.txt', 'w') as out\_f:

for indx, item in enumerate(field.value):

print('\tItem #{0}'.format(indx + 1), file=out\_f)

for item\_name, item in item.value.items():

print('\t{0}: {1}'.format(item\_name, item.value), file=out\_f)

else:

with open('JSONlog.txt', 'a') as out\_f:

print('{0}: {1}'.format(name, field.value), file=out\_f)

# for insert statements creating all the variables

for name, field in receipt.fields.items():

#print('{0}: {1}'.format(name, field.value))

if name == "MerchantAddress":

merchant\_address = field.value

if name == "MerchantName":

merchant\_name = field.value

if name == "MerchantPhoneNumber":

merchant\_number = field.value

if name == "Tax":

tax = field.value

if name == "Total":

total\_amt = field.value

if name == "TransactionDate":

purchase\_date = field.value

if name == "TransactionTime":

purchase\_time = field.value

print("-============-")

# Inserting into merchants table

if None not in (merchant\_address, merchant\_name, merchant\_number, tax,

total\_amt, purchase\_date, purchase\_time):

with pyodbc.connect('DRIVER='+driver+';SERVER=tcp:'+server+';PORT=5000;DATABASE='+database+';UID='+username+';PWD='+ password) as conn:

with conn.cursor() as cursor: #merchant\_id, exp\_merchant\_name exp\_category\_id,

cursor.execute('''Select merchant\_name \

from merchants where merchant\_name=?''', merchant\_name)

row = cursor.fetchone()

#print(row)

if row:

print("The value is already in the merchants table so we will not do any \

insert into our merchants table")

else:

cursor.execute('''INSERT into merchants (merchant\_name,\

merchant\_address, merchant\_number) \

VALUES (?, ?, ?)''', (merchant\_name, merchant\_address, merchant\_number))

print(f"Inserted values{merchant\_name, merchant\_address, merchant\_number} into merchant table")

# Inserting into total\_expenses

# first seeing if the value exists in totaL\_expense table

print(tax,total\_amt, purchase\_date,purchase\_time)

cursor.execute('''select merchant\_id from total\_expenses

where tax=? and total\_amt=? and purchase\_date=?

and purchase\_time=? ''',

(tax,total\_amt, purchase\_date.strftime('%Y-%m-%d'),

purchase\_time.strftime('%H:%M:%S')))

merchant\_id = cursor.fetchone()

# if merchant id exists dont insert else insert

if merchant\_id:

final\_merchant\_id = merchant\_id[0]

print("The row entry for the total\_expense already exists so not inserting it again")

else:

cursor.execute('''select merchant\_id as id from merchants where merchant\_name=?

''', merchant\_name)

merchant\_id\_merchant\_table = cursor.fetchone()

final\_merchant\_id = merchant\_id\_merchant\_table[0]

cursor.execute('''insert into total\_expenses (merchant\_id,\

tax, total\_amt, purchase\_date, \

purchase\_time) VALUES \

(?,?,?,?,?)''', (final\_merchant\_id,

tax, total\_amt, purchase\_date.strftime('%Y-%m-%d'),

purchase\_time.strftime('%H:%M:%S')))

print(f"Inserted values{final\_merchant\_id, tax, total\_amt, purchase\_date.strftime('%Y-%m-%d'), purchase\_time.strftime('%H:%M:%S')} into total\_expense table")

for name, field in receipt.fields.items():

if name == 'Items':

print("came here")

for indx, item in enumerate(field.value):

# print(item.value.keys())

for item\_name, item in item.value.items():

#print('\t{0}: {1}'.format(item\_name, item.value))

if item.name == 'Name': #str(item.value):

product\_name = item.value

elif item.name == 'TotalPrice':

product\_price = item.value

#check if value is null or not for product price

#to avoid null error while inserting data into expense table

if product\_price:

pass

else:

product\_price = float(re.sub('[^0-9,.]', '', item.value\_data.text)[1:])

# check category id

cursor.execute('''select category\_id as id from categories

where category\_name=?''', category\_name)

category\_id = cursor.fetchone()

final\_category\_id = category\_id[0]

if final\_category\_id:

pass

else:

cursor.execute('''insert into categories (category\_name)

values (?)''', category\_name)

cursor.execute('''select category\_id as id from categories

where category\_name=?''', category\_name)

category\_id = cursor.fetchone()

final\_category\_id = category\_id[0]

# inserting into expenses table

if str(product\_name) and float(product\_price):

#print(f"the name is {product\_name} and price is {product\_price} \

# and item is {item}")

cursor.execute('''select exp\_item\_name from expenses \

where exp\_user\_id=? and exp\_item\_name=? and \

exp\_item\_price=? and \

exp\_purchase\_date=? and exp\_purchase\_time=? and \

exp\_merchant\_id=? and exp\_merchant\_name=? and \

exp\_category\_id=? ''',

(exp\_user\_id, product\_name,

product\_price,purchase\_date.strftime('%Y-%m-%d'),

purchase\_time.strftime('%H:%M:%S')

,final\_merchant\_id, merchant\_name,

final\_category\_id))

value\_exists = cursor.fetchone()

if value\_exists:

print("The row entry for the expenses table already exists so not inserting it again")

else:

cursor.execute(''' INSERT INTO expenses

(exp\_user\_id, exp\_item\_name, exp\_item\_price,

exp\_purchase\_date, exp\_purchase\_time,

exp\_description,

exp\_merchant\_id, exp\_merchant\_name,

exp\_category\_id) VALUES

(?,?,?,?,?,?,?,?,?)''', (exp\_user\_id, product\_name,

product\_price,purchase\_date.strftime('%Y-%m-%d'),

purchase\_time.strftime('%H:%M:%S'), exp\_description

,final\_merchant\_id, merchant\_name,

final\_category\_id))

conn.commit()

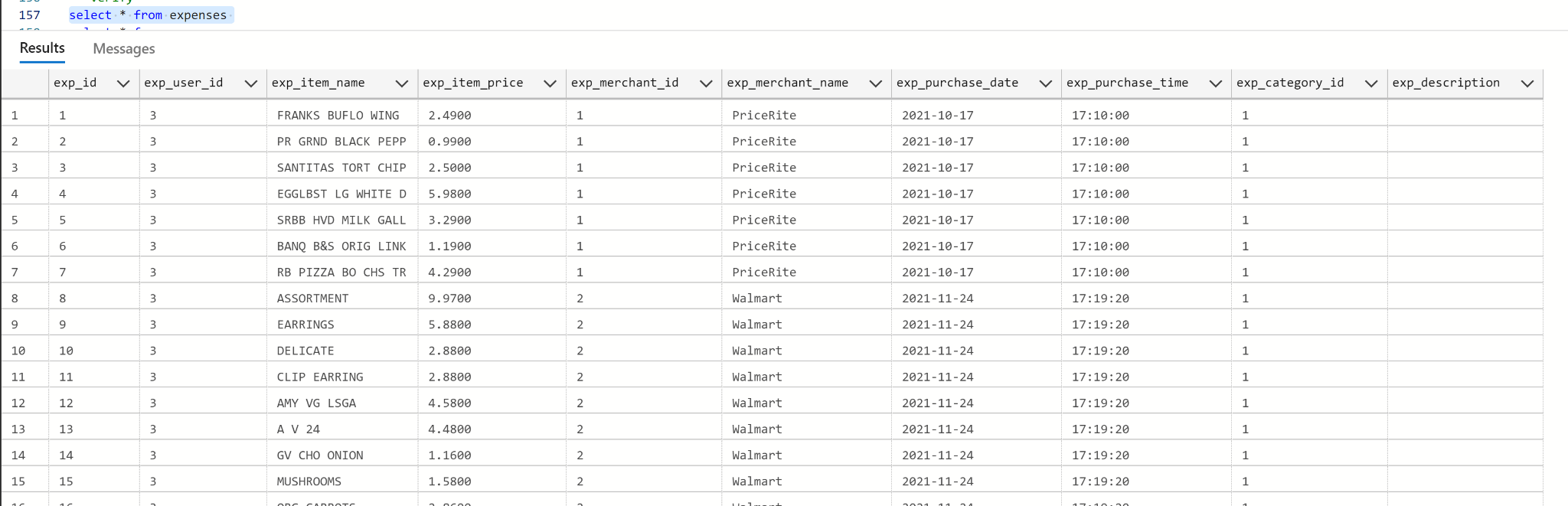
print(f"Inserted values{exp\_user\_id, product\_name,product\_price,purchase\_date.strftime('%Y-%m-%d'), purchase\_time.strftime('%H:%M:%S'), exp\_description,final\_merchant\_id, merchant\_name,final\_category\_id} into expenses table")

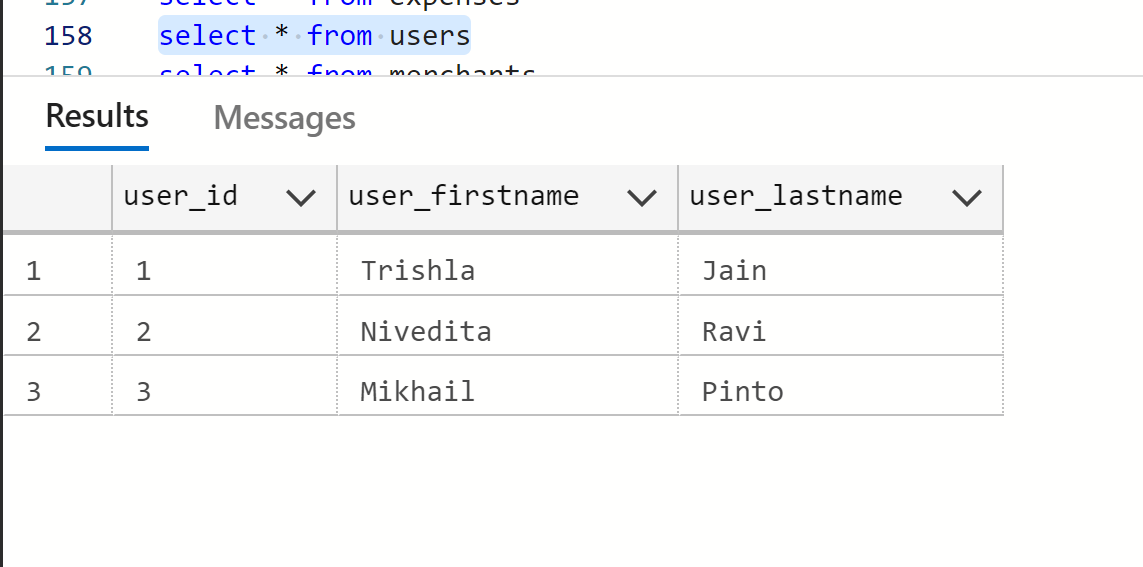
else:

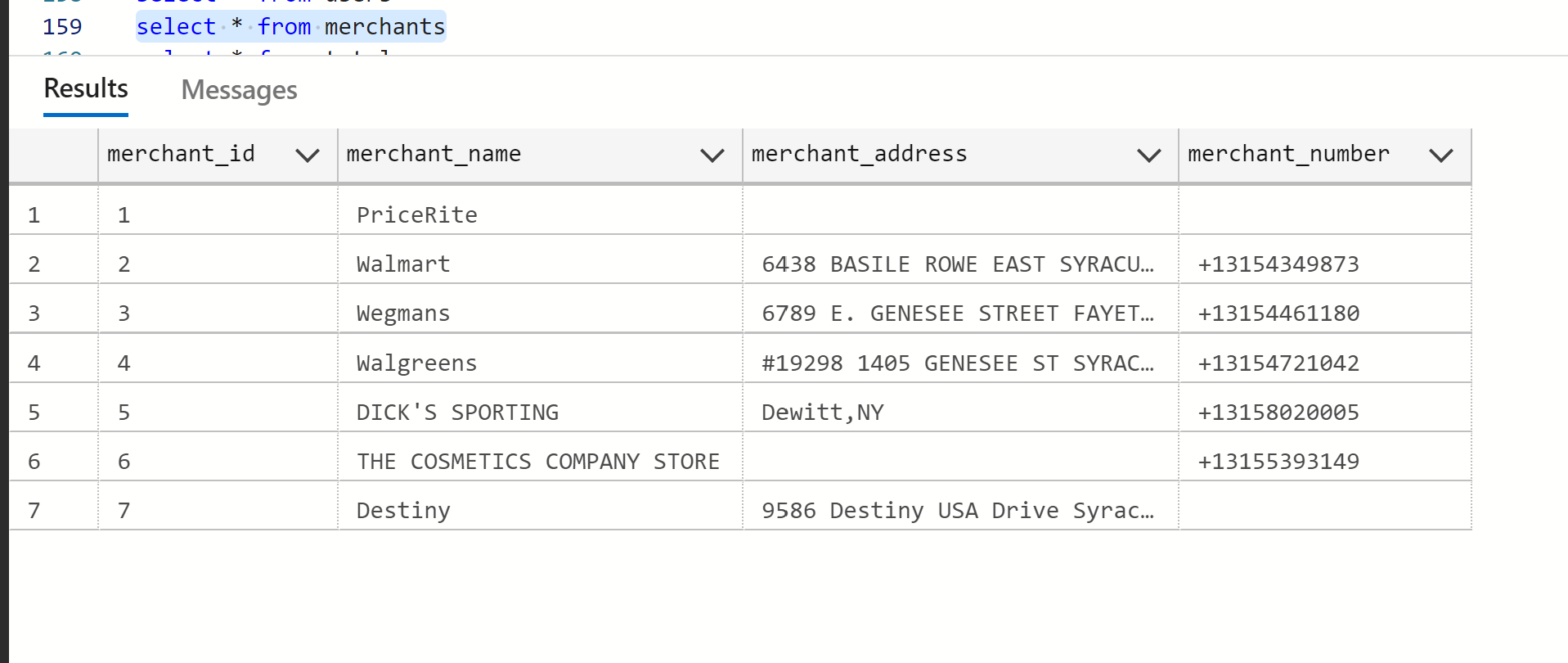
print("not running")

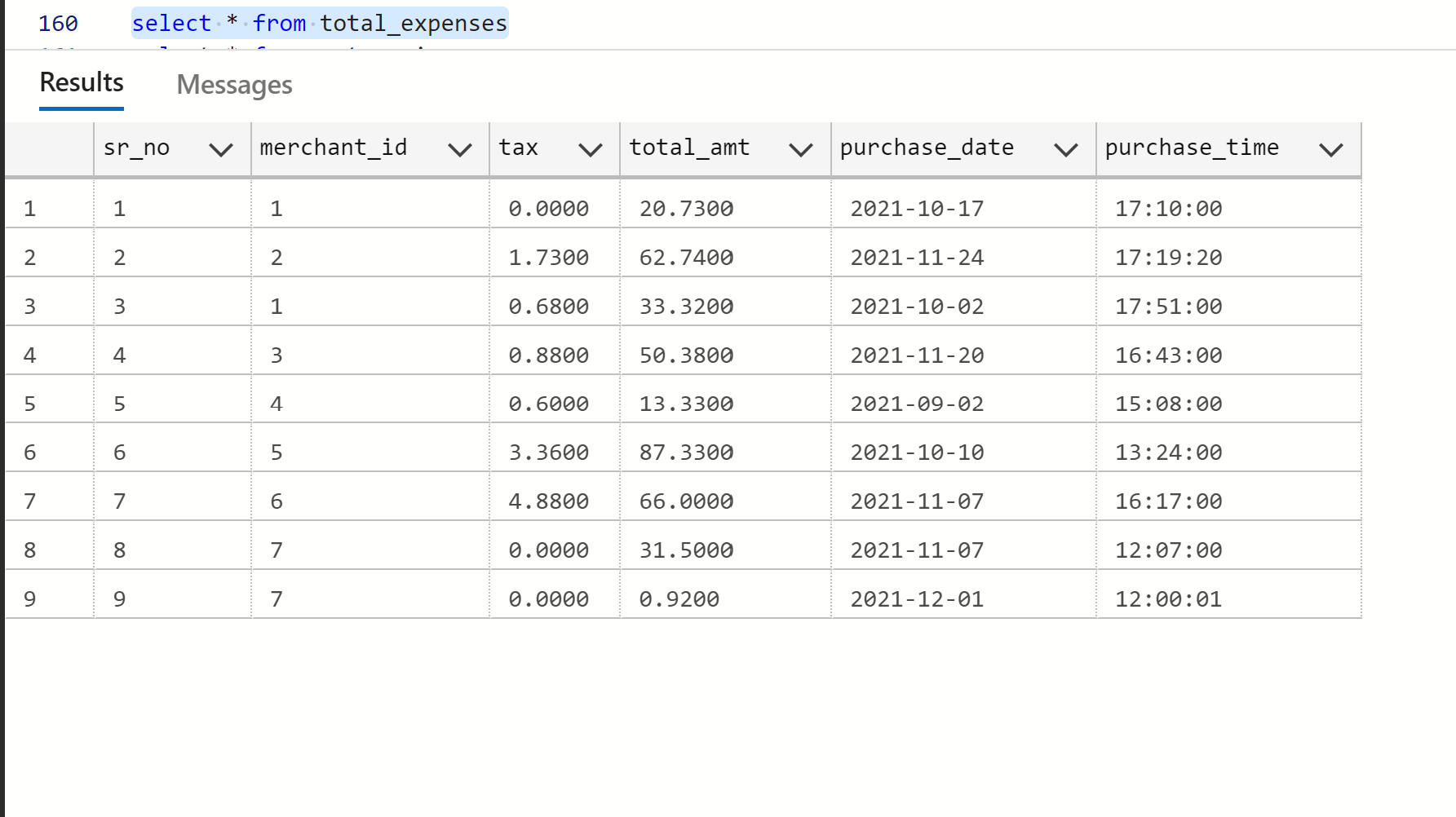
**Table Data**

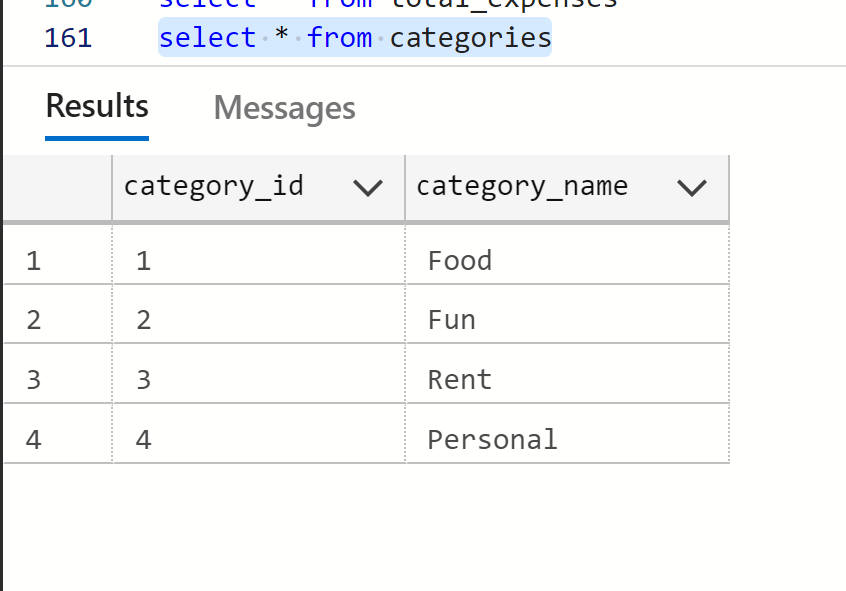
EXPENSES table



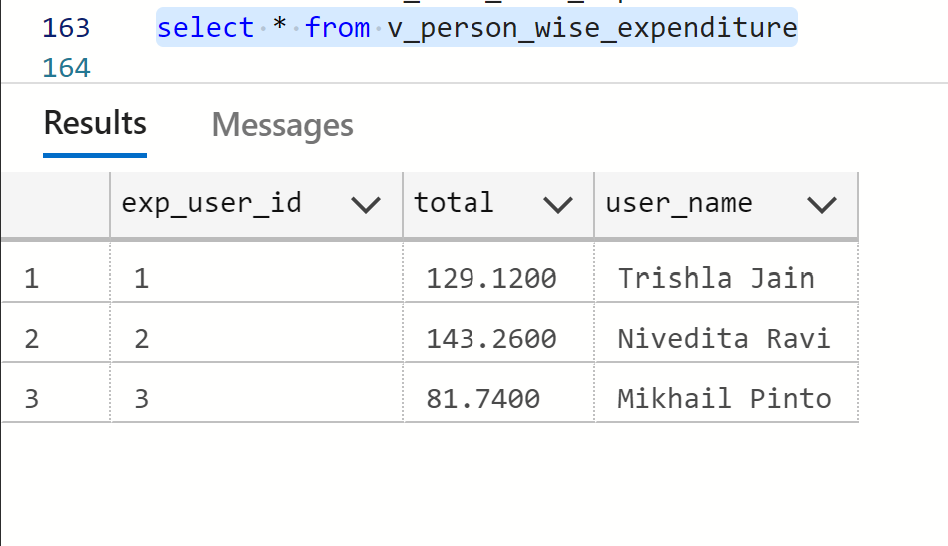
USERS tables  


MERCHANTS table  


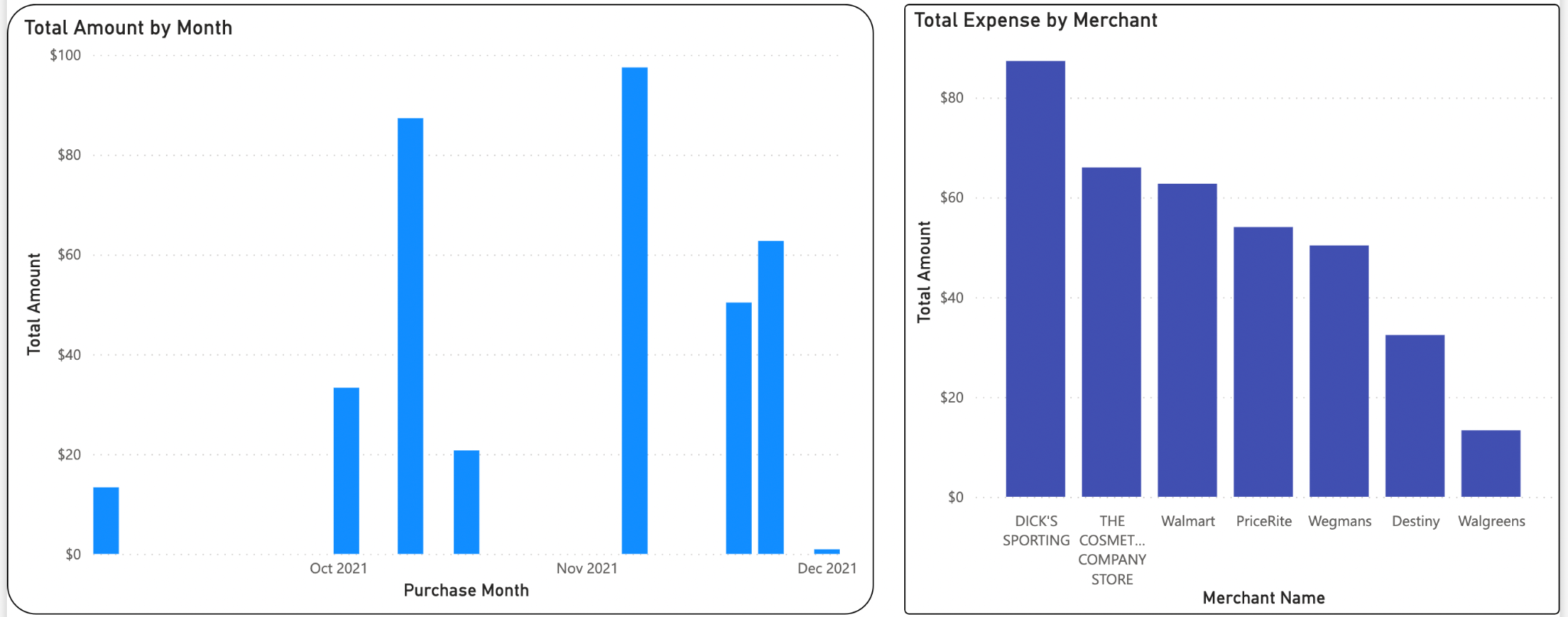
TOTAL\_EXPENSES table  


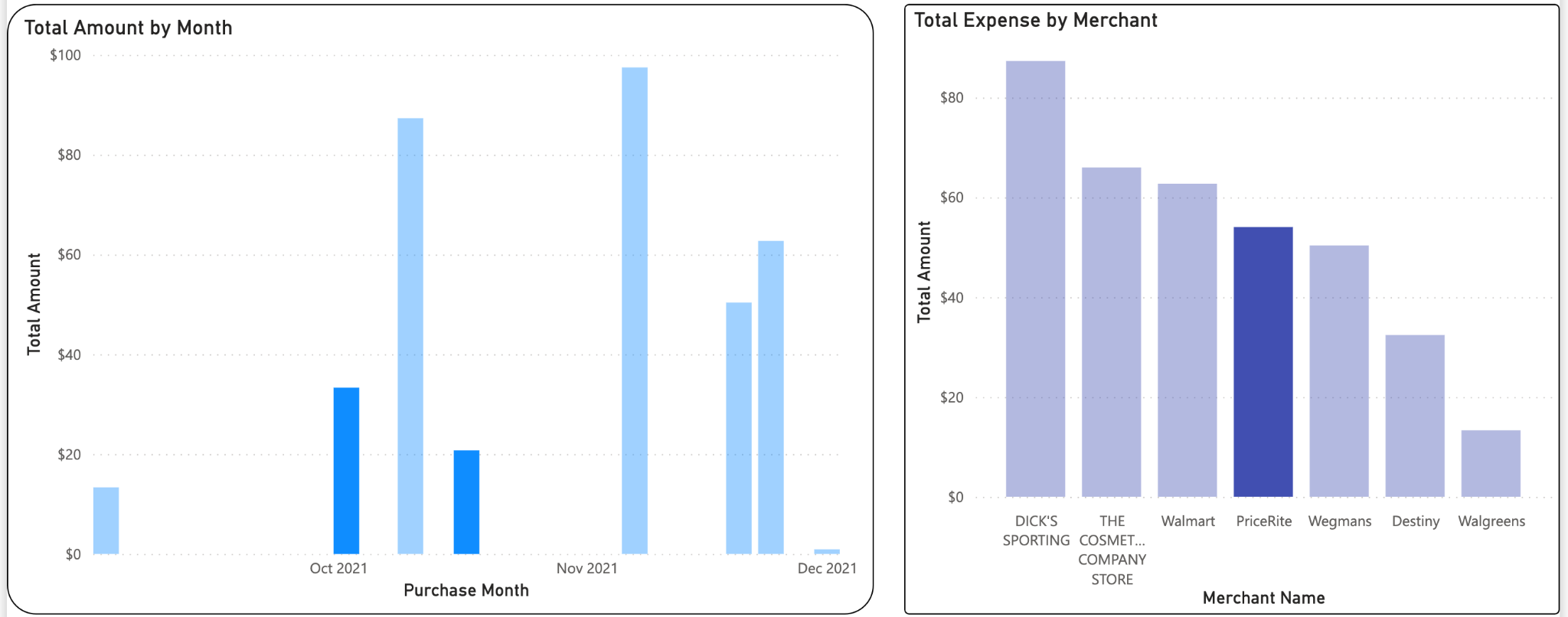
CATEGORIES table  


v\_item\_wise\_expenditure VIEW  


v\_person\_wise\_expenditure VIEW  


**Visualization**





**Time Logs**

|  |  |  |
| --- | --- | --- |
| **Tasks** | **Member’s Worked** | **Date Log** |
| Conceptual Data Model | Mikhail Pinto | Nov 17 2021 |
| Logical Data Model | Nivedita Ravi | Nov 19 2021 |
| Python Script | Trishla Jain, Nivedita Ravi | Nov 25 2021 |
| SQL UP/DOWN Scripts | Trishla Jain | Nov 25 2021 |
| Visualization (Power Bi) | Mikhail Pinto, Trishla Jain | Nov 30 2021 |
| Project Documentation | Mikhail Pinto, Nivedita Ravi, Trishla Jain | 2 Dec 2021 |
| Project Presentation | Mikhail Pinto, Nivedita Ravi, Trishla Jain | 2 Dec 2021 |

**References**

<https://azuresdkdocs.blob.core.windows.net/$web/python/azure-ai-formrecognizer/1.0.0b2/azure.ai.formrecognizer.html>

<https://docs.microsoft.com/en-us/python/api/azure-ai-formrecognizer/azure.ai.formrecognizer.formrecognizerclient?view=azure-python>

<https://learndataanalysis.org/source-code-getting-started-with-azure-form-recognizer-api-in-python/>

<https://hugoworld.wordpress.com/2019/01/06/slow-python-insert-performance-into-microsoft-sql-server/>