**Design – Program**

|  |
| --- |
| **Database** |
| Con  Cur |
| Fetch()  Insert()  Search()  Oldest\_transaction\_date()  Newest\_transaction\_date()  Get\_unique\_stock\_symbols()  Get\_cheapest\_stock()  Get\_expensive\_stock()  Get\_most\_traded\_stock()  fakeData() |

Database class consists of 2 properties and 10 methods

**Class Initialization**

DEFINE CLASS Database:

DEFINE FUNCTION \_\_init\_\_(self, db):

SET self.conn TO sqlite3.connect(db)

SET self.cur TO self.conn.cursor()

self.cur.execute(

"CREATE TABLE IF NOT EXISTS activities (id INTEGER PRIMARY KEY, date text, symbol text, t\_type text, qty text, price text)")

self.conn.commit()

**FETCH**

DEFINE FUNCTION fetch(self):

self.cur.execute("SELECT \* FROM activities")

SET rows TO self.cur.fetchall()

RETURN rows

**INSERT**

DEFINE FUNCTION insert(self, date, symbol, t\_type, qty, price):

self.cur.execute("INSERT INTO activities VALUES (NULL, ?, ?, ?, ?, ?)",

(date, symbol, t\_type, qty, price))

self.conn.commit()

**SEARCH**

DEFINE FUNCTION search(self, date, symbol, t\_type, qty, price):

SET query TO f"SELECT \* FROM activities WHERE date='{date}' AND symbol='{symbol}' AND t\_type='{t\_type}' AND qty='{qty}' AND price='{price}'"

self.cur.execute(query)

self.conn.commit()

SET rows TO self.cur.fetchall()

RETURN rows

**OLDEST TRANSACTION DATE**

DEFINE FUNCTION oldest\_transaction\_date(self):

SET query TO "SELECT MIN(date), symbol FROM activities"

self.cur.execute(query)

self.conn.commit()

SET row TO self.cur.fetchall()

RETURN row

**NEWEST TRANSACTION DATE**

DEFINE FUNCTION newest\_transaction\_date(self):

SET query TO "SELECT MAX(date), symbol FROM activities"

self.cur.execute(query)

self.conn.commit()

SET row TO self.cur.fetchall()

RETURN row

**GET UNIQUE STOCK SYMBOLS**

DEFINE FUNCTION get\_unique\_stock\_symbols(self):

SET query TO "SELECT DISTINCT symbol FROM activities"

self.cur.execute(query)

self.conn.commit()

SET row TO self.cur.fetchall()

RETURN row

**OLDEST TRANSACTION DATE**

DEFINE FUNCTION get\_cheapest\_stock(self):

SET query TO "SELECT symbol, MIN(price) FROM activities"

self.cur.execute(query)

self.conn.commit()

SET row TO self.cur.fetchall()

RETURN row

**EXPENSIVE STOCK**

DEFINE FUNCTION get\_expensive\_stock(self):

SET query TO "SELECT symbol, MAX(price) FROM activities"

self.cur.execute(query)

self.conn.commit()

SET row TO self.cur.fetchall()

RETURN row

**MOST TRADED STOCK**

DEFINE FUNCTION get\_most\_traded\_stock(self):

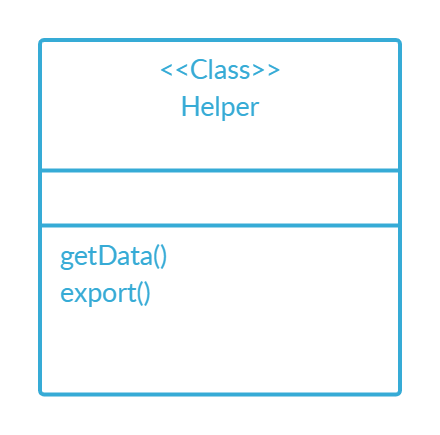
SET query TO "SELECT `symbol`, COUNT(`symbol`) AS `value\_occurrence` FROM `activities` GROUP BY `symbol` ORDER BY `value\_occurrence` DESC LIMIT 1"

self.cur.execute(query)

self.conn.commit()

SET row TO self.cur.fetchall()

RETURN row



**HELPER METHOD TO DISPLAY SUMMARY DETAILS**

DEFINE FUNCTION getData(self):

# oldest transaction date & format text

SET oldest\_transaction TO ", ".join(db.oldest\_transaction\_date()[0][:5])

self.oldest\_transaction\_date\_label.config(

text=f"Oldest Transaction date: {oldest\_transaction}")

# newest transaction date & format text

SET newest\_transaction TO ", ".join(db.newest\_transaction\_date()[0][:5])

self.newest\_transaction\_date\_label.config(

text=f"Newest Transaction date: {newest\_transaction}")

# delete existing

FOR i IN self.treeview1.get\_children():

self.treeview1.delete(i)

# get unique stock symbols & populate

SET symbols\_count TO 0

FOR row IN db.get\_unique\_stock\_symbols():

self.treeview1.insert("", 'end',

values=row)

symbols\_count += 1

self.no\_unique\_stocks\_label.config(

text=f"Number of unique stock symbols: {symbols\_count}")

# Cheapest stock

SET cheapest\_stock TO db.get\_cheapest\_stock()[0]

self.cheapest\_stock\_label.config(

text=f"Cheapest stock: {cheapest\_stock[0]} (Price - ${cheapest\_stock[1]})")

# Expensive stock

SET expensive\_stock TO db.get\_expensive\_stock()[0]

self.expensive\_stock\_label.config(

text=f"Expensive stock: {expensive\_stock[0]} (Price - ${expensive\_stock[1]})")

# Most Traded stock

SET most\_traded\_stock TO db.get\_most\_traded\_stock()[0]

self.most\_traded\_stock\_label.config(

text=f"Most traded stock: {most\_traded\_stock[0]} ({most\_traded\_stock[1]} times)")

**HELPER METHOD TO EXPORT AS TXT**

DEFINE FUNCTION export(self):

SET text\_file TO open("stock\_activity.txt", "w")

FOR row IN db.fetch():

SET line TO ' '.join(str(x) FOR x IN row)

text\_file.write(line + '\n')

text\_file.close()

messagebox.showinfo(

'Success', 'TXT file exported successfully! Check your folder')

**FUNCTION TO POPULATE DATA IN ACTIVITIES TABLE**

DEFINE FUNCTION populate\_list(self):

FOR i IN self.treeview1.get\_children():

self.treeview1.delete(i)

FOR row IN db.fetch():

self.treeview1.insert("", 'end',

values=row)