SPL 201

Assignment 4

Responsible TA: Hagit Bachmat

1 General Description

Mon Café is the largest chain and the best-known café in BGU with coffee-stands in many major building at the campus. The chain has a large number of employees and sells varied products. The chain also has contracts with various suppliers, who supply products for sell. Sells and deliveries of products are registered in an activity table.

Mon Café needs your help to manage its operation by implementing a tool using Python and SQLite.

2 Method and Technical Description

You will build a sqlite3 database that will hold the employee, supplier, product, coffee_stand and activity tables.
The database filename will be **moncafe.db**.
You will have to implement three Python modules: **`initiate.py, action.py`** and **`printdb.py`**.

2.1 The Database Structure

The database moncafe.db has five tables.

- Employees: This table holds information of the employees. The columns are:

  id INTEGER PRIMARY KEY
  name TEXT NOT NULL
  salary REAL NOT NULL
  coffee_stand INTEGER REFERENCES Coffee_stand(id)

- Suppliers: This table holds information of the suppliers. The columns are:

  id INTEGER PRIMARY KEY
  name TEXT NOT NULL
  contact_information TEXT

- Products: This table holds information of the products. The columns are:

  id INTEGER PRIMARY KEY
  description TEXT NOT NULL
  price REAL NOT NULL
  quantity INTEGER NOT NULL

- Coffee_stands: This table holds information of the coffee-stands. The columns are:

  id INTEGER PRIMARY KEY
  location TEXT NOT NULL
  number_of_employees INTEGER

- Activities: This table holds information of all activities of the chain including sells and deliveries

  product_id INTEGER INTEGER REFERENCES Product(id)
  quantity INTEGER NOT NULL
  activator_id INTEGER NOT NULL (either employee id or supplier id)
  date DATE NOT NULL (a formatted string: '`YYYYMMDD'`)

2.2 initiate.py

This module builds the database and inserts the initial data from the configuration file.
When run, it will be given a configuration file as an argument.
For example: python3 initiate.py config.txt
If the database file already exists remove it.
Initiate.py should create a "fresh" database with the tables as specified, parse the configuration file, and store the data given in the configuration file in the database appropriately.
You may assume that the configuration file exists and that the syntax and the data are valid.

2.3 action.py

This module manages the café activities, i.e. sells and delivery, in the campus. When run, it will be given an actions file as an argument. It will perform each action by the order it appears in the file then print the final database and exit.

You may assume that the configuration file exists and that the syntax and the data are valid but you need to check that the quantity of the sold product is enough. If for any reason an action may not be fulfilled do NOTHING!! (Do not print an error message, Do not "sale" part of the quantity, etc...)

2.4 printdb.py

This module prints the database and will be used to check the correctness of your work, therefore, you should follow the instruction to the letter!!!

1. For each table print in a new line the name of the table followed by its records/tuples (each in a row). The tuples should be printed in an ascending order of the primary key except for activities table in which it should be ordered by the date.
   The printing order of the tables is also by ascending order: Activity, Coffee_stand, Employee, Product and Supplier.
2. Print a detailed employees report with the following information:
   Name, Salary, Working location, total sales income. The output should be printed in an ascending order of the employee name. If two or more employees share the same name their inner print order is not important. You should print each employee in a raw using a single space between the fields.
3. Print a detailed activity report with the following information:
   date of activity, item description, quantity name of seller and the name of the supplier. If the activity is a sell the name of the supplier should be 'None', If the activity is supplying the name of the seller should be 'None'.
   The tuples should be printed from the oldest to the newest. If there is no activity do not print. You should use sql SELECT only (with 'join', 'order by' etc…) to complete this.

Note:

1. Use the Python print() function to print your data and tuples (1,3), do not format your output just use the default print behavior. However, If you need to print a list of python's objects use the __str__ function to override the default print of an object since the default print prints only the reference.
   Example:
   ```
   def __str__(self):
           return self.name + " " + str(self.salary)
   ```

   To print a list use the following:
   ```
   def print_a_list(list):
       for item in list:
           print(item)
   ```

2. You may add modules as you need but you may not remove or change the names of these three modules.
3. All three modules should run as a standalone program, i.e. can be run from the command line as the main module
   We will run each of your module from the command line, for example:
   ```
    >  initiate configfile.txt
   >  printdb
   >  action actionfile.txt
   ```

```
> printdb
```

3 Configuration and action Files

3.1 Configuration file

Each line in the configuration file represents either an employee(E), Supplier(S), product(P), coffee standor(C) or an activity(A).
For example:

`“E, 1234, Dana, 45.5, 10”` represents an employee with id 1234, named Dan with 45.5 shekel salary that is positioned at coffee stand 10.

`“S, 56, Strauss, *6860` 49 Hasivim St. Petach Tikva” represents a supplier with id is 56, named Stauss and its contact information is `*6860` 49 Hasivim St. Petach Tikva

`“P, 100, Magnum 8.5”` represents a product with id 100, its description is Magnum, the price is 8.5 shekel and the quantity is 0 by default**.**

`“C, 10,` Building `90, 2”` represents coffee stand with id 10 located at Building 90 with 2 employees.

Note:

1. E, S, P and C at the beginning of each input row define record types and should not be inserted to the database. An example of a configuration file and its output is supplied in the assignment page.
2. Employee IDs and Supplier IDs are unique, meaning an employee ID can not repeat itself as a supplier ID and vice versa (Employee-IDs ∩ Supplier-IDs = ∅).

3.2 Action file

Each line in the action file represents an activity. An activity can be either a sale or supply arrival. When quantity < 0 it is a sale activity, when quantity > 0 it is a supply arrival, quantity=0 is illegal. For example:
"100, 500, 56, 20200110" represents that supplier 56 supplied 500 units of product 100 on 10/jan/2020.
"100, -500, 1234, 20200110" represents that employee 1234 sold 500 units of product 100 on 10/jan/2020.
If current product quantity is less than the quantity in the sale activity the action should be ignored. Do not print any message.

4 Development Environment

- You should use Python 3.5 and `sqlite3` (they are available at the computers in the laboratories).
- You can use any text editor to program the assignment, but make sure your code works when running it from the terminal as specified.

5 Example

Here is an example. Note that you cannot assume the order the information appears in the configuration file. You can only assume validity of each line's syntax, and the validity of the configuration file (no double data, coffee stands exist, no illegal products, etc.)

1. Suppose you are given the following configuration file with the filename config1:

```
C, 11, Bld-90, 1
C, 12, Bld-32, 2
C, 13, Bld-71, 1
S, 101, Strauss, *6860 49 Hasivim St. Petach Tikva
S, 102, Osem, 1-700-70-76-76
E, 1001, Alice, 45.5, 11
E, 1002, Bob, 45.5, 12
E, 1003, Charlie, 45.5, 12
E, 1004, Dan, 45.5, 13
P, 9001, Latte, 6.5
P, 9002, Water, 7
P, 9003, Magnum, 8.5
P, 9004, Salad, 21
P, 9005, Cafe & Mahafe, 11.5
P, 9006, Chocolate, 5
```

And you run “python3 initiate.py config1” then the database will look as follows:

Coffee stands

id	location	#emp
11	'Bld-90'	1
12	'Bld-32'	2
13	'Bld-71'	1

Employees

id	name	salary	coffee-stand
1001	'Alice'	45.5	11
1002	'Bob'	45.5	12
1003	'Charlie'	45.5	12
1004	'Dan'	45.5	13

Products

id	description	price	quantity
9001	'Latte'	6.5	0
9002	'Water'	7	0
9003	'Magnum'	8.5	0
9004	'Salad'	21	0
9005	'Cafe & Mahafe'	11.5	0
9006	'Chocolate'	5	0

Suppliers:

id	name	contact_information
101	'Strauss'	'*6860 49 Hasivim St. Petach Tikva'
102	'Osem'	'1-700-70-76-76'

Activities:

Product_id	quantity	activator_id	date

<empty>

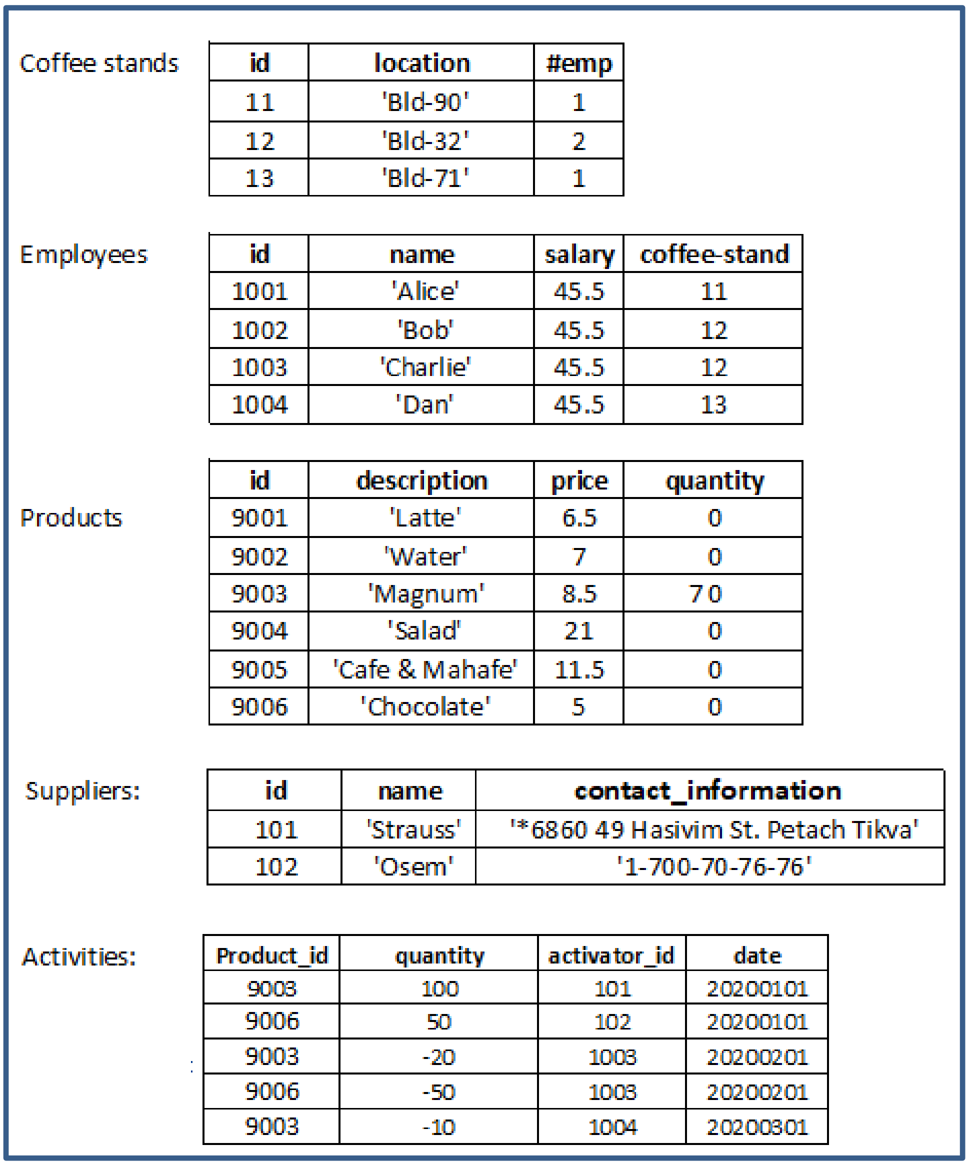
2. “python3 printdb.py” will print the following

```
Activities
Coffee stands
(11, 'Bld-90', 1)
(12, 'Bld-32', 2)
(13, 'Bld-71', 1)
Employees
(1001, 'Alice', 45.5, 11)
(1002, 'Bob', 45.5, 12)
(1003, 'Charlie', 45.5, 12)
(1004, 'Dan', 45.5, 13)
Products
(9001, 'Latte', 6.5, 0)
(9002, 'Water', 7.0, 0)
(9003, 'Magnum', 8.5, 0)
(9004, 'Salad', 21.0, 0)
(9005, 'Cafe & Mahafe', 11.5, 0)
(9006, 'Chocolate', 8.5, 0)
Suppliers
(101, 'Strauss', '*6860 49 Hasivim St. Petach Tikva')
(102, 'Osem', '1-700-70-76-76')
```

3. Suppose now you are given the following action file with the filename action1:

```
9003, 100, 101, 20200101
9006, 50, 102, 20200101
9003, -20,1003, 20200201
9003, -10, 1004, 20200301
9006, -50, 1003, 20200201
```

And you run "python3 action.py action1" then the database will look as follows:



Coffee stands

id	location	#emp
11	'Bld-90'	1
12	'Bld-32'	2
13	'Bld-71'	1

Employees

id	name	salary	coffee-stand
1001	'Alice'	45.5	11
1002	'Bob'	45.5	12
1003	'Charlie'	45.5	12
1004	'Dan'	45.5	13

Products

id	description	price	quantity
9001	'Latte'	6.5	0
9002	'Water'	7	0
9003	'Magnum'	8.5	70
9004	'Salad'	21	0
9005	'Cafe & Mahafe'	11.5	0
9006	'Chocolate'	5	0

Suppliers:

id	name	contact_information
101	'Strauss'	'*6860 49 Hasivim St. Petach Tikva'
102	'Osem'	'1-700-70-76-76'

Activities:

Product_id	quantity	activator_id	date
9003	100	101	20200101
9006	50	102	20200101
9003	-20	1003	20200201
9006	-50	1003	20200201
9003	-10	1004	20200301

4. Now "python3 printdb.py" will print the following

```
Activities
(9003, 100, 101, 20200101)
(9006, 50, 102, 20200101)
(9003, -20, 1003, 20200201)
(9006, -50, 1003, 20200201)
(9003, -10, 1004, 20200301)
Coffee stands
(11, 'Bld-90', 1)
(12, 'Bld-32', 2)
(13, 'Bld-71', 1)
Employees
(1001, 'Alice', 45.5, 11)
(1002, 'Bob', 45.5, 12)
(1003, 'Charlie', 45.5, 12)
(1004, 'Dan', 45.5, 13)
Products
(9001, 'Latte', 6.5, 0)
(9002, 'Water', 7.0, 0)
(9003, 'Magnum', 8.5, 70)
(9004, 'Salad', 21.0, 0)
(9005, 'Cafe & Mahafe', 11.5, 0)
(9006, 'Chocolate', 8.5, 0)
Suppliers
(101, 'Strauss', '*6860 49 Hasivim St. Petach Tikva')
(102, 'Osem', '1-700-70-76-76')

Employees report
Alice 45.5 Bld-90 0
Bob 45.5 Bld-32 0
Charlie 45.5 Bld-32 595.0
Dan 45.5 Bld-71 85.0

Activities
(20200101, 'Magnum', 100, None, 'Strauss')
(20200101, 'Chocolate', 50, None, 'Osem')
(20200201, 'Magnum', -20, 'Charlie', None)
(20200201, 'Chocolate', -50, 'Charlie', None)
(20200301, 'Magnum', -10, 'Dan', None)
```

6 Very (!) Important Notes

1. We will test your modules together, and independently. Independently means that we will, for example, use our own database but use your modules, or put one module of our own and use your other module. Therefore, make sure your database has the structure we specified exactly, and that the behavior of each module is precisely as specified. Also, make sure that the database filename is moncafe.db. Failing to follow these guidelines will cause tests to fail, and your grade will suffer accordingly.

2. To save you time, you may assume the validity of input. For example, an employee will not be assigned to a non-existent coffee stand. However, you may not assume that the quantity of a sell activity is enough.

3. We emphasize again, please make sure everything is as described, otherwise you will lose critical points from your grade! We care about your success as much as you do. Make sure the database filename is moncafe.db and NOT Moncafe.db and NOT moncafe.DB or any other permutation. Make sure your table and column names are also exactly as described. If you are not sure about something, then feel free to ask in the forum!

7 Bonus

Implement your assignment using persistence layer (DAO, DTO, Repository) as will be shown next week (the last week of the semester) and get 20 points bonus.

8 Submission

- The submission is done in pairs only. You cannot submit in a group larger than two, or a group smaller than two.
- You must submit one file with all your code. The file should be named id1_id2.tar.gz. This file should include the following files:
    - initiate.py
    - action.py
    - printdb.py
    - Any extra files that you need (repository, DTOs, DAOs, etc…)