numOfMonth, вычисляет разницу в месяцах между сегодняшней датой и датой транзакции, таким образом мы узнаем, когда последний раз покупатель посещал заведение. Ее так же вынес в локальную переменную для разгрузки запроса.

Далее идут запросы sql, вообще их можно было все написать в один, но в все же разбил для упрощения.

Dataset<Row> barCustomer – находим id и координаты пользователя, если ему 35 или он младше (возраст задается в переменной requiredAgeOfCustomer), так же дата посещения – последние два месяца, частота посещения более пяти раз, ну и mcc баров так же задаем в переменной BAR\_MCC. Все это при потраченных за этот период деньгах более 2000, SUM(amount)> 2000.

Второй запрос: посетители аптек, Dataset<Row> PharmacyCustomers

Фильтруем посетителей старше 35 лет, посещавших аптеки (PHARMACY\_MCC) последние полгода(6 месяцев). Получаем таблицу, в которой id, координаты, и самое главное – numberOfVisits – количество посещений, ведь нам нужно выбрать наиболее посещаемые. Здесь я воспользовался подзапросом, сохраненным в стринге helpingQuery.

Dataset<Row> PharmacyCustomers – в ней мы делаем джоин сами на себя. В итоге выбираем те варианты, где совпадают айдишники и NumberOfVisits c MAX(), таким образом выберутся данные с максимальным количеством посещений.

Наконец, финал. ResultTable – таблица, получаемая слиянием pharmacyCustomers и barCustomer. Это и есть наш конечный результат, те данные, которые и нужно было получить. Можно сделать все в одном запросе, и так было бы правильнее, однако тут еще нужно подумать, как бы не ухудшить читаемость кода. saveToCsv – сохраняем нашу таблицу в файл, в метод передается таблица и путь к файлу, думаю, тут можно было еще спрятать логику, оставив только путь.

У датасета есть метод collectAsList(), которым я и воспользовался. Он возвращает нам список обьектов Row, данные из которых можно получать простым вызывом метода row.get(i);

Инициализируем fileWriter, так как название столбцов и их количество заранее известно, я их захардкодил в хидере как client\_id,x,y. Далее просто бежим по коллекции row, вызывая у каждого ее элемента get(i). Словом, все как обычный список. Наконец, у нас есть файл с результатом, который располагается в папке data. В конце удаляем временные файлы, они нам больше не нужны. Смущает хардкод, лучше было бы как то передать сюда путь.

Парсинг csv файла в таком виде, какой он сейчас есть, не идеальное решение, возможно, все же есть методы, позволяющие загружать в датасет csv файл и задавать ему разделитель – точку с запятой, что позволило бы сэкономить много ресурсов.

Далее, объединение sql запросов в один, без отдельного запроса для посетителей бара.

Ну и еще хочется попробовать улучшить производительность при помощи многопоточности.

Вроде бы как все написал, что то может и забыл, если что, спрашивайте :)