Индивидуальный проект, автор: Егор Рудиков.
Описание алгоритма работы и деталей реализации.

15 декабря 2022 года

Список изменений:

- Запрос SQL по 4-му виду мошенничества, учитывающий больше случаев.

- Загрузка данных из локальных файлов. В первоначальной версии проекта выяснилось, что данные из файлов загружаются очень медленно при загрузке по сети. Например, загрузка файла транзакций по одному дню занимает около 10-15 минут. При запуске скрипта непосредственно на сервере все происходит быстро в течение нескольких секунд. Аналогичная проблема есть и у других сокурсников. На лекциях в качестве средства вставки нескольких записей использовалась команда cursor.executemany, но в ходе изучения документации выяснилось, что в psycopg2 есть специальные средства для потоковой загрузки однородных данных, а именно: psycopg2.extras.execute_batch. В результате произведенных изменений время загрузки данных существенно снизилось. Применительно к ранее озвученному примеру с файлом транзакций за день загрузка теперь занимает не более 10 секунд с локального ПК.

- Файлы backup дополнительно архивируются в формате zip.

От чего пришлось отказаться:

- Многопоточный режим работы. Из-за GIL одновременная работа нескольких потоков невозможна, а использование многопроцессорного режима затруднено в части синхронизации транзакций в подпроцессах и передачи данных. В psycopg2 есть поддержка асинхронного режима работы, однако ее реализация в данном драйвере накладывает существенные ограничения: используется автокоммит и в любой момент времени может исполняться только один запрос. Конечно, есть более прогрессивные средства для работы с БД, в которых есть полноценная поддержка asyncio (тот же psycopg3), однако согласно условиям задачи на проект наложены ограничения (нельзя ни поставить в целевой системе требуемые пакеты, ни предоставить проект в виде изолированного окружения со своим набором пакетов).

- Отправка отчетов по почте. Применительно к общедоступным серверам smtp не удалось добиться стабильной работы указанной функциональности как в части отправки, так и получения (антиспам-фильтр). Полагаю, что в рамках корпоративной среды данная проблема была бы решена.

04 декабря 2022 года

В соответствии с ТЗ код запуска расположен в файле main.py (исполняемый). Поддерживается запуск с параметрами командной строки: директория с файлами для загрузки (--indir), архивная директория (--outdir), файл конфигурации подключения к базам данных (--dbconf) и уровень логгирования (--log). Все аргументы командной строки имеют значения по умолчанию: для директории с входными данными - директория запуска скрипта, для архивной директории - поддиректория archive, конфигурация подключений к базам данных - файл defailt_dbconf.json в поддиректории py_scripts, уровень логгирования - INFO.

Формат json для хранения настроек подключения к БД выбран из-за его распространенности и наличия встроенных средств для работы с json в стандартной библиотеке Python. Также можно указывать дополнительные параметры подключения БД, помимо типовых, и при создании соединения они тоже будут учитываться.

В скрипте имеется 3 уровня логгирования: INFO (по умолчанию) - выводится информация об основных шагах алгоритма, WARNING - некритичные ситуации, которые требуют внимания (например, попытка загрузить файлы данных, которые уже загружены) и ERROR - при возникновении исключения в работе программы выводится

описание ошибки и stacktrace. Как обычно, логи в Python по умолчанию выводятся в stderr.

После успешной инициализации входных параметров запускается скрипт создания необходимых для работы таблиц в БД edu (если ранее они не были созданы). Таблицы создаются только для обработки загрузки из файлов и формирования отчета о мошенничестве + осуществляется базовая инициализация таблицы метаданных. Таблицы, связанные с загрузкой сведений из bank в edu создаются отдельно при кодогенерации. Дополнительно из таргетной СУБД запрашивается и сохраняется текущее время сервера, которое будет использоваться в дальнейшем вместо now() там, где это необходимо.

Следующим шагом производится загрузка данных из bank в edu таблиц accounts, cards и clients с приведением данных к формату SCD2 (таблицы в bank выполнены в формате SCD1). Учитывая, что данная операция является однотипной для всех 3-х таблиц, логика ее проведения выведена в функцию convert_scd1_to_scd2, которая принимает в качестве параметров наименование таблицы и ее ключа, словарь с переименованными в таргете полями, а также объекты соединений для двух БД и параметр текущего времени.

В функции convert_scd1_to_scd2 производится поиск в метаданных БД bank соответствующей таблицы и извлекаются наименование ее атрибутов с указанием типов и ограничений типа (максимальная длина для символьных типов). На основе полученной информации создаются (в случае необходимости) соответствующие таблицы стейджинга и таргета в БД edu - строковые типы приводятся к типу varchar (как требуется в ТЗ), вместо полей, характерных для формы SCD1 (create_dt, update_dt) создаются поля для SCD2 (effective_from, effective_to, deleted_flg). Далее извлекаются метаданные о последней загрузке соответствующей таблицы и с учетом этой информации (инкрементально) извлекается информация из БД bank, затем осуществляется стандартная процедура загрузки в формат SCD2, где в качестве effective_from выступает create_dt (update_dt) соответствующей таблицы SCD1.

Следующим этапом выполняется загрузка данных о транзакциях, заблокированных паспортах и терминалах из соответствующих файлов. Допускается пакетная загрузка, когда последовательно грузятся данные за несколько дней. Перед непосредственно загрузкой в БД происходит формирование перечня файлов для загрузки:

- сканирование директории входных данных, фильтрация по шаблону и проверка цифрового формата даты;
- группировка по дням;
- обрезка отсортированного списка дней для загрузки (слева - если информация за соответствующие дни уже загружена или имеется загрузка с более поздней датой, справа - если представлен неполный комплект файлов, отбрасываются данный и последующие дни).

Далее производится загрузка транзакций и заблокированных паспортов в фактовые таблицы, а списка терминалов — в SCD2. Теоретически, терминал может быть выведен из строя или перемещен по другому адресу (по крайней мере, среди представленных данных имеется одна удаленная запись и несколько случаев, когда адрес размещения терминала был изменен).

Необходимо отметить, что для всех операций загрузки в SCD2 в скрипте сравнение при вставке новых записей производится по всем значащим полям, а не только по ключу.

По окончании загрузки информация о загруженном дне добавляется в таблицу метаданных и стейджинговую таблицу формирования отчета rdkv_stg_rep_fraud (данная таблица содержит единственный атрибут даты, реализована в рамках поддержки возможности загрузки файлов и пакетного формирования витрины за несколько дней).

После завершения загрузки за день происходит фиксирование транзакции и перемещение соответствующих файлов в архив. Далее запускается процедура загрузки за последующие дни, если файлы таковых имеются.

Последним этапом является формирование отчетности для всех 4-х категорий мошенничества. Отчет последовательно формируется для дат, указанных в rdkv_stg_rep_fraud. Учитывая, что согласно ТЗ «витрина строится накоплением», отчет формируется по конкретной дате, затем происходит объединение с данными за предыдущие дни.

БД bank используется в режиме autocommit, так как операции изменения данных применительно к ней не осуществляются. Применительно к БД edu существует несколько точек, в которых происходит фиксирование транзакций:

- при создании таблиц и первичном заполнении таблицы метаданных.
- для каждой таблицы из bank по завершении ее загрузки в edu в формате SCD2.
- для каждого дня по завершении загрузки из всех 3-х файлов. Данные за день либо загружаются полностью, либо не загружаются вовсе.
- для каждого дня построения отчета.

Возможные последующие улучшения и финтифлюшки, если все работает правильно и работа сдана без ошибок (маловероятно, но «блаженны верующие»)):

- Основное время при запуске скрипта занимает загрузка данных из файлов transactions (около 10 минут для каждого дня на моем ПК, на сервере все быстро)). Теоретически возможно распараллелить эту операцию: построчно разбить файл на батчи, которые параллельно будут обрабатывать несколько воркеров, каждый с отдельным соединением, + отдельные воркеры, обрабатывающие список паспортов и терминалов. Затем производится опрос всех воркеров, и, в случае успешного завершения каждого из них, фиксируются транзакции принадлежащих воркерам соединений.
- Реализация возможности отправки дневного отчета о мошеннических операциях по указанному email.
- Архивирование загруженных данных (3 .backup файла в один архив, соответствующий дате загрузки).