

OTUS-Postgre-DBA-2024-09

Автоматизация ETL- процессов в PostgreSQL



**Меня хорошо видно
& слышно?**



Защита проекта

Тема: Автоматизация ETL-процессов в PostgreSQL



Александр Грешнов

ООО «Эрманн»

Отдел управления данными

План защиты

Цель и задачи проекта

Какие технологии использовались

Что получилось

Выводы

Вопросы и рекомендации

Цель и задачи проекта

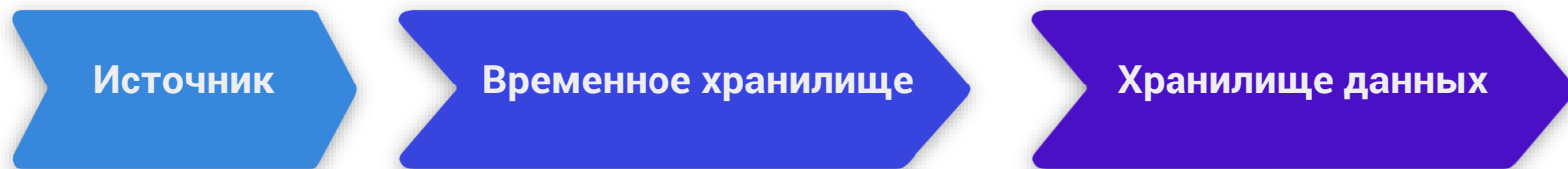
Цель проекта:

реализовать автоматическую регулярную загрузку данных из внешнего источника в базу данных PostgreSQL

1. Настроить доступ к внешнему источнику
2. Определить структуру данных источника
3. Создать инфраструктуру для хранения загруженных данных
4. Реализовать алгоритм получения данных из источника и сохранения в БД
5. Настроить автоматическое выполнение загрузки



ETL-Процесс



CSV-файлы,
размещённые на
сетевом диске

Данные в БД, загруженные
из CSV-файлов

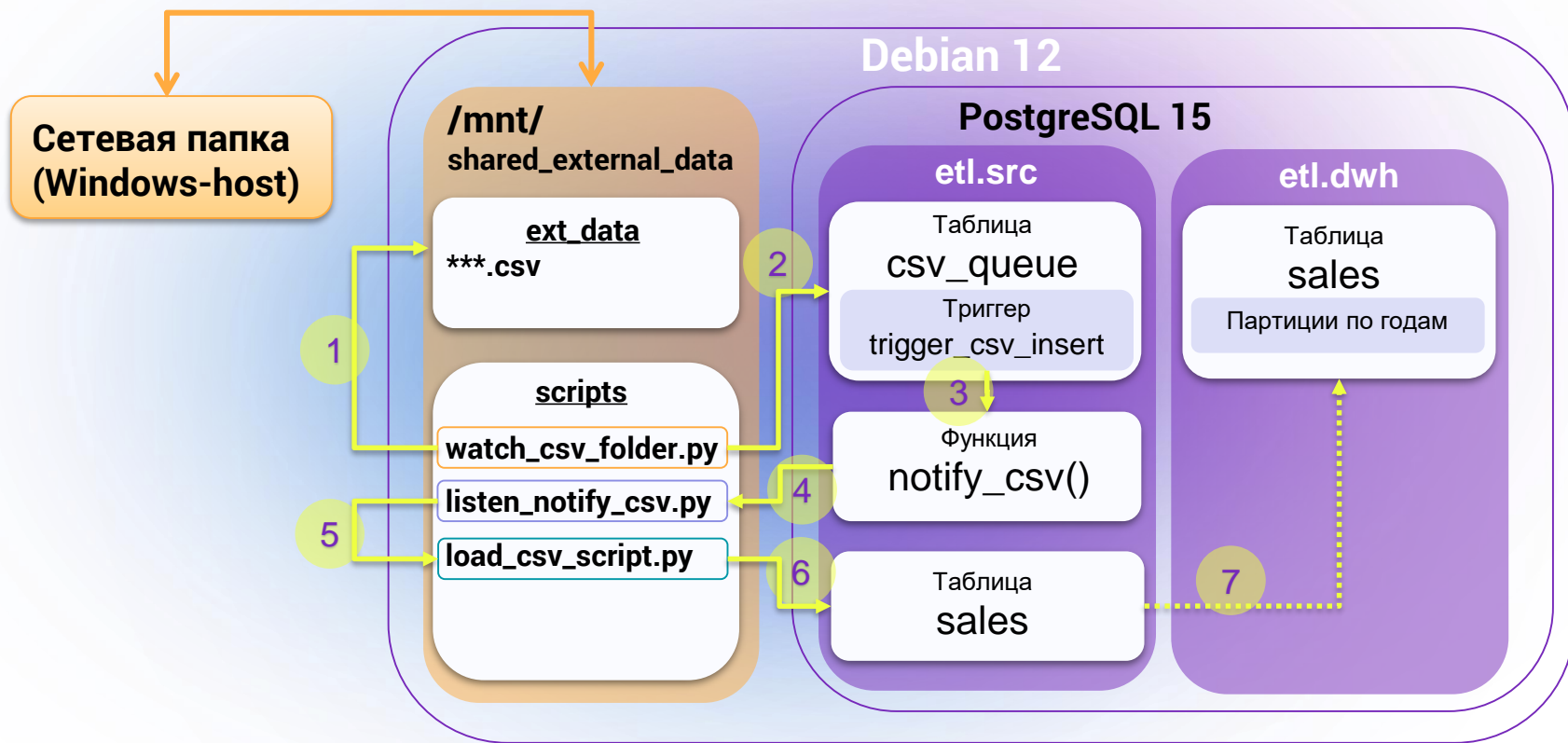
Обработанные данные в БД,
готовые для использования

Какие технологии использовались

1. **cifs-util** – Монтирование сетевого диска
2. **PostgreSQL 15** – СУБД
3. **Python** – Автоматизация процессов:
 - **watchdog** – Мониторинг файловой системы
 - **psycopg2** – Драйвер для работы с PostgreSQL
4. **LISTEN/NOTIFY** – Механизм событий в PostgreSQL
5. **subprocess** – Запуск внешних процессов
6. **cron** – планировщик задач



Что получилось



Пример csv-файла

	A	B	C	D	E	F	G	H
1	date	shop_id	shop_address	barcode	product_name	qty	price	
2	01.02.2025	1	Tests address 1	4600000000001	Test product 1	10	50.35	
3	02.02.2025	2	Tests address 2	4600000000002	Test product 2	20	55.35	
4	03.02.2025	3	Tests address 3	4600000000003	Test product 3	30	60.35	
5	04.02.2025	4	Tests address 4	4600000000004	Test product 4	40	65.35	
6	05.02.2025	5	Tests address 5	4600000000005	Test product 5	50	70.35	
7	06.02.2025	6	Tests address 6	4600000000006	Test product 6	60	75.35	
8	07.02.2025	7	Tests address 7	4600000000007	Test product 7	70	80.35	
9	08.02.2025	8	Tests address 8	4600000000008	Test product 8	80	85.35	
10	09.02.2025	9	Tests address 9	4600000000009	Test product 9	90	90.35	
11	10.02.2025	10	Tests address 10	4600000000010	Test product 10	100	95.35	
12	11.02.2025	11	Tests address 11	4600000000011	Test product 11	110	100.35	
13	12.02.2025	12	Tests address 12	4600000000012	Test product 12	120	105.35	
14	13.02.2025	13	Tests address 13	4600000000013	Test product 13	130	110.35	
15	14.02.2025	14	Tests address 14	4600000000014	Test product 14	140	115.35	
16	15.02.2025	15	Tests address 15	4600000000015	Test product 15	150	120.35	
17	16.02.2025	16	Tests address 16	4600000000016	Test product 16	160	125.35	
18	17.02.2025	17	Tests address 17	4600000000017	Test product 17	170	130.35	
19	18.02.2025	18	Tests address 18	4600000000018	Test product 18	180	135.35	
20	19.02.2025	19	Tests address 19	4600000000019	Test product 19	190	140.35	
21	20.02.2025	20	Tests address 20	4600000000020	Test product 20	200	145.35	
22	21.02.2025	21	Tests address 21	4600000000021	Test product 21	210	150.35	
23	22.02.2025	22	Tests address 22	4600000000022	Test product 22	220	155.35	
24	23.02.2025	23	Tests address 23	4600000000023	Test product 23	230	160.35	
25	24.02.2025	24	Tests address 24	4600000000024	Test product 24	240	165.35	
26	25.02.2025	25	Tests address 25	4600000000025	Test product 25	250	170.35	
27	26.02.2025	26	Tests address 26	4600000000026	Test product 26	260	175.35	
28	27.02.2025	27	Tests address 27	4600000000027	Test product 27	270	180.35	
29	28.02.2025	28	Tests address 28	4600000000028	Test product 28	280	185.35	
30								

1	date;shop_id;shop_address;barcode;product_name;qty;price
2	01.02.2025;1;Tests address 1;4600000000001;Test product 1;10;50.35
3	02.02.2025;2;Tests address 2;4600000000002;Test product 2;20;55.35
4	03.02.2025;3;Tests address 3;4600000000003;Test product 3;30;60.35
5	04.02.2025;4;Tests address 4;4600000000004;Test product 4;40;65.35
6	05.02.2025;5;Tests address 5;4600000000005;Test product 5;50;70.35
7	06.02.2025;6;Tests address 6;4600000000006;Test product 6;60;75.35
8	07.02.2025;7;Tests address 7;4600000000007;Test product 7;70;80.35
9	08.02.2025;8;Tests address 8;4600000000008;Test product 8;80;85.35
10	09.02.2025;9;Tests address 9;4600000000009;Test product 9;90;90.35
11	10.02.2025;10;Tests address 10;4600000000010;Test product 10;100;95.35
12	11.02.2025;11;Tests address 11;4600000000011;Test product 11;110;100.35
13	12.02.2025;12;Tests address 12;4600000000012;Test product 12;120;105.35
14	13.02.2025;13;Tests address 13;4600000000013;Test product 13;130;110.35
15	14.02.2025;14;Tests address 14;4600000000014;Test product 14;140;115.35
16	15.02.2025;15;Tests address 15;4600000000015;Test product 15;150;120.35
17	16.02.2025;16;Tests address 16;4600000000016;Test product 16;160;125.35
18	17.02.2025;17;Tests address 17;4600000000017;Test product 17;170;130.35
19	18.02.2025;18;Tests address 18;4600000000018;Test product 18;180;135.35
20	19.02.2025;19;Tests address 19;4600000000019;Test product 19;190;140.35
21	20.02.2025;20;Tests address 20;4600000000020;Test product 20;200;145.35
22	21.02.2025;21;Tests address 21;4600000000021;Test product 21;210;150.35
23	22.02.2025;22;Tests address 22;4600000000022;Test product 22;220;155.35
24	23.02.2025;23;Tests address 23;4600000000023;Test product 23;230;160.35
25	24.02.2025;24;Tests address 24;4600000000024;Test product 24;240;165.35
26	25.02.2025;25;Tests address 25;4600000000025;Test product 25;250;170.35
27	26.02.2025;26;Tests address 26;4600000000026;Test product 26;260;175.35
28	27.02.2025;27;Tests address 27;4600000000027;Test product 27;270;180.35
29	28.02.2025;28;Tests address 28;4600000000028;Test product 28;280;185.35
30	

Структура БД etl.src

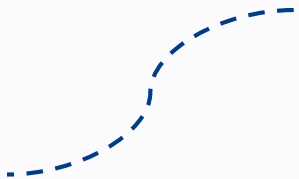
```
CREATE TABLE csv_queue (  
    id SERIAL PRIMARY KEY,  
    file_path TEXT NOT NULL,  
    processed BOOLEAN DEFAULT FALSE  
);
```

Сюда Python-скрипт `watch_csv_folder.py`
добавляет информацию о новых файлах

```
CREATE OR REPLACE FUNCTION notify_csv() RETURNS TRIGGER AS $$  
BEGIN  
    PERFORM pg_notify('load_csv', '/mnt/shared_external_data/scripts/load_csv_script.py ' ||  
NEW.file_path);  
    RETURN NEW;  
END;  
$$ LANGUAGE plpgsql;  
  
CREATE TRIGGER trigger_csv_insert  
AFTER INSERT ON csv_queue  
FOR EACH ROW  
EXECUTE FUNCTION notify_csv();
```

Структура БД etl.src

```
CREATE TABLE sales (  
    date DATE NOT NULL,  
    shop_id INTEGER NOT NULL,  
    shop_address TEXT,  
    barcode VARCHAR(50) NOT NULL,  
    product_name TEXT,  
    qty INTEGER NOT NULL,  
    price NUMERIC(10,2) NOT NULL  
);
```



load_csv_script.py загружает сюда
обработанные из csv данные

Скрипт Python - watch_csv_folder.py

```
WATCH_FOLDER = "/mnt/shared_external_data/ext_data/"
```

Добавляем запись о файле в БД

```
def add_file_to_queue(file_path):  
    """Добавляем новый файл в csv_queue"""  
    conn = get_db_connection()  
    cursor = conn.cursor()  
    cursor.execute("INSERT INTO src.csv_queue (file_path) VALUES (%s)", (file_path,))  
    conn.commit()  
    cursor.close()  
    conn.close()  
    print(f"Файл {file_path} добавлен в csv_queue.")
```

```
class CSVFileHandler(FileSystemEventHandler):  
    """Отслеживает появление новых файлов"""  
    def on_created(self, event):  
        if event.is_directory:  
            return  
        if event.src_path.endswith(".csv"):  
            time.sleep(1) # Ожидание полной записи файла  
            print(f"Файл создан: {event.src_path}")  
            add_file_to_queue(event.src_path)
```

Событие появления нового файла

```
observer = PollingObserver()  
observer.schedule(CSVFileHandler(), path=WATCH_FOLDER, recursive=False)  
observer.start()
```

Запуск сканирования

Скрипт Python - listen_notify_csv.py

```
def load_csv_to_postgres(script_path, file_path):  
    """Запускаем внешний Python-скрипт для загрузки CSV"""  
    print(f"Запущен скрипт {script_path} для обработки файла {file_path}")  
    subprocess.run(["python3", script_path, file_path], check=True)  
  
# Подключаемся к БД и слушаем события  
conn = get_db_connection()  
conn.set_isolation_level(psycopg2.extensions.ISOLATION_LEVEL_AUTOCOMMIT)  
cursor = conn.cursor()  
cursor.execute("LISTEN load_csv;")  
  
print("Ожидание новых файлов...")  
while True:  
    select.select([conn], [], [], None)  
    conn.poll()  
    while conn.notifies:  
        notify = conn.notifies.pop(0)  
        message = notify.payload.split(" ") # Разделяем путь к скрипту и путь к файлу  
        script_path = message[0]  
        file_path = message[1]  
        print(f"Получено уведомление о файле: {file_path}, запуск скрипта: {script_path}")  
        load_csv_to_postgres(script_path, file_path)
```

Запуск скрипта загрузки данных из csv

Прослушивание события

Получен сигнал о наступлении события

Скрипт Python - load_csv_script.py.py

```
def convert_date(date_str):
    """Преобразует дату из формата DD.MM.YYYY в формат YYYY-MM-DD"""
    return datetime.strptime(date_str, "%d.%m.%Y").strftime("%Y-%m-%d")

def load_csv(file_path):
    """Загружает CSV-файл в PostgreSQL"""
    conn = get_db_connection()
    cursor = conn.cursor()

    with open(file_path, "r", encoding="utf-8") as f:
        reader = csv.reader(f, delimiter=";")
        next(reader) # Пропускаем заголовок

        for row in reader:
            row[0] = convert_date(row[0]) # Преобразуем дату в ISO-формат

            cursor.execute(
                "INSERT INTO src.sales (date, shop_id, shop_address, barcode, product_name, qty, price) "
                "VALUES (%s, %s, %s, %s, %s, %s, %s)", row
            )

    conn.commit()
    cursor.close()
    conn.close()
    print(f"Файл {file_path} успешно загружен в БД.")

if __name__ == "__main__":
    file_path = sys.argv[1] # Получаем путь к файлу из аргументов командной строки
    load_csv(file_path)
```

Открываем файл csv

Парсинг даты

Добавление записи в БД

Demo

```
alex@debian: ~  
postgres@debian:/mnt/shared_external_data/ext_data$ python3 /mnt/shared_external_data/scripts/listen_notify_csv.py  
Ожидание новых файлов...  
Получено уведомление о файле: /mnt/shared_external_data/ext_data/data-2025-02-01-10.csv, запуск скрипта: /mnt/shared_external_data/scripts/load_csv_script.py  
Запущен скрипт /mnt/shared_external_data/scripts/load_csv_script.py для обработки файла /mnt/shared_external_data/ext_data/data-2025-02-01-10.csv  
Файл /mnt/shared_external_data/ext_data/data-2025-02-01-10.csv успешно загружен в БД.
```

```
alex@debian: /home  
postgres@debian:/mnt/shared_external_data/ext_data$ python3 /mnt/shared_external_data/scripts/watch_csv_folder.py  
Наблюдение за папкой /mnt/shared_external_data/ext_data/ запущено...  
Файл создан: /mnt/shared_external_data/ext_data/data-2025-02-01-10.csv  
Файл /mnt/shared_external_data/ext_data/data-2025-02-01-10.csv добавлен в csv_queue.
```

id	file_path
1	/mnt/shared_external_data/ext_data/data-2025-02-01 - Copy (2) - Copy - Copy.csv
2	/mnt/shared_external_data/ext_data/data-2025-02-01 - Copy (2) - Copy - Copy - Copy.csv
3	/mnt/shared_external_data/ext_data/data-2025-02-01 - Copy (2) - Copy - Copy - Copy - Copy.csv
4	/mnt/shared_external_data/ext_data/data-2025-02-01 - Copy (2) - Copy - Copy - Copy - Copy - Copy.csv
5	/mnt/shared_external_data/ext_data/data-2025-02-01 - Copy (2) - Copy - Copy - Copy - Copy - Copy - Copy.csv
6	/mnt/shared_external_data/ext_data/data-2025-02-01-1.csv
7	/mnt/shared_external_data/ext_data/data-2025-02-01 - Copy (3) - Copy - Copy.csv
8	/mnt/shared_external_data/ext_data/data-2025-02-01 - Copy (3) - Copy - Copy - Copy.csv
9	/mnt/shared_external_data/ext_data/data-2025-02-01 - Copy (3) - Copy - Copy - Copy - Copy.csv
10	/mnt/shared_external_data/ext_data/data-2025-02-01-2.csv
11	/mnt/shared_external_data/ext_data/data-2025-02-01-3.csv
12	/mnt/shared_external_data/ext_data/data-2025-02-01-4.csv
13	/mnt/shared_external_data/ext_data/data-2025-02-01-4.csv
14	/mnt/shared_external_data/ext_data/data-2025-02-01-10.csv
15	/mnt/shared_external_data/ext_data/data-2025-02-01-10.csv

date	shop_id	shop_address	barcode	product_name	qty	price
2025-02-14	14	Tests address 14	4,600,000,000,014	Test product 14	140	115.35
2025-02-15	15	Tests address 15	4,600,000,000,015	Test product 15	150	120.35
2025-02-16	16	Tests address 16	4,600,000,000,016	Test product 16	160	125.35
2025-02-17	17	Tests address 17	4,600,000,000,017	Test product 17	170	130.35
2025-02-18	18	Tests address 18	4,600,000,000,018	Test product 18	180	135.35
2025-02-19	19	Tests address 19	4,600,000,000,019	Test product 19	190	140.35
2025-02-20	20	Tests address 20	4,600,000,000,020	Test product 20	200	145.35
2025-02-21	21	Tests address 21	4,600,000,000,021	Test product 21	210	150.35
2025-02-22	22	Tests address 22	4,600,000,000,022	Test product 22	220	155.35
2025-02-23	23	Tests address 23	4,600,000,000,023	Test product 23	230	160.35
2025-02-24	24	Tests address 24	4,600,000,000,024	Test product 24	240	165.35
2025-02-25	25	Tests address 25	4,600,000,000,025	Test product 25	250	170.35
2025-02-26	26	Tests address 26	4,600,000,000,026	Test product 26	260	175.35
2025-02-27	27	Tests address 27	4,600,000,000,027	Test product 27	270	180.35
2025-02-28	28	Tests address 28	4,600,000,000,028	Test product 28	280	185.35

Выводы

1. Цели частично достигнуты, задачи выполнены
2. Сначала были попытки с COPY, сложности с мониторингом сетевой папки, структура БД сложностей не вызвала
3. Время реализации 12 часов
4. Полезность проекта 10/10
5. Текущих вопросов нет
6. Обработка ошибок, перезаливка данных



Вопросы и рекомендации



если есть вопросы



если вопросов нет

Спасибо за внимание!

