Департамент образования города Москвы

Государственное автономное образовательное учреждение

высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

ОТЧЕТ

по дисциплине «Проектный практикум по разработке ETL-решений»

Направление подготовки 38.03.05 – бизнес-информатика

Профиль подготовки «Аналитика данных и эффективное управление»

(очная форма обучения)

Самостоятельная работа 1 ETL-процессинг в Pentaho

Выполнил:

st-89

Руководитель:

Москва  
2025

**Цель работы:** необходимо провести предобработку данных по данным клиентов.

**Задачи:** Вариант 2:

1. Создать таблицу customers (id, first\_name, last\_name, email, phone, registration\_date)
2. Создать таблицу active\_customers с полями исходной таблицы + status, last\_order\_date
3. Удаление дубликатов по email
4. Группировка по дате регистрации
5. Обогащение данными о статусе активности

**Ход работы:**

Описание архитектуры решения: Загрузка данных из Postgres – далее их обработка и загрузка в phpmyadmin.

1. Для начала создадим таблицы в postgresql – таблица customers и phpmyadmin – active\_customers (Рис. 1 и Рис. 2)

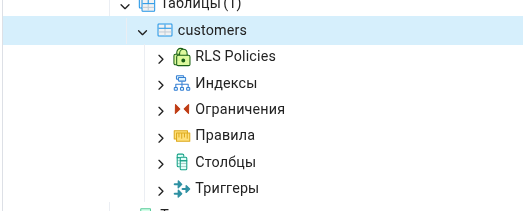


Рис. 1 – Создание таблица в PgAdmin

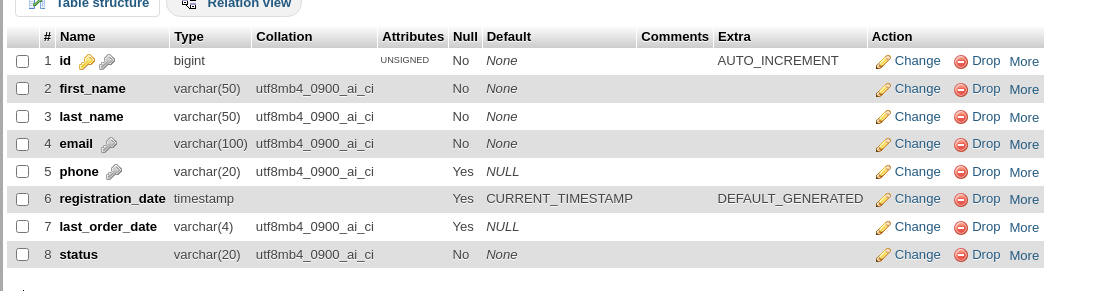


Рис. 2 – Создание таблицы в phpmyadmin

SQL скрипты для создания таблиц:

CREATE TABLE customers (

id SERIAL PRIMARY KEY,

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(20) UNIQUE,

registration\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE active\_customers (

id SERIAL PRIMARY KEY,

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(20) UNIQUE,

registration\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

last\_order\_date VARCHAR(4),

status VARCHAR(20) NOT NULL CHECK (status IN ('active', 'inactive'))

);

1. Создаем трансформацию и по итогам ее загружаем данные в MySQL, предварительно сгенерировав данные для Postgres и загрузив их туда INSERT-ом.

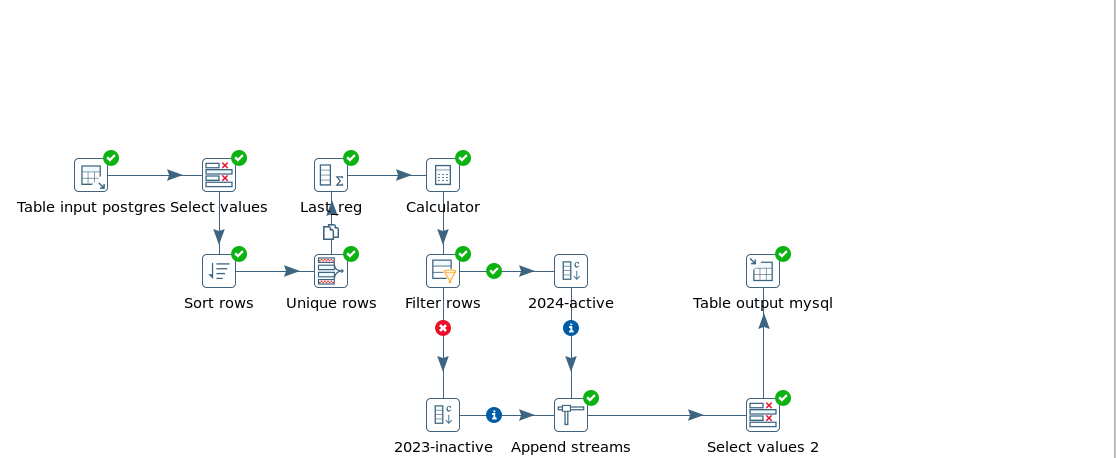


Рис. 3 - Общий вид трансформации

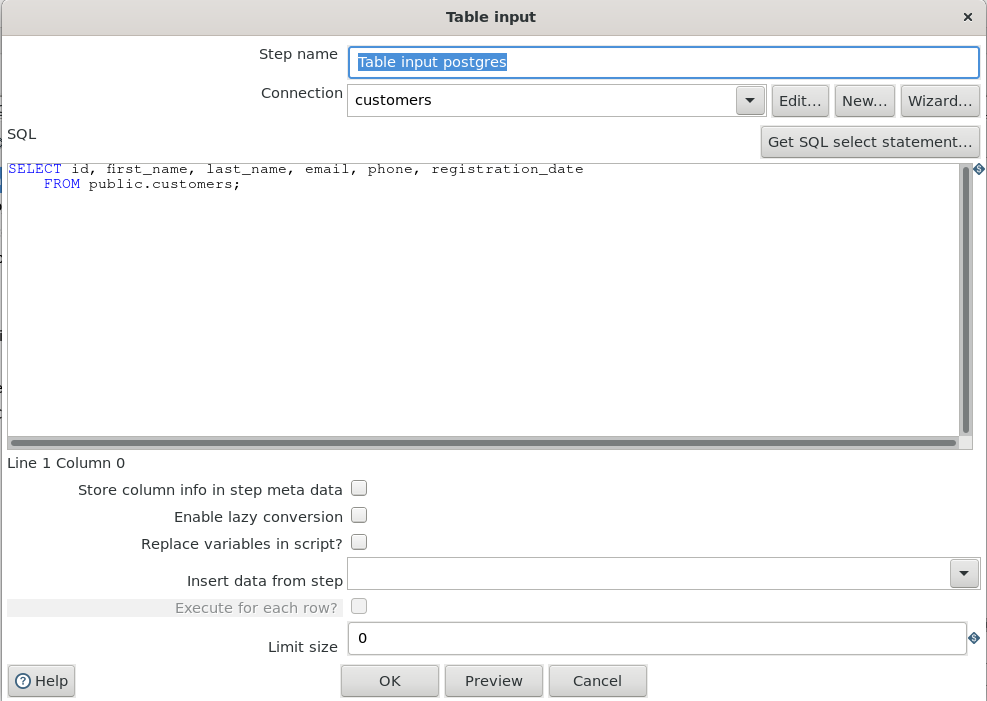


Рис. 4 – Забираем данные из postgresql

Далее по узлу через Select value проверяем типы данных. Sort rows – сортируем их для нахождения дублей, чтобы убрать их в узле Unique Rows.

Далее группируем по дате регистрации в узле Last Reg с параметром по последнему зарегистрировавшемуся пользователю в конкретную дату.

В узле Calculator (Рис. 5) добавляем поле last\_order\_date посредством registration\_date и выставляем год.

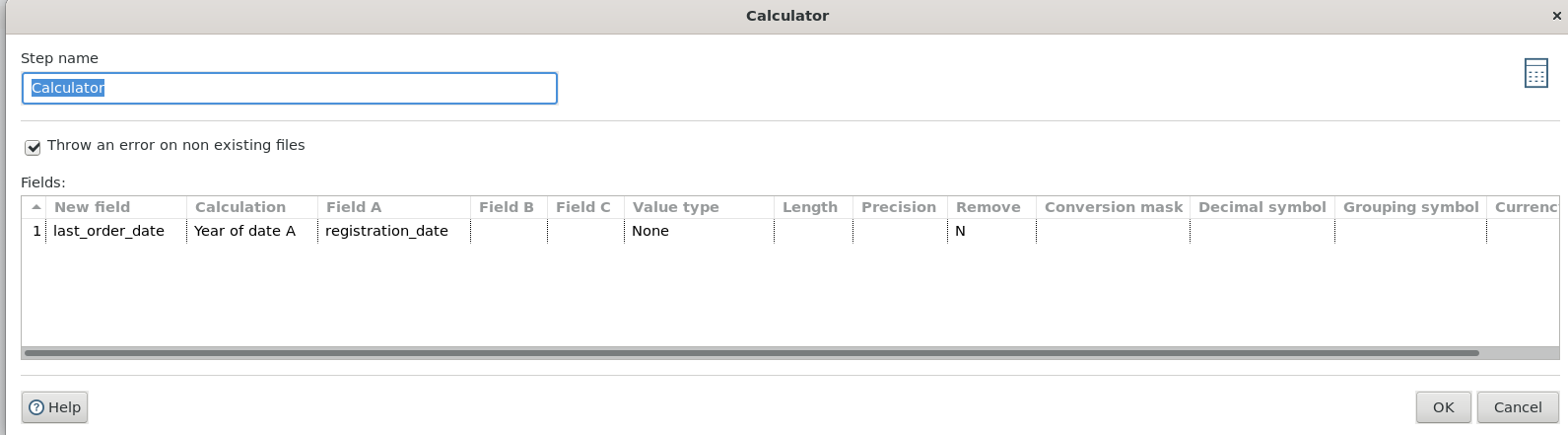


Рис. 5 - настройки Calculator

С помощью узла Filter Rows выставляем фильтр, если год 2024 – то active, то далее в узел Add constants (2024-active). Если год 2023 – inactive и в узел Add constants (2023-inactive)

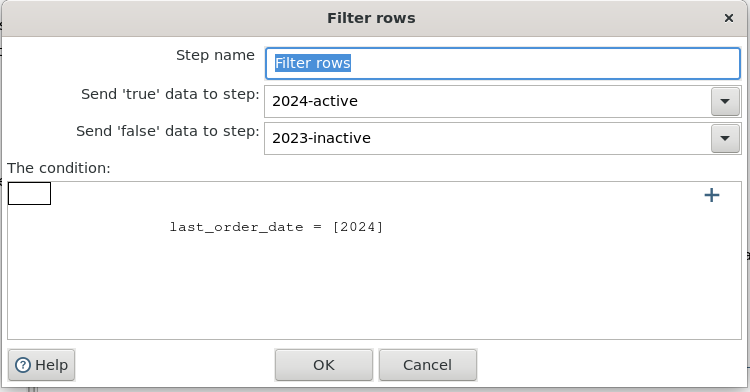


Рис. 6 - настройка Filter Rows

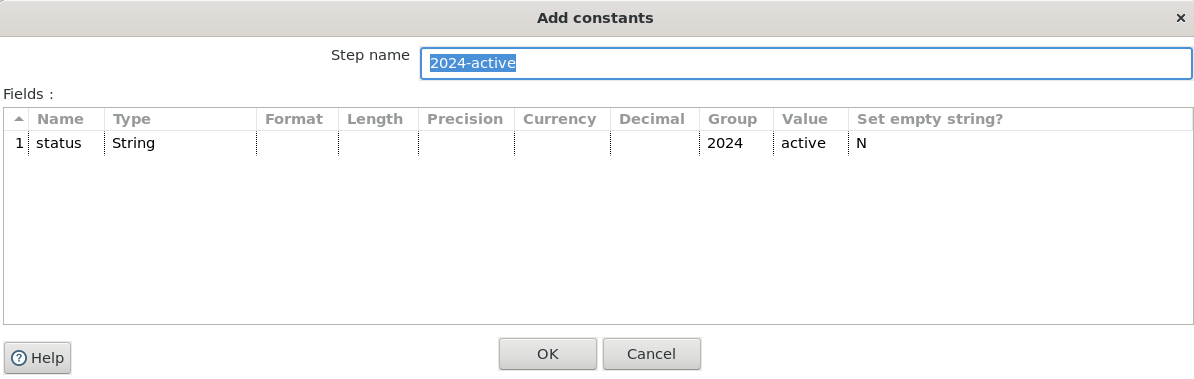


Рис. 7 - Настройка Add constants для 2024

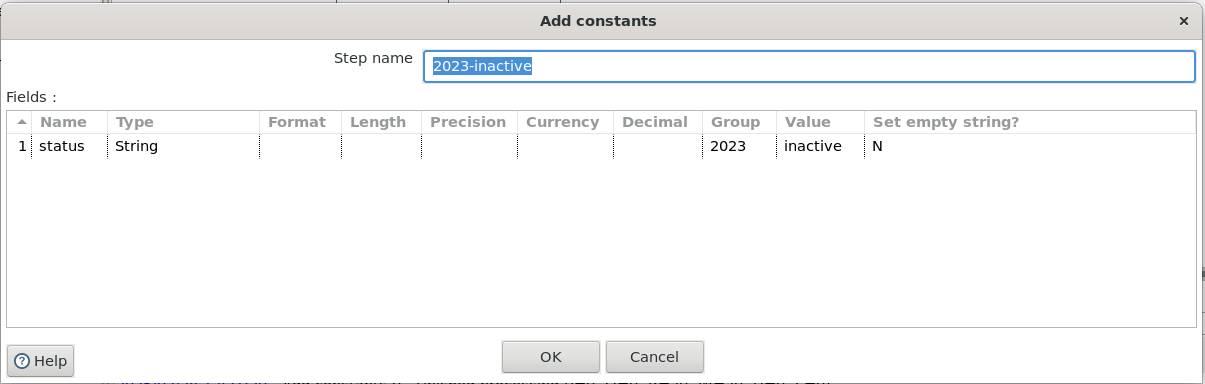


Рис. 8 - Настройка Add constants 2023

В узле Append streams соединяем данные и далее загружаем в MySQL

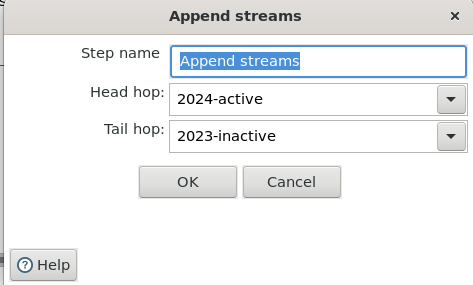


Рис. 9 - Соединение данных

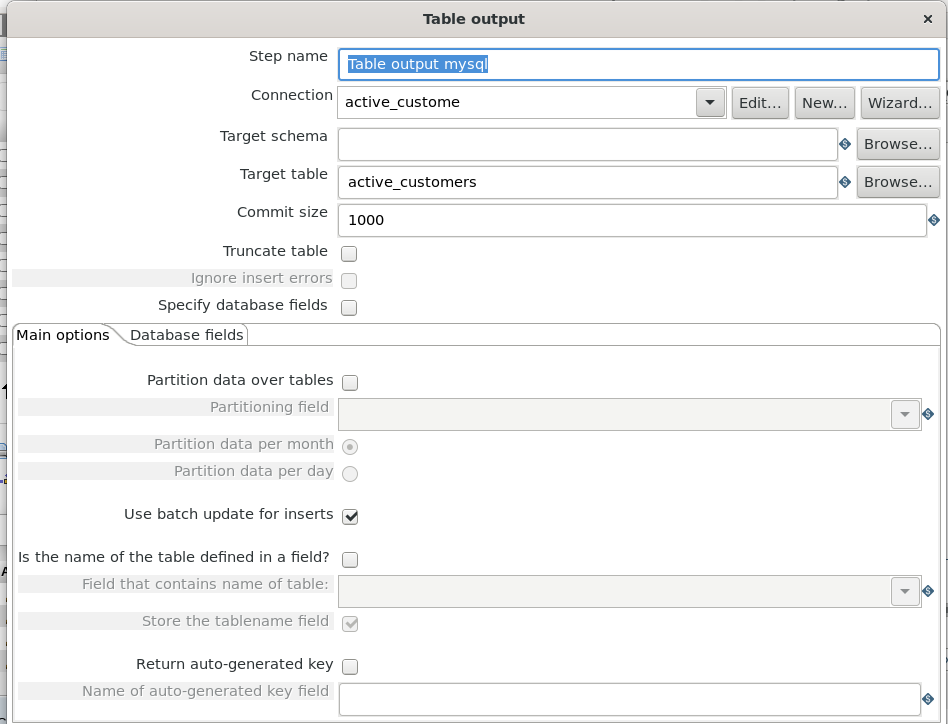


Рис. 10 - Подключение к MySQL

По итогу получается вот такие загруженные данные в Mysql

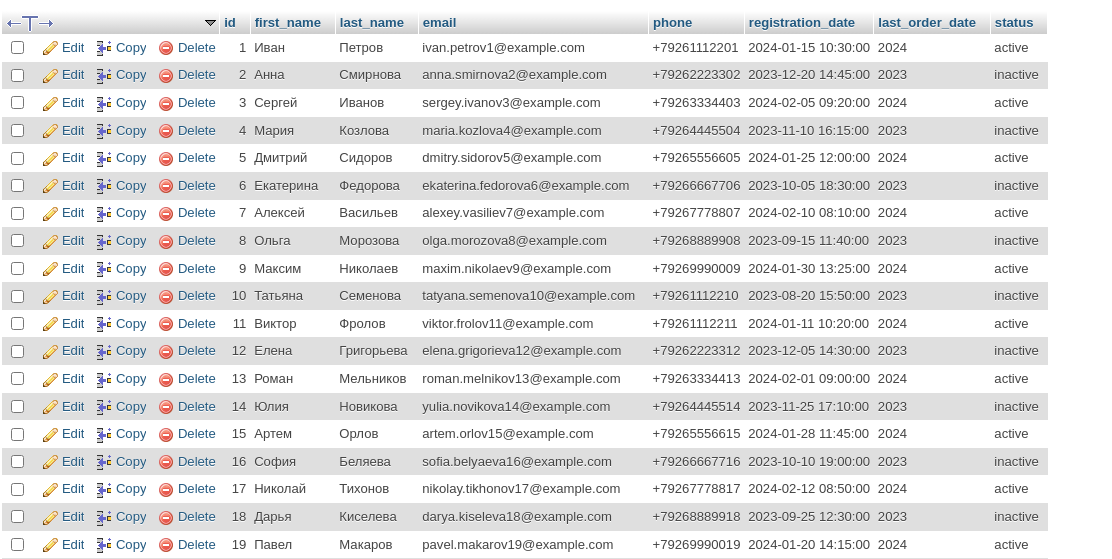


Рис. 11 - Данные в phpmyadmin

**Вывод:** Была проведена выгрузка данных из PostgreSQL в ETL процесс для предобработки и далее загрузка в phpmyadmin

**ПРИЛОЖЕНИЕ**

SQL код для создания таблиц

-- Таблица заказов (основная информация о продажах)

CREATE TABLE orders (

row\_id INT PRIMARY KEY,

order\_date DATE,

ship\_date DATE,

ship\_mode VARCHAR(50),

sales DECIMAL(10,2),

quantity INT,

discount DECIMAL(4,2),

profit DECIMAL(10,2),

returned TINYINT(1) DEFAULT 0 -- 1 = Yes, 0 = No

);

-- Таблица клиентов

-- Пересоздаем таблицу customers

DROP TABLE IF EXISTS customers;

CREATE TABLE customers (

id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id VARCHAR(20) NOT NULL,

customer\_name VARCHAR(100),

segment VARCHAR(50),

country VARCHAR(100),

city VARCHAR(100),

state VARCHAR(100),

postal\_code VARCHAR(20),

region VARCHAR(50),

INDEX idx\_customer\_id (customer\_id),

INDEX idx\_region (region)

);

-- Пересоздаем таблицу products

DROP TABLE IF EXISTS products;

CREATE TABLE products (

id INT AUTO\_INCREMENT PRIMARY KEY,

product\_id VARCHAR(20) NOT NULL,

category VARCHAR(50),

sub\_category VARCHAR(50),

product\_name VARCHAR(255),

person VARCHAR(100),

INDEX idx\_product\_id (product\_id), -- Обычный индекс вместо UNIQUE

INDEX idx\_category (category),

INDEX idx\_subcategory (sub\_category)

);