Департамент образования и науки города Москвы

Государственное автономное образовательное учреждение высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

Проектный практикум по разработке ETL-решений

**Лабораторная работа 2.1**

**Динамические соединения с базами данных**

Выполнила: Сергеева А. И., группа: АДЭУ-211

Преподаватель: Босенко Т.М.

Москва

2025

**Цель работы:** получить практические навыки создания ETL-процесса для интеграции данных из различных источников с использованием динамических соединений в Pentaho Data Integration, включая обработку повторяющихся данных.

**Задачи:**

1. Создать динамические подключения к различным источникам данных.
2. Разработать процесс выявления и обработки дублирующихся записей.
3. Реализовать механизм объединения данных в единое хранилище.
4. Настроить обработку ошибок при выполнении трансформации.

**Программное обеспечение:**

- Pentaho Data Integration 9.4.

- MySQL или PostgreSQL.

- CSV или Excel файлы с тестовыми данными.

**Ход работы:**

В начале работы были созданы 3 таблицы, в которые будет загружаться информация: customers, products, orders, были созданы индексы и применена кодировка, как видно на рисунках 1-4.

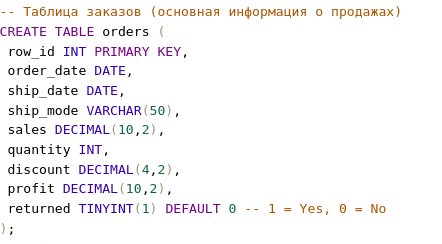


Рисунок 1 – Создание таблицы orders

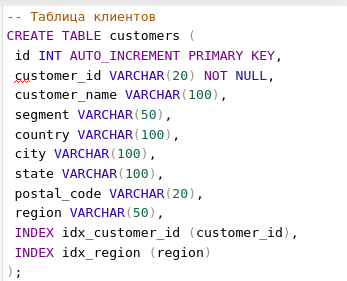


Рисунок 2 – Создание таблицы customers

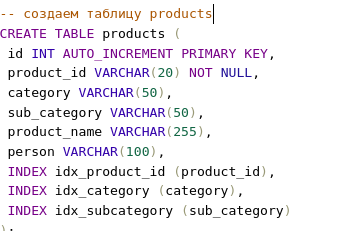


Рисунок 3 – Создание таблицы products

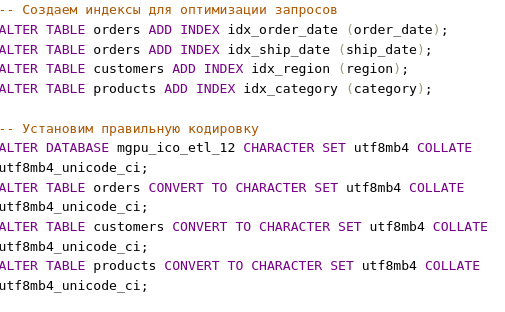


Рисунок 4 – Создание индексов и установка кодировки

Далее была создана папка datain, в которой и будут храниться все трансформации, а также job.

Первая трансформация загружает данные в таблицу orders. В объекте Select values настраиваются типы данных столбцов, для столбцов с датой назначен тип даты, а для row id назначен целочисленный тип. Данный блок отображен на рисунке 5.

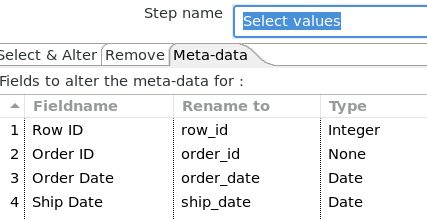


Рисунок 5 – Настройка Select values для трансформации orders

В объекте Memory group by были выбраны необходимые столбцы для будущей таблицы, что продемонстрировано на рисунке 6.

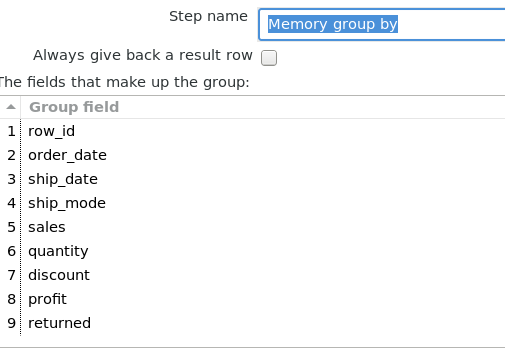


Рисунок 6 – Настройка Memory group by для трансформации orders

В объекте Filter row отбираются данные, у которых есть дата заказа и дата доставки, как показывает рисунок 7.

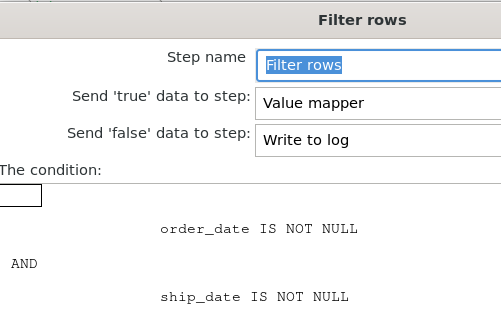


Рисунок 7 - Настройка Filter row для трансформации orders

Объект Value mapper на рисунке 8 меняет значения столбца returned, если yes, то 1, если no, то 0, если пусто, то также присваивается 0.

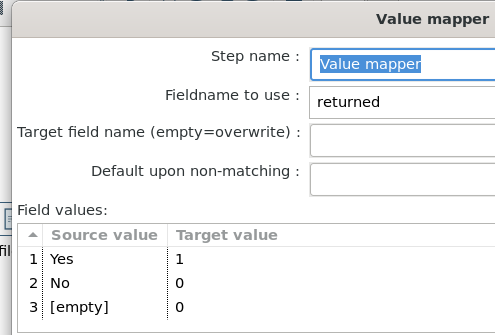


Рисунок 8 – Настройка Value mapper для трансформации orders

В рамках **12 варианта** был создан дополнительный фильтр, который отбирает только те заказы, срок доставки которых больше 5 дней. Для этого был создан калькулятор, считающий дни доставки на рисунке 9.

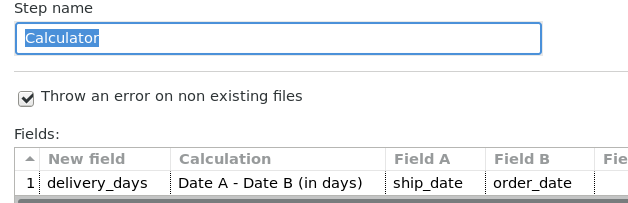


Рисунок 9 – Настройка калькулятора для фильтра срока доставки

После создания калькулятора был уже создан сам фильтр, представленный на рисунке 10.

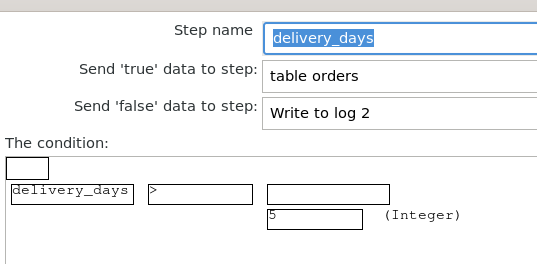


Рисунок 10 – Настройка фильтра для дней доставки

Последним этапом трансформации является подключение к базе для внесения данных, как видно на рисунке 11.

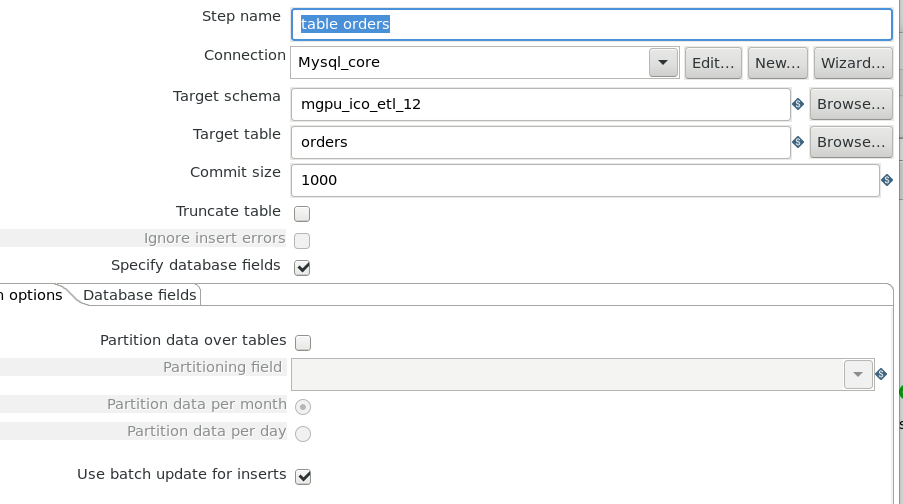


Рисунок 11 – Настройка подключения к базе для внесения данных в таблицу orders

Схема трансформации показана на рисунке 12.

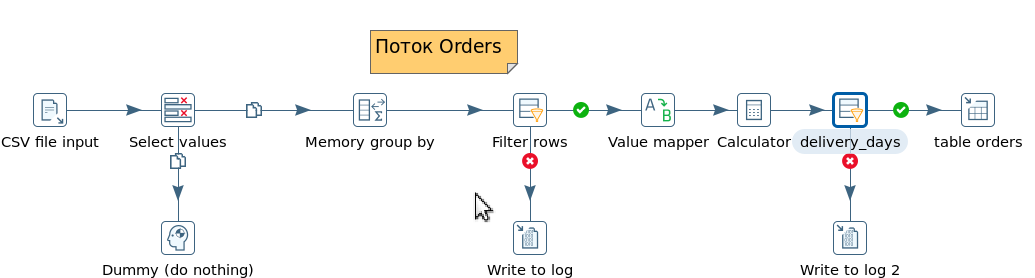


Рисунок 12 – Трансформация для таблицы orders

Подобные операции были выполнены и для трансформаций для таблиц customers, products на рисунках 13-14.

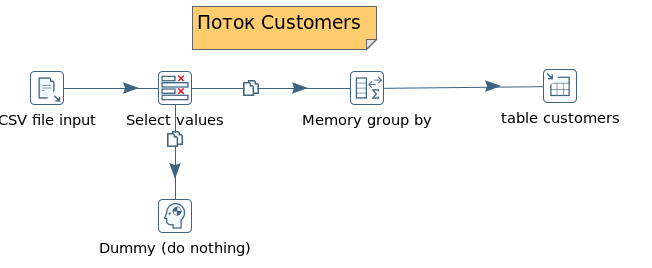


Рисунок 13 – Трансформация для таблицы customers

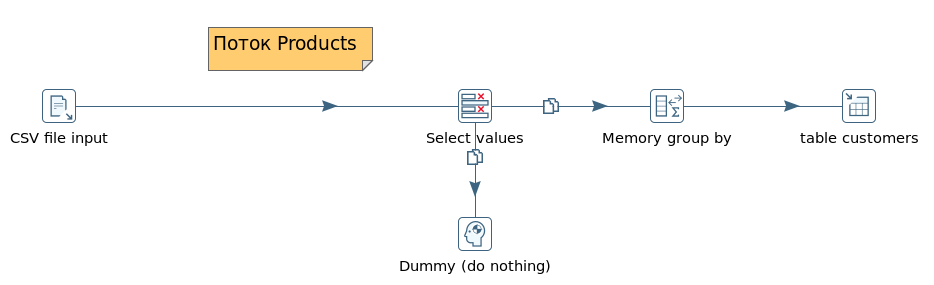


Рисунок 14 – Трансформация для таблицы products

Фрагменты загруженных таблиц продемонстрированы на рисунках 15-17.

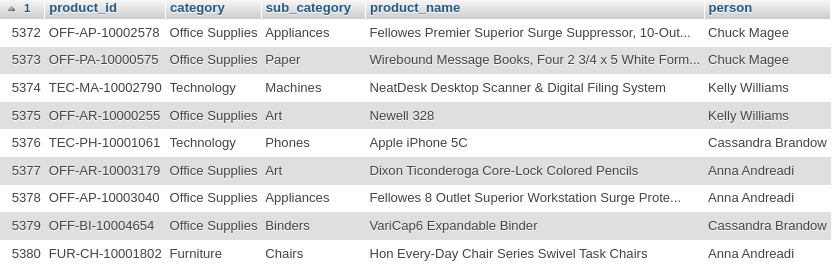


Рисунок 15 - Таблица products

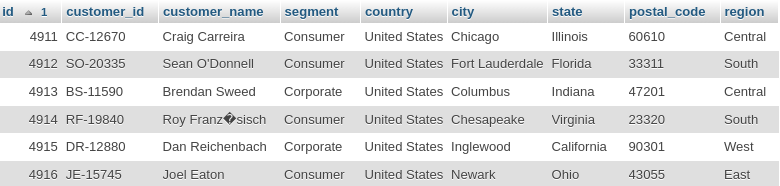


Рисунок 16 – Таблица customers

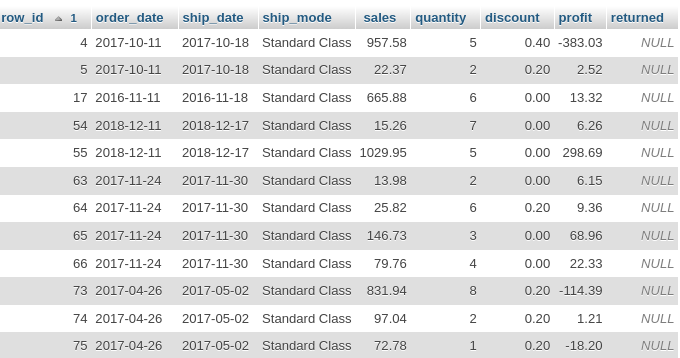


Рисунок 17 – Таблица orders

В рамках дополнительных заданий необходимо было подготовить отчёт о прибыли и анализ продаж, для этого была создана еще одна трансформация. Кроме стандартных операций был добавлен объект Memory Group by, группирующий прибыль, продажи и число продаж по категориям, регионам, городам, штатам, сегментам, странам, его настройка показана на рисунке 18.

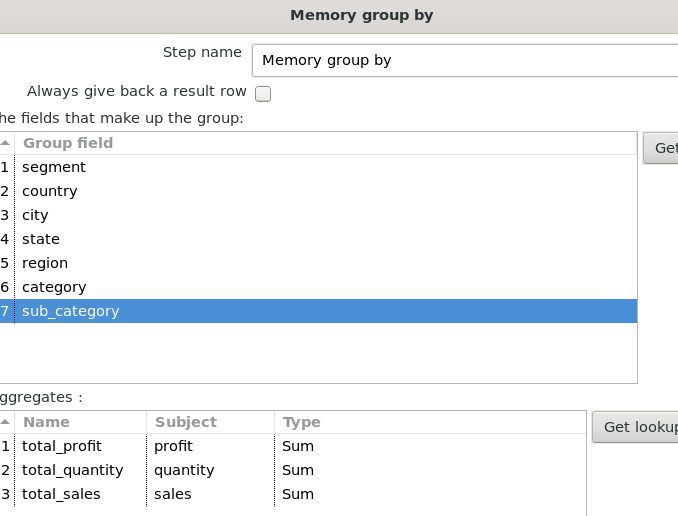


Рисунок 18 – Настройка объекта Memory Group By для отчёта о прибыли

Конечная трансформация показана на рисунке 19.

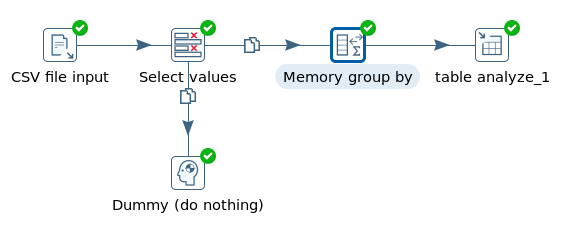


Рисунок 19 – Трансформация для отчёта о прибыли

Для хранения информации была создана дополнительная таблица, представленная на рисунке 20, которую в дальнейшем можно обобщать по одному из полей, например, посмотреть группировку только по странам.

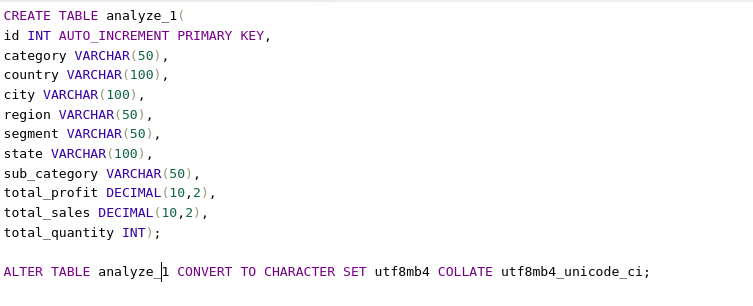


Рисунок 20 – Создание таблицы для отчета о прибыли

Конечная таблица показана на рисунке 21.

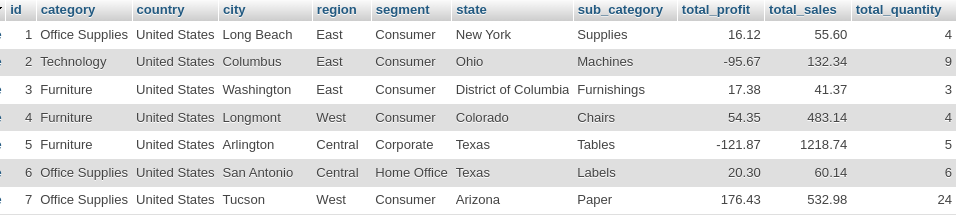


Рисунок 21 – Отчёт о прибыли и продажам

Для примера была выполнена группировка по штатам на рисунках 22-23.

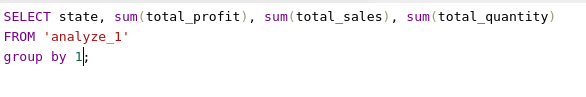


Рисунок 22 – SQL запрос для группировки по штатам

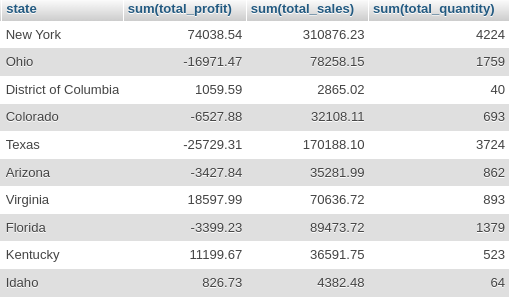


Рисунок 23 – Фрагмент результата запроса

Job представлен на рисунке 24.

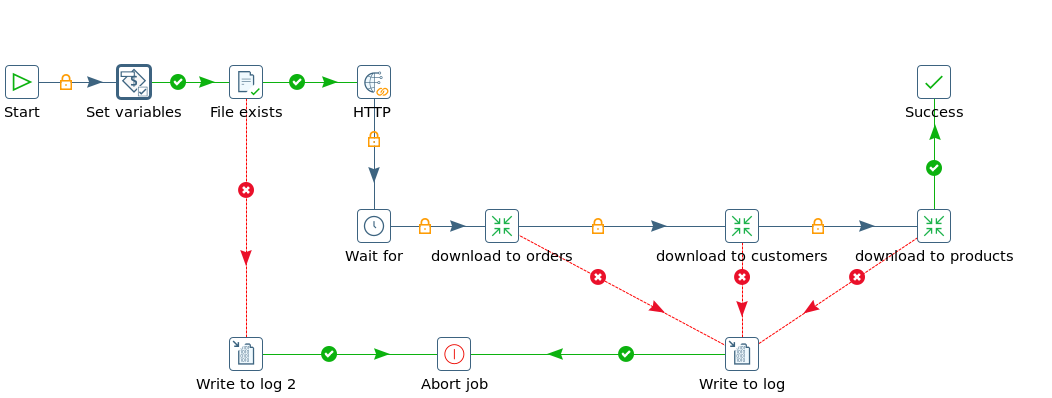


Рисунок 24 – Job с загрузкой файла из веб-источника, примененными трансформациями и выгрузкой в базу данных

**Общий вывод**:

В ходе выполнения работы были успешно реализованы все этапы создания ETL-процесса для интеграции данных из различных источников. Благодаря продвинутым возможностям Pentaho Data Integration удалось настроить динамические соединения к различным источникам данных, таким как база данных MySQL, а также к внешнему файлу в формате CSV. Это обеспечило гибкость в работе с разнообразными источниками, позволяя легко адаптироваться к изменению данных. А также изучены возможности фильтрации и группировки. При фильтрации можно ненужные данные отправлять в файлы с логами, а объект Dummy (do nothing) в Pentaho Data Integration позволяет создавать фиктивные шаги. Для группировки использовался объект Memory Gorup By, а не Gorup By, ведь для работы Group By необходимы отсортированные данные, а для Memory Group By — нет. Memory Group By внутренне сортирует данные и выдаёт результат на основе выбранной агрегатной опции. При этом данные не настолько большие, чтобы Memory Group By пришлось заменять Group By в сочетании с сортировкой. Цель и задачи были выполнены.

**Ответы на контрольные вопросы:**

1. Что такое динамические соединения в PDI?

Динамические соединения в Pentaho Data представляют собой механизм, позволяющий устанавливать соединения к источникам данных (как базам данных, так и другим типам источников).

1. Как организовать обработку ошибок в трансформации?

Чтобы организовать обработку ошибок в трансформации в Pentaho Data Integration, нужно:

* Нажать правой кнопкой мыши на шаг и выбрать «Определить обработку ошибок».
* Установить целевой шаг для потока данных с ошибками и проверить опцию «Включить обработку ошибок».

Обработка ошибок позволит настроить шаг так, чтобы вместо остановки трансформации при возникновении ошибки строки, вызвавшие ошибку, передавались на другой шаг.

Также для мониторинга ошибок можно использовать ведение логов в базе данных для трансформаций или заданий (вкладка «Лог» в диалоге параметров задания или трансформации). Так можно получать актуальную информацию о статусе выполнения и, например, писать задание, которое периодически сканирует базу логов и отправляет отчёты об ошибках куда нужно.

1. Какие методы выявления дублей существуют?

Чтобы избавиться от дубликатов в Pentaho Data Integration, можно использовать шаг «Unique rows». Он удаляет дублирующиеся строки из входного потока. Также для удаления дубликатов после сохранения их значений в выходной таблице можно использовать шаг «Execute SQL», где определяется SQL, который удаляет дублирующиеся записи и оставляет только уникальные строки.

1. Как настроить параметризацию подключений?

Чтобы настроить параметризацию подключений в Pentaho Data Integration, нужно:

* Выбрать следующую доступную строку в таблице параметров.
* Указать тип базы данных и ввести допустимое имя параметра и его соответствующее значение.

Также для больших проектов можно использовать файлы свойств. У каждой задачи есть свои параметры, которые хранятся в файле свойств с таким же именем, как и у задачи. В этом файле нужно записывать только те параметры, которые требуются конкретной задаче, чтобы не перезаписывать параметры дочерней задачи.

Ещё один способ — передавать параметры из командной строки или скриптов оболочки и настраивать их в настройках задачи или преобразования. Этот подход подходит для небольших процессов.

1. Какие компоненты PDI используются для объединения данных?

Для объединения данных в Pentaho Data Integration (PDI) используются следующие компоненты:

* Step Add XML. Позволяет создавать простые фрагменты XML. С его помощью можно определить корневой элемент и вложить в него другие элементы XML, а также опционально определить для них атрибуты.
* Step XML Join. С его помощью можно объединять фрагменты XML.
* Step Database Join. Используется для объединения двух таблиц в разных базах данных. Для объединения двух таблиц из одной базы данных применяется метод «Table Input», который выполняет объединение только на уровне SQL.
* Step Merge Join. Для его использования нужно, чтобы строки были идеально отсортированы, так как он использует SQL-запрос с условием «ORDER BY».