Лабораторная работа 4.2. Динамические соединения с базами данных

Цель работы: получить практические навыки создания ETL-процесса для загрузки данных из CSV-файла в базу данных MySQL с использованием Pentaho Data Integration.

Задачи:

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

Программное обеспечение:

- Pentaho Data Integration 9.4.
- MySQL или PostgreSQL.
- CSV или Excel файлы с тестовыми данными.

Теоретические сведения

Динамические соединения в PDI позволяют:

- Использовать параметры подключения из внешних источников.
- Менять настройки соединения во время выполнения.
- Обрабатывать множество источников данных в одном процессе.

Компоненты обработки ошибок.

- **wrt-execution_error** - запись информации об ошибках.
- **abrt-execution_error** - прерывание выполнения при критических ошибках.

Ход работы

1. Подготовка.

   Создайте новую базу данных MySQL используя предоставленный SQL скрипт.

   Убедитесь, что файл **samplestore-general.csv** доступен (https://github.com/BosenkoTM/workshop-on-ETL/blob/main/data_for_lessons/samplestore-general.csv).

   Настройте подключение к MySQL в Pentaho.

2. Порядок выполнения.

   Трансформация 1 загружает **customers**.

   Трансформация 2 загружает данные **products**.

   Трансформация 3 загружает **Output**.

   Job «**CSV_to_MYsql**. **kjb**» контролирует весь процесс.

3. Особенности решения:

   Нормализованная структура БД.

   Правильные связи между таблицами.

   Обработка ошибок.

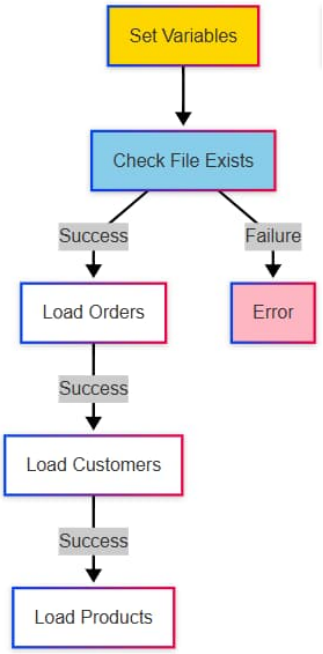
   Логирование процесса.

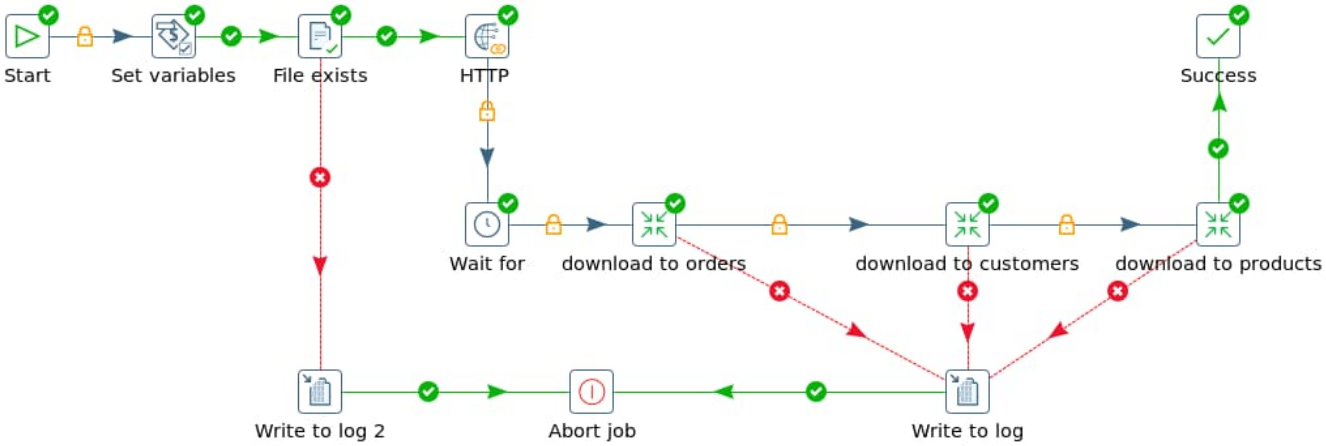
Загрузка данных из веб-источника. Job CSV_to_MYsql.kjb

Структура Job с проверками:

Set Variables (установка CSV_FILE_PATH).

Write To Log (вывод пути для проверки).
Check File Exists (проверка файла).
Transformation 1: Load Orders.
Transformation 2: Load Customers.
Transformation 3: Load Products.





Настройка Job:
1. Добавьте шаг "Set Variables" в начало Job:
   Имя переменной: CSV_FILE_PATH
   Значение: /home/dba/Downloads/datain/samplestore-general.csv
   Тип: String
2. В "Check File Exists" используйте:
   Filename: ${CSV_FILE_PATH}
В каждой трансформации измените CSV Input:
Filename: ${CSV_FILE_PATH}
Загрузка файла по протоколу HTTP:

В поле "URL" укажите:
https://raw.githubusercontent.com/BosenkoTM/workshop-on-ETL/main/data_for_lessons/
/samplestore-general.csv

убедитесь, что:

- Директория ~/Downloads/datain существует.
- У вас есть права на запись в эту директорию.
- Если директория не существует, создайте её:

mkdir -p ~/Downloads/datain

Если возникают проблемы с правами доступа, можно проверить и установить права:

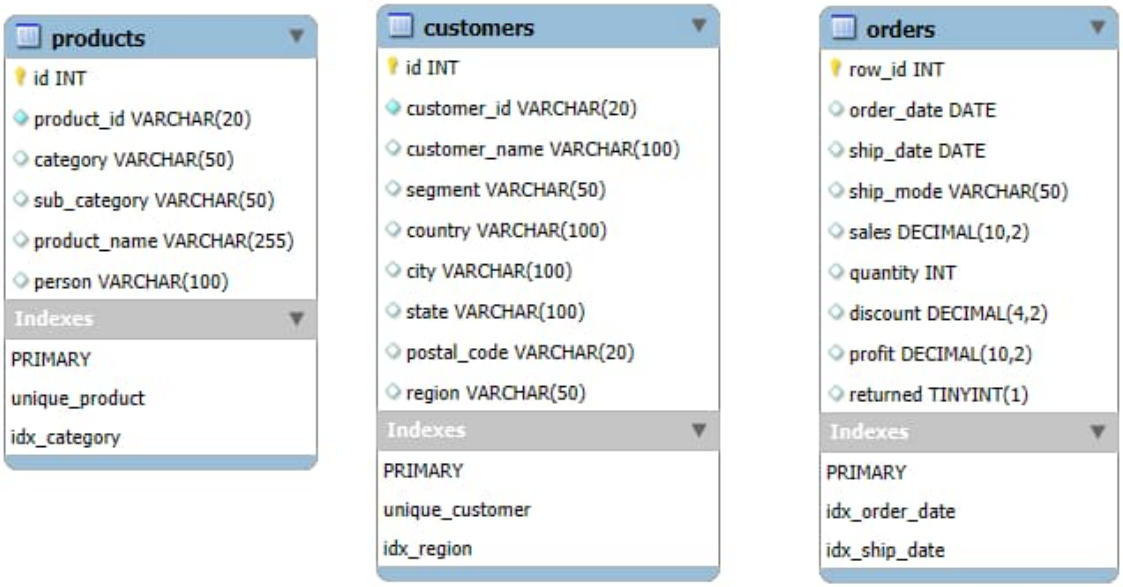
chmod 755 ~/Downloads/datain

Общая настройка объекта HTTP представлена на рисунке ниже.

Целевой объект хранения – прописать полный путь, где будут храниться данные после загрузки.

file:///home/dba/Downloads/datain/samplestore-general.csv

Создание базы данные или таблиц в PostgreSQL/MySQL



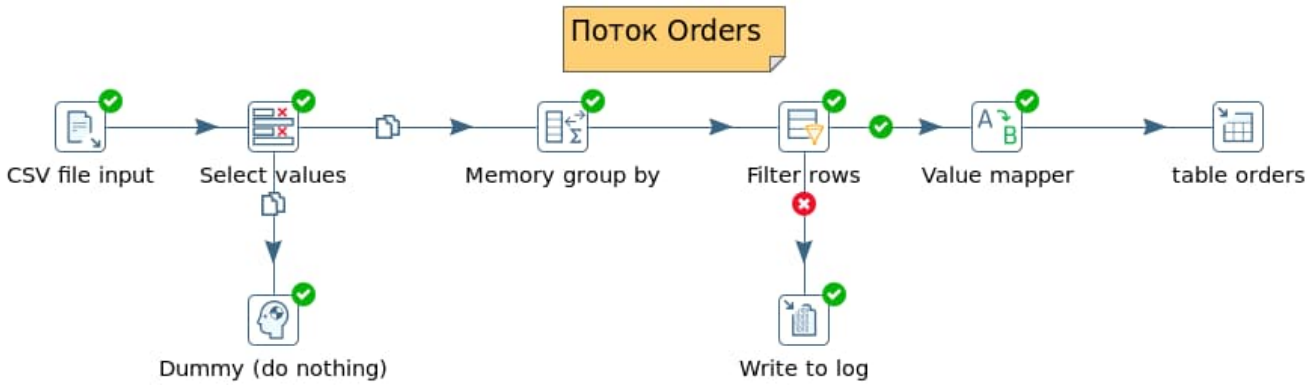

```
-- Таблица заказов (основная информация о продажах)
CREATE TABLE orders (
    row_id INT PRIMARY KEY,
    order_date DATE,
    ship_date DATE,
    ship_mode VARCHAR(50),
```

```
    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
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)
);

-- создаем 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),
    INDEX idx_category (category),
    INDEX idx_subcategory (sub_category)
);

-- Создаем индексы для оптимизации запросов
ALTER TABLE orders ADD INDEX idx_order_date (order_date);
ALTER TABLE orders ADD INDEX idx_ship_date (ship_date);
ALTER TABLE customers ADD INDEX idx_region (region);
```

ALTER TABLE products ADD INDEX idx_category (category);

-- Установим правильную кодировку
ALTER DATABASE mgpu_ico_etl_prepod CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
ALTER TABLE orders CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
ALTER TABLE customers CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
ALTER TABLE products CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;

Трансформация 1. lab_02_1_csv_orders.ktr



Объект **Select values** (конвертация типов):
Выбрать все поля, для явного преобразование типов.
В закладке Meta-data

- Row ID: Integer
- Order Date: Date (format: dd.MM.yyyy)
- Ship Date: Date (format: dd.MM.yyyy)
- Остальные поля не изменять.


Select & Alter | Remove | Meta-data

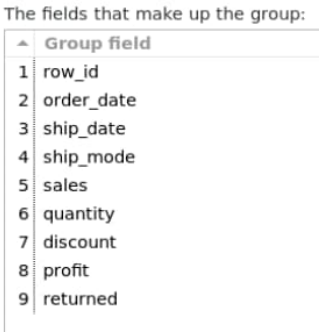
Fields to alter the meta-data for :

	Fieldname	Rename to	Type
1	Row ID	row_id	Integer
2	Order ID	order_id	None
3	Order Date	order_date	Date
4	Ship Date	ship_date	Date
5	Ship Mode	ship_mode	None
6	Customer ID	customer_id	None
7	Customer Name	customer_name	None
8	Segment	segment	None
9	Country	country	None
10	City	city	None
11	State	state	None
12	Postal Code	postal_code	None
13	Region	region	None
14	Product ID	product_id	None
15	Category	category	None
16	Sub-Category	sub_category	None
17	Product Name	product_name	None
18	Sales	sales	None
19	Quantity	quantity	None
20	Discount	discount	None
21	Profit	profit	None
22	Returned	returned	None
23	Person	person	None

Объект **Memory group by:**

Выбрать поля для таблицы БД **orders**:

- row_id (из Row ID)
- order_date (из Order Date)
- ship_date (из Ship Date)
- ship_mode (из Ship Mode)
- sales
- quantity
- discount
- profit
- returned

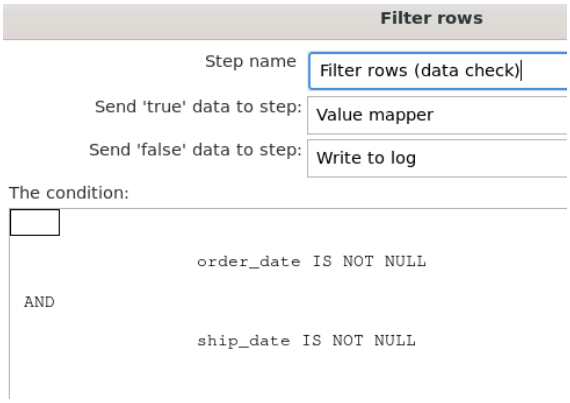


Объект **Filter row:**

Проверка даты

order_date IS NOT NULL

ship_date IS NOT NULL



Если true, то переходим в объект **Table output, false – объект Write to log**

Объект **Value mapper**

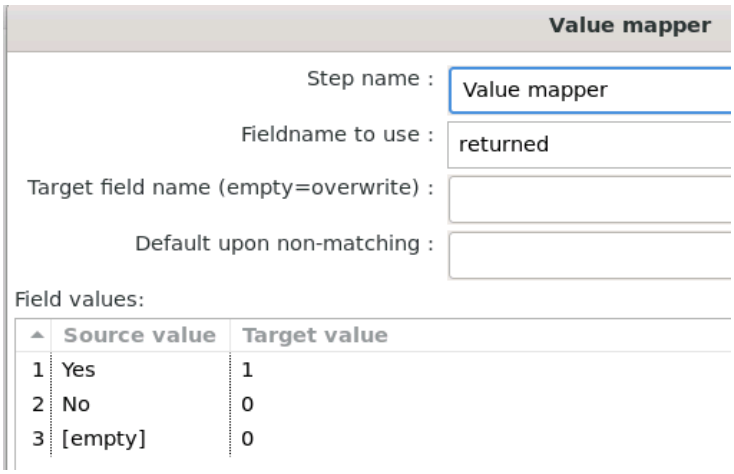
- Fieldname to use: returned
- Target fieldname: -

- Mappings:

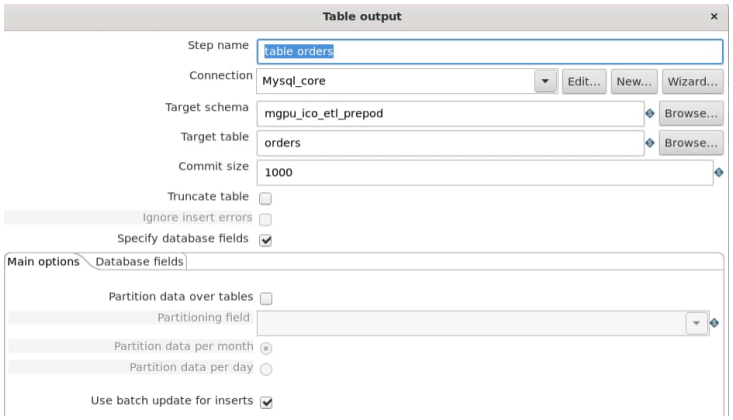
Yes -> 1

No -> 0

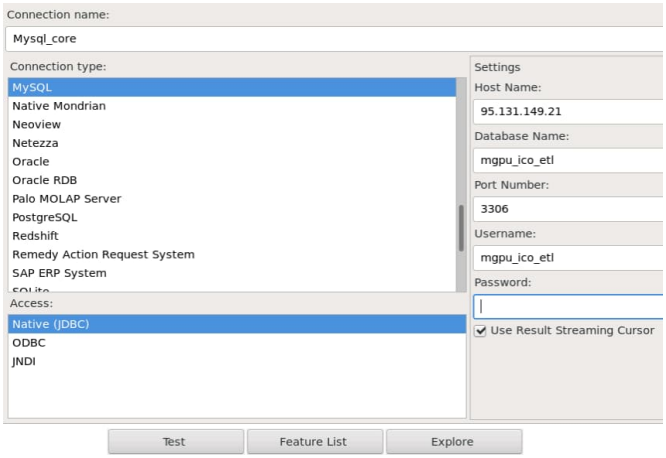
[empty] -> 0



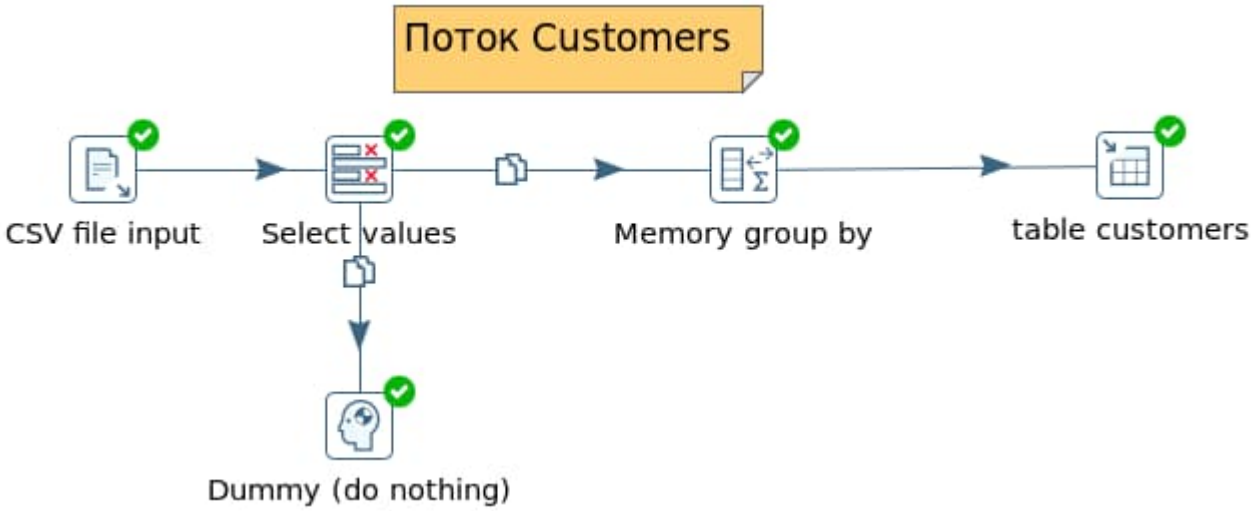
Объект **Table output:**



Настраиваем коннектор согласно Вашим учетным данным подключения к БД.



Трансформация 2. lab_02_2_csv_**customers**.ktr

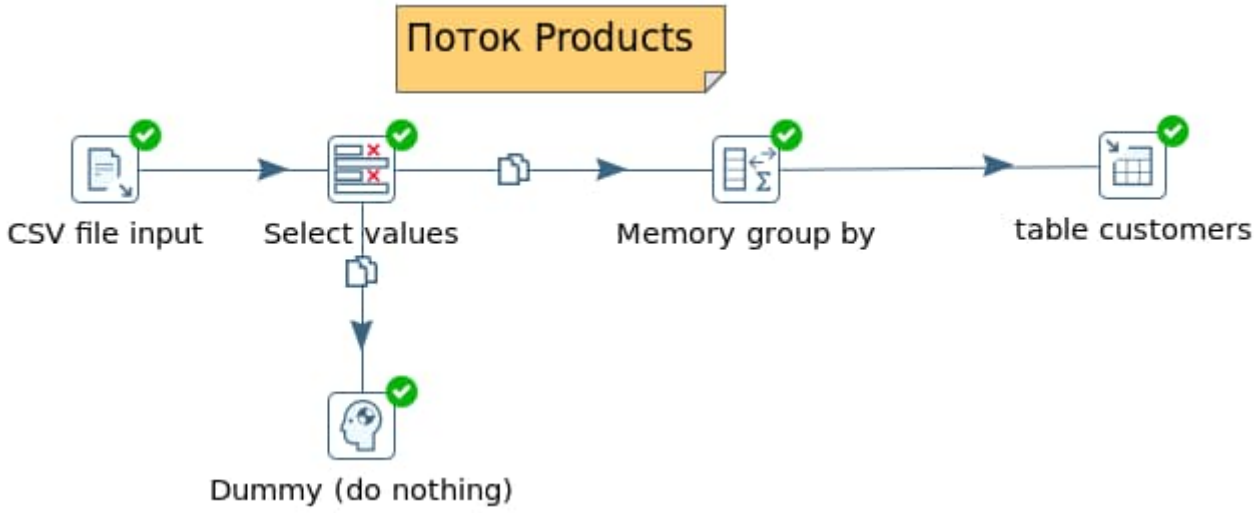


Настройка **Select Fields** для **customers**:

Выбрать поля:

- customer_id
- customer_name
- segment
- country
- city
- state
- postal_code
- region

Трансформация 3. lab_02_3_csv_products.ktr



Настройка **Select Fields** для **products**:

Выбрать поля:

- product_id
- category
- sub_category
- product_name
- person

Настройка **Table Output** для каждой таблицы:

Database connection:

- Type: MySQL
- Host: localhost
- Database: samplestore
- User/Password: ваши данные

Settings:

- Target table: соответствующая таблица
- Specify database fields: ✓
- Use batch update: ✓

Структура Job:

```
Start
  ├── Set Variables (FILE_PATH)
  ├── Check File Exists
  ├── HTTP Download
  ├── Transform: Load Orders (lab_02_1)
  ├── Transform: Load Customers (lab_02_2)
  ├── Transform: Load Products (lab_02_3)
  ├── Transform: Analytics 1 (lab_02_4)
  └── Transform: Analytics 2 (lab_02_5)
```

Варианты индивидуальных заданий 4.2.

№	Основной фильтр для загрузки в БД	Доп. задание 1	Доп. задание 2
1	Фильтр по дате: заказы за 2016 год	Анализ прибыльности по категориям	Отчет по менеджерам
2	Фильтр по региону: только Central	Статистика по способам доставки	Анализ возвратов
3	Фильтр по сегменту: только Consumer	Отчет по скидкам	Анализ по штатам
4	Фильтр по категории: только Office Supplies	Анализ продаж по месяцам	Отчет по клиентам
5	Фильтр по размеру заказа: Quantity > 3	Статистика по регионам	Анализ прибыльности
6	Фильтр по прибыли: только Profit > 0	Отчет по категориям	Анализ доставки
7	Фильтр по скидке: Discount > 0	Анализ по сегментам	Отчет по возвратам
8	Фильтр по доставке: только Standard Class	Статистика продаж	Анализ по городам
9	Фильтр: только заказы без возвратов	Отчет по менеджерам	Анализ категорий
10	Фильтр по стране: только United States	Анализ скидок	Отчет по регионам
11	Фильтр по сумме: Sales > 100	Статистика по категориям	Анализ клиентов
12	Фильтр: срок доставки > 5 дней	Отчет по прибыли	Анализ продаж
13	Фильтр по штату: только Texas	Анализ возвратов	Отчет по доставке
14	Фильтр: только заказы с возвратами	Статистика по менеджерам	Анализ регионов
15	Фильтр по подкатегории: только Art	Отчет по городам	Анализ скидок
16	Фильтр: только заказы со скидкой > 15%	Анализ категорий	Отчет по сегментам
17	Фильтр по менеджеру: конкретное Person	Статистика доставки	Анализ прибыли
18	Фильтр: заказы 1-го квартала 2016	Отчет по регионам	Анализ возвратов
19	Фильтр по марже: (Profit/Sales) > 0.2	Анализ по штатам	Отчет по категориям
20	Фильтр: срочная доставка (Same Day)	Статистика по клиентам	Анализ продаж

Требования к отчету

1. Титульный лист.
2. Цель и задачи работы.
3. Описание архитектуры решения.
4. Скриншоты настроек компонентов.
5. SQL-скрипты создания структур данных.
6. Примеры обработанных данных. Схему трансформаций в Pentaho.
7. Выводы.

Критерии оценки

- Корректность работы с источниками (25 баллов).
- Обработка дублей (25 баллов).

- Обработка ошибок (25 баллов).
- Оформление отчета (25 баллов).

Контрольные вопросы

1. Что такое динамические соединения в PDI?
2. Как организовать обработку ошибок в трансформации?
3. Какие методы выявления дублей существуют?
4. Как настроить параметризацию подключений?
5. Какие компоненты PDI используются для объединения данных?