Домашнее задание 8. Специфика применения ETL в различных предметных сферах

1. Скачайте файлы boking.csv, client.csv и hotel.csv;

2. Создайте новый dag;

3. Создайте три оператора для получения данных и загрузите файлы. Передайте дата фреймы в оператор трансформации;

4. Создайте оператор который будет трансформировать данные:

— Объедините все таблицы в одну;

— Приведите даты к одному виду;

— Удалите невалидные колонки;

— Приведите все валюты к одной;

5. Создайте оператор загрузки в базу данных;

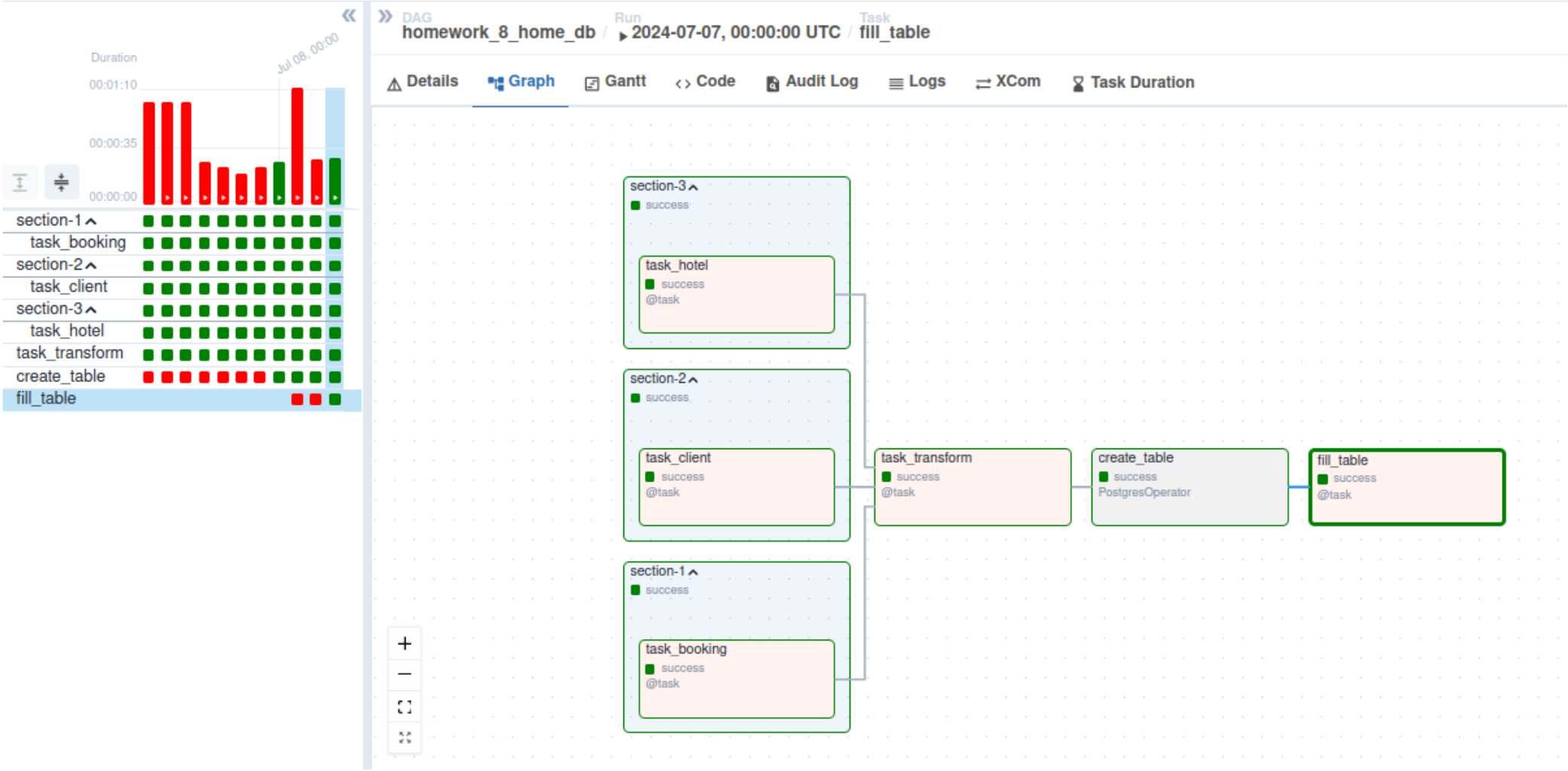
6. Запустите dag.

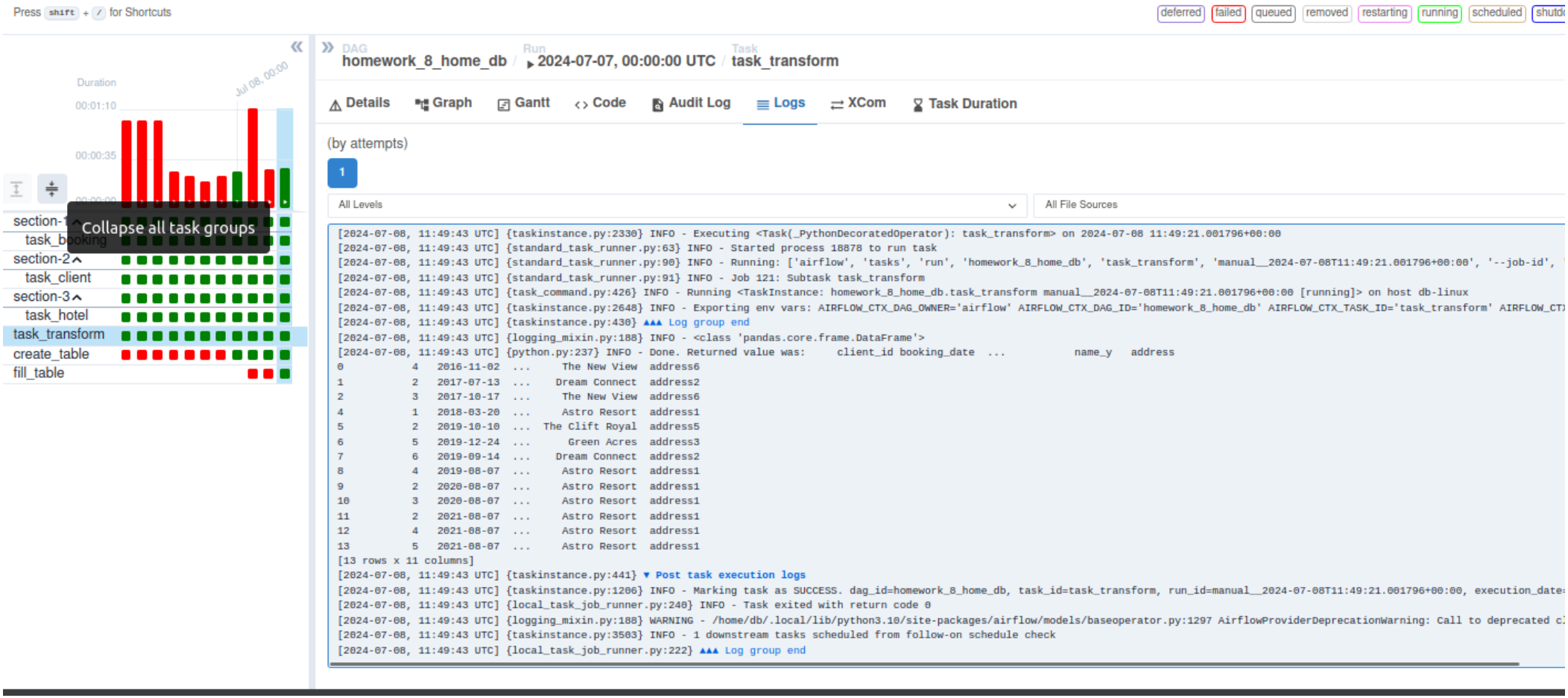
```
from airflow import DAG
from airflow.decorators import task
from datetime import datetime
import pandas as pd
import os
from airflow.utils.task_group import TaskGroup
from airflow.providers.postgres.operators.postgres import PostgresOperator
from airflow.providers.postgres.hooks.postgres import PostgresHook
from airflow.hooks.base_hook import BaseHook

connection = BaseHook.get_connection("postgres_conn")
with DAG(
    "homework_8_home_db",
    start_date=datetime(2024, 6, 18),
    schedule="@daily",
    catchup=False,
```

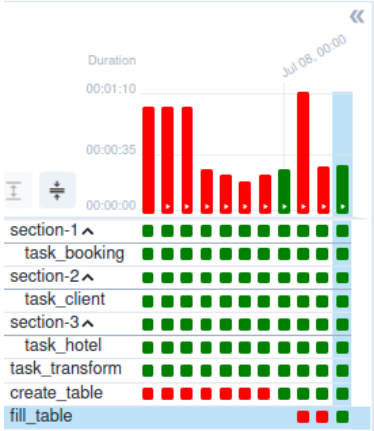
```
    tags=["homework"],
) as dag:
    @task(task_id="task_booking")
    def get_data_booking():
        try:
            df_booking = pd.read_csv(
                "https://gbcdn.mrgcdn.ru/uploads/asset/5551670/attachment/6257a083503973164c0bb0571d41d9e8.csv"
            )
            return df_booking
        except Exception as e:
            print("Exception booking:", e)

    @task(task_id="task_client")
    def get_data_client():
        try:
            df_client = pd.read_csv(
                "https://gbcdn.mrgcdn.ru/uploads/asset/5551674/attachment/7c6bf202bd10996ca60a2593f755d4f4.csv"
            )
            return df_client
        except Exception as e:
            print("Exception client:", e)

    @task(task_id="task_hotel")
    def get_data_hotel():
        try:
            df_hotel = pd.read_csv(
                "https://gbcdn.mrgcdn.ru/uploads/asset/5551688/attachment/3ed446d2c750d05b6c177f62641af670.csv"
            )
            return df_hotel
        except Exception as e:
            print("Exception hotel:", e)
```

```python
    @task(task_id="task_transform")
    def transform_data(data_1, data_2, data_3):
        df_new = data_1.merge(data_2, left_on="client_id", right_on="client_id").merge(
            data_3, left_on="hotel_id", right_on="hotel_id"
        )
        euro_exchange_rate = 0.845522
        df_new["booking_date"] = df_new["booking_date"].str.replace("/", "-")
        df_new.loc[df_new["currency"] == "EUR", "booking_cost"] = (
            df_new["booking_cost"] * euro_exchange_rate
        )
        df_new.loc[df_new["currency"] == "EUR", "currency"] = "GBP"
        df_new = df_new[df_new["booking_cost"].notna()]
        filename = "/home/db/airflow/hw/hw_8.csv"
        if not os.path.exists("/home/db/airflow/hw"):
            os.mkdir("/home/db/airflow/hw")
        df_new.to_csv(filename)
        return df_new

    def dag_with_taskgroup():
        with TaskGroup("section-1") as data_booking:
            data_1 = get_data_booking()
        with TaskGroup("section-2") as data_client:
            data_2 = get_data_client()
        with TaskGroup("section-3") as data_hotel:
            data_3 = get_data_hotel()
        result = transform_data(data_1, data_2, data_3)
        print(type(result))
        create_table = PostgresOperator(
            task_id="create_table",
            postgres_conn_id="postgres_conn",
            sql="""
            DROP TABLE IF EXISTS data_hW_8;
```

```
        CREATE TABLE IF NOT EXISTS data_hW_8 (
            "index" NUMERIC PRIMARY KEY,
            "client_id" INTEGER,
            "booking_date" VARCHAR(100),
            "room_type" VARCHAR(100),
            "hotel_id" INTEGER,
            "booking_cost" FLOAT,
            "currency" VARCHAR(100),
            "age" FLOAT,
            "name_x" VARCHAR(100),
            "type" VARCHAR(100),
            "name_y" VARCHAR(100),
            "address" VARCHAR(100)
            );""",
    )

    @task(task_id="fill_table")
    def fill_table():
        postgres_hook = PostgresHook(postgres_conn_id="postgres_conn")
        conn = postgres_hook.get_conn()
        cur = conn.cursor()
        with open("/home/db/airflow/hw/hw_8.csv", "r") as file:
            cur.copy_expert(
                "COPY data_hW_8 FROM STDIN WITH CSV HEADER DELIMITER AS ',' QUOTE '\"'",
                file,
            )
            conn.commit()

    result >> create_table >> fill_table()
dag = dag_with_taskgroup()
```


DAG
homework_8_home_db
Run
2024-07-07, 00:00:00 UTC
Task
fill_table
Details
Graph
Gantt
Code
Audit Log
Logs
XCom
Task Duration
Duration
Jul 08, 00:00
00:01:10
00:00:35
00:00:00
section-1
task_booking
section-2
task_client
section-3
task_hotel
task_transform
create_table
fill_table
section-3
success
task_hotel
success
@task
section-2
success
task_client
success
@task
section-1
success
task_booking
success
@task
task_transform
success
@task
create_table
success
PostgresOperator
fill_table
success
@task


Press shift + / for Shortcuts
deferred
failed
queued
removed
restarting
running
scheduled
DAG
homework_8_home_db
Run
2024-07-07, 00:00:00 UTC
Task
task_transform
Duration
00:01:10
00:00:35
Jul 08, 00:00
Collapse all task groups
section-1
task_booking
section-2
task_client
section-3
task_hotel
task_transform
create_table
fill_table
Details
Graph
Gantt
Code
Audit Log
Logs
XCom
Task Duration
(by attempts)
1
All Levels
All File Sources
[2024-07-08, 11:49:43 UTC] {taskinstance.py:2330} INFO - Executing <Task(_PythonDecoratedOperator): task_transform> on 2024-07-08 11:49:21.001796+00:00
[2024-07-08, 11:49:43 UTC] {standard_task_runner.py:63} INFO - Started process 18878 to run task
[2024-07-08, 11:49:43 UTC] {standard_task_runner.py:90} INFO - Running: ['airflow', 'tasks', 'run', 'homework_8_home_db', 'task_transform', 'manual__2024-07-08T11:49:21.001796+00:00', '--job-id',
[2024-07-08, 11:49:43 UTC] {standard_task_runner.py:91} INFO - Job 121: Subtask task_transform
[2024-07-08, 11:49:43 UTC] {task_command.py:426} INFO - Running <TaskInstance: homework_8_home_db.task_transform manual__2024-07-08T11:49:21.001796+00:00 [running]> on host db-linux
[2024-07-08, 11:49:43 UTC] {taskinstance.py:2648} INFO - Exporting env vars: AIRFLOW_CTX_DAG_OWNER='airflow' AIRFLOW_CTX_DAG_ID='homework_8_home_db' AIRFLOW_CTX_TASK_ID='task_transform' AIRFLOW_CT
[2024-07-08, 11:49:43 UTC] {taskinstance.py:430} ▲▲▲ Log group end
[2024-07-08, 11:49:43 UTC] {logging_mixin.py:188} INFO - <class 'pandas.core.frame.DataFrame'>
[2024-07-08, 11:49:43 UTC] {python.py:237} INFO - Done. Returned value was: client_id booking_date ... name_y address
0 4 2016-11-02 ... The New View address6
1 2 2017-07-13 ... Dream Connect address2
2 3 2017-10-17 ... The New View address6
4 1 2018-03-20 ... Astro Resort address1
5 2 2019-10-10 ... The Clift Royal address5
6 5 2019-12-24 ... Green Acres address3
7 6 2019-09-14 ... Dream Connect address2
8 4 2019-08-07 ... Astro Resort address1
9 2 2020-08-07 ... Astro Resort address1
10 3 2020-08-07 ... Astro Resort address1
11 2 2021-08-07 ... Astro Resort address1
12 4 2021-08-07 ... Astro Resort address1
13 5 2021-08-07 ... Astro Resort address1
[13 rows x 11 columns]
[2024-07-08, 11:49:43 UTC] {taskinstance.py:441} ▼ Post task execution logs
[2024-07-08, 11:49:43 UTC] {taskinstance.py:1206} INFO - Marking task as SUCCESS. dag_id=homework_8_home_db, task_id=task_transform, run_id=manual__2024-07-08T11:49:21.001796+00:00, execution_date=
[2024-07-08, 11:49:43 UTC] {local_task_job_runner.py:240} INFO - Task exited with return code 0
[2024-07-08, 11:49:43 UTC] {logging_mixin.py:188} WARNING - /home/db/.local/lib/python3.10/site-packages/airflow/models/baseoperator.py:1297 AirflowProviderDeprecationWarning: Call to deprecated c
[2024-07-08, 11:49:43 UTC] {taskinstance.py:3503} INFO - 1 downstream tasks scheduled from follow-on schedule check
[2024-07-08, 11:49:43 UTC] {local_task_job_runner.py:222} ▲▲▲ Log group end



DAG homework_8_home_db / Run 2024-07-07, 00:00:00 UTC / Task fill_table

Clear t

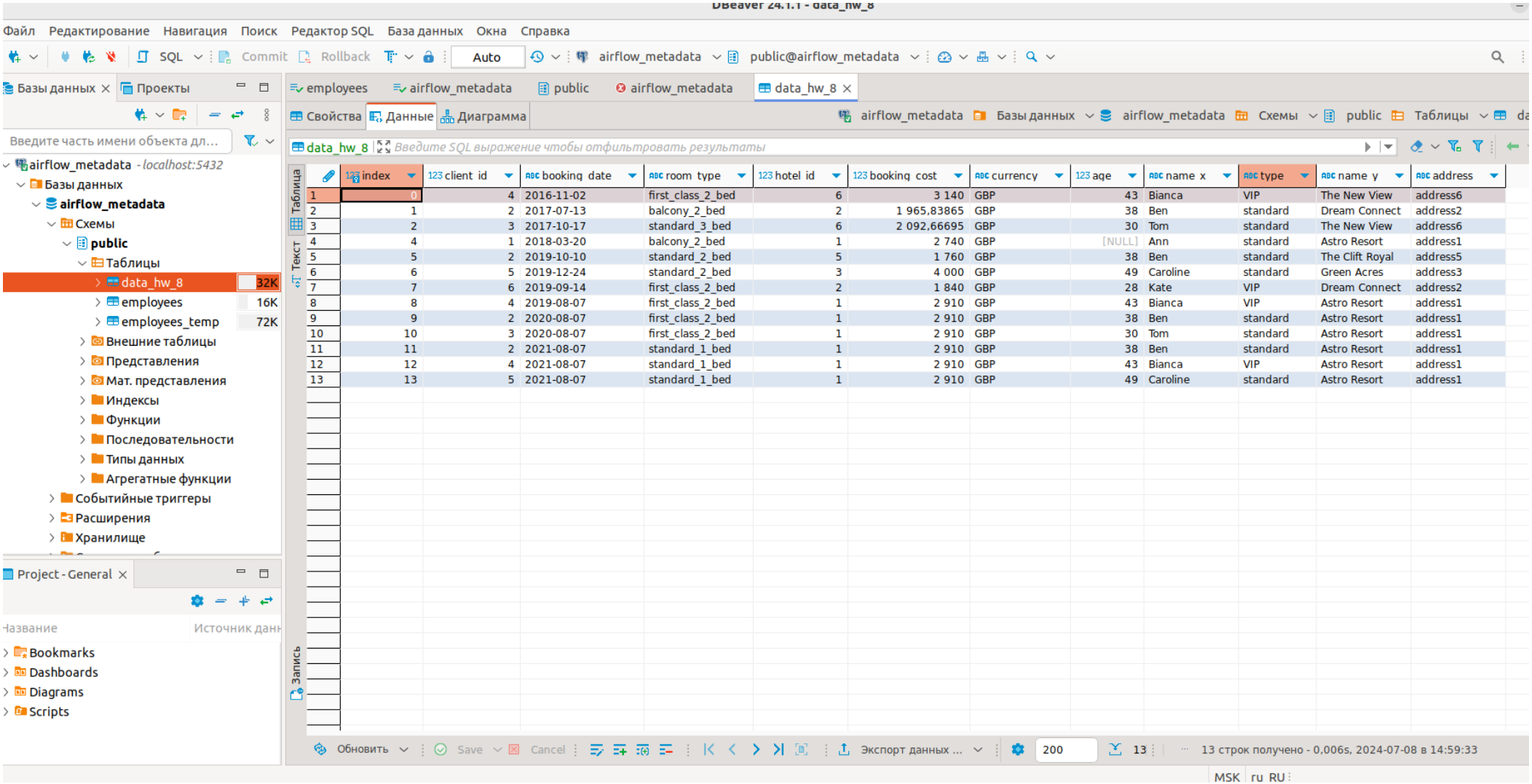
Details | Graph | Gantt | Code | Audit Log | Logs | XCom | Task Duration

(by attempts)

1

All Levels | All File Sources

```
db-linux
*** Found local files:
***   * /home/db/airflow/logs/dag_id=homework_8_home_db/run_id=manual__2024-07-08T11:49:21.001796+00:00/task_id=fill_table/attempt=1.log
[2024-07-08, 11:49:50 UTC] {local_task_job_runner.py:120} ▼ Pre task execution logs
[2024-07-08, 11:49:50 UTC] {taskinstance.py:2076} INFO - Dependencies all met for dep_context=non-requeueable deps ti=<TaskInstance: homework_8_home_db.fill_table manual__2024-07-08T11:49:21.001796+00:00 [queued]>
[2024-07-08, 11:49:50 UTC] {taskinstance.py:2076} INFO - Dependencies all met for dep_context=requeueable deps ti=<TaskInstance: homework_8_home_db.fill_table manual__2024-07-08T11:49:21.001796+00:00 [queued]>
[2024-07-08, 11:49:50 UTC] {taskinstance.py:2306} INFO - Starting attempt 1 of 1
[2024-07-08, 11:49:50 UTC] {taskinstance.py:2330} INFO - Executing <Task(_PythonDecoratedOperator): fill_table> on 2024-07-08 11:49:21.001796+00:00
[2024-07-08, 11:49:50 UTC] {standard_task_runner.py:63} INFO - Started process 18892 to run task
[2024-07-08, 11:49:50 UTC] {standard_task_runner.py:90} INFO - Running: ['airflow', 'tasks', 'run', 'homework_8_home_db', 'fill_table', 'manual__2024-07-08T11:49:21.001796+00:00', '--job-id', '123', '--raw', '--subdir', '
[2024-07-08, 11:49:50 UTC] {standard_task_runner.py:91} INFO - Job 123: Subtask fill_table
[2024-07-08, 11:49:50 UTC] {task_command.py:426} INFO - Running <TaskInstance: homework_8_home_db.fill_table manual__2024-07-08T11:49:21.001796+00:00 [running]> on host db-linux
[2024-07-08, 11:49:50 UTC] {taskinstance.py:2648} INFO - Exporting env vars: AIRFLOW_CTX_DAG_OWNER='airflow' AIRFLOW_CTX_DAG_ID='homework_8_home_db' AIRFLOW_CTX_TASK_ID='fill_table' AIRFLOW_CTX_EXECUTION_DATE='2024-07-08T
[2024-07-08, 11:49:50 UTC] {taskinstance.py:430} ▲▲▲ Log group end
[2024-07-08, 11:49:50 UTC] {base.py:84} INFO - Using connection ID 'postgres_conn' for task execution.
[2024-07-08, 11:49:50 UTC] {python.py:237} INFO - Done. Returned value was: None
[2024-07-08, 11:49:50 UTC] {taskinstance.py:441} ▼ Post task execution logs
[2024-07-08, 11:49:50 UTC] {taskinstance.py:1206} INFO - Marking task as SUCCESS. dag_id=homework_8_home_db, task_id=fill_table, run_id=manual__2024-07-08T11:49:21.001796+00:00, execution_date=20240708T114921, start_date=
[2024-07-08, 11:49:50 UTC] {local_task_job_runner.py:240} INFO - Task exited with return code 0
[2024-07-08, 11:49:50 UTC] {taskinstance.py:3503} INFO - 0 downstream tasks scheduled from follow-on schedule check
[2024-07-08, 11:49:50 UTC] {local_task_job_runner.py:222} ▲▲▲ Log group end
```



Решила оставить столбцы с id, убрала лишь строки где были отсутствующие значения по стоимости.