Чек-лист по установке ClickHouse + PostgreSQL + Python

Академия Информационных Систем

Подготовка окружения

Windows

✅ Установлен WSL2 (рекомендуется) ИЛИ используется нативный Docker Desktop

```
# В PowerShell (администратор)
wsl --install

# Перезагрузить компьютер
```

✅ Docker Desktop установлен и запущен (иконка в трее)

✅ В настройках Docker включен WSL2 backend

✅ Python установлен (проверить: `python --version` в cmd)

✅ Git Bash или Windows Terminal (опционально, для удобства)

Linux (Ubuntu/Debian)

✅ Docker установлен

```
sudo apt update
sudo apt install docker.io docker-compose
sudo systemctl start docker

sudo usermod -aG docker $USER  # выйти/зайти заново
```

✔ Python 3.9+ установлен

```
python3 --version  # должно быть 3.9+

sudo apt install python3-pip python3-venv
```

✔ PostgreSQL клиент (опционально)

```
sudo apt install postgresql-client
```

macOS

✔ Docker Desktop for Mac установлен

✔ Домашняя директория расшарена в Docker (Settings → Resources → File Sharing)

✔ Python установлен через brew: `brew install python@3.11`

Установка ClickHouse (через Docker)

Шаг 1 Создание docker-compose.yml

✔ Создана папка проекта (например, bigdata_course)

✔ В папке создан файл docker-compose.yml со следующим содержимым:

```
version: '3.8'

services:
  clickhouse:
    image: clickhouse/clickhouse-server:latest
    container_name: clickhouse_server
    hostname: clickhouse
    ports:
      - "8123:8123"   # HTTP API
```

```
    - "9000:9000"   # Native protocol (clickhouse-client)
  volumes:
    - ./clickhouse_data:/var/lib/clickhouse
    - ./clickhouse_logs:/var/log/clickhouse-server
  environment:
    CLICKHOUSE_DB: course_db
    CLICKHOUSE_USER: course_user
    CLICKHOUSE_PASSWORD: course_pass
  ulimits:
    nofile:
      soft: 262144
      hard: 262144

postgres:
  image: postgres:15
  container_name: postgres_server
  environment:
    POSTGRES_DB: course_db
    POSTGRES_USER: course_user
    POSTGRES_PASSWORD: course_pass
  ports:
    - "5432:5432"
  volumes:
    - ./postgres_data:/var/lib/postgresql/data

adminer:
  image: adminer
  container_name: adminer_ui
  ports:
    - "8080:8080"
  depends_on:
    - postgres
    - clickhouse
```

Шаг 2 Запуск контейнеров

✔ В терминале перейти в папку с docker-compose.yml

✔ Выполнить: docker-compose up -d

✔ Проверить, что контейнеры запущены: docker ps
Должны быть три контейнера: clickhouse_server, postgres_server, adminer_ui

Шаг 3 Проверка ClickHouse

✔ Подключиться через клиент: docker exec -it clickhouse_server clickhouse-client --user course_user --password course_pass

✔ Выполнить тестовый запрос:

```
SHOW DATABASES;

SELECT version();
```

✔ Выйти: EXIT;

Шаг 4 Проверка PostgreSQL

✔ Подключиться:

```
docker exec -it postgres_server psql -U course_user -d course_db
```

✔ Выполнить:

```
\l  -- список БД

SELECT version();
```

✔ Выйти: `\q`

Шаг 5 Проверка Adminer (веб-интерфейс)

✔ Открыть браузер: `http://localhost:8080`

Для ClickHouse:

✔ Система: ClickHouse

✔ Сервер: `clickhouse`

✔ Имя пользователя: `course_user`

✔ Пароль: `course_pass`

✔ База данных: `course_db`

Для PostgreSQL:

✔ Система: PostgreSQL

✔ Сервер: `postgres`

✔ Остальное аналогично

Установка Python и библиотек

Шаг 1 Создание виртуального окружения

В папке проекта создать venv:

```
# Windows (cmd)
python -m venv venv
venv\Scripts\activate

# Linux/macOS
python3 -m venv venv

source venv/bin/activate
```

Шаг 2 Установка необходимых пакетов

✔ Создать файл requirements.txt:

```
clickhouse-driver==0.2.6
psycopg2-binary==2.9.9
pandas==2.1.4
prophet==1.1.5
sqlalchemy==2.0.23
python-dotenv==1.0.0
jupyter==1.0.0
matplotlib==3.8.2

seaborn==0.13.0
```

✔ Установить:

```
pip install -r requirements.txt
```

Шаг 3 Проверка соединения из Python

✔ Создать файл test_connection.py:

```
import os
from clickhouse_driver import Client
import psycopg2

# ClickHouse
ch_client = Client(
    host='localhost',
    port=9000,
    user='course_user',
    password='course_pass',
    database='course_db'
)

try:
    result = ch_client.execute('SELECT 1 as test')
    print(f"✅ ClickHouse OK: {result}")
except Exception as e:
    print(f"❌ ClickHouse error: {e}")

# PostgreSQL
try:
    pg_conn = psycopg2.connect(
        host='localhost',
        port=5432,
        database='course_db',
        user='course_user',
        password='course_pass'
    )
    cur = pg_conn.cursor()
    cur.execute('SELECT 1 as test')
    print(f"✅ PostgreSQL OK: {cur.fetchone()}")
    cur.close()
```

```
    pg_conn.close()
except Exception as e:
    print(f"❌ PostgreSQL error: {e}")
```

✔ Запустить: python test_connection.py

✔ Должны увидеть два "✅"

Создание тестовых данных

Шаг 1 Создание таблиц

✔ Файл init_tables.py:

```
from clickhouse_driver import Client
import psycopg2

ch = Client(host='localhost', user='course_user', password='course_pass',
database='course_db')

# ClickHouse таблица
ch.execute('''
CREATE TABLE IF NOT EXISTS sales (
    sale_id UInt64,
    product_id UInt64,
    sale_date Date,
    amount Float64,
    quantity UInt32
) ENGINE = MergeTree()
ORDER BY (sale_date, product_id)
PARTITION BY toYYYYMM(sale_date)
''')

# PostgreSQL таблица
pg_conn = psycopg2.connect(
```

```
    host='localhost', database='course_db',
    user='course_user', password='course_pass'
)
pg_conn.autocommit = True
pg_cur = pg_conn.cursor()
pg_cur.execute('''
CREATE TABLE IF NOT EXISTS products (
    product_id SERIAL PRIMARY KEY,
    product_name VARCHAR(200),
    category VARCHAR(100),
    price DECIMAL(10,2)
)
''')
pg_cur.close()
pg_conn.close()

print("✅ Таблицы созданы")
```

Шаг 2 Загрузка тестовых данных

✔ Файл `load_sample_data.py`:

```
from clickhouse_driver import Client
import psycopg2
import pandas as pd
from datetime import datetime, timedelta

# PostgreSQL - товары
pg_conn = psycopg2.connect(
    host='localhost', database='course_db',
    user='course_user', password='course_pass'
)
pg_cur = pg_conn.cursor()

products_data = [
```

```
    (1, 'Ноутбук', 'Электроника', 50000),
    (2, 'Мышь', 'Электроника', 1500),
    (3, 'Книга SQL', 'Книги', 1200),
    (4, 'Наушники', 'Электроника', 3500),
]

for p in products_data:
    pg_cur.execute('INSERT INTO products VALUES (%s,%s,%s,%s) ON CONFLICT DO
NOTHING', p)
pg_conn.commit()
pg_cur.close()
pg_conn.close()

# ClickHouse - продажи
ch = Client(host='localhost', user='course_user', password='course_pass',
database='course_db')

# Генерация 1000 тестовых продаж
sales = []
start_date = datetime(2024, 1, 1)
for i in range(1, 1001):
    sale_date = start_date + timedelta(days=i % 90)
    sales.append({
        'sale_id': i,
        'product_id': (i % 4) + 1,
        'sale_date': sale_date,
        'amount': products_data[(i % 4)][3] * ((i % 5) + 1) * 0.5,
        'quantity': (i % 10) + 1
    })

ch.execute('INSERT INTO sales VALUES', sales)

print(f"✅ Загружено {len(sales)} записей в ClickHouse")
```

Проверка полного цикла

✔ Запустить последовательно:

python init_tables.py

python load_sample_data.py

✔ Проверить данные в ClickHouse:

```
SELECT count(*) FROM sales;
```

```
SELECT sale_date, sum(amount) FROM sales GROUP BY sale_date ORDER BY sale_date LIMIT 10;
```

✔ Проверить данные в PostgreSQL:

```
SELECT * FROM products;
```

✔ Выполнить JOIN-запрос через Python:

```
from clickhouse_driver import Client
import psycopg2

ch = Client(host='localhost', user='course_user', password='course_pass', database='course_db')

# Получаем продукты из PG через словарь ClickHouse
# или делаем два запроса и объединяем в Python
ch_data = ch.execute('''
    SELECT product_id, sum(amount) as total_sales
    FROM sales
    GROUP BY product_id
''')


print("Продажи по товарам:", ch_data)
```