МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

Санкт-Петербургский национальный исследовательский университет ИТМО

Мегафакультет трансляционных информационных технологий

Факультет информационных технологий и программирования

Лабораторная работа №4

По дисциплине «Технологии управления данными»

Выполнил студент группы №M3306

Бочагова Екатерина

Проверил

Повышев Владислав Вячеславович

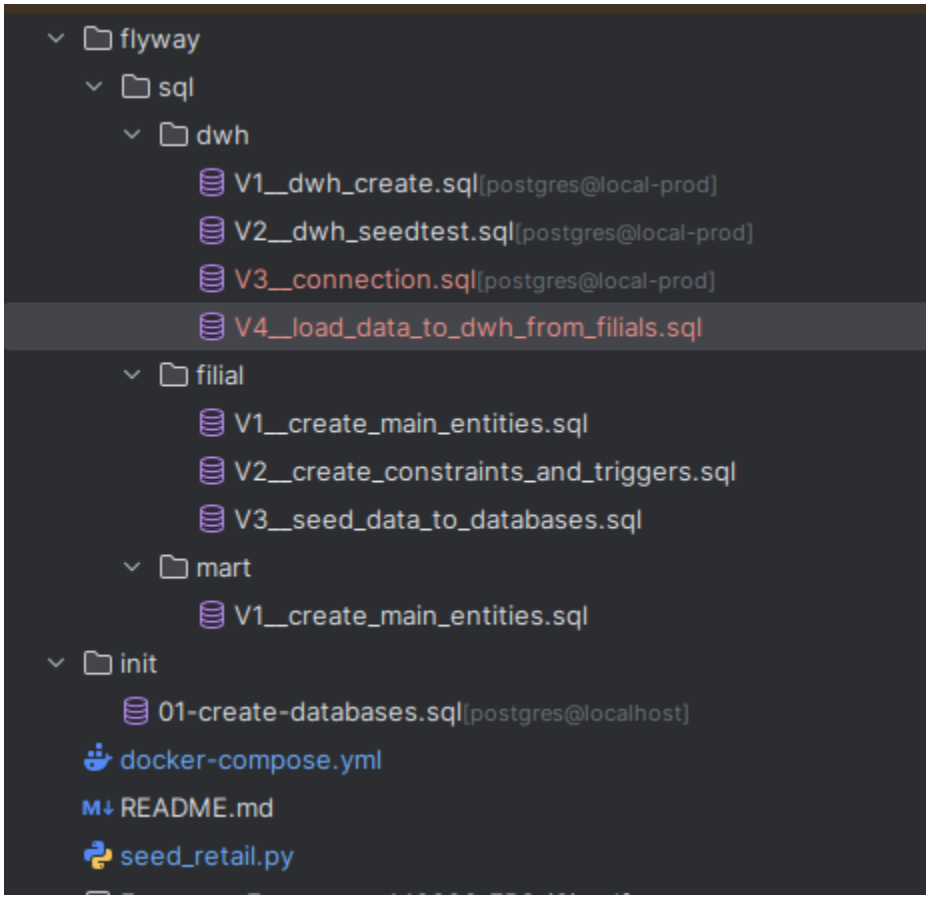
Санкт-Петербург

2025

Условие лабораторной работы

Цель работы: создать хранимую процедуру, или скрипт, обеспечивающую автоматизированный перенос данных из Филиалов в Хранилище.
Процедура должна обеспечить добавление вновь поступившей информации из Филиала Запад, и Филиала Восток в центральное хранилище данных.
Процедура должна исключить дублирование информации.
Разработчик должен продумать механизм обеспечивающий минимализацию нагрузки на продуктивные сервера филиалов.
В процедуре можно использовать операции множественной вставки.

* все скрипты поместим в миграции для dwh - итоговая структура проекта



Подготовка к выполнению работы:

1) Необходимо написать скрипт, при помощи которого будет выполняться подключение к базам данных филиалов
2) Разработаем скрипт, который будет игнорировать дублирующуюся информацию

Что необходимо, чтобы работать с другими базами данных:

- Расширение для подключения к внешним бд (postgres_fdw)
- Создадим удаленный сервер (create server ‘название сервера’ foreign data wrapper postgres_fdw) и пропишем креды через options (host ‘’, dbname ‘’, port ‘’)

* этот удаленный сервер по большому счету не сервер, а конфигурационный объект, который хранит информацию о том, к какой бд нужно подключиться и как

- Создадим пользователя, при помощи которого будем переносить эти данные (create user mapping for postgres server ‘название сервера, созданного на предыдущем этапе’) и так же через options(user ‘’, password ‘’) пропишем креды
- Создадим схемы в которые будем импортировать таблицы на следующем этапе
- Импортируем таблицы (тк в каждой бд все таблицы находятся в схеме public то просто импортируем эту схему)

Итоговый скрипт для подключения к базам данных филиалов:

```
create extension if not exists postgres_fdw;

create server if not exists west foreign data wrapper postgres_fdw
    options (host 'postgres', dbname 'filial_west', port '5436');
create user  mapping if not exists for postgres server west
    options(user 'postgres', password 'password');

create server if not exists east foreign data wrapper postgres_fdw
    options (host 'postgres', dbname 'filial_east', port '5436');
create user mapping if not exists for postgres server east
    options(user 'postgres', password 'password');

create schema if not exists filial_west;
create schema if not exists filial_east;

import foreign schema public from server west into filial_west;
import foreign schema public from server east into filial_east;
```

Проверим работоспособность:
Остановим контейнер командой docker compose down
И перебилдим его: docker compose up --build

```
flyway_west-1  | Schema "public" is up to date. No migration necessary.
flyway_west-1 exited with code 0
flyway_dwh-1 exited with code 0
flyway_east-1 exited with code 0
flyway_mart-1  | Flyway OSS Edition 11.14.1 by Redgate
flyway_mart-1  |
flyway_mart-1  | See release notes here: https://rd.gt/416ObMi
flyway_mart-1  | Database: jdbc:postgresql://postgres:5432/dw_retail (PostgreSQL 16.9)
flyway_mart-1  | Current version of schema "public": 3
flyway_mart-1  | WARNING: Schema "public" has a version (3) that is newer than the latest available migration (1) !
flyway_mart-1  | Schema "public" is up to date. No migration necessary.
flyway_mart-1 exited with code 0
```

Все миграции завершились с кодом 0! Значит, все правильно отработало

Напишем процедуру наполнения хранилища данными из филиалов (с наших серверов)

```
CREATE OR REPLACE PROCEDURE dwh.load_from_branches()
LANGUAGE plpgsql
AS $$
DECLARE
    branch RECORD;
    old_search_path text;
BEGIN
    old_search_path := current_setting('search_path');

    FOR branch IN
        SELECT * FROM (
            VALUES
                ('West', 'filial_west'),
                ('East', 'filial_east')
        ) AS t(branch_name, schema_name)
    LOOP
        RAISE NOTICE '==== Загрузка из филиала: % (% схема) ====',
branch.branch_name, branch.schema_name;

        PERFORM set_config('search_path', format('%I,public',
branch.schema_name), true);

        INSERT INTO dwh.dim_customer (customer_uid, customer_name,
source_system)
        SELECT c.customer_uid, c.customer_name, branch.branch_name
        FROM   customer AS c
        WHERE NOT EXISTS (
            SELECT 1 FROM dwh.dim_customer dc
            WHERE dc.customer_uid = c.customer_uid
        );

        INSERT INTO dwh.dim_category (category_uid, category_name,
source_system)
        SELECT c.category_uid, c.category_name, branch.branch_name
        FROM   category AS c
        WHERE NOT EXISTS (
            SELECT 1 FROM dwh.dim_category dc
            WHERE dc.category_uid = c.category_uid
        );

        INSERT INTO dwh.dim_product (product_uid, product_name, sku,
source_system)
        SELECT p.product_uid, p.product_name, p.sku, branch.branch_name
        FROM   product AS p
        WHERE NOT EXISTS (
            SELECT 1 FROM dwh.dim_product dp
            WHERE dp.product_uid = p.product_uid
        );
```

```
        INSERT INTO dwh.dim_product_category (product_key, category_key,
source_uid)
        SELECT
            dp.product_key,
            dc.category_key,
            pc.uid_product_category
        FROM product_category AS pc
        JOIN dwh.dim_product dp ON dp.product_uid = (
            SELECT product_uid
            FROM product p2
            WHERE p2.id = pc.product_id
        )
        JOIN dwh.dim_category dc ON dc.category_uid = (
            SELECT category_uid
            FROM category c2
            WHERE c2.id = pc.category_id
        )
        ON CONFLICT DO NOTHING;

        INSERT INTO dwh.dim_sale (sale_uid, sale_number, source_system)
        SELECT s.sale_uid, s.id::text, branch.branch_name
        FROM sale AS s
        WHERE NOT EXISTS (
            SELECT 1 FROM dwh.dim_sale ds
            WHERE ds.sale_uid = s.sale_uid
        );

        INSERT INTO dwh.fact_sales (
            customer_key,
            product_key,
            branch_key,
            sale_key,
            sale_date_key,
            sale_uid,
            sale_item_uid,
            quantity,
            unit_price,
            sale_date,
            source_system
        )
        SELECT
            dc.customer_key,
            dp.product_key,
            (SELECT branch_key FROM dwh.dim_branch b WHERE b.database_name =
branch.schema_name),
            ds.sale_key,
            TO_CHAR(s.sale_date, 'YYYYMMDD')::int,
            s.sale_uid,
```

```
            si.sale_item_uid,
            si.quantity,
            si.unit_price,
            s.sale_date,
            branch.branch_name
        FROM sale_item AS si
        JOIN sale AS s ON s.id = si.sale_id
        JOIN dwh.dim_customer dc ON dc.customer_uid = (
            SELECT customer_uid FROM customer c WHERE c.id = s.customer_id
        )
        JOIN dwh.dim_product dp ON dp.product_uid = (
            SELECT product_uid FROM product p WHERE p.id = si.product_id
        )
        JOIN dwh.dim_sale ds ON ds.sale_uid = s.sale_uid
        WHERE NOT EXISTS (
            SELECT 1 FROM dwh.fact_sales fs
            WHERE fs.sale_item_uid = si.sale_item_uid
        );

    END LOOP;

    PERFORM set_config('search_path', old_search_path, true);

    RAISE NOTICE 'Загрузка успешно завершена!';
END;
$$;
```

Аналогично первому пункту проверим работоспособность:

```
 ✔ Container db-branch-of-a-trading-enterprise-flyway_west-1  Created                                    0.
 ✔ Container db-branch-of-a-trading-enterprise-flyway_east-1  Created                                    0.
 ✔ Container db-branch-of-a-trading-enterprise-flyway_mart-1  Created                                    0.
Attaching to flyway_dwh-1, flyway_east-1, flyway_mart-1, flyway_west-1, postgres-1
postgres-1  |
postgres-1  | PostgreSQL Database directory appears to contain a database; Skipping initialization
postgres-1  |
postgres-1  | 2025-11-18 17:47:29.941 UTC [1] LOG:  starting PostgreSQL 16.9 (Debian 16.9-1.pgdg130+1) on x8
4-pc-linux-gnu, compiled by gcc (Debian 14.2.0-19) 14.2.0, 64-bit
postgres-1  | 2025-11-18 17:47:29.941 UTC [1] LOG:  listening on IPv4 address "0.0.0.0", port 5432
postgres-1  | 2025-11-18 17:47:29.941 UTC [1] LOG:  listening on IPv6 address "::", port 5432
postgres-1  | 2025-11-18 17:47:29.943 UTC [1] LOG:  listening on Unix socket "/var/run/postgresql/.s.PGSQL.5
"
postgres-1  | 2025-11-18 17:47:29.946 UTC [29] LOG:  database system was shut down at 2025-11-18 17:47:09 UT
postgres-1  | 2025-11-18 17:47:29.949 UTC [1] LOG:  database system is ready to accept connections
flyway_west-1  | Flyway OSS Edition 11.14.1 by Redgate
flyway_west-1  |
flyway_west-1  | See release notes here: https://rd.gt/416ObMi
flyway_east-1  | Flyway OSS Edition 11.14.1 by Redgate
flyway_east-1  |
flyway_east-1  | See release notes here: https://rd.gt/416ObMi
flyway_dwh-1   | Flyway OSS Edition 11.14.1 by Redgate
flyway_dwh-1   |
flyway_dwh-1   | See release notes here: https://rd.gt/416ObMi
flyway_west-1  | Database: jdbc:postgresql://postgres:5432/filial_west (PostgreSQL 16.9)
flyway_west-1  | Successfully validated 3 migrations (execution time 00:00.022s)
flyway_dwh-1   | Database: jdbc:postgresql://postgres:5432/dw_retail (PostgreSQL 16.9)
flyway_east-1  | Database: jdbc:postgresql://postgres:5432/filial_east (PostgreSQL 16.9)
flyway_west-1  | Current version of schema "public": 3
flyway_west-1  | Schema "public" is up to date. No migration necessary.
flyway_east-1  | Successfully validated 3 migrations (execution time 00:00.019s)
flyway_dwh-1   | Successfully validated 4 migrations (execution time 00:00.022s)
flyway_east-1  | Current version of schema "public": 3
flyway_east-1  | Schema "public" is up to date. No migration necessary.
flyway_dwh-1   | Current version of schema "public": 4
flyway_dwh-1   | Schema "public" is up to date. No migration necessary.
flyway_east-1 exited with code 0
flyway_west-1 exited with code 0
flyway_dwh-1 exited with code 0
flyway_mart-1  | Flyway OSS Edition 11.14.1 by Redgate
flyway_mart-1  |
flyway_mart-1  | See release notes here: https://rd.gt/416ObMi
flyway_mart-1  | Database: jdbc:postgresql://postgres:5432/dw_retail (PostgreSQL 16.9)
flyway_mart-1  | Current version of schema "public": 4
flyway_mart-1  | WARNING: Schema "public" has a version (4) that is newer than the latest available migratio
1) !
flyway_mart-1  | Schema "public" is up to date. No migration necessary.
flyway_mart-1 exited with code 0
```