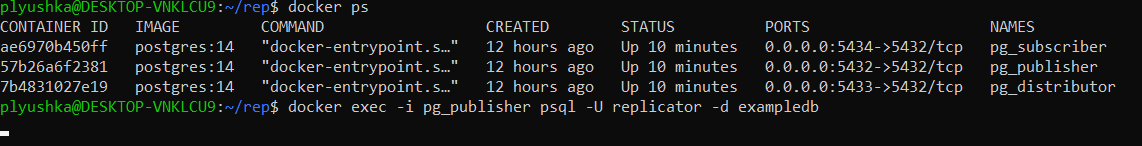
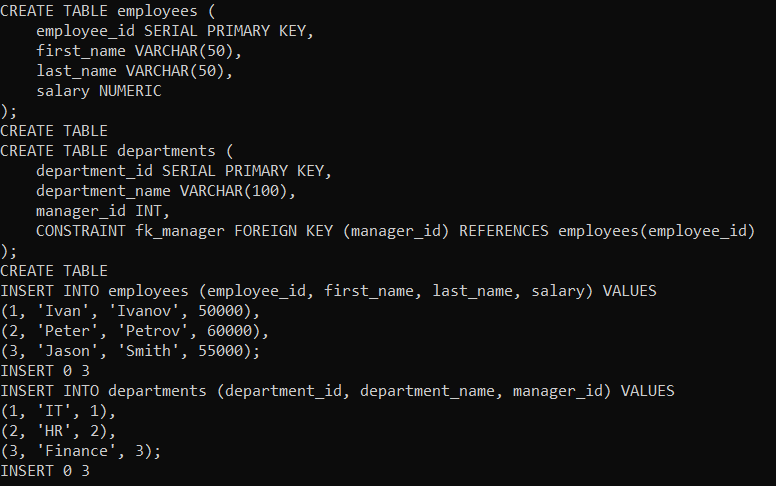
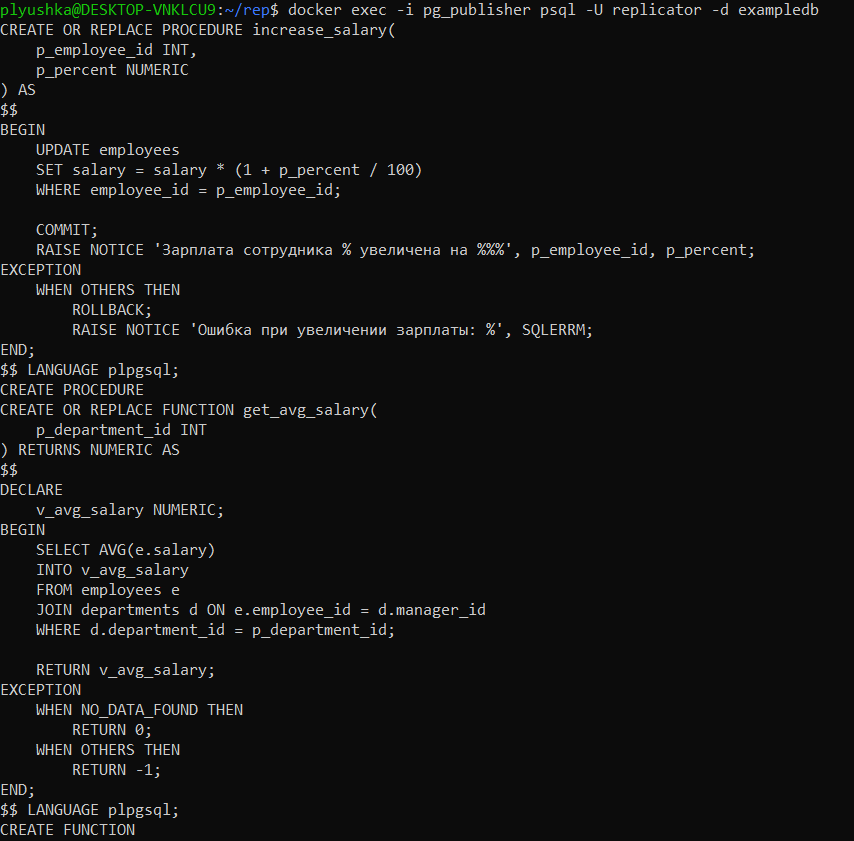
В данной лабораторной работе была настроена логическая репликация между несколькими контейнерами PostgreSQL. Логическая репликация позволяет реплицировать данные на уровне строк и таблиц, что полезно для распределённых систем и резервного копирования.



Были созданы две таблицы: employees и departments. Данные были вставлены в таблицы. Была создана процедура для увеличения зарплаты сотрудника на определённый процент.

|  |
| --- |
| CREATE TABLE employees (  employee\_id SERIAL PRIMARY KEY,  first\_name VARCHAR(50),  last\_name VARCHAR(50),  salary NUMERIC  );  CREATE TABLE departments (  department\_id SERIAL PRIMARY KEY,  department\_name VARCHAR(100),  manager\_id INT,  CONSTRAINT fk\_manager FOREIGN KEY (manager\_id) REFERENCES employees(employee\_id)  );  INSERT INTO employees (employee\_id, first\_name, last\_name, salary) VALUES  (1, 'Ivan', 'Ivanov', 50000),  (2, 'Peter', 'Petrov', 60000),  (3, 'Jason', 'Smith', 55000);  INSERT INTO departments (department\_id, department\_name, manager\_id) VALUES  (1, 'IT', 1),  (2, 'HR', 2),  (3, 'Finance', 3);  CREATE OR REPLACE PROCEDURE increase\_salary(  p\_employee\_id INT,  p\_percent NUMERIC  ) AS  $$  BEGIN  UPDATE employees  SET salary = salary \* (1 + p\_percent / 100)  WHERE employee\_id = p\_employee\_id;    COMMIT;  RAISE NOTICE 'Зарплата сотрудника % увеличена на %%%', p\_employee\_id, p\_percent;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK;  RAISE NOTICE 'Ошибка при увеличении зарплаты: %', SQLERRM;  END;  $$ LANGUAGE plpgsql; |





**Настройка логической репликации**

На стороне **издателя** (Publisher)была создана публикация для таблиц employees и departments.

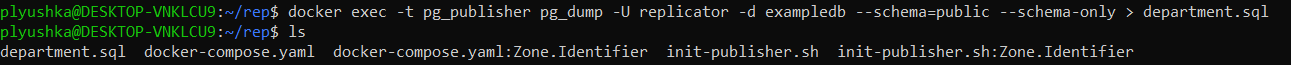
CREATE PUBLICATION pub\_orders FOR TABLE employees, departments;



Схема таблиц была экспортирована из издателя и импортирована в **распределитель** (Distributor).

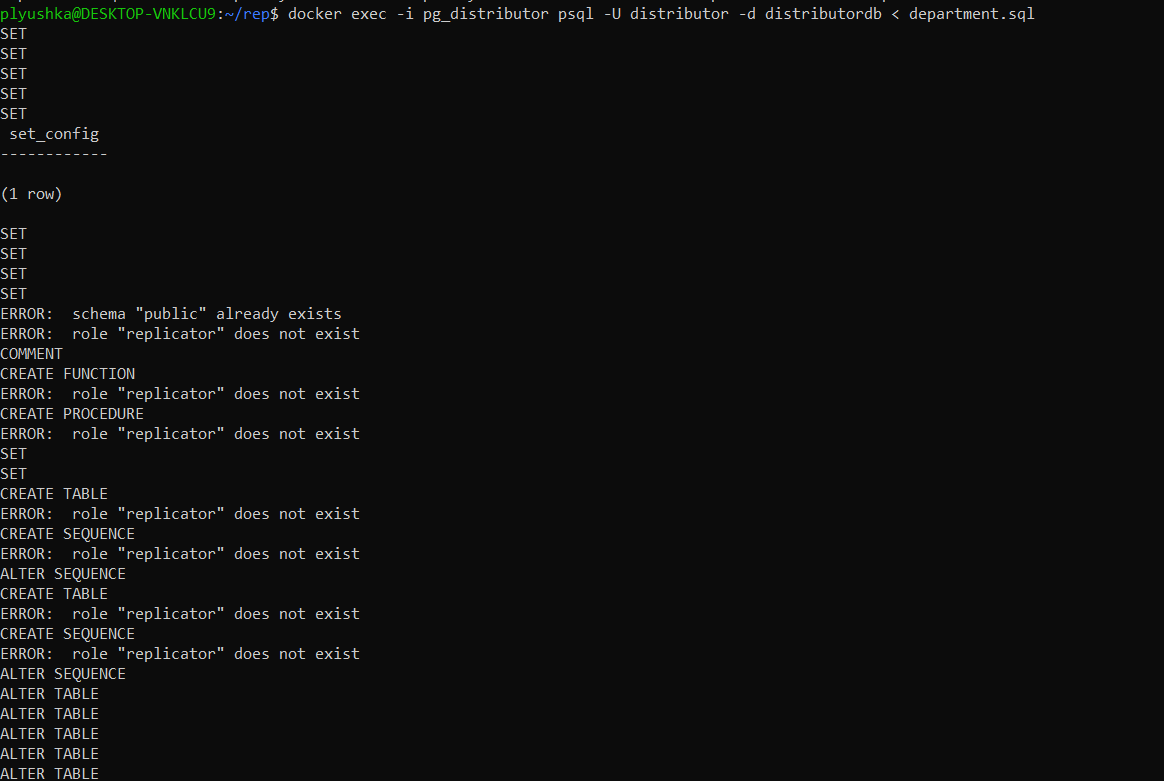
docker exec -t pg\_publisher pg\_dump -U replicator -d exampledb --schema=public --schema-only > department.sql

**pg\_dump** экспортирует схему таблиц в файл department.sql.



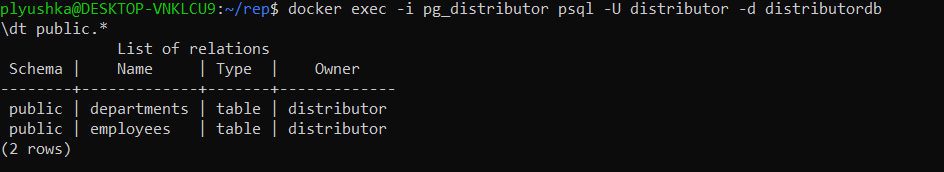
docker exec -i pg\_distributor psql -U distributor -d distributordb < department.sql

**psql**импортирует схему в базу данных распределителя.



docker exec -i pg\_distributor psql -U distributor -d distributordb

\dt public.\*

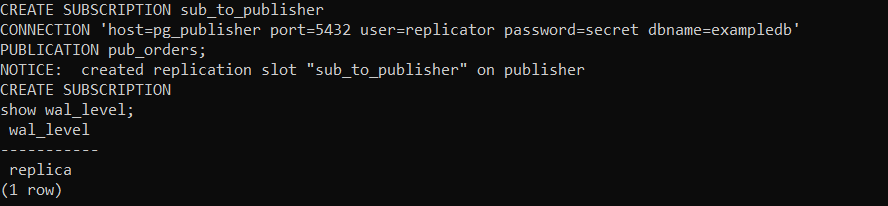


На стороне распределителя была создана **подписка** на публикацию издателя.

CREATE SUBSCRIPTION sub\_to\_publisher

CONNECTION 'host=pg\_publisher port=5432 user=replicator password=secret dbname=exampledb'

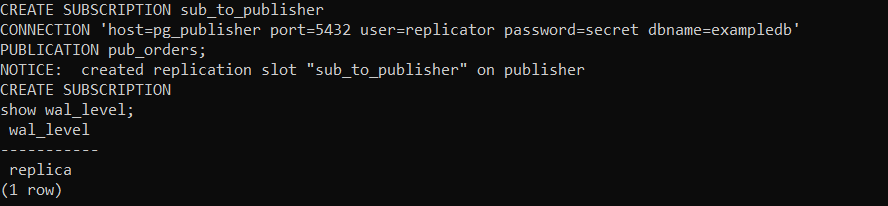
PUBLICATION pub\_orders;



**Установка уровня WAL**

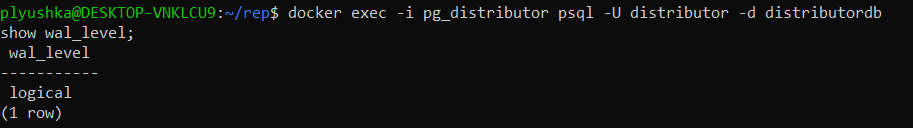
Для работы логической репликации необходимо установить уровень WAL (Write-Ahead Logging) в значение logical.

**wal\_level** = 'logical' позволяет PostgreSQL записывать изменения в WAL в формате, пригодном для логической репликации.



ALTER SYSTEM SET wal\_level = 'logical';





**minimal -** Минимальный уровень, используется для восстановления после сбоев, но не поддерживает репликацию.

**replica -** Используется для потоковой (физической) репликации, включает данные для воспроизведения операций.

**logical -** Требуется для логической репликации, включает всю информацию о DML (INSERT, UPDATE, DELETE).

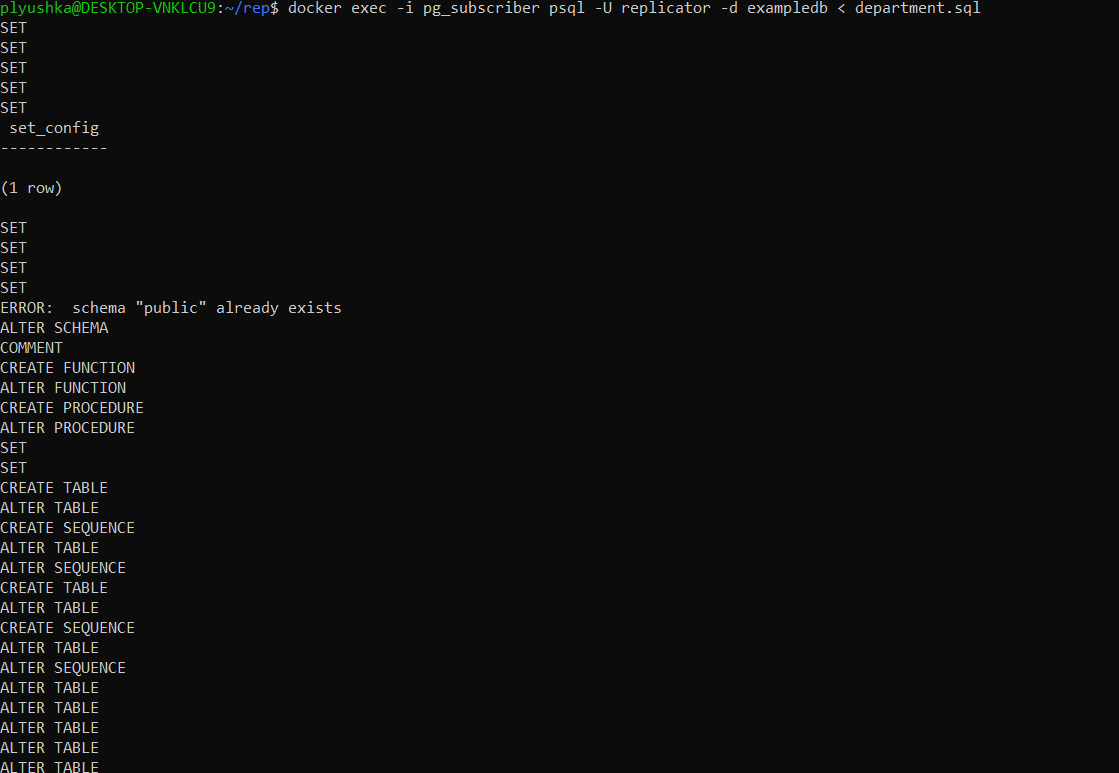
CREATE PUBLICATION pub\_distro

FOR TABLE employees, departments;



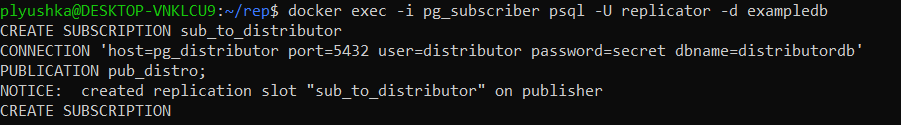
**Настройка подписчика**

docker exec -i pg\_subscriber psql -U replicator -d exampledb < department.sql

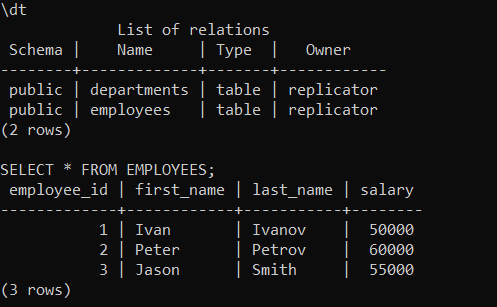


docker exec -i pg\_subscriber psql -U replicator -d exampledb

CREATE SUBSCRIPTION sub\_to\_distributor CONNECTION 'host=pg\_distributor port=5432 user=distributor password=secret dbname=distributordb' PUBLICATION pub\_distro;



Проверка:



SELECT proname, proargtypes, prorettype::regtype

FROM pg\_proc

JOIN pg\_namespace ON pg\_proc.pronamespace = pg\_namespace.oid

WHERE nspname = 'public';

