

Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики

Факультет инфокоммуникационных технологий

Направление подготовки 11.03.02

Практическая работа №1

Выполнил:

Доценников Никита Андреевич

Группа: К3221

Проверила:

Татьяна Евгеньевна Войтюк

Санкт-Петербург

2025

Цель работы:

Освоить базовые приёмы установки и настройки субд “postgresql”, научиться использовать pgadmin для администрирования, развернуть учебную базу данных “demo” и изучить ее структуру, научиться создавать базы данных и схемы. Научиться создавать, изменять связи таблиц, настройки ограничений, индексов и строить ER-диаграммы.

Задачи, решаемые при выполнении работы.

- Установить и настроить субд postgresql и pgadmin.
- Проверить сервер с помощью выполнения запроса.
- Развернуть базу demo и проверить что ее объекты доступны.
- Изучить структуру базы demo.
- Создать базу данных двумя способами: через графический интерфейс и с помощью `CREATE DATABASE`.
- Создать в базе данных hr схемы.
- Научиться создавать таблицы в базе данных с использованием графического интерфейса и запросов.
- Научиться задавать параметры таблиц.
- Узнать применение команды `ALTER TABLE`.
- Создать связи между таблицами как через интерфейс, так и с помощью sql.
- Научиться создавать индексы.
- Изучить построение и редактирование er-диаграмм.
- Научиться добавлять, изменять и проверять данные в таблицах с помощью интерфейса и запросов.
- Изучить назначение и реализацию ограничений.
- Научиться очищать таблицы с помощью команды `TRUNCATE`.
- Понимать создание и сопровождение баз данных.

Исходные данные.

- Сервер с установленными docker и контейнерами:
 - postgres:16 — сервер субд postgresql,
 - dpage/pgadmin4 — веб-интерфейс.
- Учебная база данных demo.
- Субд postgresql и pgadmin.
- Бд hr, содержащая схемы и объекты для работы.
- Схема employeesDepartments.
- SQL-скрипты, из методички.

Выполнение работы.

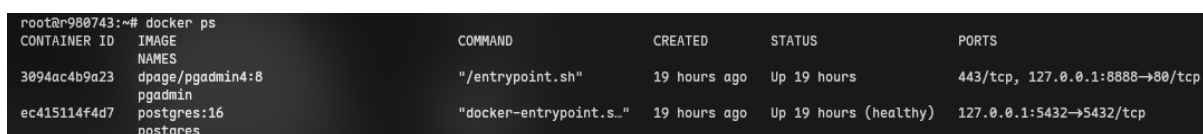
Часть 1.

Задание 1. Установка необходимого ПО.

Для выполнения работы была установлена субд postgresql и приложение pgadmin. Установка производилась в среде docker.

На сервере был установлен docker и запущены контейнеры:

postgres:16 , dpape/pgadmin4 . На рис. 1 показан вывод команды `docker ps` , который показывает запущенные контейнеры `pgadmin` и `postgres` .



CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
3094ac4b9a23	dpape/pgadmin4:8	"/entrypoint.sh"	19 hours ago	Up 19 hours	443/tcp, 127.0.0.1:8888→80/tcp
ec415114f4d7	postgres:16	"docker-entrypoint.s..."	19 hours ago	Up 19 hours (healthy)	127.0.0.1:5432→5432/tcp

Рис 1: Результат команды `docker ps` .

Контейнеры были объединены в общую сеть docker, чтобы pgadmin мог напрямую подключаться к postgres. На рис. 2 показаны настройки сети `dbnet` .

```

root@r980743:~# docker network inspect dbnet
[
  {
    "Name": "dbnet",
    "Id": "8d62db233b593ccfd61d93a73946d4bdf9188b9d2288e8f39285c62f3592ef82",
    "Created": "2025-09-23T17:17:25.408657713Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.23.0.0/16",
          "Gateway": "172.23.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "3094ac4b9a23e2557f2496c274d727040fc5aa4eeb8b1686e8ede9818e617fd4": {
        "Name": "pgadmin",
        "EndpointID": "1707ced2ae7b442ebdf9cccb6162a86d105305693b2deb19fc0994821840e84f",
        "MacAddress": "02:42:ac:17:00:03",
        "IPv4Address": "172.23.0.3/16",
        "IPv6Address": ""
      },
      "ec415114f4d7be443192c5ec81cc67a89f38dcb3c6dcff686f90f4ccd788b16b": {
        "Name": "postgres",
        "EndpointID": "ecaa8951afe60e34569b2dedd7d84a7e3c365283f3cf0b54fed708bf6bf5a6a8",
        "MacAddress": "02:42:ac:17:00:02",
        "IPv4Address": "172.23.0.2/16",
        "IPv6Address": ""
      }
    },
    "Options": {},
    "Labels": {}
  }
]

```

Рис 2: Настройки сети dbnet .

pgadmin стал доступен по адресу db.fymio.us

На рис. 3 показан фрагмент Caddyfile, ответственный за db.fymio.us.

```

db.fymio.us {
  reverse_proxy 127.0.0.1:8888
  encode gzip
  header {
    Strict-Transport-Security max-age=31536000;
    X-Content-Type-Options nosniff
    X-Frame-Options SAMEORIGIN
  }
  tls {
    protocols tls1.2 tls1.3
  }
}
root@r980743:~#

```

Рис 3: Часть вывода команды `cat /etc/caddy/Caddyfile`.

В pgadmin был зарегистрирован новый сервер. В настройках подключения были указаны следующие параметры. На рис. 4, 5 показаны соответствующие настройки.

The screenshot shows the 'Web PostgreSQL' configuration window in pgAdmin. The 'General' tab is selected. The 'Name' field contains 'Web PostgreSQL'. The 'Server group' dropdown is set to 'Servers'. The 'Background' and 'Foreground' checkboxes are both unchecked. The 'Shared?' toggle switch is turned off. The 'Shared Username' and 'Comments' fields are empty. At the bottom, there are buttons for 'Close', 'Reset', and 'Save'.

Web PostgreSQL	
General Connection Parameters SSH Tunnel Advanced >	
Name	Web PostgreSQL
Server group	Servers
Background	☐
Foreground	☐
Shared?	☐
Shared Username	
Comments	
Close Reset Save	

Рис 4: Настройки General сервера Web PostgreSQL.

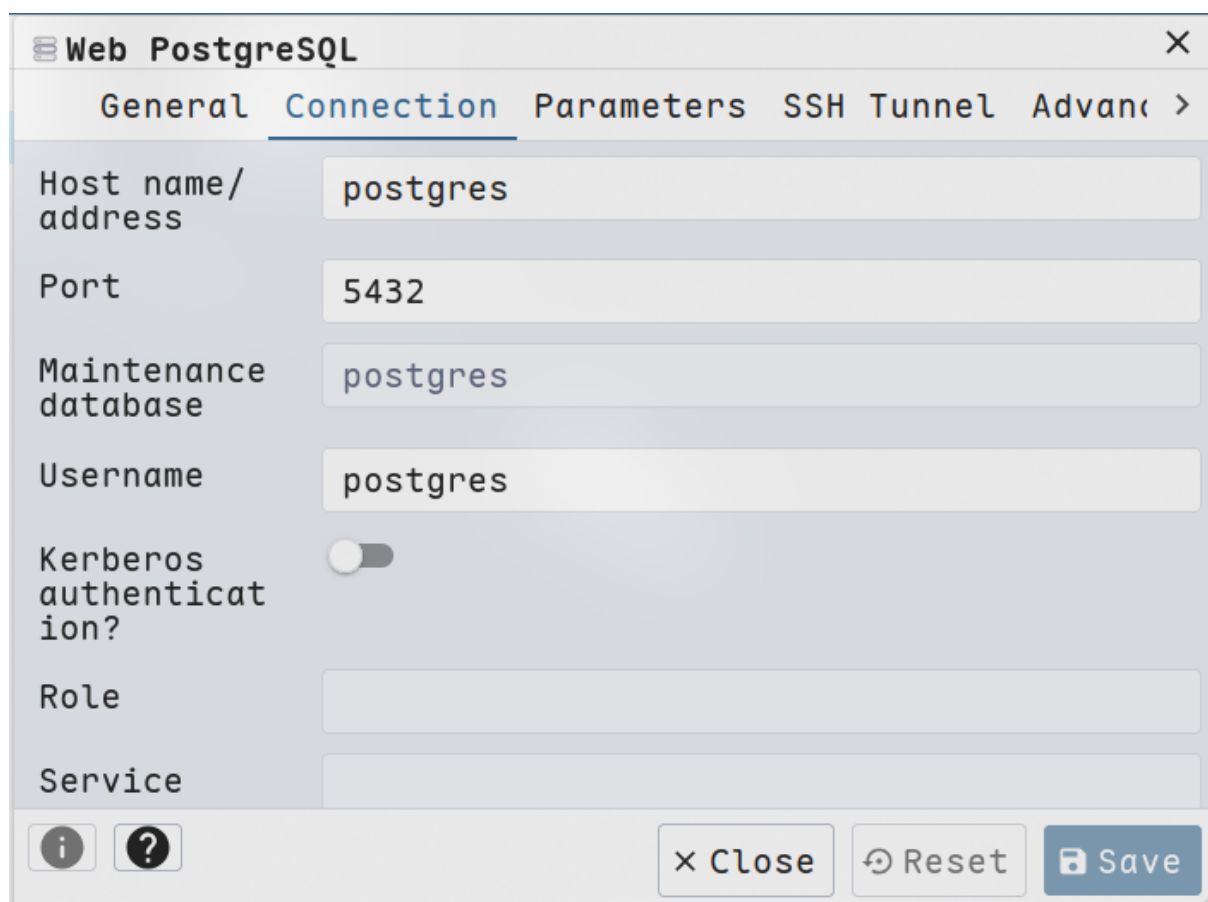


Рис 5: Настройки Connection сервера Web PostgreSQL.

В "Object Explorer" слева я раскрыл узел "Databases" (Рис. 6):

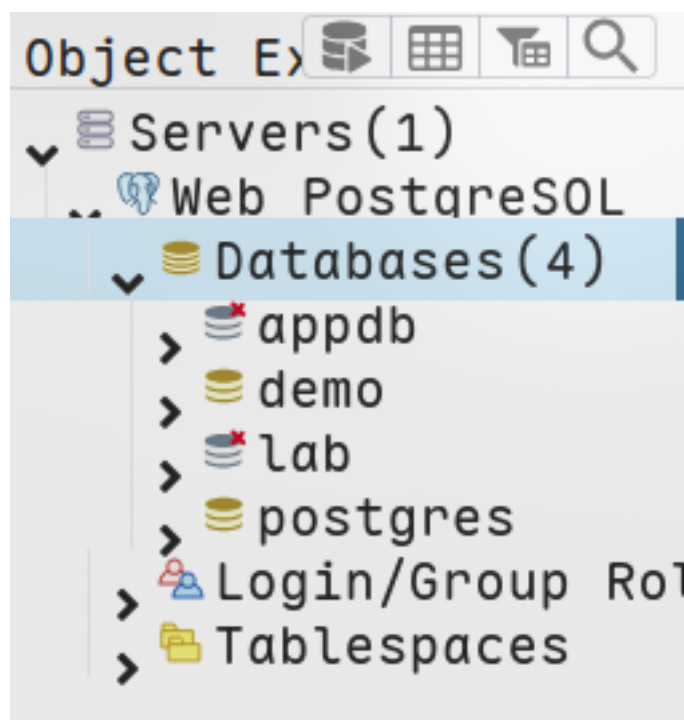


Рис 6: Object Explorer -> Databases.

Затем я нажал ПКМ по базе “postgres” и выбрал инструмент “Query Tool” (Рис. 7):

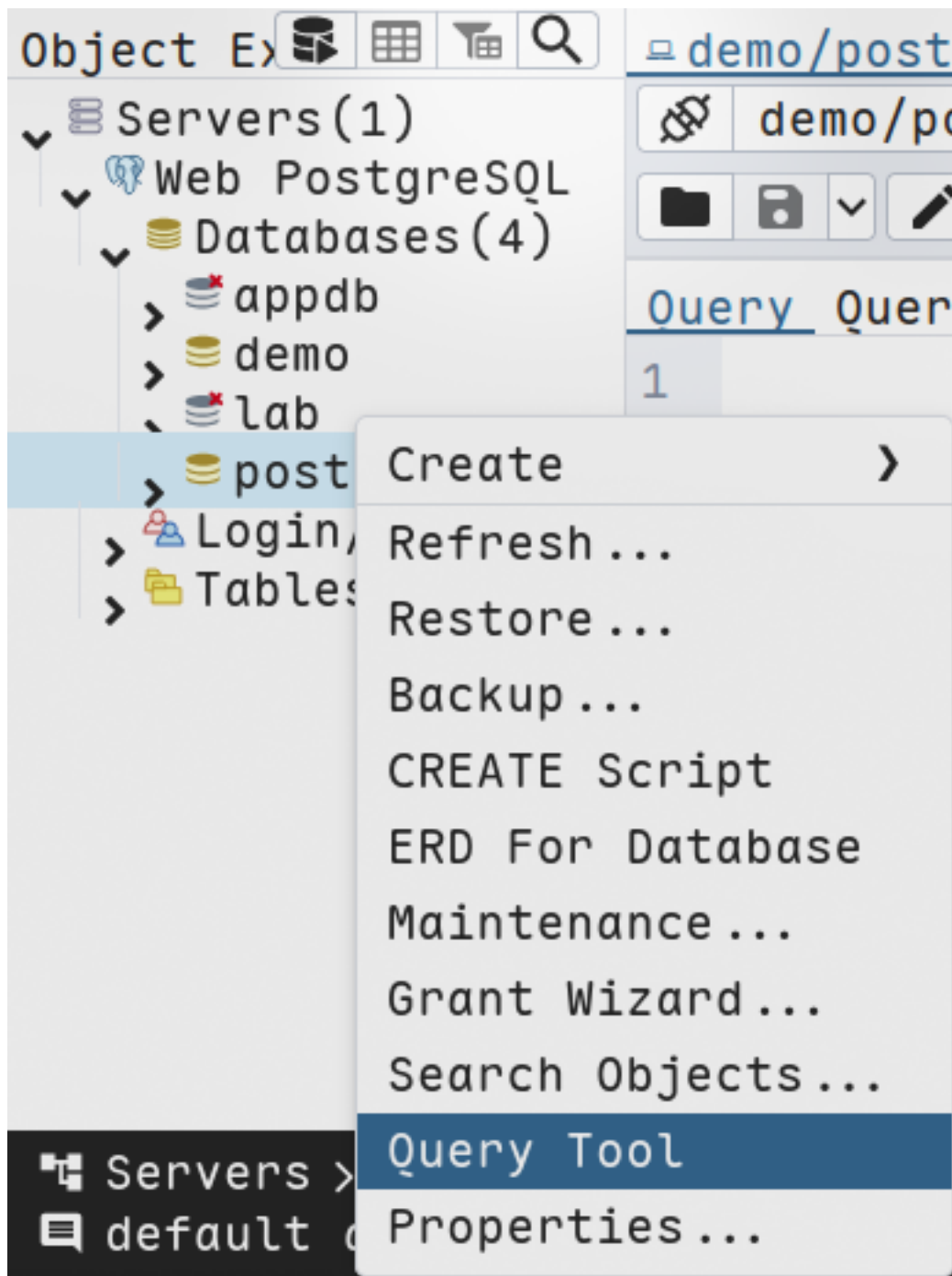


Рис 7: Открытие Query Tool.

В открывшемся окне написал запрос проверки версии сервера `SELECT version();`. После этого нажал кнопку “Execute script” (Рис. 8).

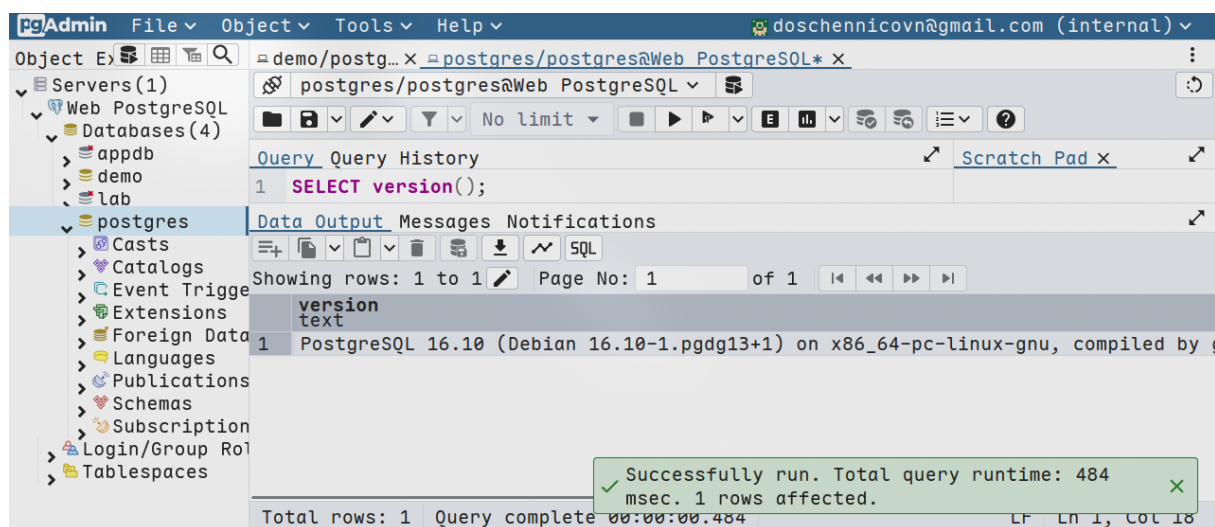


Рис 8: Выполнение скрипта проверки версии.

В поле “Data Output” я увидел версию PostgreSQL 16.10.

Задание 2. Подключение учебной базы данных.

После выполнения следующих команд на сервере учебная база данных появилась в “Object Explorer”.

```
wget https://edu.postgrespro.ru/demo-medium.zip
unzip demo-medium.zip
docker exec -i postgres psql -U postgres -d postgres < demo-
medium-20170815.sql
```

После появления базы данных “demo”, в “Query Tool” я выполнил следующий скрипт (Рис. 9):

```
SELECT * FROM bookings.aircrafts_data;
```

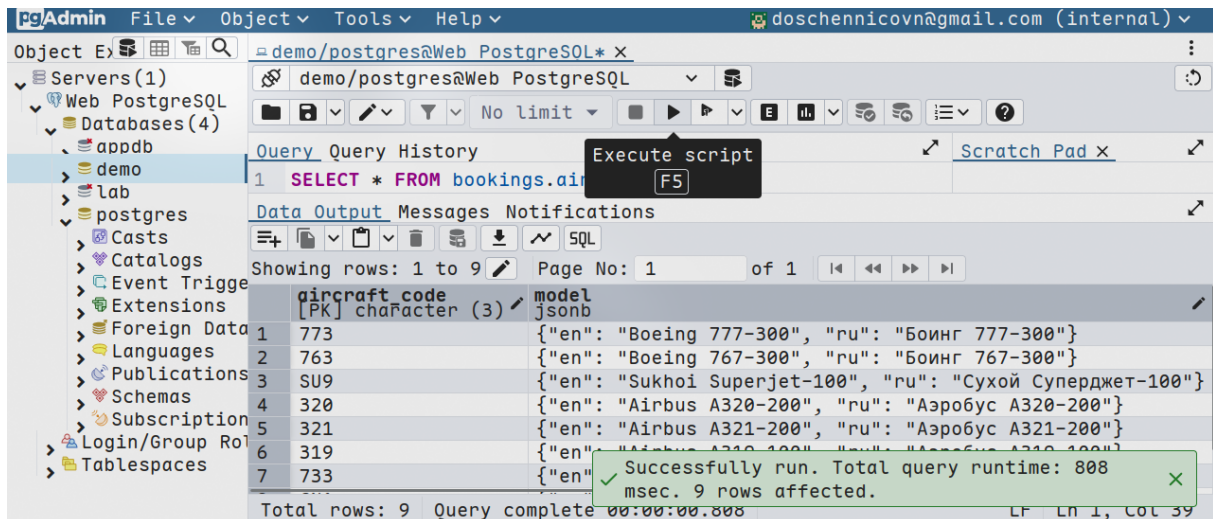


Рис 9: Выполнение скрипта `SELECT * FROM bookings.aircrafts_data`.

Задание 3. Ознакомление с учебной БД.

Я ознакомился с описанием БД на сайте postgrespro.ru/education/demodb

Задание 4. Создание учебной базы данных HR средствами pgAdmin.

Я нажал ПКМ на узле “Databases” и выбрал меню “Create” затем “Database...” (Рис. 10).

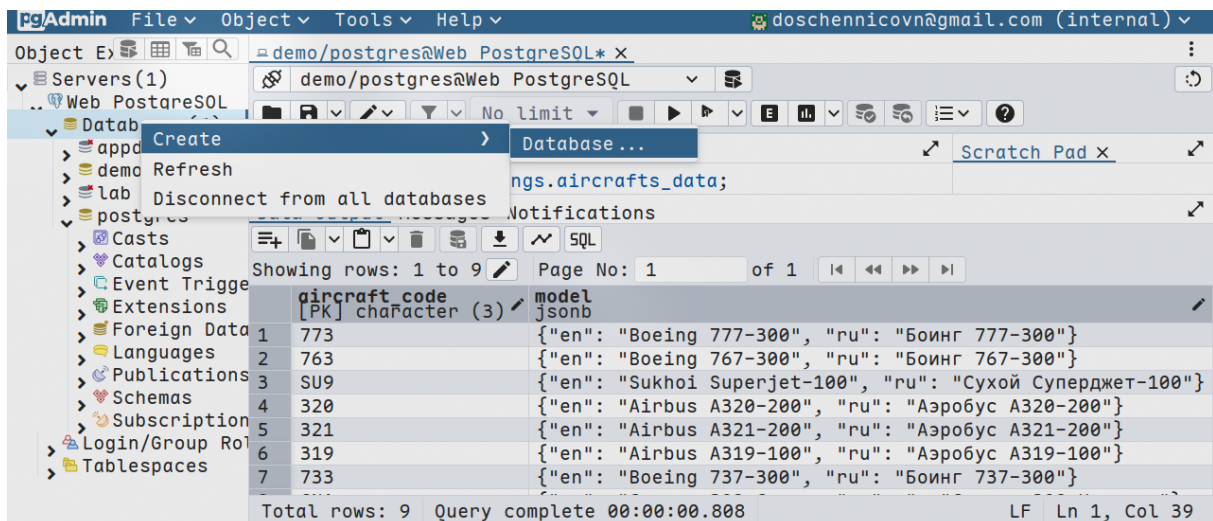


Рис 10: Создание новой базы данных.

Затем в открывшемся окне я выставил следующие настройки (Рис. 10, 11).

Create - Database

General Definition Security Parameters Advanced SQL

Database

OID

Owner

Comment

Close Reset Save

Рис 11: Настройки новой базы. Вкладка General.

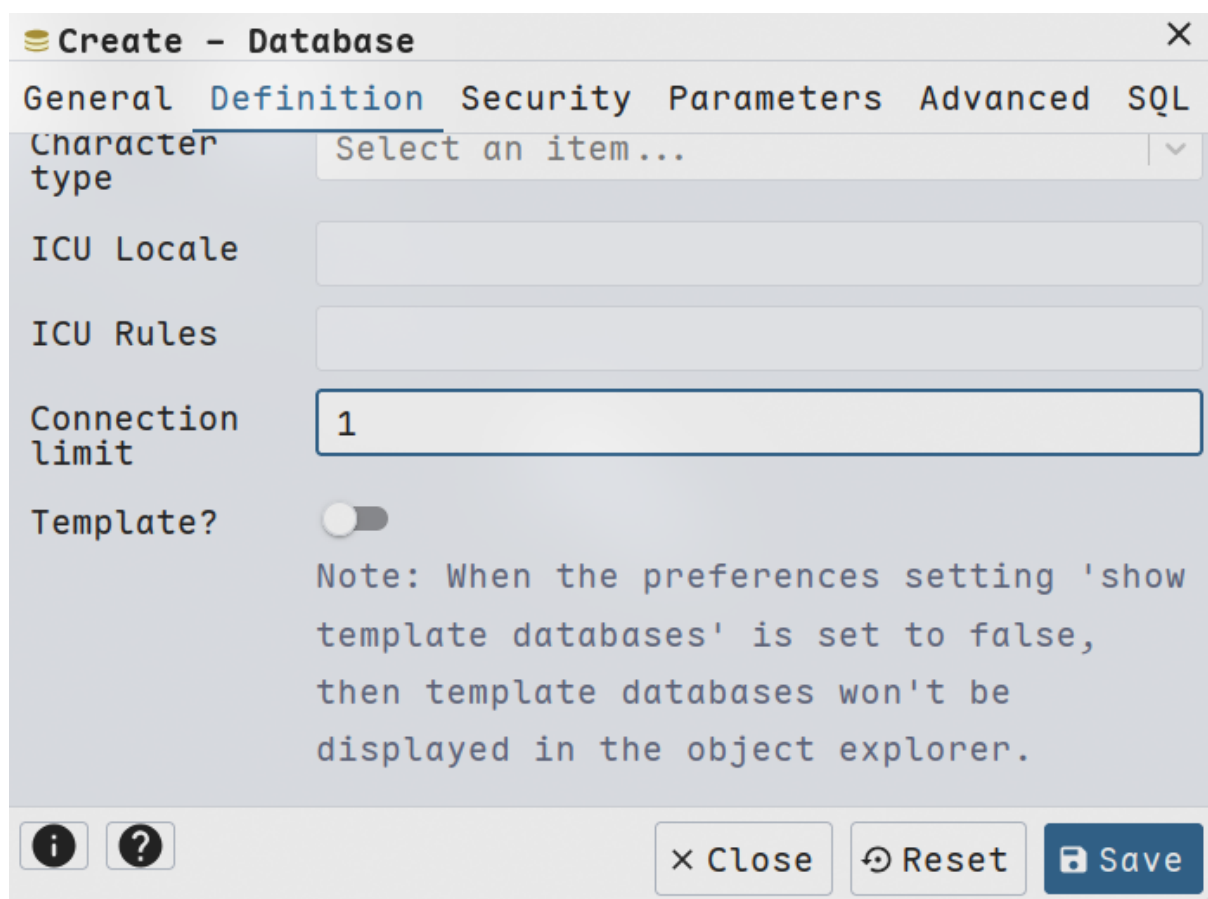


Рис 12: Настройки новой базы. Вкладка Definition.

Затем я сохранил настройки. В “Object Explorer” появилась созданная база (Рис. 12).

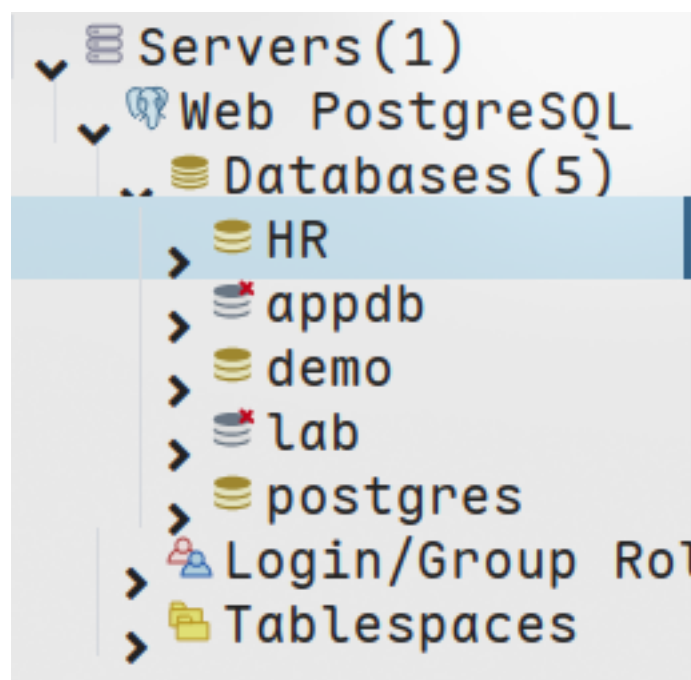


Рис 13: База HR в Object Explorer.

После этого я удалил базу через ПКМ (Рис. 14).

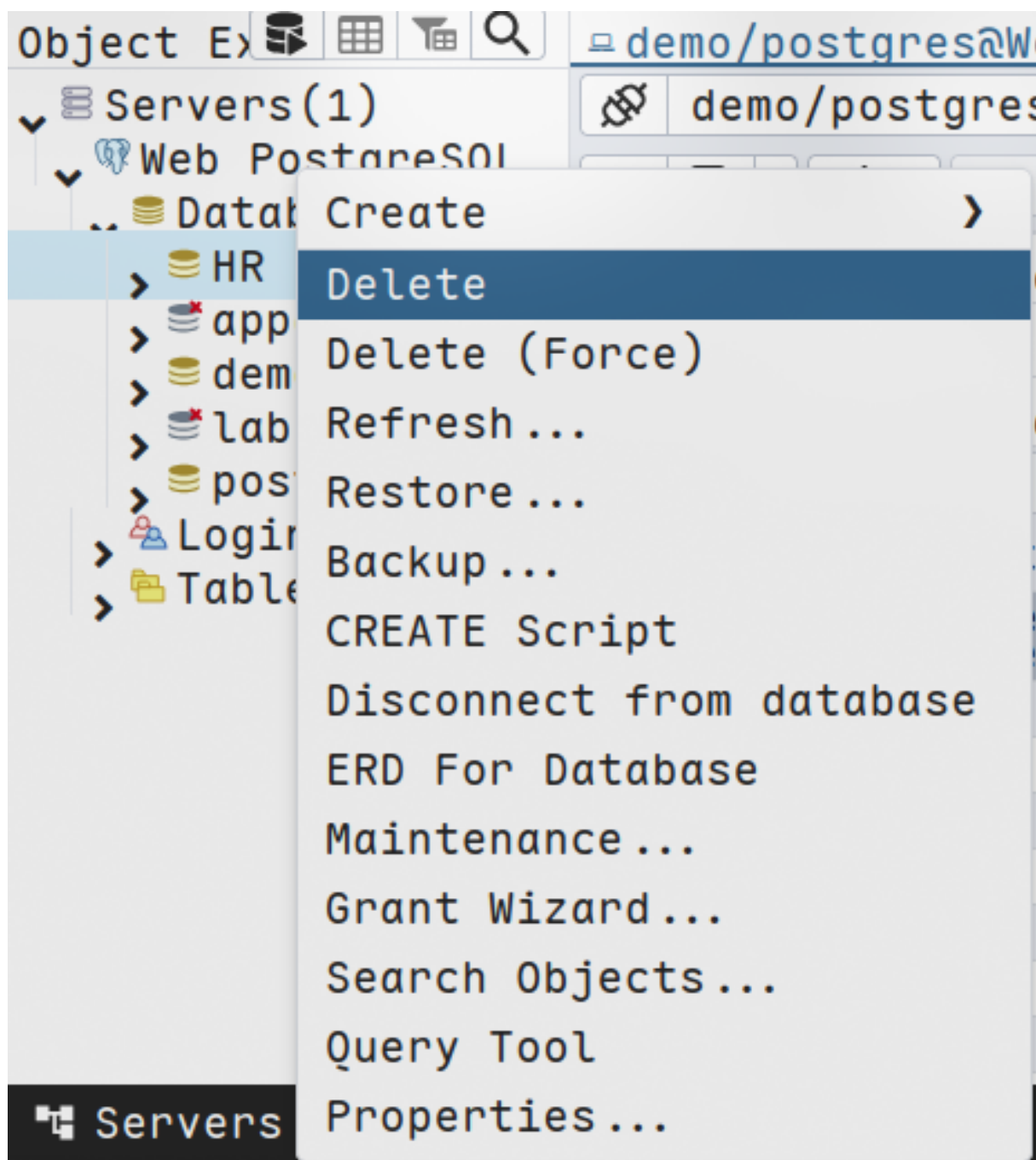


Рис 14: Удаление базы HR.

Задание 5. Создание учебной базы данных HR средствами PL/pgSQL

Я открыл "Query Tool" базы данных postgres. И прописал следующий скрипт (Рис. 15).

```
CREATE DATABASE hr
WITH ENCODING 'UTF8'
LC_COLLATE 'C'
LC_CTYPE 'C'
TEMPLATE template0
CONNECTION LIMIT 1;
```

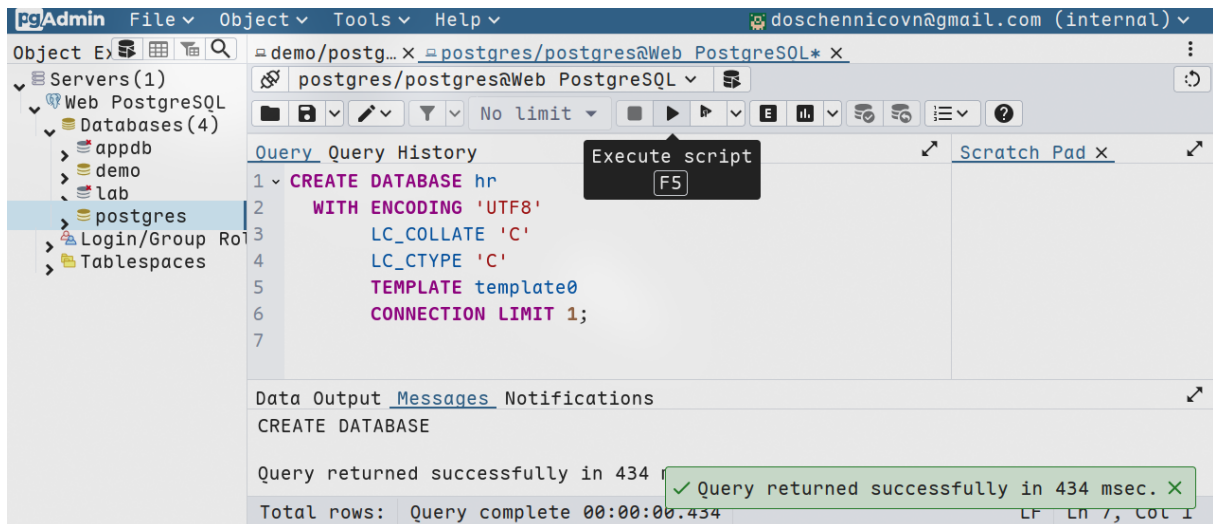


Рис 15: Создание базы `hr` при помощи скрипта.

После этого в списке баз появилась созданная (Рис. 16).

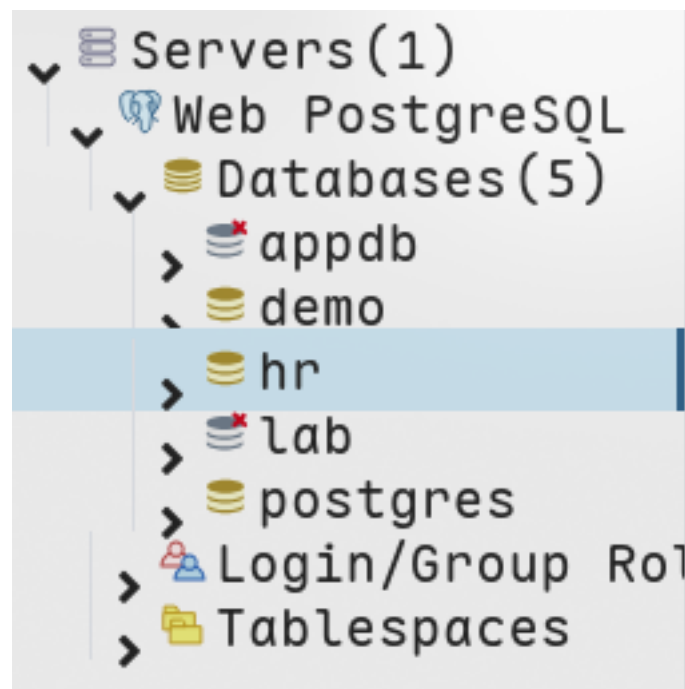


Рис 16: Новая база в Object Explorer.

Задание 6. Создание схем.

Я развернул вкладку для создания схем. Рис. 17 показывает меню создание схем.

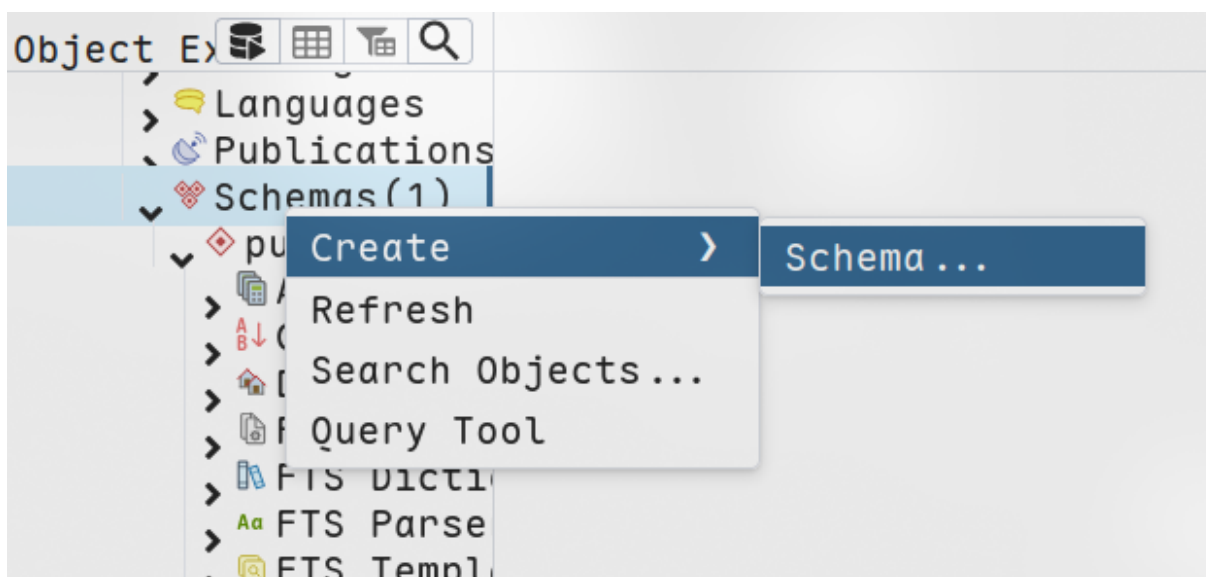


Рис 17: Создание схемы.

И создал схему "EmployeesDepartments". В "Object Explorer" видно что схема создана (Рис. 18).

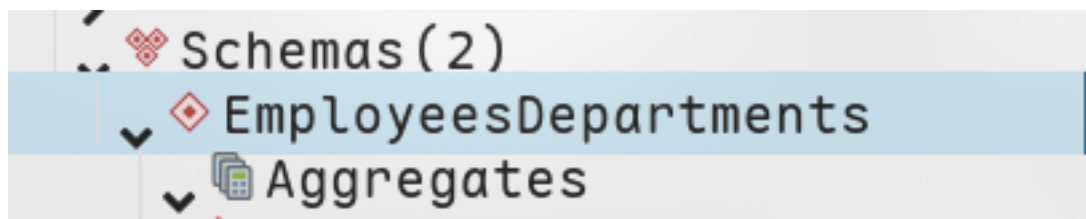


Рис 18: Новая схема появилась.

Чтобы создать вторую схему, я открыл "Query Tool" к базе hr. И прописал следующие запросы (Рис. 19):

```
CREATE SCHEMA IF NOT EXISTS "countries"  
AUTHORIZATION postgres;
```

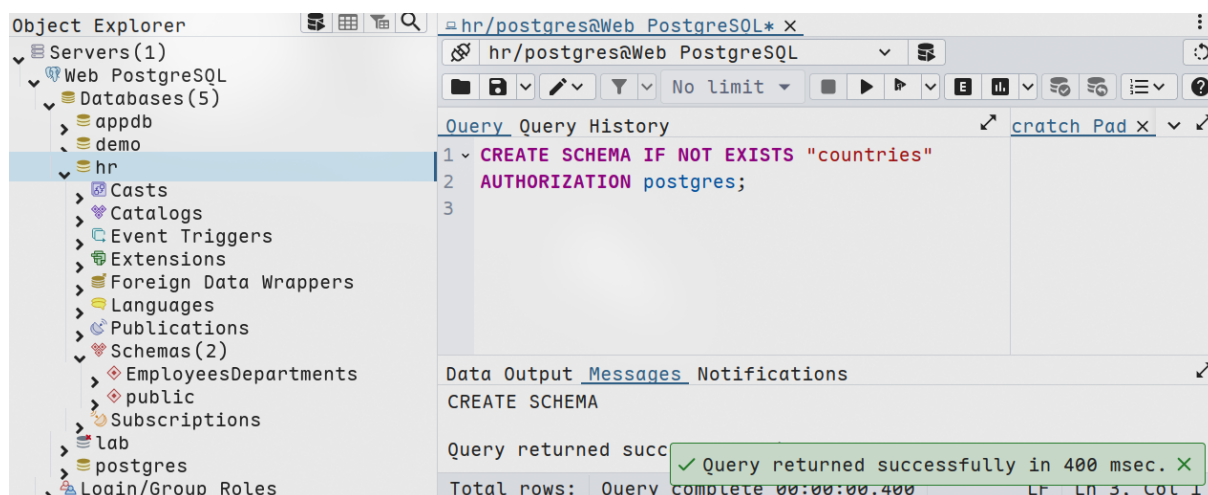


Рис 19: Исполнение скрипта по созданию схемы.

В “Object Explorer” видна добавленная схема (Рис. 20):

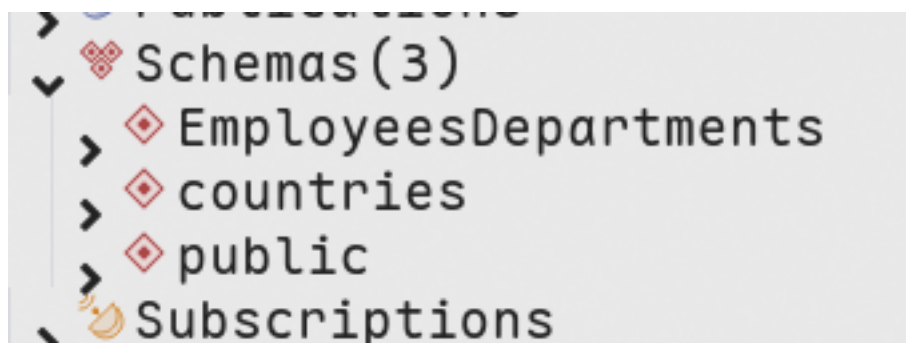


Рис 20: Появление новой схемы.

Часть 2.

Задание 1. Создание таблицы в графической среде pgadmin.

Я раскрыл базу hr до узла tables и выбрал опцию create (Рис. 21):

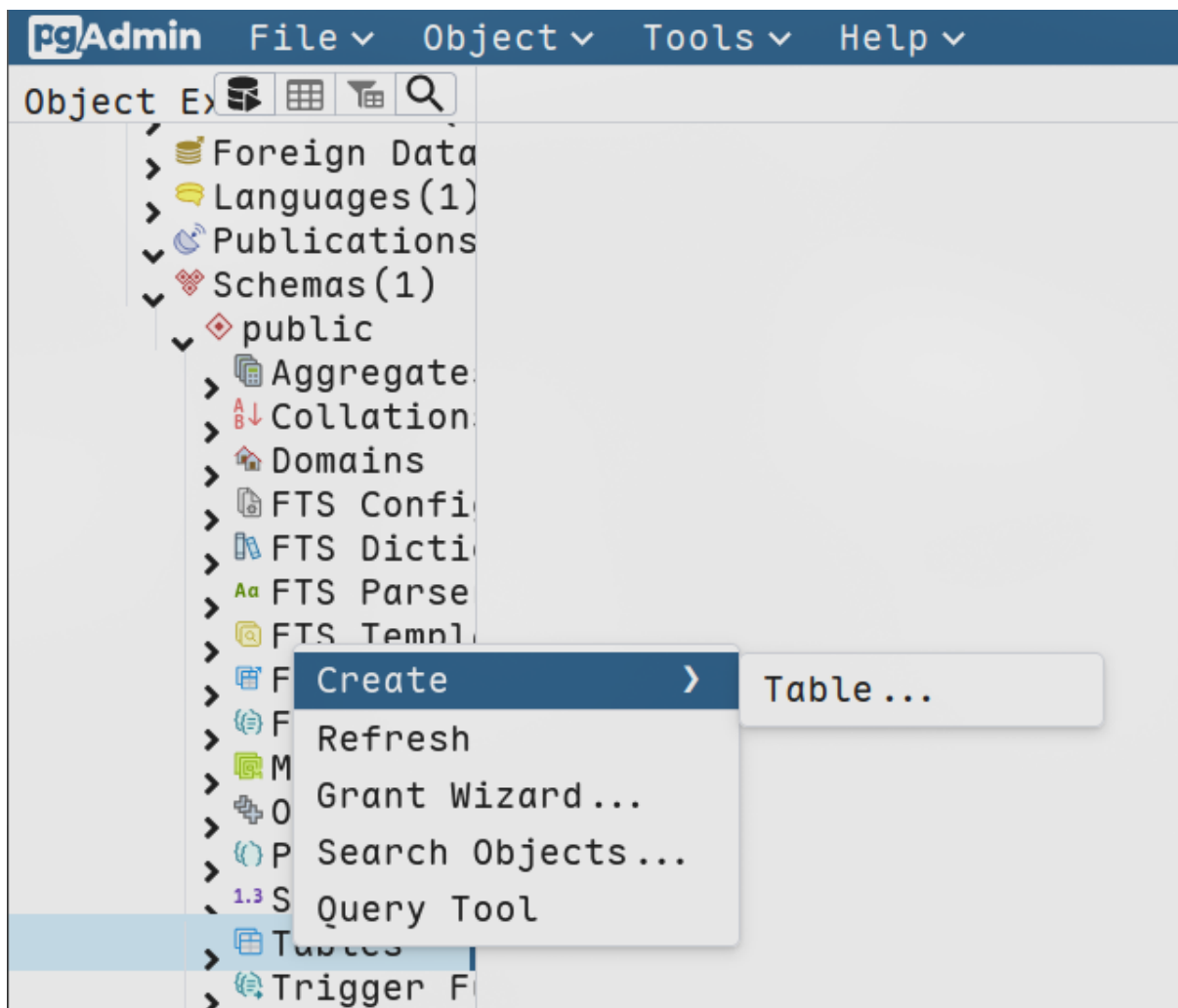


Рис 21: Меню создания новой таблицы.

И создал таблицу EMPLOYEES (Рис. 22):

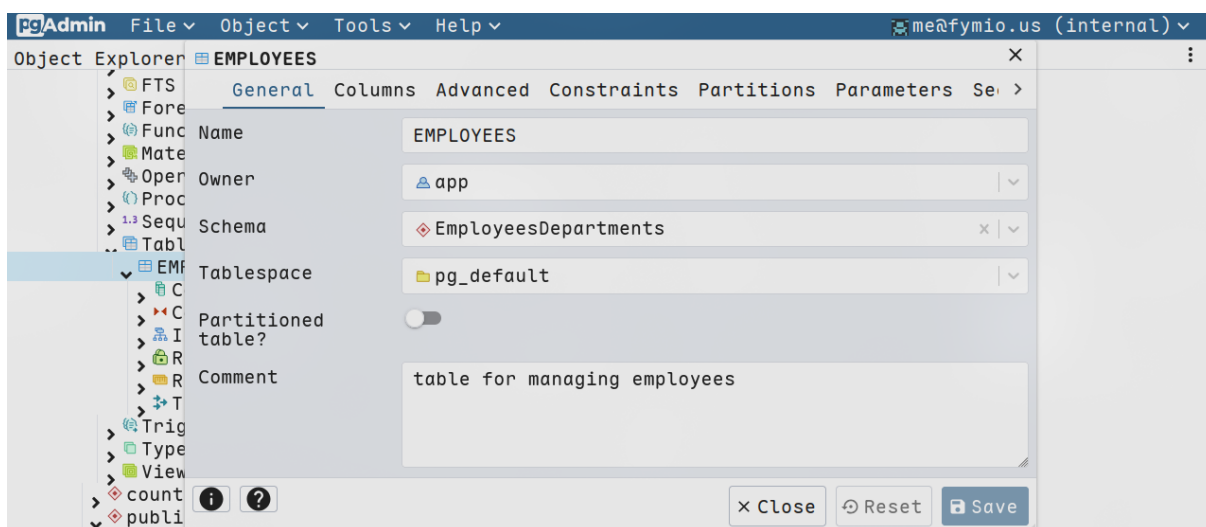


Рис 22: Настройки таблицы EMPLOYEES.

Затем я заполнил вкладку `columns` как показано на рисунке в условии (Рис. 23):

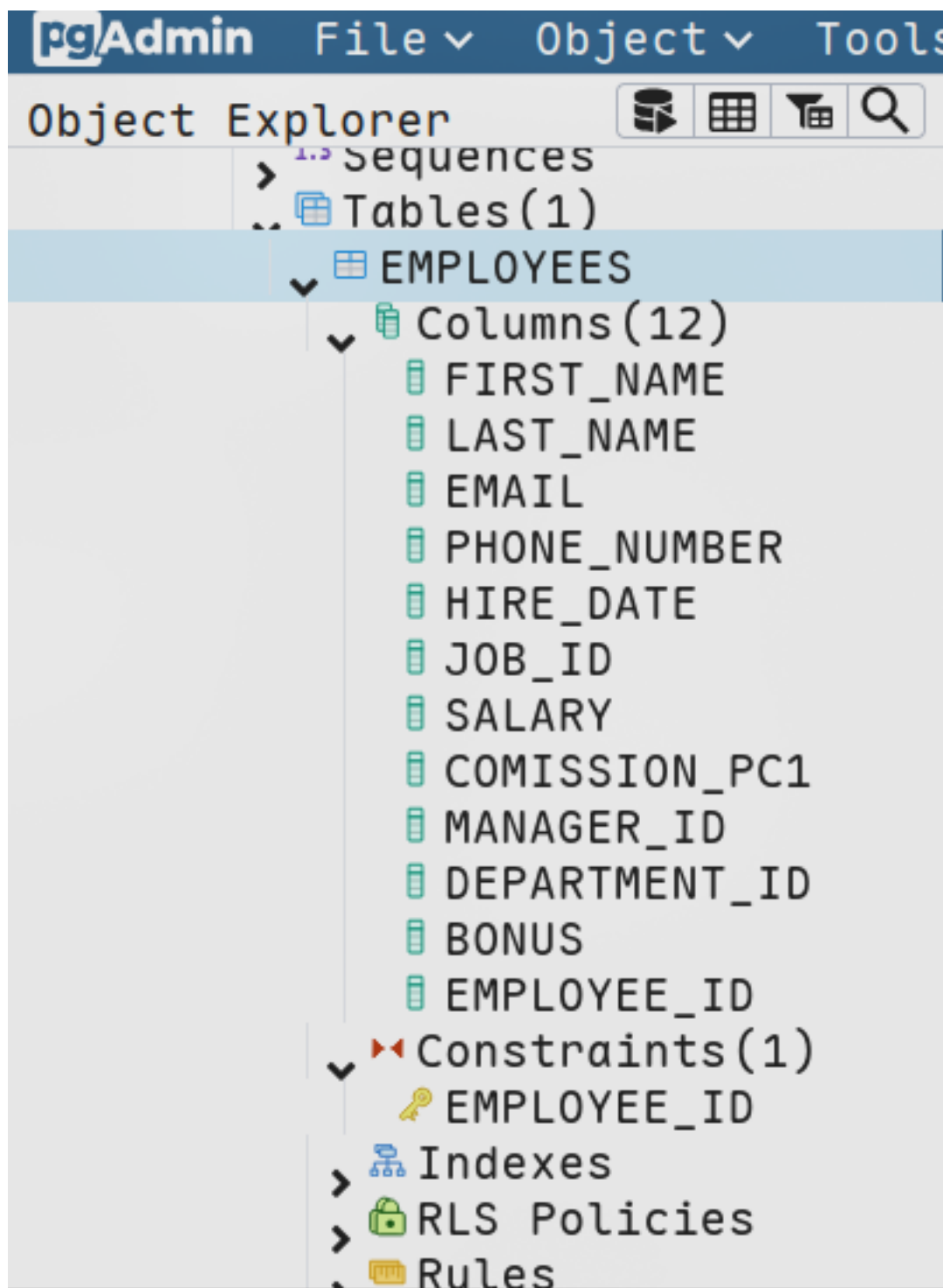


Рис 23: Столбцы таблицы EMPLOYEES.

Для поля `employee_id`, я выставил следующие параметры (Рис. 24):

EMPLOYEE_ID X

General Definition Constraints >

Default

Not NULL? ☒

Type NONE ✓ IDENTITY GENERATED

Identity ALWAYS | v

Increment 1

Start 1

Minimum

i ? X Close ↺ Reset Save

Рис 24: Настройки столбца EMPLOYEE_ID -> Constraints.

Для столбца hire_date я поставил значение по умолчанию current_date (Рис. 25).

HIRE_DATE [X]

General Definition Constraints >

Default: current_date

Not NULL? ☒

Type: ☒ NONE ☐ IDENTITY ☐ GENERATED

[i] [?] [Close] [Reset] [Save]

Рис 25: Обновленное значение по умолчанию для столбца HIRE_DATE .

После этого, я сохранил таблицу.

Задание 2. Создание таблицы в Query Editor.

Я открыл query tool для hr и вписал туда следующий скрипт (Рис. 26):

```
CREATE TABLE "EmployeesDepartments"."DEPARTMENTS" (
  DEPARTMENT_ID integer GENERATED ALWAYS AS IDENTITY
(INCREMENT 1 START 1) PRIMARY KEY NOT NULL,
  DEPARTMENT_NAME character varying(30)[] NOT NULL,
  MANAGER_ID bigint NULL,
```

```
LOCATION_ID integer NULL
);
```

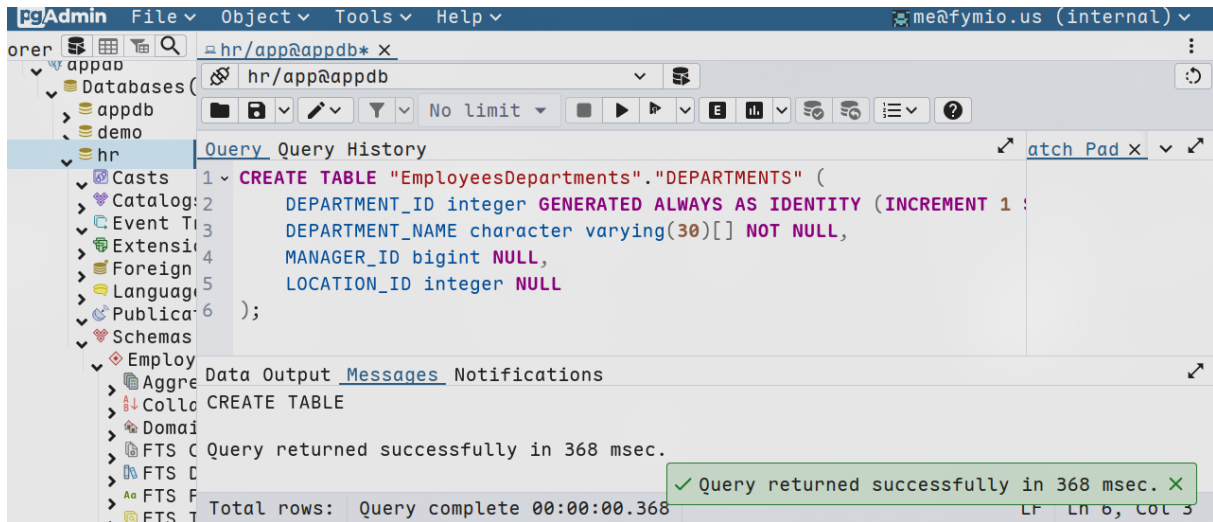


Рис 26: Исполнение скрипта по созданию таблицы.

Таблица появилась в object explorer (Рис. 27):

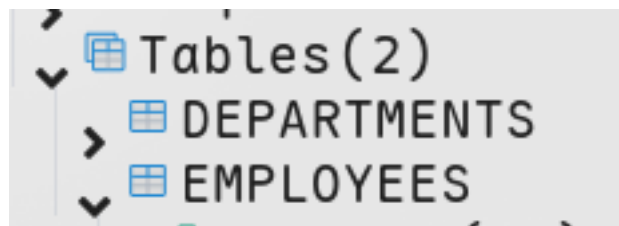


Рис 27: Новая таблица в Object Explorer.

Затем я создал еще одну таблицу скриптом (Рис. 28):

```
CREATE TABLE "EmployeesDepartments"."LOCATIONS"
  (LOCATION_ID smallint GENERATED ALWAYS AS IDENTITY NOT
  NULL,
  STREET_ADDRESS character varying(40),
  POSTAL_CODE character varying(12),
  CITY character varying(30),
  COUNTRY_ID CHAR(2)
  );
```

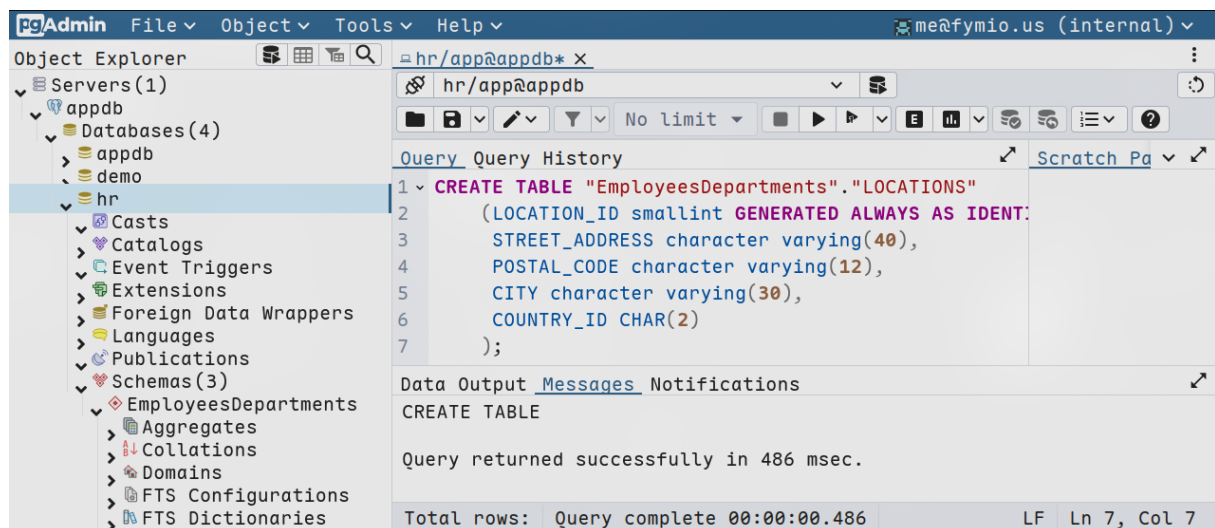


Рис 28: Исполнение скрипта по созданию таблицы.

Как можно видеть, таблица была создана (Рис. 29):

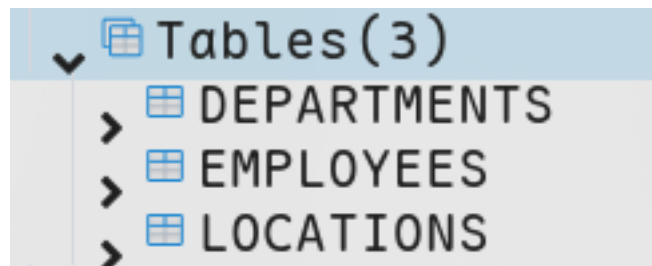


Рис 29: Новая таблица в Object Explorer.

Задание 3. Изменение таблицы.

В query tool я ввел следующее (Рис. 30):

```

ALTER TABLE "EmployeesDepartments"."LOCATIONS"
ADD STATE_PROVINCE character varying(25) NULL;

```

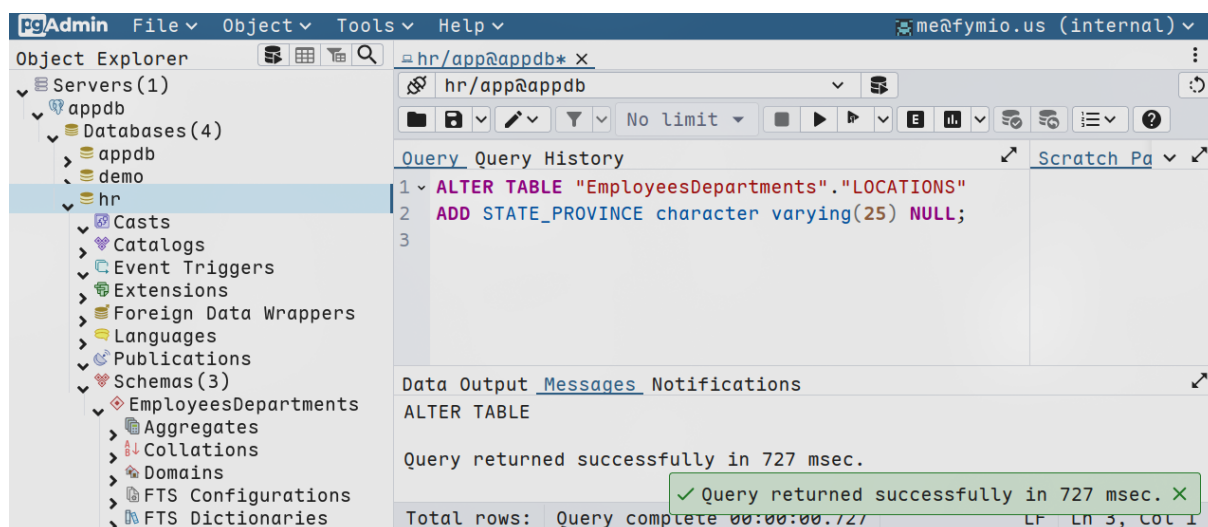


Рис 30: Исполнение скрипта в query tool.

Изменил столбец CITY следующим кодом (Рис. 31):

```
ALTER TABLE "EmployeesDepartments"."LOCATIONS"
ADD CONSTRAINT "PK_LokationId" PRIMARY KEY (LOCATION_ID);
```

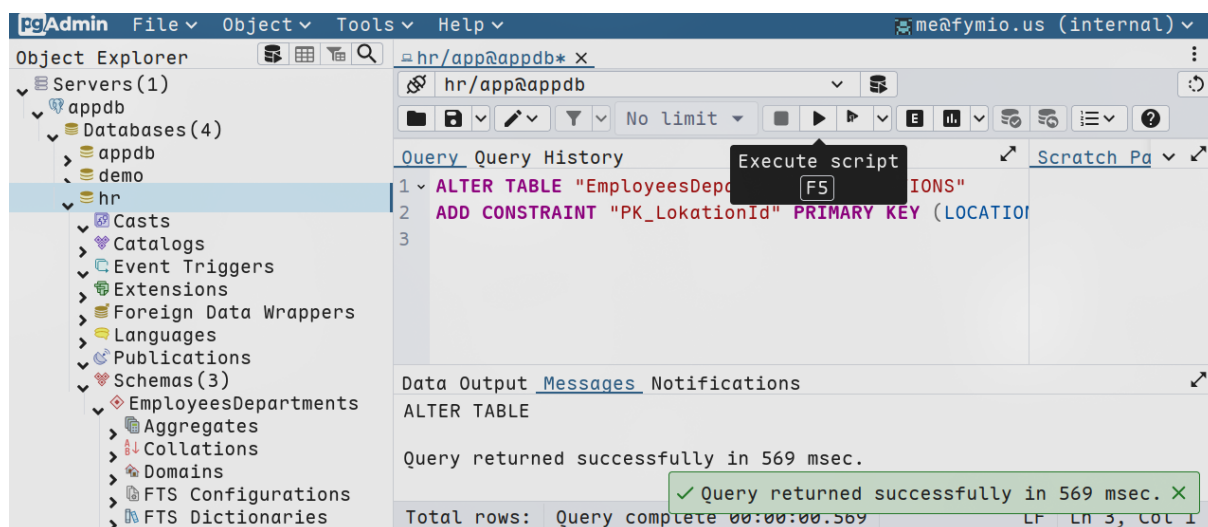


Рис 31: Исполнение скрипта query tool.

Таблица locations (Рис. 32):

location_id	street_address	postal_code	city
[PK] smallint	character varying (40)	character varying (12)	char

Рис 32: Таблица locations.

Затем я прописал текст скрипта script1_create.sql (Рис. 33, 34):

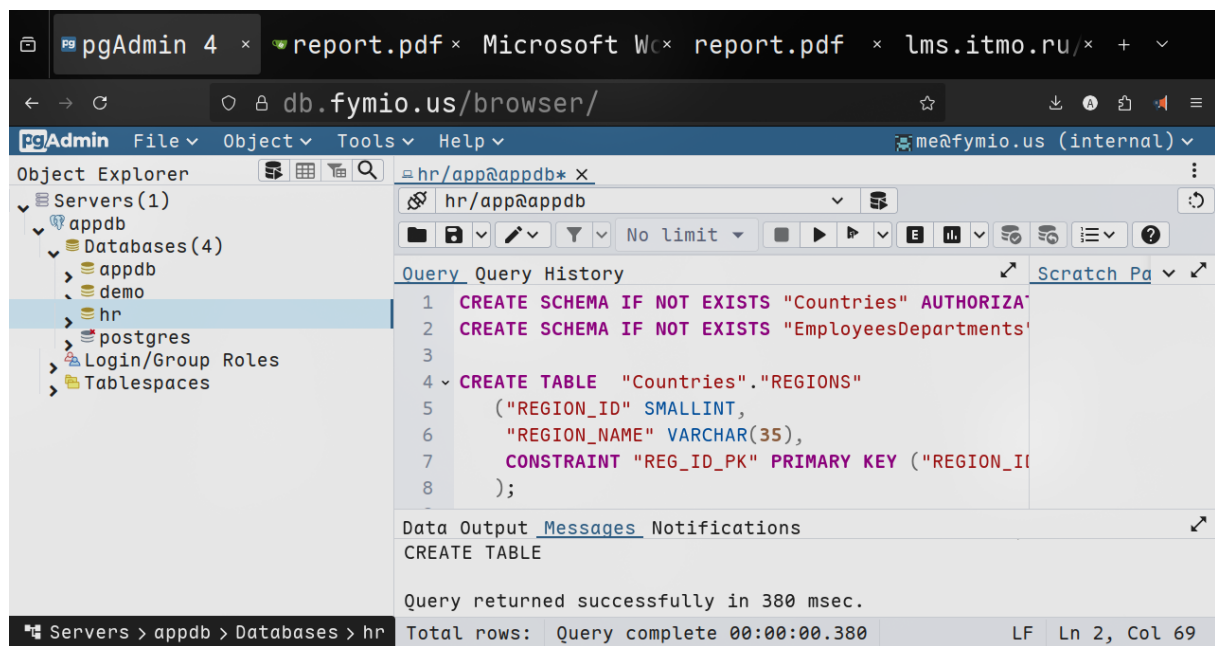


Рис 33: Скрипт `script1_create.sql` в query tool.

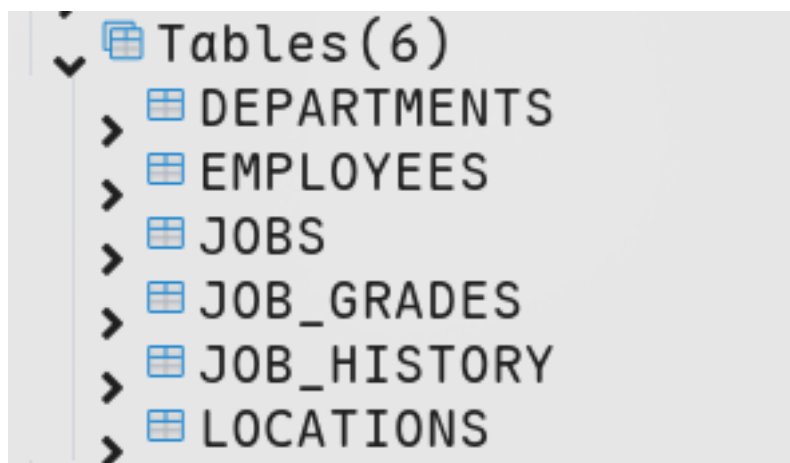


Рис 34: Новые таблицы в Object Explorer.

Задание 4. Создание отношения графическим интерфейсом pgadmin.

Я нажал ПКМ по таблице `EmployeesDepartments.LOCATIONS` и выбрал меню создания внешнего ключа (Рис. 35):

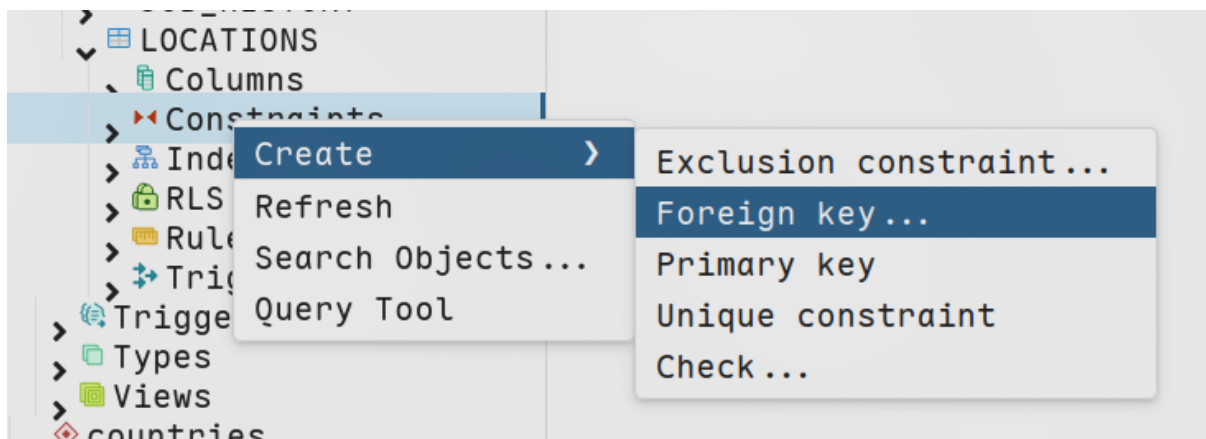


Рис 35: Меню создания внешнего ключа.

Затем я ввел задание название отношения (Рис. 36):



Рис 36: Настройки создания Foreign Key.

После, я добавил столбец кнопкой add (Рис. 37):

Create - Foreign key
×

General
Definition
Columns
Acti >

Columns

Local column

Select an item... | ▾

References

Select an item... | ▾

Referencing

Select an item... | ▾

Add

	Local	Referenced	Referenced...
	LOCATION_ID	COUNTRY_ID	"Countries".

i

?

×

Close

↺

Reset

💾

Save

Рис 37: Настройки создания Foreign Key.

Вкладка sql (Рис. 38):

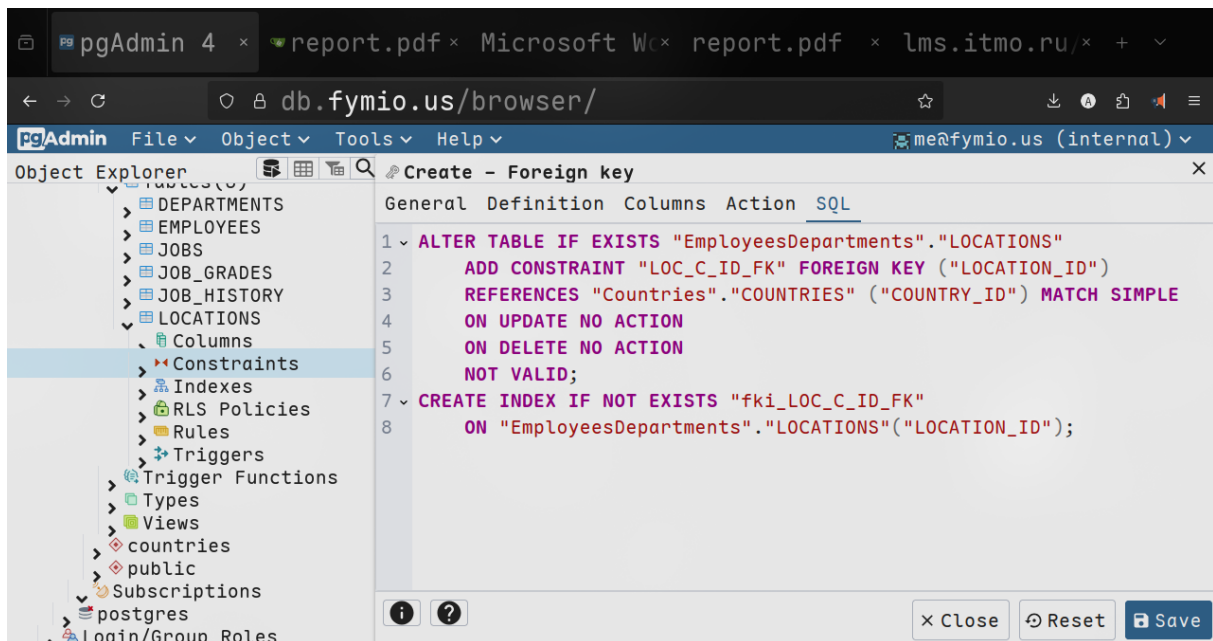


Рис 38: SQL код создания Foreign Key.

Ограничения locations (Рис. 39):

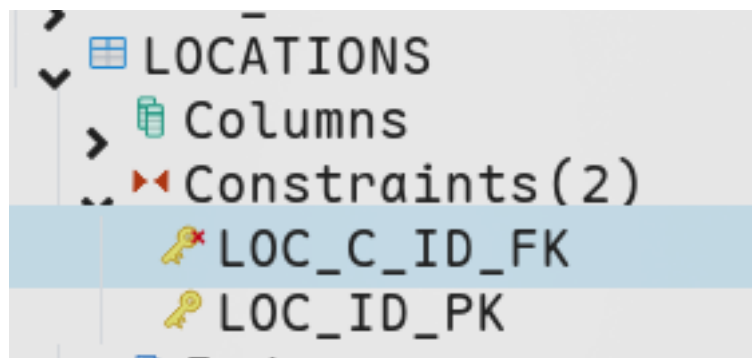


Рис 39: Ограничения LOCATIONS.

Затем я создал отношение с помощью ddl оператора alter table (Рис. 40):

```

ALTER TABLE "Countries"."COUNTRIES"
ADD CONSTRAINT "COUNTR_REG_ID_FK1" FOREIGN KEY ("REGION_ID")
REFERENCES "Countries"."REGIONS" ("REGION_ID");

```

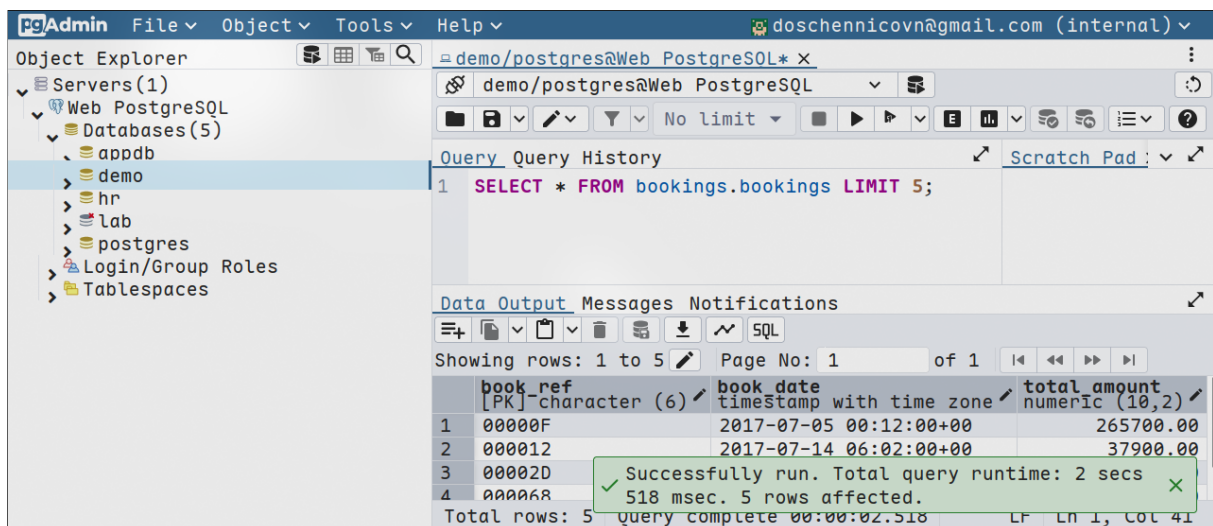


Рис 40: Выполнение скрипта в query tool.

Задание 5. Создания индекса с помощью графического интерфейса pgadmin.

В таблице EmployeesDepartments.LOCATIONS, я выбрал раздел меню для создания индекса (Рис. 41):

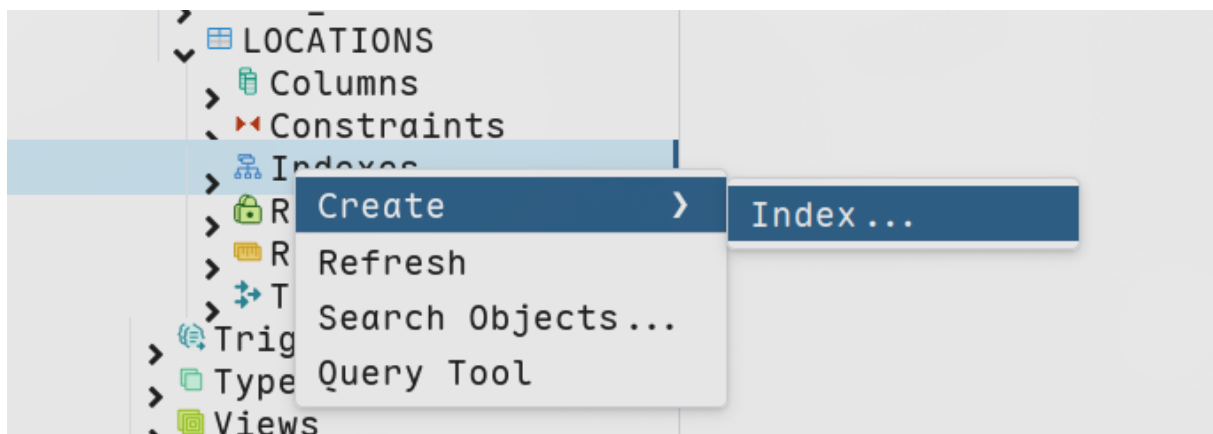


Рис 41: Меню для создания индекса.

Я в диалоге создания индекса я передал следующие параметры (Рис. 42, 43, 44):

Create - Index

General Definition Columns SQL

Name: LOC_CITY_IX

Tablespace: Select an item...

Comment:

❗ You must specify at least one column/expression.

Close Reset Save

Рис 42: Параметры создания индекса. Вкладка General.

Create - Index

General Definition Columns SQL

Access Method: btree

With

Fill factor:

Gin pending list limit:

This value is specified in kilobytes.

Pages per range:

Number of table blocks that make up one block range for each entry of a BRIN index.

❗ You must specify at least one column/expression.

Close Reset Save

Рис 43: Параметры создания индекса. Вкладка Definition.

Create - Index

General Definition Columns SQL

Columns/Expressions

Is expression ☒

Column

Expression

Add

	Col/Exp	Operator ...	Sort order	NULLs	Collation
☒	C CITY	Select v	ASC v	LAST v	Select v

Include columns

Close Reset Save

Рис 44: Параметры создания индекса. Вкладка Columns.

sql:

Create - Index

General Definition Columns SQL

```

1 CREATE INDEX "LOC_CITY_IX"
2   ON "EmployeesDepartments"."LOCATIONS" USING btree
3   ("CITY")
4   WITH (deduplicate_items=True)
5 ;

```

Close Reset Save

Рис 45: SQL скрипт создания индекса.

Задание 6. Построение диаграмм базы данных.

Для создания диаграммы я нажал ПКМ на узел базы данных и выбрал параметр “ERD For Database” (Рис. 46):

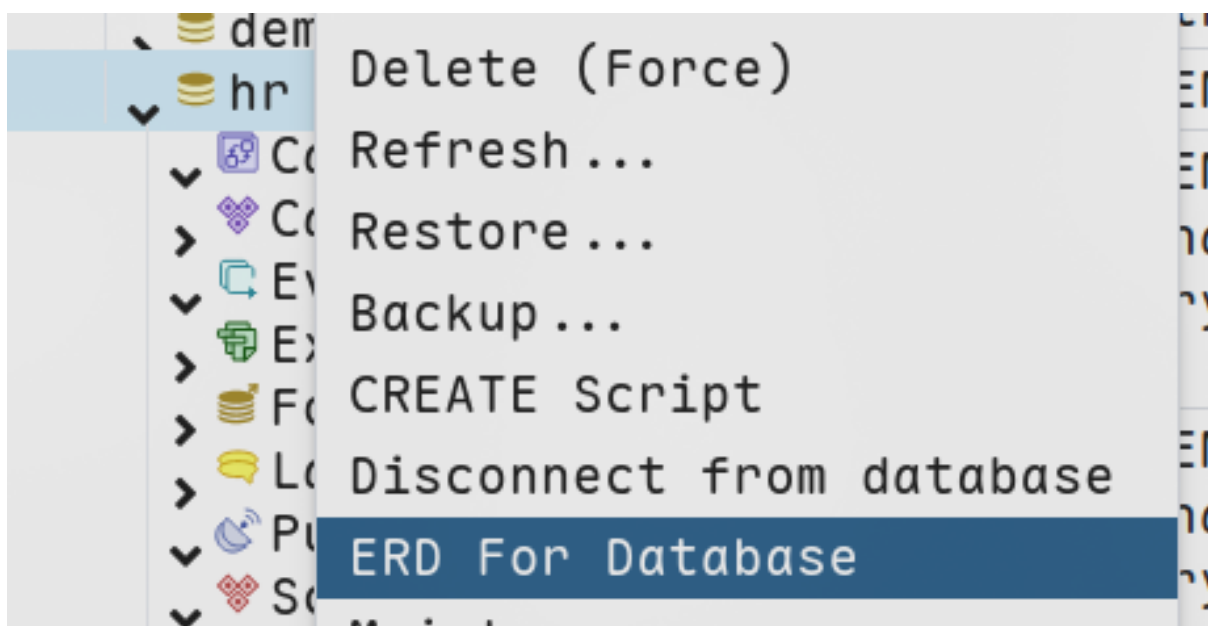


Рис 46: Меню создание ERD-диаграммы.

Затем я скорректировал вид своей диаграммы в соответствии с условием (Рис. 47):

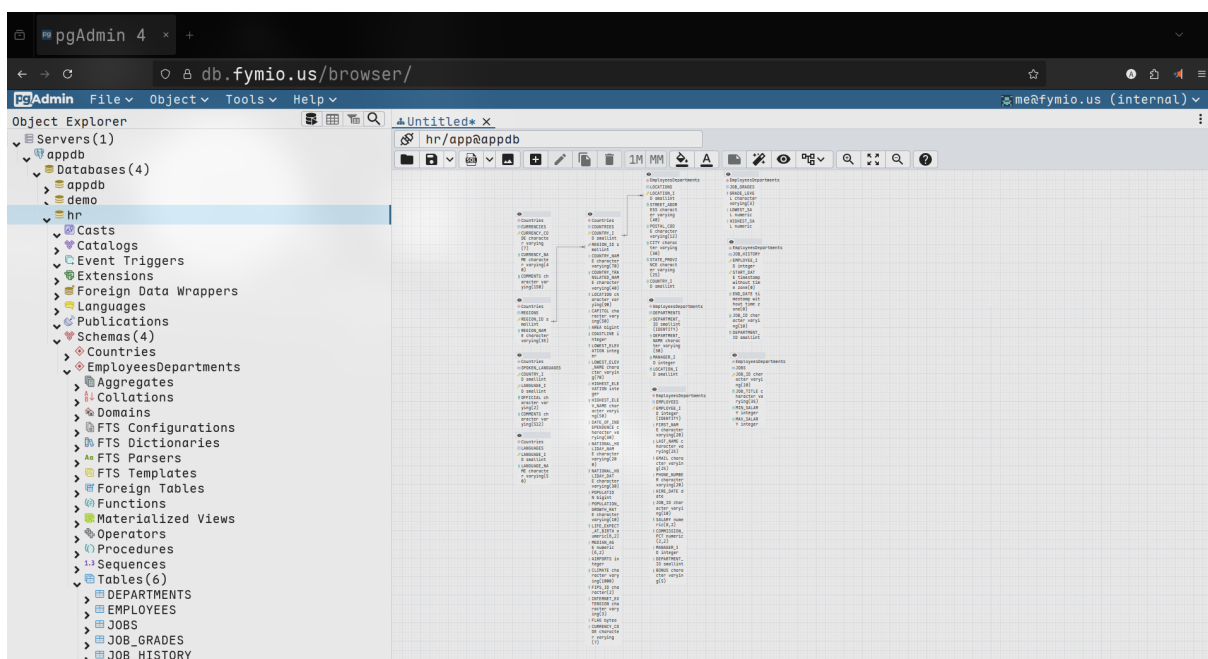


Рис 47: Вид ERD-диаграммы.

Я экспортировал снимок диаграммы, нажав на соответствующую кнопку. Результат (Рис. 48):

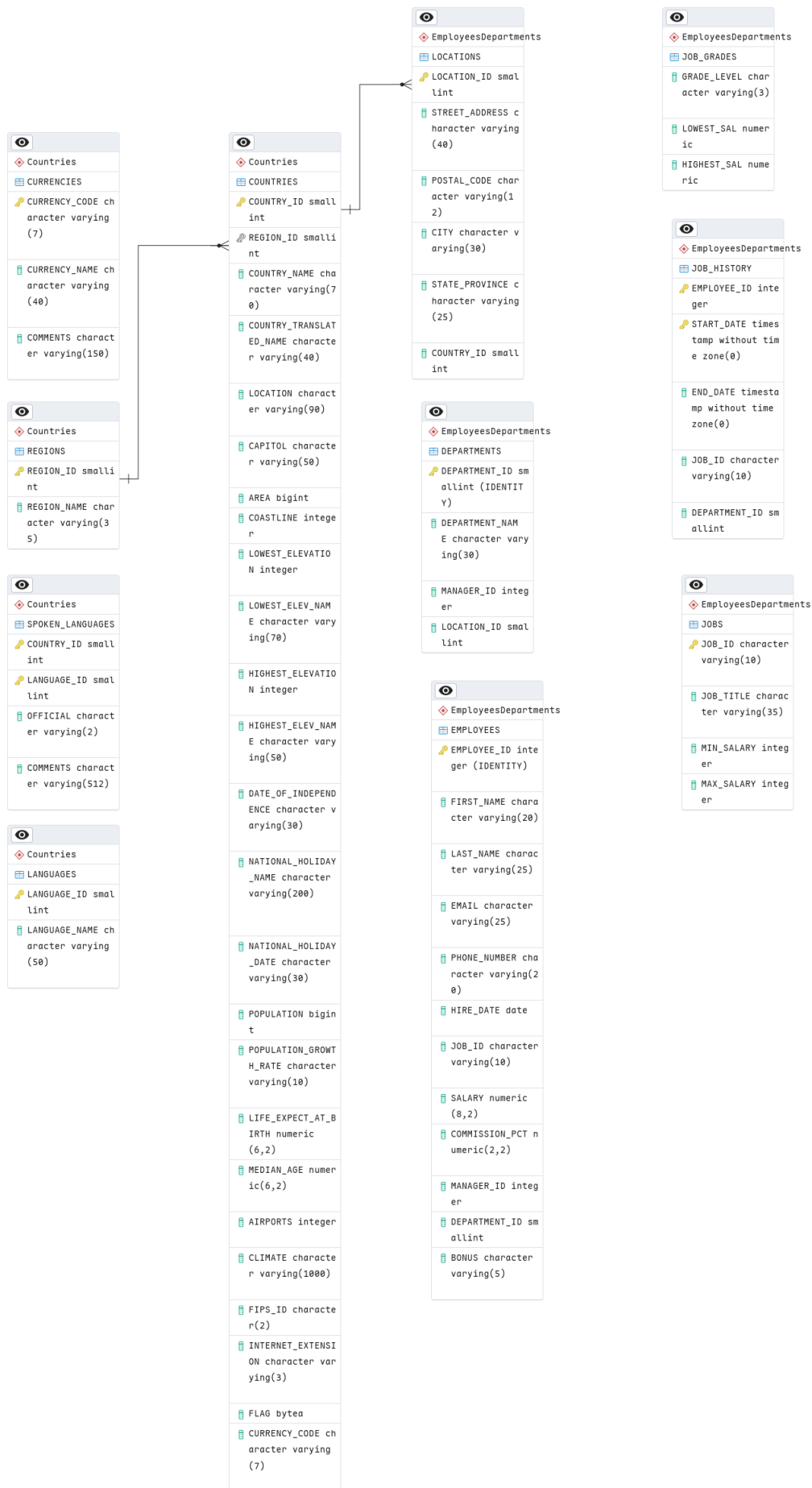


Рис 48: Рисунок диаграммы.

Сохранил проект диаграммы (Рис. 49):

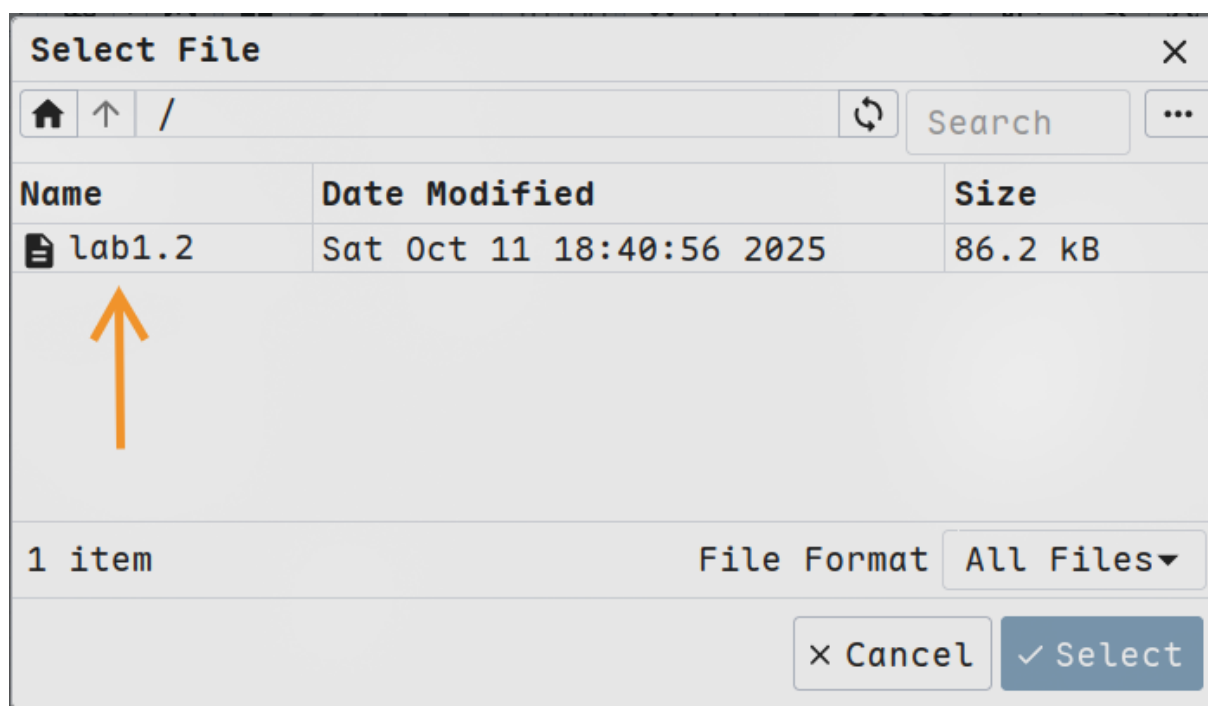


Рис 49: Сохраненный проект диаграммы.

Задание 7. Добавление данных в таблицу.

Я раскрыл меню для генерирования скрипта по вставке данных в таблицу Countries.CURRENCIES (Рис. 50):

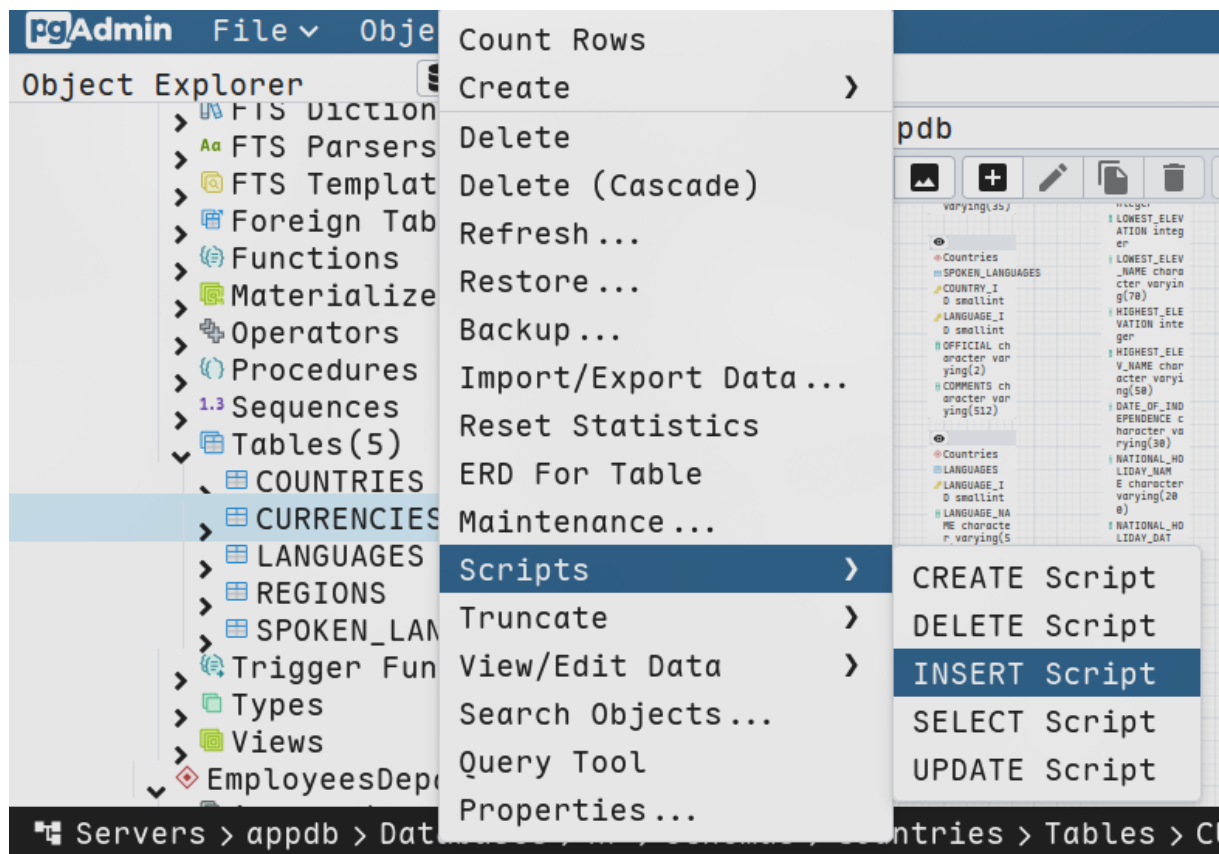


Рис 50: Меню для генерирования скрипта.

Затем я добавил следующий код в скрипт (Рис. 51):

```
INSERT INTO "Countries"."CURRENCIES" ("COMMENTS",
"CURRENCY_NAME", "CURRENCY_CODE")
VALUES ('EuroZone', 'Euro', 'EUR');
```

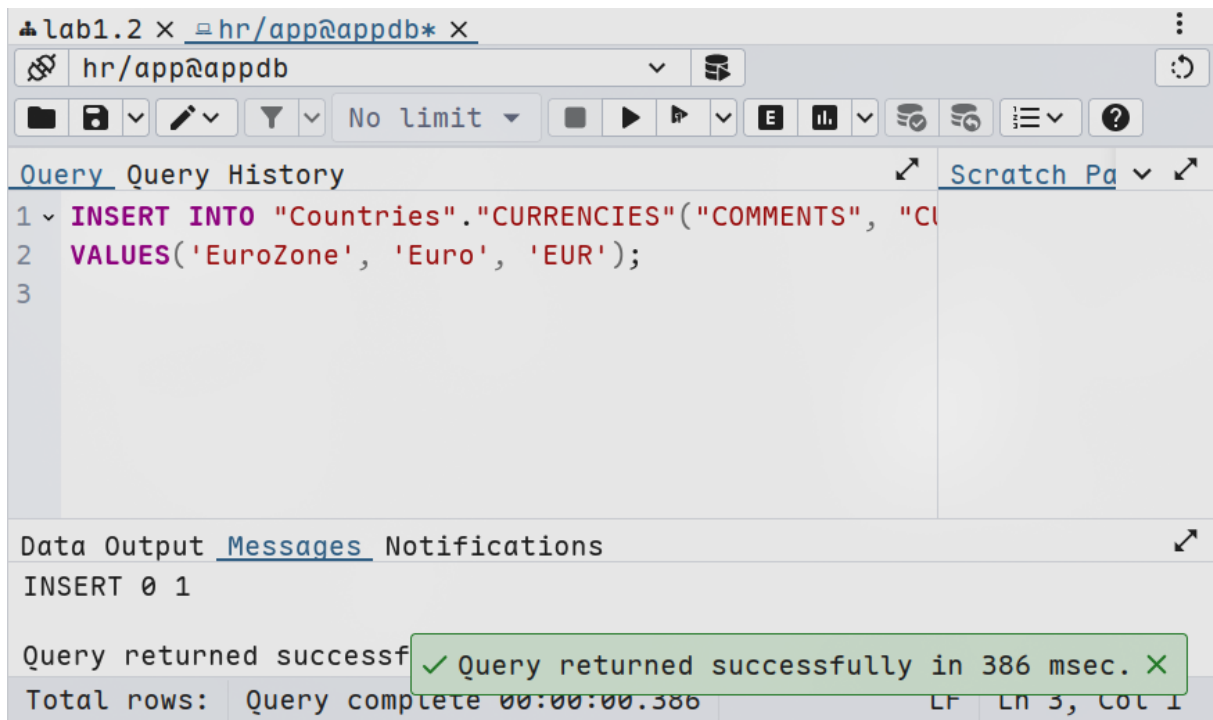


Рис 51: Скрипт в query tool.

После запуска скрипта проверил при помощи (Рис. 52):

```
select * from "Countries"."CURRENCIES";
```

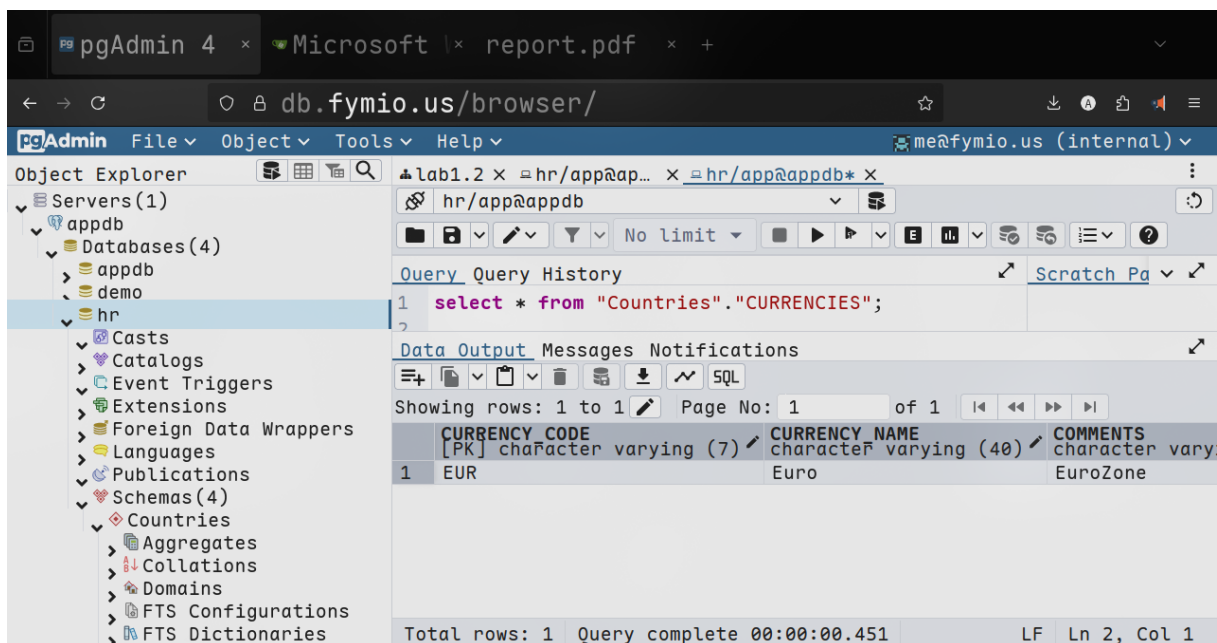


Рис 52: Скрипт в query tool.

В столбце EMPLOYEE_ID я заменил identity с always на by default (Рис. 53).

EMPLOYEE_ID [X]

General Definition Constraints >

Default:

Not NULL? ☒

Type: NONE ✓ **IDENTITY** GENERATED

Identity: BY DEFAULT | v

Increment: 1

Start: 1

Minimum: 1

[i] [?] [X Close] [↺ Reset] [Save]

Рис 53: Изменение параметра identity.

Затем при помощи графического интерфейса pgadmin, я добавил запись в таблицу “EmployeesDepartments”. “EMPLOYEES” (Рис. 54, 55):

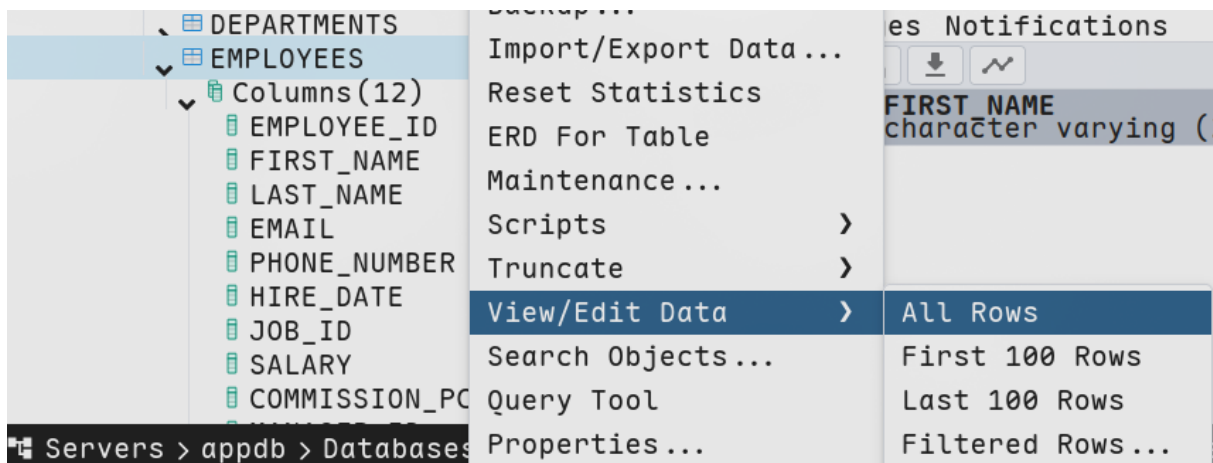


Рис 54: Просмотр строк таблицы.

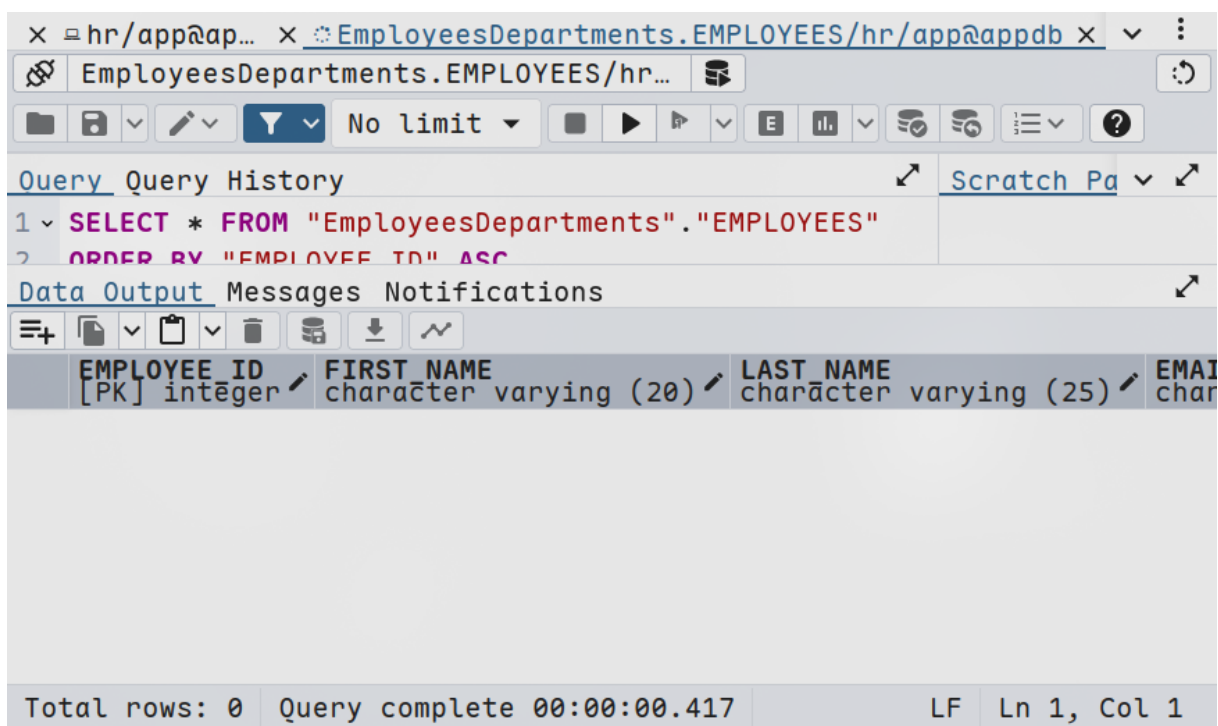


Рис 55: Пустые строки таблицы.

Затем я добавил строку и сохранил данные (Рис. 56, 57, 58):

	EMPLOYEE ID [PK] integer	FIRST_NAME character varying (20)	LAST_NAME character varying (25)	EMAIL character varying (25)	PHONE NUMBER character varying (20)	HIRE_DATE date	JOB_ID character varying (10)	SALARY numeric (8, 2)
0+	1	Steven	King	SKING	515.123.4567	2025-05-11	AD_PRES	24000.00

Рис 56: Добавление строки.

COMMI	MANAGER	DEP/	BONUS
number	integer	small	charo
0.23	[null]	90	[null]

Рис 57: Добавление строки.

PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMI	MANAGER	DEP/	BONUS
character var	date	character	numeric (	number	integer	small	charo
515.123.4567	2025-05-11	AD_PRES	24000.00	0.23	[null]	90	[null]

✓ Data saved successfully. ✕

Total rows: 0 Query complete 00:00:00.417

Рис 58: Новая строка сохранена.

Затем, в query tool я ввел следующий код (Рис. 59):

```
INSERT INTO "EmployeesDepartments"."EMPLOYEES" ("FIRST_NAME",
"LAST_NAME", "EMAIL", "PHONE_NUMBER", "JOB_ID", "SALARY",
"COMMISSION_PCT", "MANAGER_ID", "DEPARTMENT_ID")
VALUES(2, 'Steven', 'King', 'SKING', '515.123.4567',
'AD_PRES', 24000, null, null, 90);
```

hr/app@appdb*	hr/app@appdb	No limit	Query	Query History	Scratch Pa
<pre>1 INSERT INTO "EmployeesDepartments"."EMPLOYEES" 2 ("EMPLOYEE_ID", "FIRST_NAME", "LAST_NAME", "EMAIL", 3 VALUES (2, 'Steven', 'King', 'SKING', '515.123.4567 4</pre>					
Data Output Messages Notifications INSERT 0 1 Query returned successfully in 518 msec. ✓ Query returned successfully in 518 msec. ✕ Total rows: Query complete 00:00:00.518					

Рис 59: Код в query tool.

Проверим (Рис. 60):

```
SELECT * FROM "EmployeesDepartments"."EMPLOYEES";
```

The screenshot shows a database query tool interface. At the top, there's a toolbar with various icons for file operations, query execution, and settings. Below the toolbar, the 'Query' tab is active, displaying the SQL query: `SELECT * FROM "EmployeesDepartments"."EMPLOYEES";`. The 'Data Output' tab is also visible, showing the results of the query. The results are displayed in a table with four columns: `EMPLOYEE_ID` (integer), `FIRST_NAME` (character varying (20)), `LAST_NAME` (character varying (25)), and `EMAIL` (character varying (25)). The table contains two rows of data. The first row has `EMPLOYEE_ID` 1, `FIRST_NAME` Steven, `LAST_NAME` King, and `EMAIL` SKIN. The second row has `EMPLOYEE_ID` 2, `FIRST_NAME` Steven, `LAST_NAME` King, and `EMAIL` SKIN. At the bottom, a status bar indicates 'Total rows: 2' and 'Query complete 00:00:00.407'.

	EMPLOYEE_ID [PK] integer	FIRST_NAME character varying (20)	LAST_NAME character varying (25)	EMAIL character varying (25)
1	1	Steven	King	SKIN
2	2	Steven	King	SKIN

Total rows: 2 Query complete 00:00:00.407 LF Ln 2, Col 1

Рис 60: Скрипт в query tool.

Задание 8. Ограничения для столбцов.

Я добавил ограничения к таблицам "Countries"."SPOKEN_LANGUAGES" и "EmployeesDepartments"."EMPLOYEES" при помощи следующего кода (Рис. 61, 62):

```
ALTER TABLE "Countries"."SPOKEN_LANGUAGES"  
ADD CONSTRAINT "CTRY_NUM_FK1" FOREIGN KEY ("COUNTRY_ID")  
REFERENCES "Countries"."COUNTRIES" ("COUNTRY_ID");
```

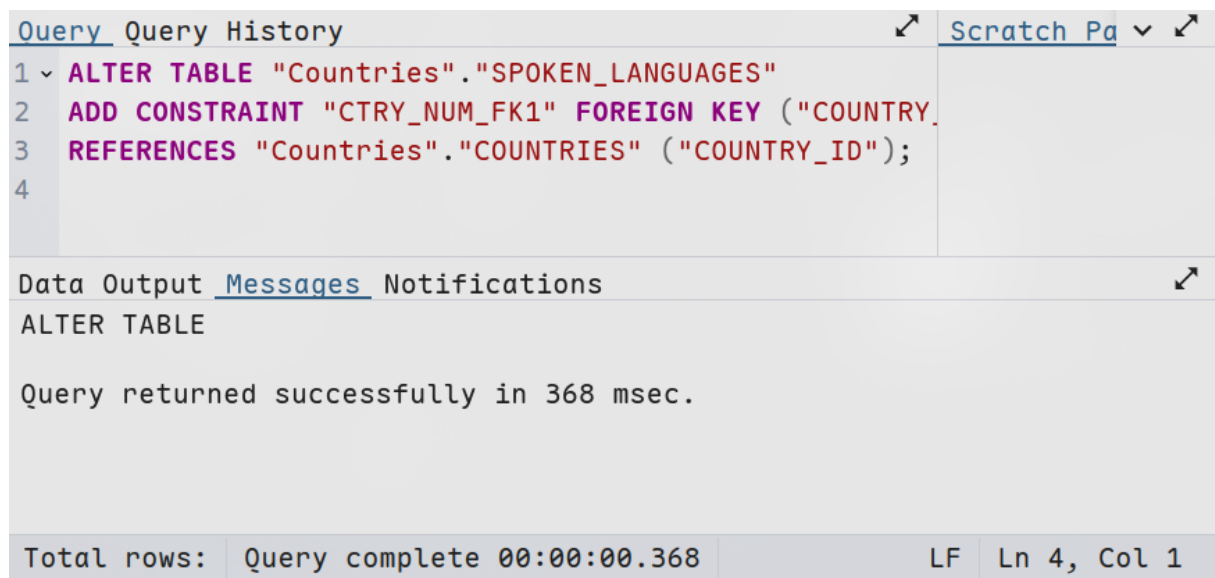


Рис 61: Добавления ограничений через query tool.

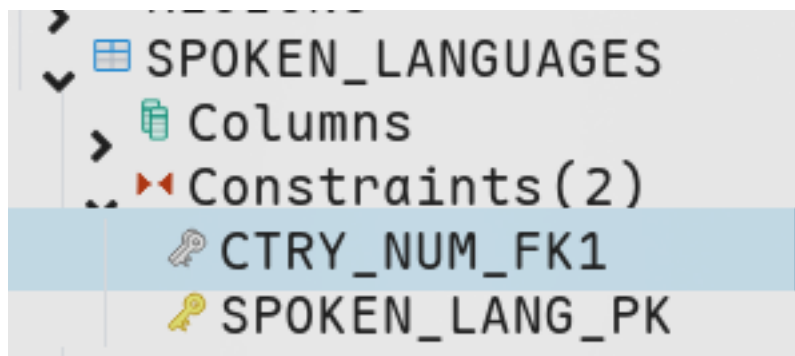


Рис 62: Ограничения таблицы SPOKEN_LANGUAGES.

и (Рис. 63, 64)

```
ALTER TABLE "EmployeesDepartments"."EMPLOYEES"
ADD CONSTRAINT "EMP_SALARY_MIN"
CHECK ("SALARY" > 0);
```

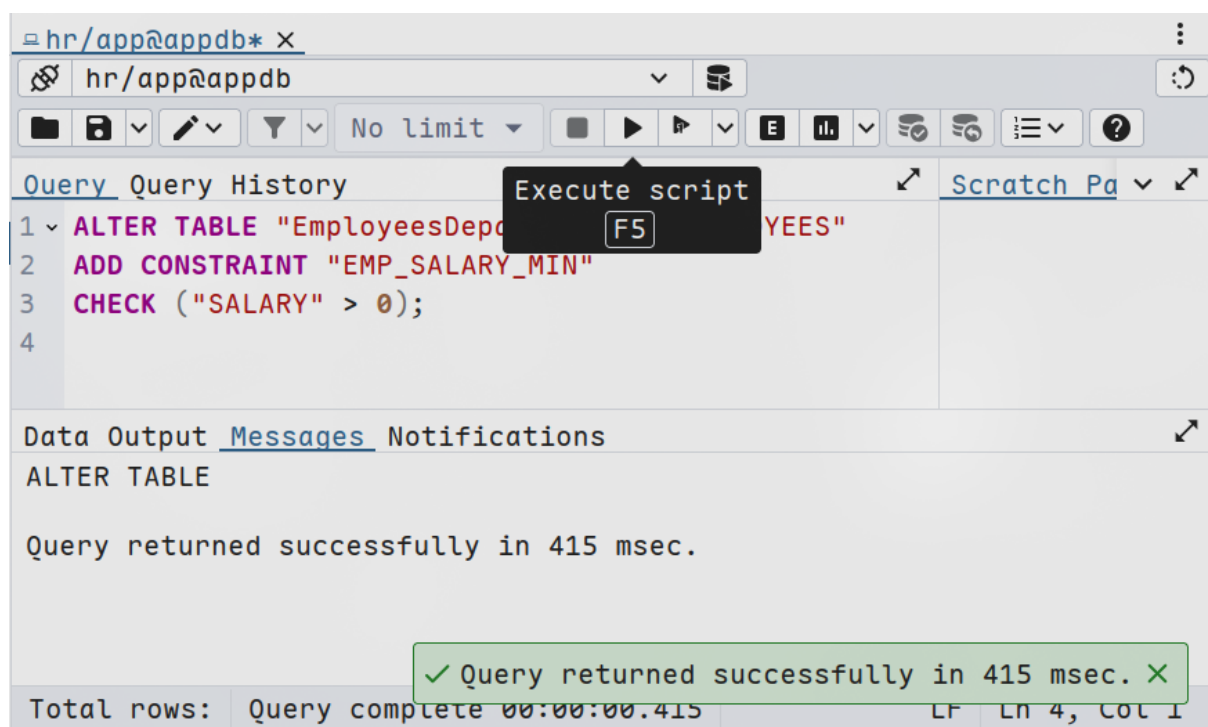


Рис 63: Добавление ограничения через query tool.

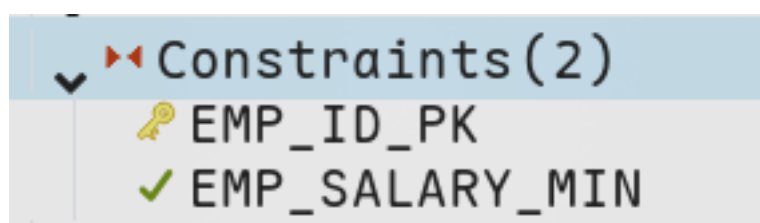


Рис 64: Добавленное ограничение.

Затем я добавил еще одно ограничение для таблицы JOB_HISTORY. Для этого в свойствах таблицы в разделе constraints добавил check с условием, что "END_DATE" > "START_DATE" (Рис. 65, 66):

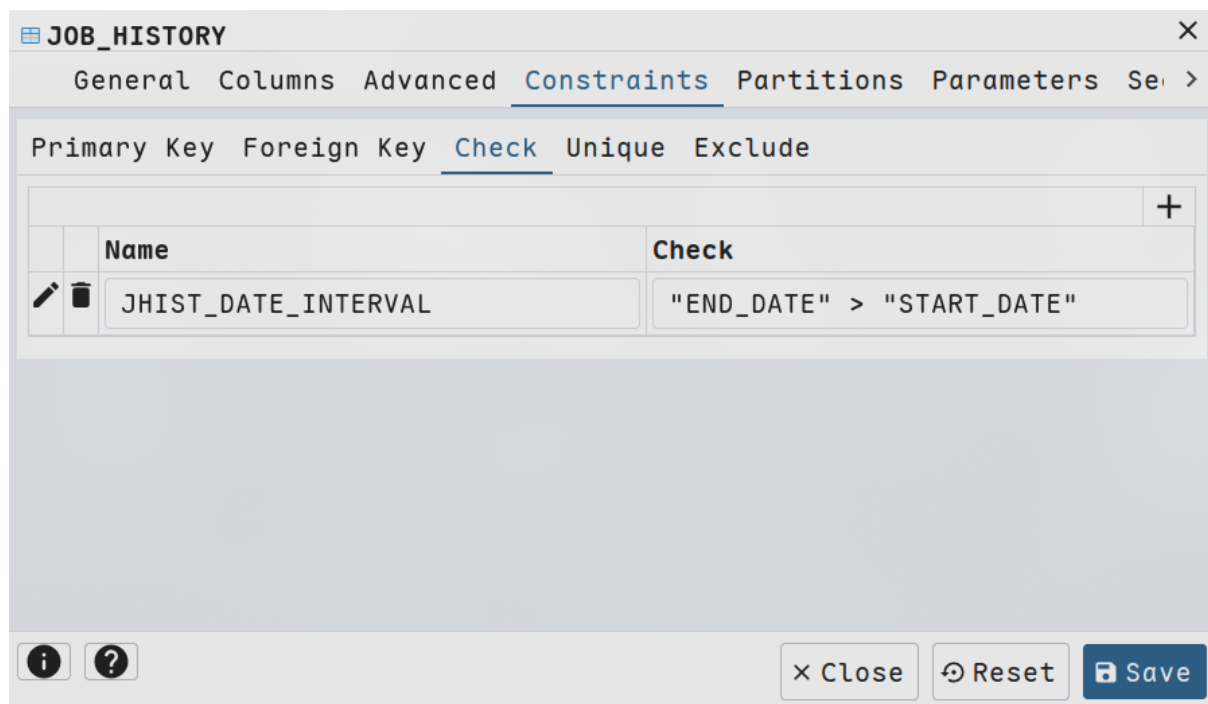


Рис 65: Добавление ограничения для JOB_HISTORY.

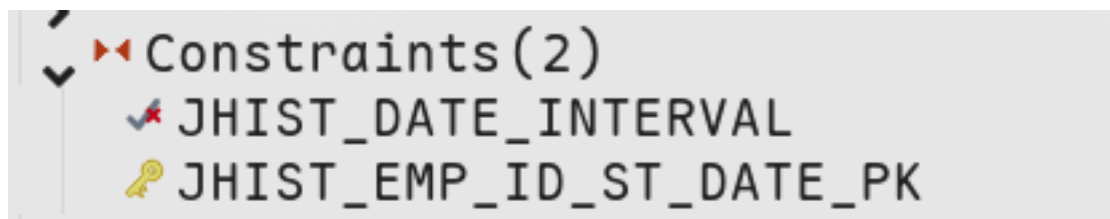


Рис 66: Добавленное ограничение.

Я включил это ограничение выключив соответствующую опцию (Рис. 67, 68):

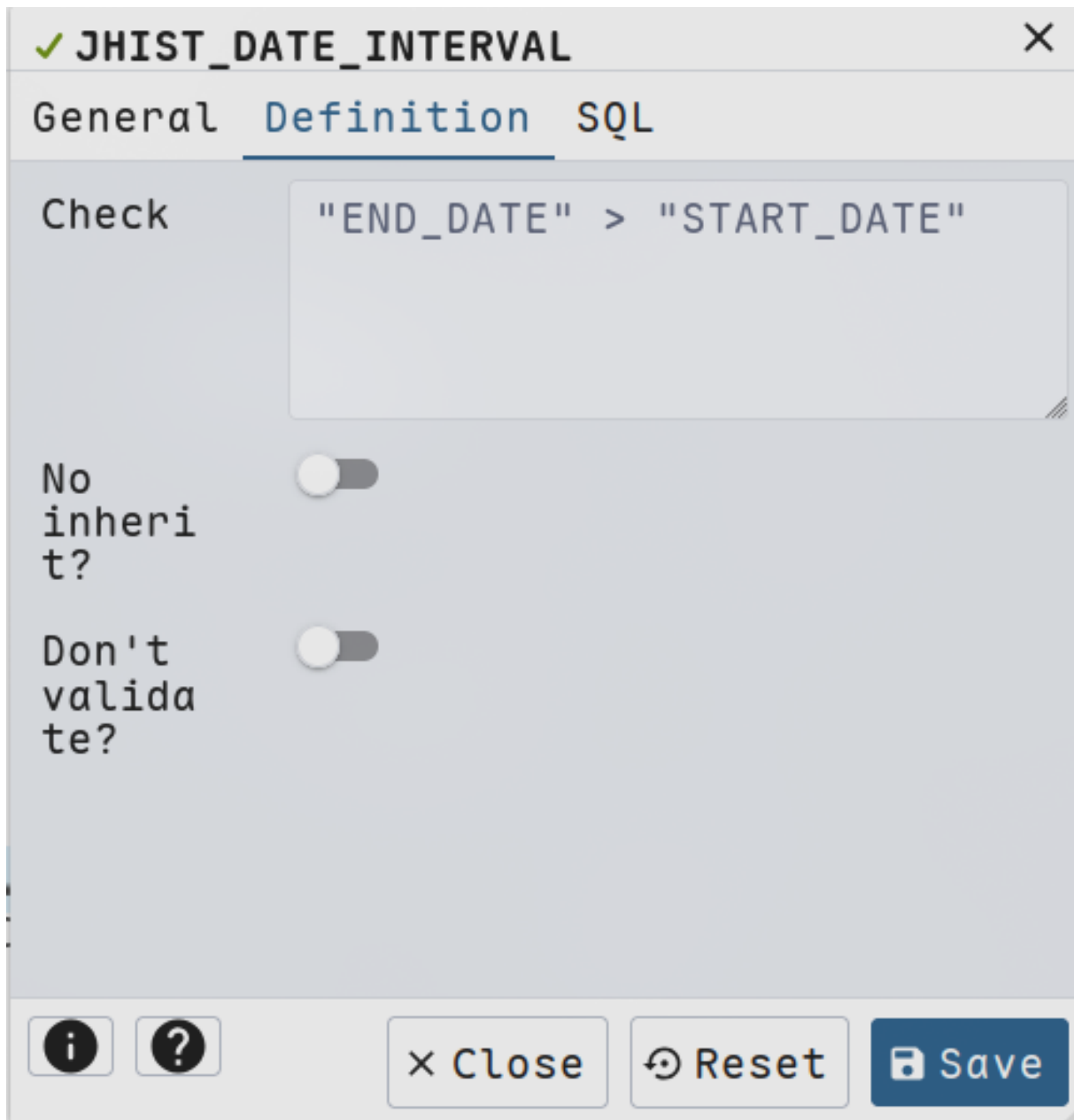


Рис 67: Выключенная функция "Don't validate?".

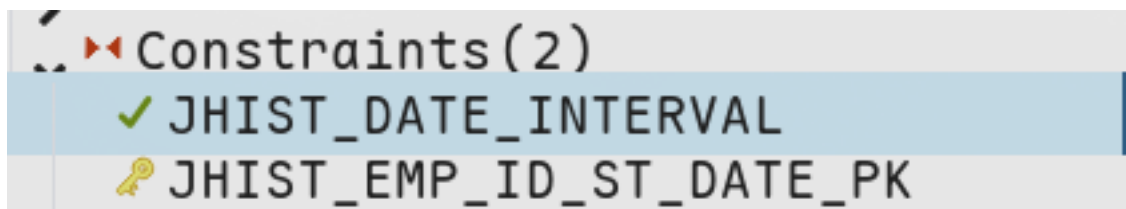


Рис 68: Включенное ограничение.

Также я проверил работоспособность ограничения (Рис. 69, 70):

```
INSERT INTO "EmployeesDepartments"."JOB_HISTORY"
("EMPLOYEE_ID", "START_DATE", "END_DATE", "JOB_ID",
```

```
"DEPARTMENT_ID")  
VALUES(200, '06-17-1993', '09-17-1987', 'AD_ASST', 90);
```

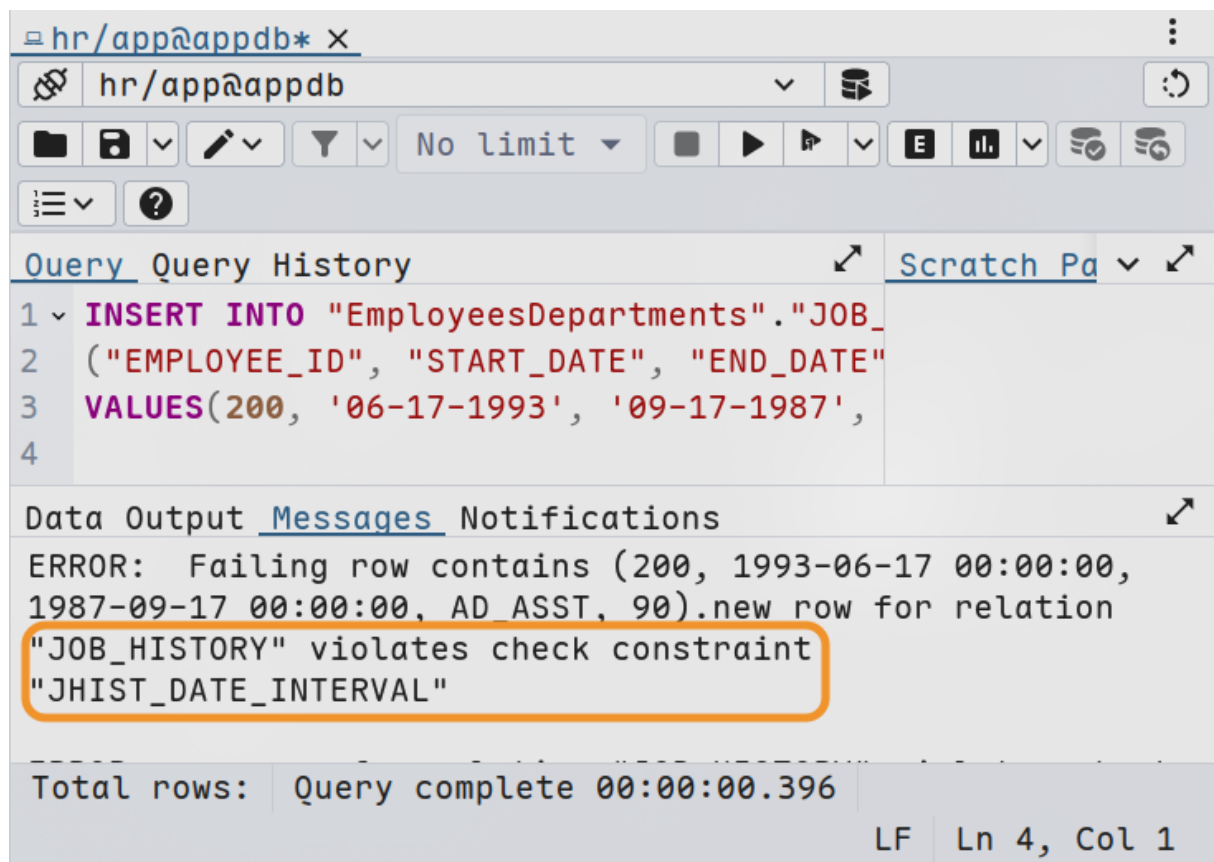


Рис 69: Ошибка. Скрипт не отрабатывает.

И

```
INSERT INTO "EmployeesDepartments"."JOB_HISTORY"  
("EMPLOYEE_ID", "START_DATE", "END_DATE", "JOB_ID",  
"DEPARTMENT_ID")  
VALUES(200, '06-17-1981', '06-17-1993', 'AD_ASST', 90);
```

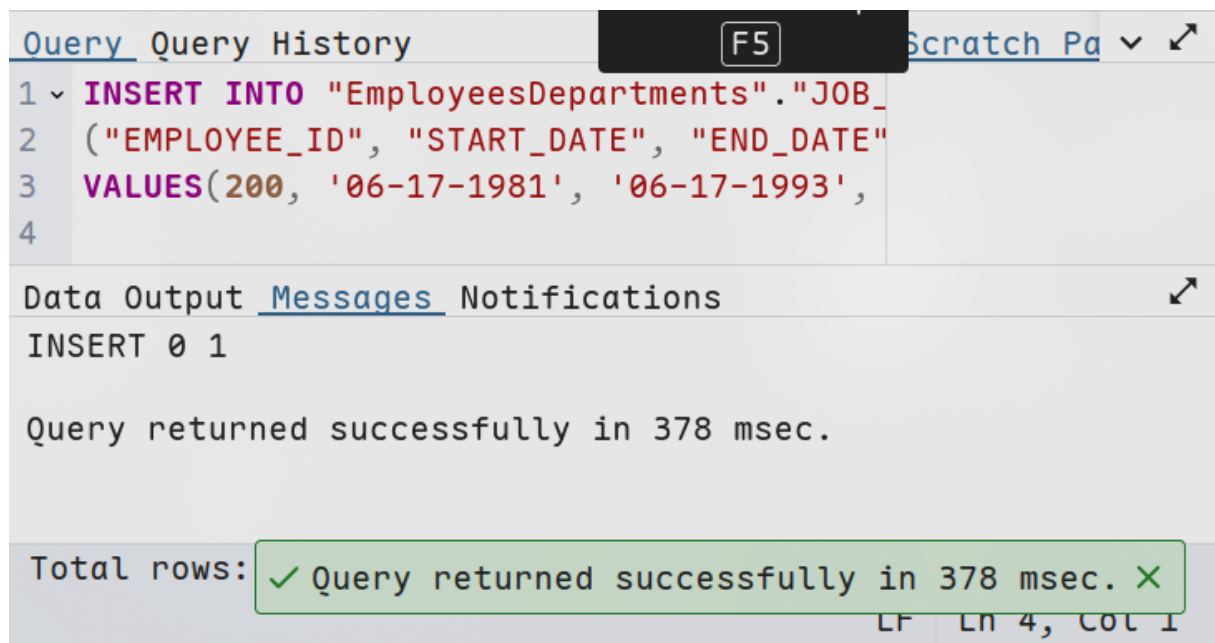


Рис 70: Скрипт отрабатывает без ошибки.

Задание 9. Очистка таблиц и завершение создания БД HR.

Я произвел очистку таблицы EMPLOYEES с помощью DDL команды TRUNCATE (Рис. 71):

```
TRUNCATE "EmployeesDepartments"."EMPLOYEES";
```

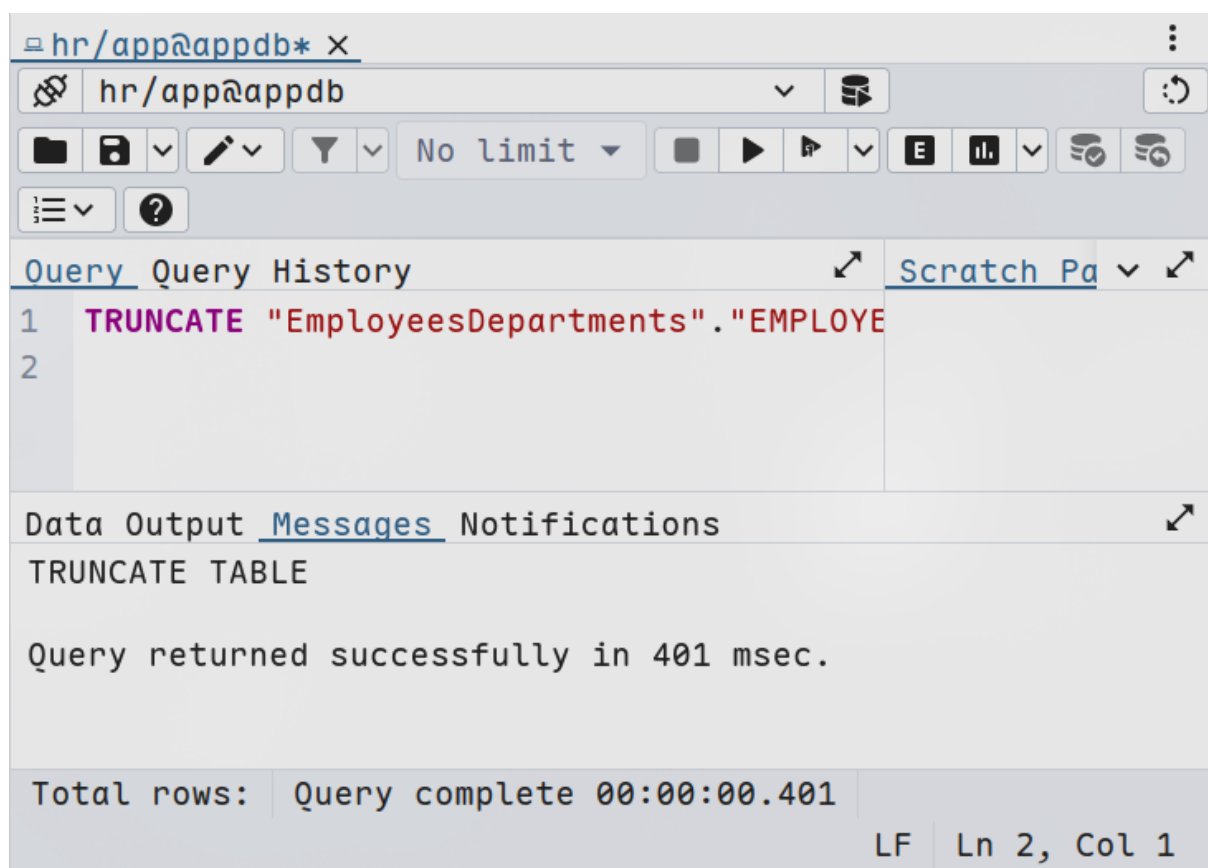


Рис 71: Выполнение команды `TRUNCATE`.

Также я очистил `JOB_HISTORY` и `CURRENCIES` (Рис. 72, 73):

```
TRUNCATE "EmployeesDepartments"."JOB_HISTORY";
```

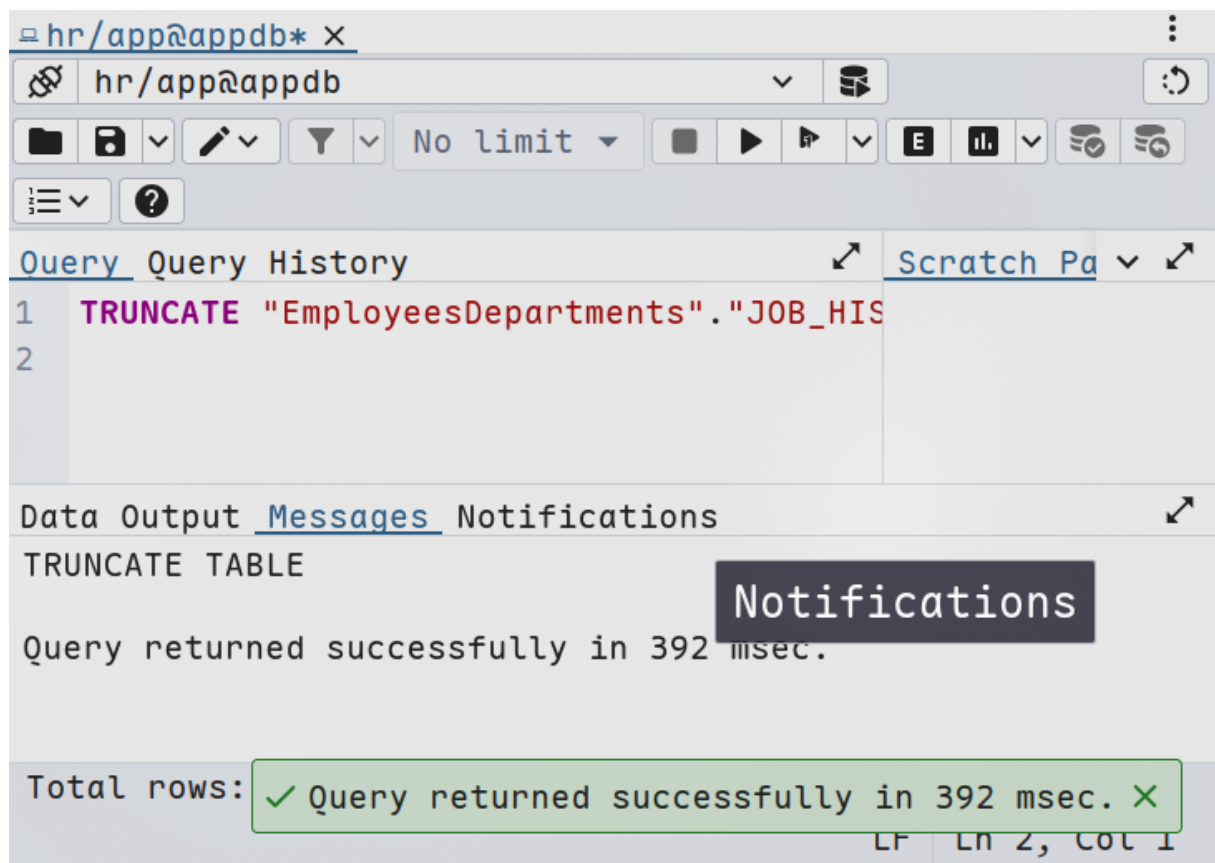


Рис 72: Выполнение команды TRUNCATE.

и

```
TRUNCATE "Countries"."CURRENCIES";
```

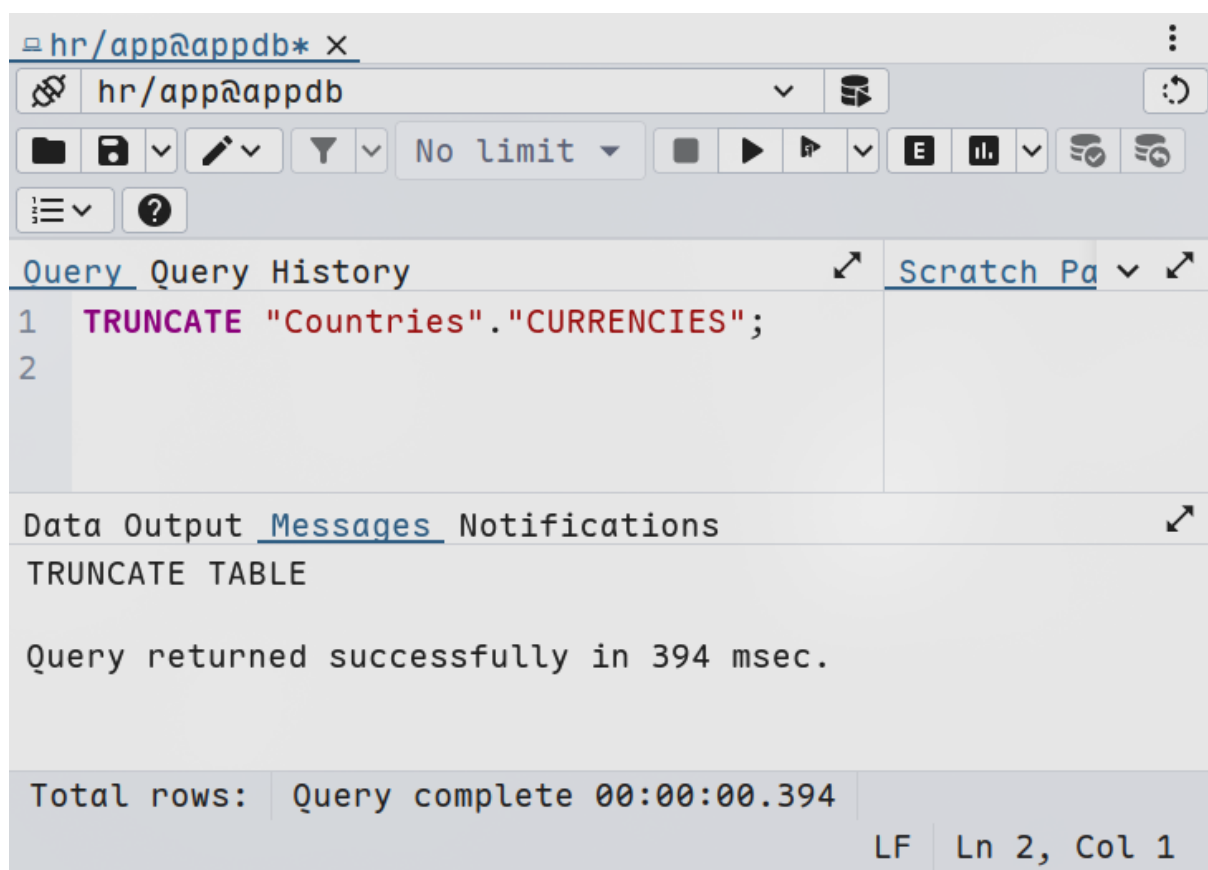


Рис 73: Исполнение команды `TRUNCATE` .

После очистки я выполнил код из файла `script2_alter.sql` (Рис. 74):

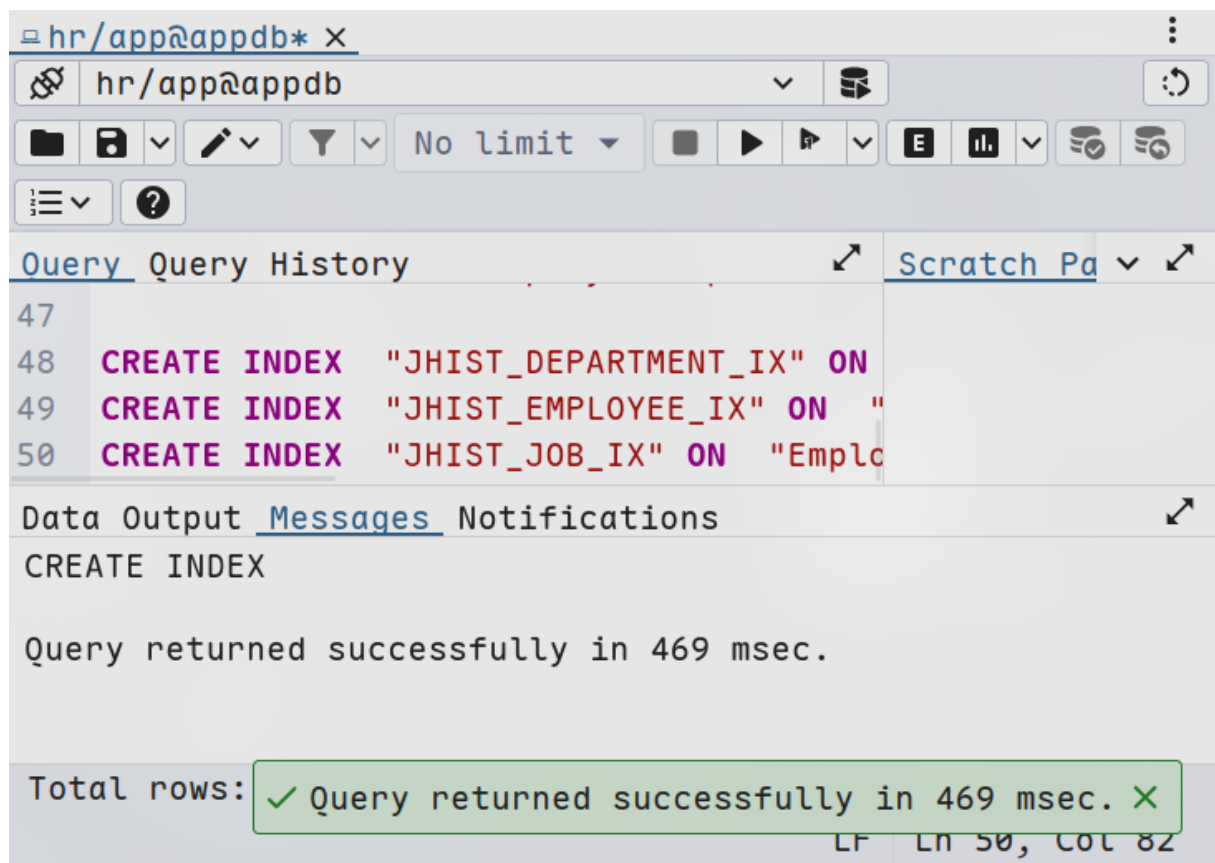


Рис 74: Исполнение скрипта `script2_alter.sql`.

И для добавления данных, я выполнил скрипт `script3_insert.sql` (Рис. 75):

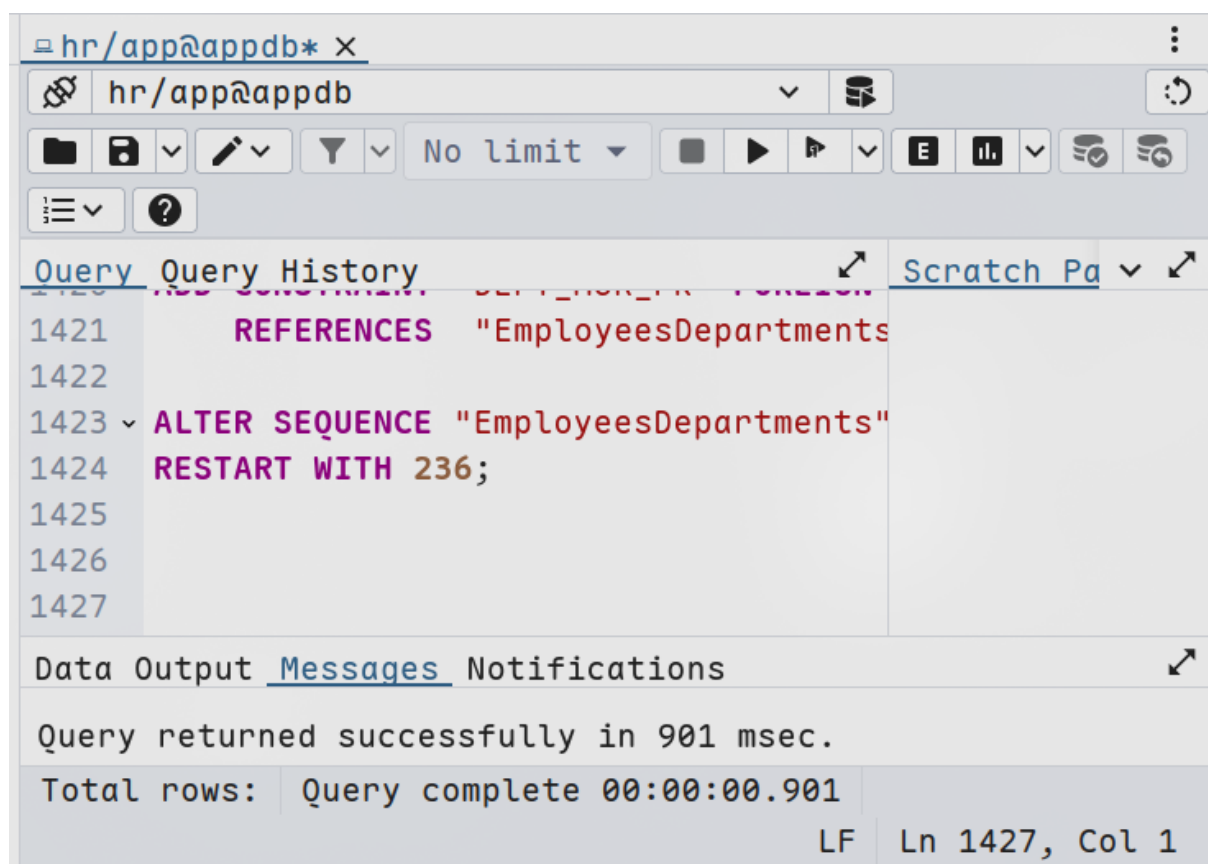
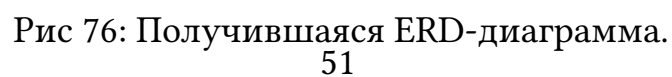


Рис 75: Исполнение скрипта `script3_insert.sql`.

Результат (Рис. 76):



Выводы.

В ходе выполнения работы я изучил установку и настройки postgresql, и освоил работу с pgadmin. При помощи docker я сделал сервер postgresql и pgadmin.

В процессе работы я научился:

- создавать и удалять базы данных как через pgadmin, так и с помощью sql;
- использовать настройки локализации и шаблоны при создании баз данных;
- создавать схемы для объединения объектов базы и разделения их по группам;
- создавать таблицы с pgadmin и SQL, задавая типы данных, значения по умолчанию, первичные ключи;
- изменять таблицы с помощью `ALTER TABLE`, добавлять столбцы, ключи, а также ограничения, проверки;
- создавать связи между таблицами и обеспечивать целостность данных с помощью внешних ключей;
- добавлять индексы;
- использовать команды dml;
- очищать таблицы при помощи `TRUNCATE` и выполнять внешние sql-скрипты для массового изменения и заполнения данных;
- визуализировать структуру базы данных при помощи ег-диаграмм, редактировать и экспортировать их.

Кроме того, в процессе работы я разобрался с ошибками, возникающими при создании баз данных с разными локалями и при несогласованности ключей, и научился устранять их настройкой параметров и порядка выполнения операций.