

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
Факультет инфокоммуникационных технологий

ОТЧЕТ
О ЛАБОРАТОРНОЙ РАБОТЕ № 3
по теме: Создание таблиц базы данных POSTGRESQL
по дисциплине: Проектирование и реализация баз данных

Специальность:

09.03.03 Мобильные и сетевые технологии

Проверил:

Говорова М.М. _____

Дата: «__» ____ 2021 г.

Оценка _____

Выполнил:

студент группы К3241

Касаткин Д.А.

Санкт-Петербург 2021 г.

ЦЕЛЬ РАБОТЫ

Овладеть практическими навыками создания таблиц базы данных PostgreSQL 1X, заполнения их рабочими данными, резервного копирования и восстановления БД.

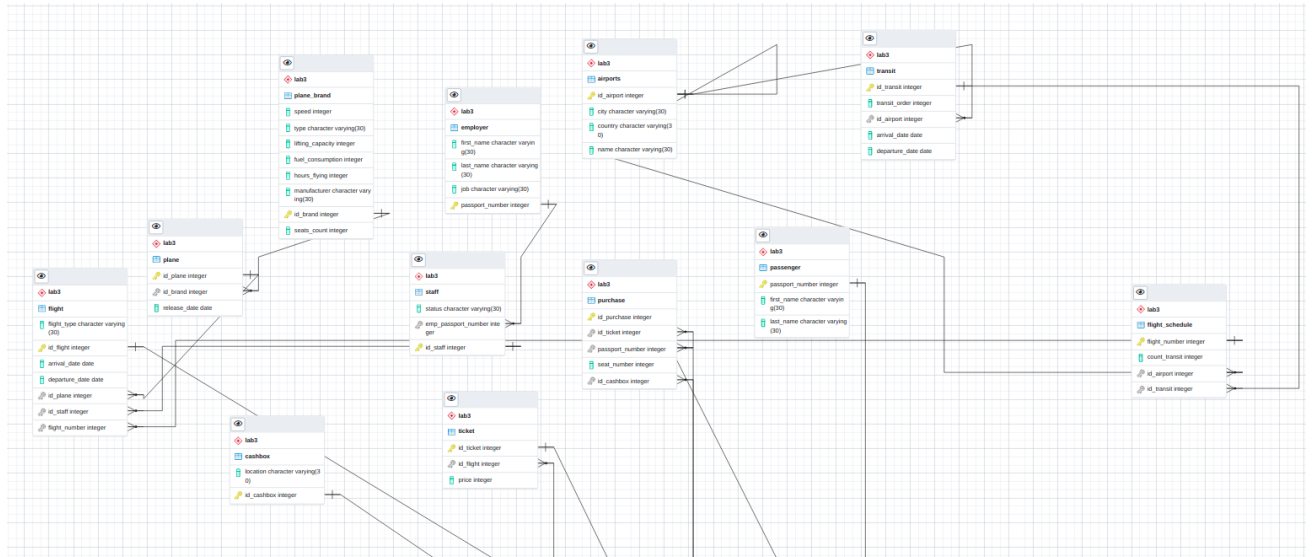
ПРАКТИЧЕСКОЕ ЗАДАНИЕ

Создать базу данных с использованием pgAdmin 4 (Вариант 8. БД «Аэропорт»).

1. Создать схему в составе базы данных.
2. Создать таблицы базы данных.
3. Установить ограничения на данные: *Primary Key, Unique, Check, Foreign Key*.
4. Заполнить таблицы БД рабочими данными.
5. Создать резервную копию БД.
Указание:
Создать две резервные копии:
 - с расширением *CUSTOM* для восстановления БД;
 - с расширением *PLAIN* для листинга (в отчете);
 - при создании резервных копий БД настроить параметры *Dump options* для *Type of objects* и *Queries* .
6. Восстановить БД.

ВЫПОЛНЕНИЕ

1. Наименование БД “postgres”
2. Схема логической модели базы данных, сгенерированная в Generate ERD



3. Создание базы данных

```
1 -- Database: postgres
2
3 -- DROP DATABASE postgres;
4
5 CREATE DATABASE postgres
6     WITH
7     OWNER = postgres
8     ENCODING = 'UTF8'
9     LC_COLLATE = 'en_US.UTF-8'
10    LC_CTYPE = 'en_US.UTF-8'
11    TABLESPACE = pg_default
12    CONNECTION LIMIT = -1;
13
14 COMMENT ON DATABASE postgres
15     IS 'default administrative connection database';
```

4. Создание схемы

```
1 -- SCHEMA: lab3
2
3 -- DROP SCHEMA lab3 ;
4
5 CREATE SCHEMA lab3
6     AUTHORIZATION daniil;
```

5. Создание таблиц, колонок и ограничений

```
1 -- Table: lab3.airports
2
3 -- DROP TABLE lab3.airports;
4
5 CREATE TABLE IF NOT EXISTS lab3.airports
6 (
7     id_airport integer NOT NULL,
8     city character varying(30) COLLATE pg_catalog."default" NOT NULL,
9     country character varying(30) COLLATE pg_catalog."default" NOT NULL,
10    name character varying(30) COLLATE pg_catalog."default" NOT NULL,
11    CONSTRAINT airports_pkey PRIMARY KEY (id_airport)
12 )
13 WITH (
14     OIDS = FALSE
15 )
16 TABLESPACE pg_default;
17
18 ALTER TABLE lab3.airports
19     OWNER to daniil;
```

```

1 -- Table: lab3.cashbox
2
3 -- DROP TABLE lab3.cashbox;
4
5 CREATE TABLE IF NOT EXISTS lab3.cashbox
6 (
7     location character varying(30) COLLATE pg_catalog."default" NOT NULL,
8     id_cashbox integer NOT NULL,
9     CONSTRAINT cashbox_pkey PRIMARY KEY (id_cashbox)
10 )
11 WITH (
12     OIDS = FALSE
13 )
14 TABLESPACE pg_default;
15
16 ALTER TABLE lab3.cashbox
17     OWNER to daniil;

```

```

1 -- Table: lab3.employer
2
3 -- DROP TABLE lab3.employer;
4
5 CREATE TABLE IF NOT EXISTS lab3.employer
6 (
7     first_name character varying(30) COLLATE pg_catalog."default" NOT NULL,
8     last_name character varying(30) COLLATE pg_catalog."default" NOT NULL,
9     job character varying(30) COLLATE pg_catalog."default" NOT NULL,
10    passport_number integer NOT NULL,
11    CONSTRAINT employer_pkey PRIMARY KEY (passport_number),
12    CONSTRAINT job_check CHECK (job::text = ANY (ARRAY['первый пилот'::character varying::text, 'второй пилот'::character varying::text, 'стюард'::character varying::text, 'стюардесса'::ch
13 )
14 WITH (
15     OIDS = FALSE
16 )
17 TABLESPACE pg_default;
18
19 ALTER TABLE lab3.employer
20     OWNER to daniil;

```

Dashboard Properties SQL Statistics Dependencies Dependents postgres/daniil...

```

1 -- Table: lab3.flight
2
3 -- DROP TABLE lab3.flight;
4
5 CREATE TABLE IF NOT EXISTS lab3.flight
6 (
7     flight_type character varying(30) COLLATE pg_catalog."default" NOT NULL,
8     id_flight integer NOT NULL,
9     arrival_date date NOT NULL,
10    departure_date date NOT NULL,
11    id_plane integer NOT NULL,
12    id_staff integer NOT NULL,
13    flight_number integer NOT NULL,
14    CONSTRAINT flight_pkey PRIMARY KEY (id_flight),
15    CONSTRAINT flight_id_plane_fkey FOREIGN KEY (id_plane)
16        REFERENCES lab3.plane (id_plane) MATCH SIMPLE
17        ON UPDATE NO ACTION
18        ON DELETE NO ACTION
19        NOT VALID,
20    CONSTRAINT flight_id_staff_fkey FOREIGN KEY (id_staff)
21        REFERENCES lab3.staff (id_staff) MATCH SIMPLE
22        ON UPDATE NO ACTION
23        ON DELETE NO ACTION
24        NOT VALID,
25    CONSTRAINT flight_number FOREIGN KEY (flight_number)
26        REFERENCES lab3.flight_schedule (flight_number) MATCH SIMPLE
27        ON UPDATE NO ACTION
28        ON DELETE NO ACTION
29        NOT VALID,
30    CONSTRAINT departure_date_check CHECK (departure_date > '1900-01-01'::date) NOT VALID,
31    CONSTRAINT arrival_date_check CHECK (arrival_date > departure_date) NOT VALID
32 )
33 WITH (
34     OIDS = FALSE
35 )
36 TABLESPACE pg_default;
37
38 ALTER TABLE lab3.flight
39     OWNER to daniil;
40 -- Index: fki_flight_number
41
42 -- DROP INDEX lab3.fki_flight_number;
43
44 CREATE INDEX fki_flight_number
45     ON lab3.flight USING btree
46     (flight_number ASC NULLS LAST)
47     TABLESPACE pg_default;
48 -- Index: fki_id_plane
49
50 -- DROP INDEX lab3.fki_id_plane;

```

```

14 CONSTRAINT flight_pkey PRIMARY KEY (id_flight),
15 CONSTRAINT flight_id_plane_fkey FOREIGN KEY (id_plane)
16 REFERENCES lab3_plane (id_plane) MATCH SIMPLE
17 ON UPDATE NO ACTION
18 ON DELETE NO ACTION
19 NOT VALID,
20 CONSTRAINT flight_id_staff_fkey FOREIGN KEY (id_staff)
21 REFERENCES lab3_staff (id_staff) MATCH SIMPLE
22 ON UPDATE NO ACTION
23 ON DELETE NO ACTION
24 NOT VALID,
25 CONSTRAINT flight_number FOREIGN KEY (flight_number)
26 REFERENCES lab3_flight_schedule (flight_number) MATCH SIMPLE
27 ON UPDATE NO ACTION
28 ON DELETE NO ACTION
29 NOT VALID,
30 CONSTRAINT departure_date_check CHECK (departure_date > '1980-01-01'::date) NOT VALID,
31 CONSTRAINT arrival_date_check CHECK (arrival_date > departure_date) NOT VALID
32 )
33 WITH (
34 OIDS = FALSE
35 )
36 TABLESPACE pg_default;
37
38 ALTER TABLE lab3_flight
39 OWNER to daniel;
40 -- Index: fki_flight_number
41
42 -- DROP INDEX lab3.fki_flight_number;
43
44 CREATE INDEX fki_flight_number
45 ON lab3_flight USING btree
46 (flight_number ASC NULLS LAST)
47 TABLESPACE pg_default;
48 -- Index: fki_id_plane
49
50 -- DROP INDEX lab3.fki_id_plane;
51
52 CREATE INDEX fki_id_plane
53 ON lab3_flight USING btree
54 (id_plane ASC NULLS LAST)
55 TABLESPACE pg_default;
56 -- Index: fki_id_staff
57
58 -- DROP INDEX lab3.fki_id_staff;
59
60 CREATE INDEX fki_id_staff
61 ON lab3_flight USING btree
62 (id_staff ASC NULLS LAST)
63 TABLESPACE pg_default;

```

```

1 -- Table: lab3_flight_schedule
2
3 -- DROP TABLE lab3_flight_schedule;
4
5 CREATE TABLE IF NOT EXISTS lab3_flight_schedule
6 (
7     flight_number integer NOT NULL,
8     count_transit integer NOT NULL,
9     id_airport integer NOT NULL,
10    id_transit integer NOT NULL,
11    CONSTRAINT flight_schedule_pkey PRIMARY KEY (flight_number),
12    CONSTRAINT flight_schedule_id_airport_fkey FOREIGN KEY (id_airport)
13    REFERENCES lab3_airports (id_airport) MATCH SIMPLE
14    ON UPDATE NO ACTION
15    ON DELETE NO ACTION
16    NOT VALID,
17    CONSTRAINT flight_schedule_id_transit_fkey FOREIGN KEY (id_transit)
18    REFERENCES lab3_transit (id_transit) MATCH SIMPLE
19    ON UPDATE NO ACTION
20    ON DELETE NO ACTION
21    NOT VALID
22 )
23 WITH (
24 OIDS = FALSE
25 )
26 TABLESPACE pg_default;
27
28 ALTER TABLE lab3_flight_schedule
29 OWNER to daniel;
30 -- Index: fki_id_airport
31
32 -- DROP INDEX lab3.fki_id_airport;
33
34 CREATE INDEX fki_id_airport
35 ON lab3_flight_schedule USING btree
36 (id_airport ASC NULLS LAST)
37 TABLESPACE pg_default;
38 -- Index: fki_id_transit
39
40 -- DROP INDEX lab3.fki_id_transit;
41
42 CREATE INDEX fki_id_transit
43 ON lab3_flight_schedule USING btree
44 (id_transit ASC NULLS LAST)
45 TABLESPACE pg_default;

```

[Database](#) [Properties](#) [SQL](#) [Statistics](#) [Dependencies](#) [Dependents](#) [postgres/ramm...](#)

```

1 -- Table: lab3.passenger
2
3 -- DROP TABLE lab3.passenger;
4
5 CREATE TABLE IF NOT EXISTS lab3.passenger
6 (
7     passport_number integer NOT NULL,
8     first_name character varying(30) COLLATE pg_catalog."default" NOT NULL,
9     last_name character varying(30) COLLATE pg_catalog."default" NOT NULL,
10    CONSTRAINT passenger_pkey PRIMARY KEY (passport_number)
11 )
12 WITH (
13     OIDS = FALSE
14 )
15 TABLESPACE pg_default;
16
17 ALTER TABLE lab3.passenger
18     OWNER to daniil;

```

```

1 -- Table: lab3.plane
2
3 -- DROP TABLE lab3.plane;
4
5 CREATE TABLE IF NOT EXISTS lab3.plane
6 (
7     id_plane integer NOT NULL,
8     id_brand integer NOT NULL,
9     release_date date NOT NULL,
10    CONSTRAINT plane_pkey PRIMARY KEY (id_plane),
11    CONSTRAINT plane_id_brand_fkey FOREIGN KEY (id_brand)
12        REFERENCES lab3.plane_brand (id_brand) MATCH SIMPLE
13        ON UPDATE NO ACTION
14        ON DELETE NO ACTION
15        NOT VALID
16 )
17 WITH (
18     OIDS = FALSE
19 )
20 TABLESPACE pg_default;
21
22 ALTER TABLE lab3.plane
23     OWNER to daniil;
24 -- Index: fki_id_brand
25
26 -- DROP INDEX lab3.fki_id_brand;
27
28 CREATE INDEX fki_id_brand
29     ON lab3.plane USING btree
30     (id_brand ASC NULLS LAST)
31     TABLESPACE pg_default;

```



```

1 -- Table: lab3.plane_brand
2
3 -- DROP TABLE lab3.plane_brand;
4
5 CREATE TABLE IF NOT EXISTS lab3.plane_brand
6 (
7     speed integer NOT NULL,
8     type character varying(30) COLLATE pg_catalog."default" NOT NULL,
9     lifting_capacity integer NOT NULL,
10    fuel_consumption integer NOT NULL,
11    hours_flying integer NOT NULL,
12    manufacturer character varying(30) COLLATE pg_catalog."default" NOT NULL,
13    id_brand integer NOT NULL,
14    seats_count integer NOT NULL,
15    CONSTRAINT plane_brand_pkey PRIMARY KEY (id_brand),
16    CONSTRAINT type_check CHECK (type::text = ANY (ARRAY['крылово́й'::character varying::text, 'пассажирский'::character varying::text])) NOT VALID,
17    CONSTRAINT man_check CHECK (manufacturer::text = ANY (ARRAY['Airbus'::character varying::text, 'Boeing'::character varying::text, 'WN'::character varying::text, 'TV'::character varying::text]))
18    CONSTRAINT seats_check CHECK (seats_count > 0 AND seats_count < 100) NOT VALID
19 )
20 WITH (
21     OIDS = FALSE
22 )
23 TABLESPACE pg_default;
24
25 ALTER TABLE lab3.plane_brand
26     OWNER to danil;

```

```

1 -- Table: lab3.purchase
2
3 -- DROP TABLE lab3.purchase;
4
5 CREATE TABLE IF NOT EXISTS lab3.purchase
6 (
7     id_purchase integer NOT NULL,
8     id_ticket integer NOT NULL,
9     passport_number integer NOT NULL,
10    seat_number integer NOT NULL,
11    id_cashbox integer NOT NULL,
12    CONSTRAINT purchase_pkey PRIMARY KEY (id_purchase),
13    CONSTRAINT purchase_id_cashbox_fkey FOREIGN KEY (id_cashbox)
14        REFERENCES lab3.cashbox (id_cashbox) MATCH SIMPLE
15        ON UPDATE NO ACTION
16        ON DELETE NO ACTION
17        NOT VALID,
18    CONSTRAINT purchase_id_ticket_fkey FOREIGN KEY (id_ticket)
19        REFERENCES lab3.ticket (id_ticket) MATCH SIMPLE
20        ON UPDATE NO ACTION
21        ON DELETE NO ACTION
22        NOT VALID,
23    CONSTRAINT purchase_passport_number_fkey FOREIGN KEY (passport_number)
24        REFERENCES lab3.passenger (passport_number) MATCH SIMPLE
25        ON UPDATE NO ACTION
26        ON DELETE NO ACTION
27        NOT VALID
28 )
29 WITH (
30     OIDS = FALSE
31 )
32 TABLESPACE pg_default;
33
34 ALTER TABLE lab3.purchase
35     OWNER to daniil;
36 -- Index: fki_id_cashbox
37
38 -- DROP INDEX lab3.fki_id_cashbox;
39
40 CREATE INDEX fki_id_cashbox
41     ON lab3.purchase USING btree
42     (id_cashbox ASC NULLS LAST)
43     TABLESPACE pg_default;
44 -- Index: fki_id_pas
45
46 -- DROP INDEX lab3.fki_id_pas;
47
48 CREATE INDEX fki_id_pas
49     ON lab3.purchase USING btree
50     (passport_number ASC NULLS LAST)

```

```

1 -- Table: lab3.staff
2
3 -- DROP TABLE lab3.staff;
4
5 CREATE TABLE IF NOT EXISTS lab3.staff
6 (
7     status character varying(30) COLLATE pg_catalog."default" NOT NULL,
8     emp_passport_number integer NOT NULL,
9     id_staff integer NOT NULL,
10    CONSTRAINT staff_pkey PRIMARY KEY (id_staff),
11    CONSTRAINT id_emp FOREIGN KEY (emp_passport_number)
12        REFERENCES lab3.employer (passport_number) MATCH SIMPLE
13        ON UPDATE NO ACTION
14        ON DELETE NO ACTION
15        NOT VALID,
16    CONSTRAINT stauts_check CHECK (status::text = ANY (ARRAY['донуцен'::character varying::text, 'не донуцен'::character varying::text])) NOT VALID
17 )
18 WITH (
19     OIDS = FALSE
20 )
21 TABLESPACE pg_default;
22
23 ALTER TABLE lab3.staff
24     OWNER to daniil;
25 -- Index: fki_emp_passport_number
26
27 -- DROP INDEX lab3.fki_emp_passport_number;
28
29 CREATE INDEX fki_emp_passport_number
30     ON lab3.staff USING btree
31     (emp_passport_number ASC NULLS LAST)
32     TABLESPACE pg_default;

```

```

1 -- Table: lab3.ticket
2
3 -- DROP TABLE lab3.ticket;
4
5 CREATE TABLE IF NOT EXISTS lab3.ticket
6 (
7     id_ticket integer NOT NULL,
8     id_flight integer NOT NULL,
9     price integer NOT NULL,
10    CONSTRAINT ticket_pkey PRIMARY KEY (id_ticket),
11    CONSTRAINT id_flight FOREIGN KEY (id_flight)
12        REFERENCES lab3.flight (id_flight) MATCH SIMPLE
13        ON UPDATE NO ACTION
14        ON DELETE NO ACTION
15        NOT VALID
16 )
17 WITH (
18     OIDS = FALSE
19 )
20 TABLESPACE pg_default;
21
22 ALTER TABLE lab3.ticket
23     OWNER to daniil;
24 -- Index: fki_id_flight
25
26 -- DROP INDEX lab3.fki_id_flight;
27
28 CREATE INDEX fki_id_flight
29     ON lab3.ticket USING btree
30     (id_flight ASC NULLS LAST)
31     TABLESPACE pg_default;

```

```

1 -- Table: lab3.transit
2
3 -- DROP TABLE lab3.transit;
4
5 CREATE TABLE IF NOT EXISTS lab3.transit
6 (
7     id_transit integer NOT NULL,
8     transit_order integer NOT NULL,
9     id_airport integer NOT NULL,
10    arrival_date date NOT NULL,
11    departure_date date NOT NULL,
12    CONSTRAINT transit_pkey PRIMARY KEY (id_transit),
13    CONSTRAINT id_airport FOREIGN KEY (id_airport)
14        REFERENCES lab3.airports (id_airport) MATCH SIMPLE
15        ON UPDATE NO ACTION
16        ON DELETE NO ACTION
17        NOT VALID,
18    CONSTRAINT dep_date_check CHECK (departure_date > '1900-01-01'::date) NOT VALID,
19    CONSTRAINT ar_date_check CHECK (arrival_date > departure_date) NOT VALID
20 )
21 WITH (
22     OIDS = FALSE
23 )
24 TABLESPACE pg_default;
25
26 ALTER TABLE lab3.transit
27     OWNER to daniil;

```

6. Заполнение данными

Query Editor	Query History
1	INSERT INTO lab3.employer(first_name,last_name, job, passport_number)
2	VALUES ('Артем', 'Галкин','первый пилот', 0218321673),('Наталья', 'Леонова','стюардесса', 0243233233),('Олег', 'Ковалев','второй пилот', 0756409567);
3	
4	
5	INSERT INTO lab3.staff(status, emp_passport_number, id_staff)
6	VALUES ('допущен',0218321673, 98),('не допущен',0243233233, 97),('допущен',0756409567, 96);
7	
8	INSERT INTO lab3.plane(id_brand,id_plane,release_date)
9	VALUES (103, 1, '2016-03-21'),(102, 2, '2013-07-05');
10	
11	
12	INSERT INTO lab3.transit(arrival_date,departure_date,id_airport,id_transit,transit_order)
13	VALUES ('2021-03-21','2021-03-20', 13, 657, 0),('2021-07-03','2021-07-01', 14, 1038, 1);
14	
15	INSERT INTO lab3.flight_schedule(count_transit,flight_number,id_airport,id_transit)
16	VALUES (1, 7, 13, 657), (2, 12, 14, 1038);
17	
18	
19	INSERT INTO lab3.flight(id_flight, flight_type, arrival_date, departure_date, id_plane, id_staff, flight_number)
20	VALUES (123, 'Regular', '2021-05-16', '2021-05-15', 1, 98, 7),(124, 'Regular', '2021-06-01', '2021-05-31', 1, 97, 7),(127, 'Transfer', '2021-04-21', '2021-04-20', 2, 96, 7);
21	
22	

```

Query Editor Query History
4
5 INSERT INTO lab3.flight_schedule(count_transit,flight_number,id_airport,id_transit)
6 VALUES (1, 7, 13, 657), (2, 12, 14, 1038);
7
8
9 INSERT INTO lab3.flight(id_flight, flight_type, arrival_date, departure_date, id_plane, id_staff, flight_number)
10 VALUES (123, 'Regular', '2021-05-16', '2021-05-15', 1, 98, 7),(124, 'Regular', '2021-06-01', '2021-05-31', 1, 97, 7),(127, 'Transfer', '2021-04-21', '2021-04-20', 2, 9
11
12
13 INSERT INTO lab3.ticket(id_flight,id_ticket,price)
14 VALUES (123, 456, 2397),(124, 749, 5789);
15
16
17 INSERT INTO lab3.purchase(id_cashbox,id_purchase,id_ticket,passport_number,seat_number)
18 VALUES (13, 798, 456, 0308075673, 56);
19
20
21 INSERT INTO lab3.purchase(id_cashbox,id_purchase,id_ticket,passport_number,seat_number)
22 VALUES (14,581, 749, 0308043267, 13);
23
24

```

ВЫВОДЫ

Программа pgAdmin позволяет создавать базы данных на высоком уровне: либо напрямую посредством взаимодействия с ее GUI, или же через работу на встроенном генераторе ER диаграмм. Однако, каким способом диаграмма не была создана, всегда сохраняется возможность увидеть какими SQL командами была создана диаграмма. Видеть код, создаваемый нажатием кнопки в интерфейсе, оказалось полезным для нахождения ошибок и общего понимания работы программы.