



**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря
Сікорського»
ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ**

**Кафедра системного програмування та спеціалізованих
комп'ютерних систем**

**Лабораторна робота № 1
з дисципліни «Бази даних і засоби управління»**

«Ознайомлення з базовими операціями СУБД PostgreSQL»

Виконав:
студент групи КВ-82
Ященко Іван Васильович

Перевірив:

Метою роботи є здобуття практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

У звіті щодо пункту №1 завдання має бути:

- перелік сутностей з описом їх призначення;
- графічний файл розробленої моделі «сутність-зв'язок»;
- назва нотації.

У звіті щодо пункту №2 завдання має бути:

- опис процесу перетворення (наприклад, “сутність А було перетворено у таблицю А, а зв'язок R (M:N) зумовив появу додаткової таблиці R1 тощо);
- схему бази даних у графічному вигляді з **назвами таблиць (!) та зв'язками між ними.**

У звіті щодо пункту №3 завдання має бути:

- пояснення щодо відповідності схеми бази даних нормальним формам НФ1, НФ2 та НФ3. У випадку невідповідності надати опис необхідних змін у схемі;
- У випадку проведення змін у схемі бази даних надати оновлену версію схеми, інакше - не наводити схему.

У звіті щодо пункту №4 завдання має бути:

- навести копії екрану з pgAdmin4, що відображають назви та типи стовпців (доступне у закладці “Columns” властивостей “Properties” таблиць дерева об'єктів у pgAdmin4);
- навести копії екрану з pgAdmin4, що відображають вміст таблиць бази даних у PostgreSQL. Таблиці на зображенні обов'язково **повинні мати назву!**

Опис предметної галузі

(Сервіс продажів квитків в кіно). При проектуванні даної галузі можна виділити такі сутності.

Зал (Room) – призначений для ідентифікації кінозали де будуть проходити сеанси кінофільмів і також розподіл кімнат на кольори для їх відмінності(зроблено на прикладі кінозалів Butterfly у м. Київ), **сеанс(show_time)** створений для ідентифікації сеансу кінофільму де також зберігається ідентифікатори фільму та дата проведення сеансу.

фільм (Film) – створений для ідентифікації фільму, його імені та час його проходження.

місце/ряд (Seat/Row) – створений для ідентифікації місця для сидіння, включає в себе ідентифікатори місця і ряду,

клієнт (Client) - створений для ідентифікації клієнта, його імені та віку.

номер телефону клієнту (Telephone) – створений для ідентифікації номера телефону клієнту , що включає в себе номер телефону , ідентифікатор клієнту та сам номер телефону.

квиток(Ticket) - створений для ідентифікації квитка на кіно, що

включає в себе ідентифікатор квитка, ідентифікатор місця для сидіння, ідентифікатор сеансу, та ідентифікатор клієнту.

В багатьох кіно-залах може відбуватися декілька сеансів за один день (M:N);

В кожному сеансі може відтворюватись лише один фільм (1:1);

В кожному залі є багато місць(1:N);

В кожному квитку може бути одне місце, сеанс та зал (1:1);

У одного клієнта може бути багато квитків (1:N) або 0 квитків;

У одного клієнта може бути декілька телефонів (1:N);

Концептуальна модель учбової предметної області " Сервіс продажу квитків кіно "

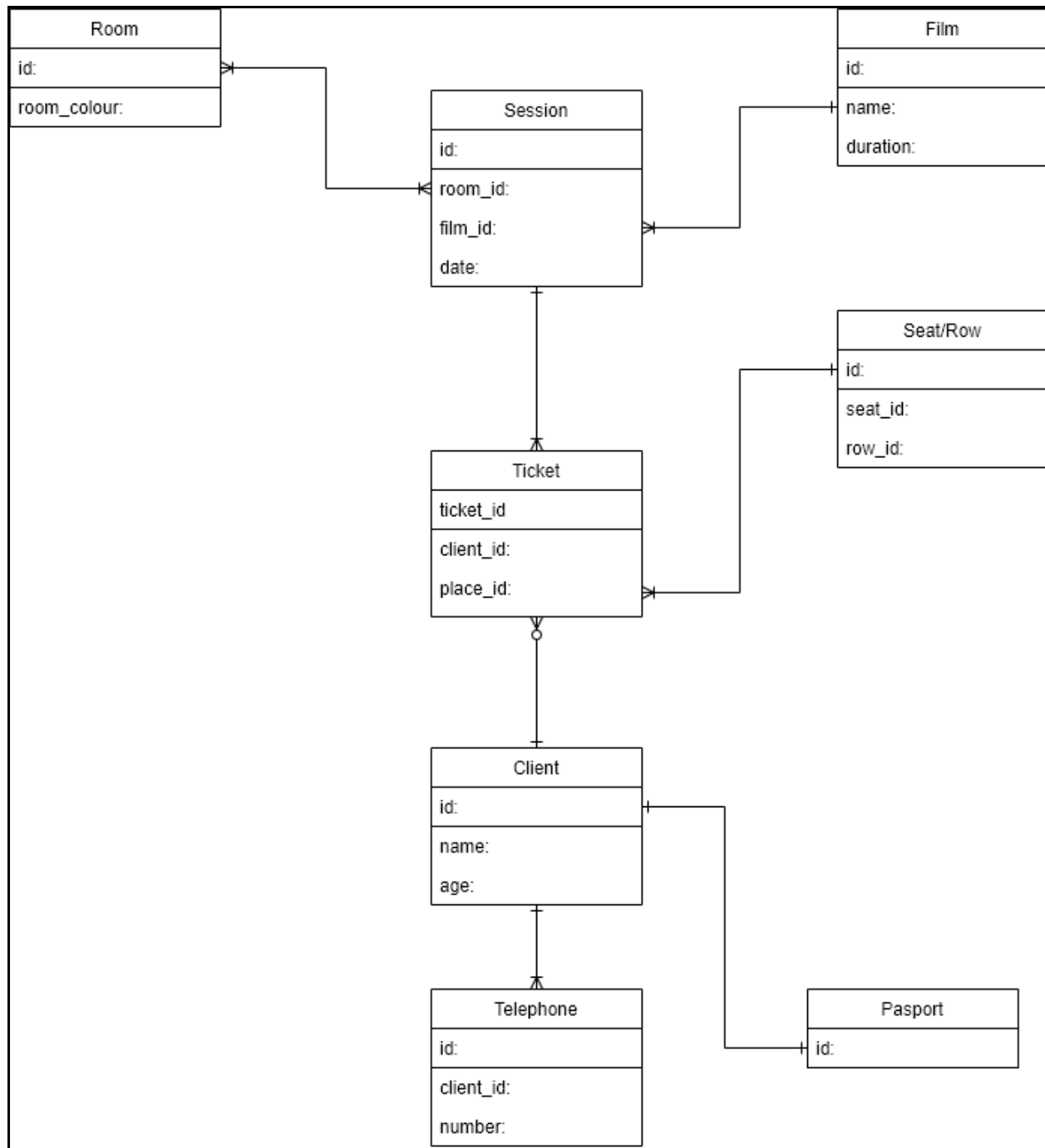


Рисунок 1 - Концептуальна модель предметної області " Сервіс продажу квитків кіно "

Нотація: Пташина лапка + засоби програми draw.io

Пояснення щодо відповідності схеми бази даних нормальним формам:

Схема бази даних відповідає 1НФ тому, що схема передбачає 1 елемент в кожній комірці.

Схема бази даних відповідає 2НФ тому, що по-перше відповідає 1НФ, а по-друге

Схема не включає в собі залежності від декількох потенційних ключів, а залежить лише від одного.

Схема бази даних відповідає 3НФ тому, що по-перше вона відповідає 2НФ, а по-друге відсутні транзитивні функціональні залежності неключових атрибутів від ключових.

Логічна модель (Структура) БД “ Сервіс продажу квитків кіно ” (засобами SqlDMB)

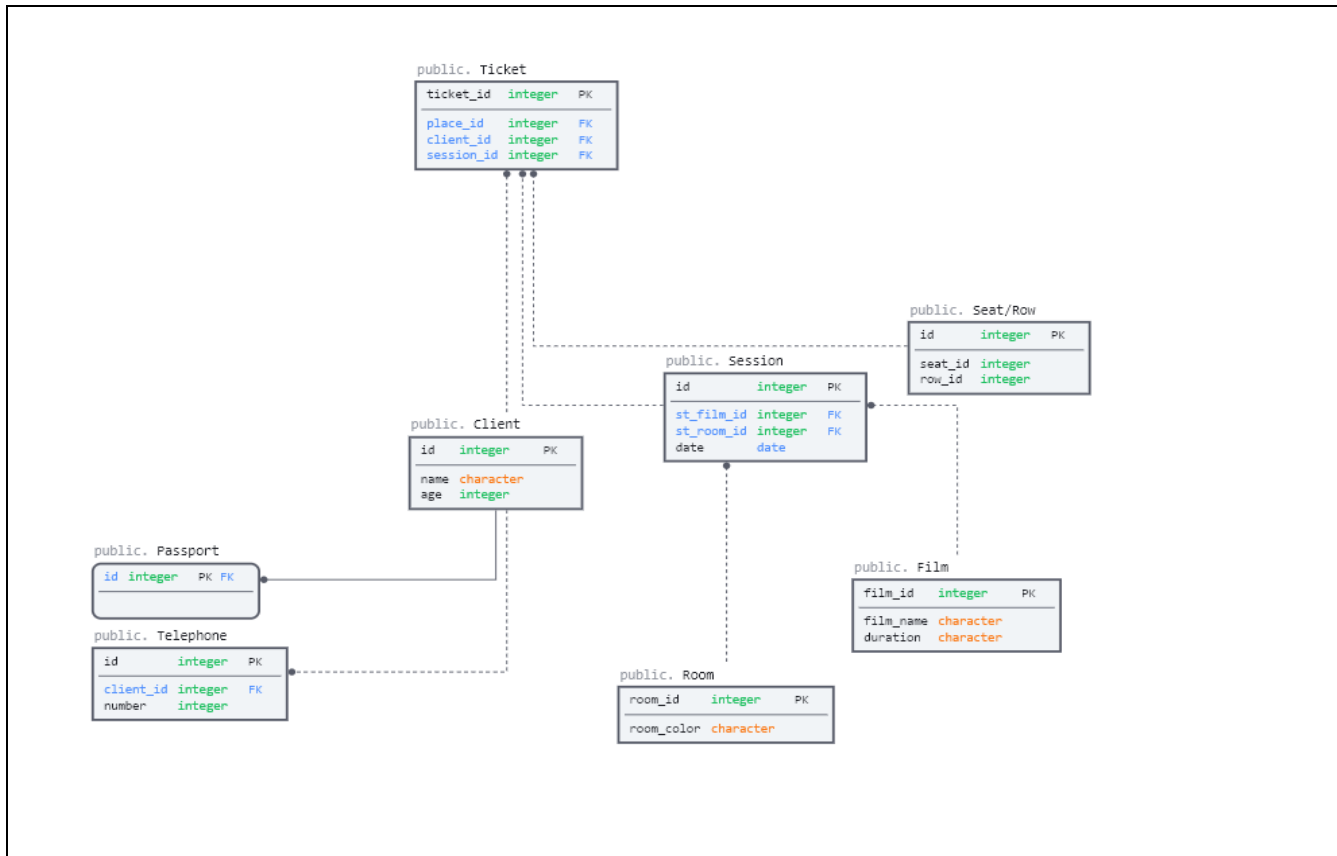


Рисунок 2 - Логічна модель (Структура) БД “ Сервіс продажу квитків кіно ”
(засобами SqlDMB)

Опис структури БД “Сервіс продажу квитків кіно”

СУТНІСТЬ	АТРИБУТ	ТИП(Розмір)
Сутність “ Room ” Вміщує інформацію про залу кінотеатру	id – унікальний ID категорії в БД room_colour – колір зали в кінотеатрі для розпізнавання(зроблено на прикладі кінозалів Butterfly у м.Київ)	Числовий Текстовий(30)
Сутність “ Session ” Вміщує інформацію про сеанс у кінотеатрі	id - унікальний ID сеансу date – дата проходження заходу room_id – ID зали, де буде проходити сеанс film_id – ID фільму, що буде проходити під час сеансу.	Числовий Дата Числовий Числовий
Сутність “ Film ” Вміщує інформацію про фільм	id – унікальний ID фільму name – ім’я фільму duration – протяжність фільму	Числовий Текстовий(30) Текстовий(30)
Сутність “ Seat/Row ” Вміщує інформацію про місце сидіння	id – унікальний ID місця seat_id – ID місця для сидіння row_id – ID ряду для сидіння	Числовий Числовий Числовий
Сутність “ Ticket ” Вміщує інформацію щодо білету у кіно	ticket_id – унікальний ID білету client_id – ID клієнта place_id - ID місця	Числовий Числовий Числовий
Сутність “ Client ” Вміщує у собі інформацію про покупця	id - унікальний ID клієнта name – ім’я клієнта age – вік клієнта	Числовий Текстовий(30) Числовий
Сутність “ Phone ” Вміщує інформацію щодо телефонів покупців	id – унікальний ID мобільного номера client_id - ID клієнта чий телефон number – номер телефону	Числовий Числовий Числовий
Сутність “ Passport ” Вміщує інформацію паспортних даних користувача	id унікальний ID паспортних даних користувача	Числовий

Додаток Б1. Структура БД " Сервіс продажу квитків кіно "

Telephone

<u>id</u>	client_id	number
0	1	985621213
1	1	983622154
2	0	981625214

Client

<u>id</u>	name	age
0	Ivan	54
1	Victor Barinov	100

Ticket

session_id	place_id	client_id
0	11	0
0	12	0
0	13	1
1	11	0
1	12	1
1	15	0

Passport

<u>id</u>
0
1

Session

<u>id</u>	room_id	film_id	data
0	0	1	2020-09-09
1	1	0	2020-09-15

Seat/Row

<u>id</u>	seat_id	row_id
11	1	1
12	2	1
13	3	1
14	4	1
15	5	1

Film

<u>film_id</u>	film_name	duration
0	Mother	1г30хв
1	DreamHack	2г0хв

Room

<u>room_id</u>	room_colour
0	Red
1	Green
2	Blue
3	Brown

Рисунок 3 - Структура БД " Сервіс продажу квитків кіно "

Структура БД “Сервіс продажу квитків” в pgAdmin IV

Client		Data Output	Explain	Messages	Notifications
Columns (3)		id [PK] integer	name character varying[] (30)	age integer	
id		1	0 {Ivan}	54	
name		2	1 {'Victor Barinov'}	100	
age					

```
1 -- Table: public.Client
2
3 -- DROP TABLE public."Client";
4
5 CREATE TABLE public."Client"
6 (
7     id integer NOT NULL,
8     name character varying(30)[] COLLATE pg_catalog."default" NOT NULL,
9     age integer NOT NULL,
10    CONSTRAINT "Client_pkey" PRIMARY KEY (id)
11 )
12
13 TABLESPACE pg_default;
14
15 ALTER TABLE public."Client"
16     OWNER to postgres;
```

Film			
Columns (3)			
	film_name		
	film_id		
	duration		
Constraints			

	film_name character varying[] (30)	film_id [PK] integer	duration character varying[] (30)
1	{Mother}		0 ('1 година : 30 хвилини')
2	{DreamHack}		1 ('2 години : 0 хвилини')

```

1 -- Table: public.Film
2
3 -- DROP TABLE public."Film";
4
5 CREATE TABLE public."Film"
6 (
7     film_name character varying(30)[] COLLATE pg_catalog."default" NOT NULL,
8     film_id integer NOT NULL DEFAULT nextval('"Film_film_id_seq"'::regclass),
9     duration character varying(30)[] COLLATE pg_catalog."default" NOT NULL,
10    CONSTRAINT "Film_pkey" PRIMARY KEY (film_id)
11 )
12
13 TABLESPACE pg_default;
14
15 ALTER TABLE public."Film"
16     OWNER to postgres;

```

▼ Tables (7)

► Client

► Film

▼ Room

▼ Columns (2)

room_id

room_color

► Constraints

► Indexes

Data Output Explain Messages Notifications

	room_id [PK] integer	room_color character varying[] (30)
1	0	{Red}
2	1	{Green}
3	2	{Blue}
4	3	{Brown}

```
1 -- Table: public.Room
2
3 -- DROP TABLE public."Room";
4
5 CREATE TABLE public."Room"
6 (
7     room_id integer NOT NULL DEFAULT nextval('"Room_room_id_seq"'::regclass),
8     room_color character varying(30)[] COLLATE pg_catalog."default" NOT NULL,
9     CONSTRAINT "Room_pkey" PRIMARY KEY (room_id)
10 )
11
12 TABLESPACE pg_default;
13
14 ALTER TABLE public."Room"
15     OWNER to postgres;
```

- ▼ Tables (7)
 - Client
 - Film
 - Room
 - ▼ Seat/Row
 - ▼ Columns (3)
 - id
 - seat_id
 - row_id
 - Constraints (1)

Data Output Explain Messages Notifications

	id [PK] integer	seat_id integer	row_id integer	
1		11	1	1
2		12	2	1
3		13	3	1
4		14	4	1
5		15	5	1

```

1 -- Table: public.Seat/Row
2
3 -- DROP TABLE public."Seat/Row";
4
5 CREATE TABLE public."Seat/Row"
6 (
7     id integer NOT NULL,
8     seat_id integer NOT NULL,
9     row_id integer NOT NULL,
10    CONSTRAINT "Seat/Row_pkey" PRIMARY KEY (id)
11 )
12
13 TABLESPACE pg_default;
14
15 ALTER TABLE public."Seat/Row"
16     OWNER to postgres;

```

▼ Tables (7)

- Client
- Film
- Room
- Seat/Row
- Session
 - Columns (4)
 - id
 - st_film_id
 - st_room_id
 - date

	Data Output	Explain	Messages	Notifications
	id [PK] integer	st_film_id integer	st_room_id integer	date date
1		0	1	0 2020-09-...
2		1	0	1 2020-09-...

```

1 -- Table: public.Session
2
3 -- DROP TABLE public."Session";
4
5 CREATE TABLE public."Session"
6 (
7     id integer NOT NULL DEFAULT nextval('"ShowTime_id_seq"'::regclass),
8     st_film_id integer NOT NULL DEFAULT nextval('"ShowTime_film_id_seq"'::regclass),
9     st_room_id integer NOT NULL DEFAULT nextval('"ShowTime_room_id_seq"'::regclass),
10    date date NOT NULL,
11    CONSTRAINT "ShowTime_pkey" PRIMARY KEY (id),
12    CONSTRAINT st_film_id FOREIGN KEY (st_film_id)
13        REFERENCES public."Film" (film_id) MATCH SIMPLE
14        ON UPDATE NO ACTION
15        ON DELETE NO ACTION
16        NOT VALID,
17    CONSTRAINT st_room_id FOREIGN KEY (st_room_id)
18        REFERENCES public."Room" (room_id) MATCH SIMPLE
19        ON UPDATE NO ACTION
20        ON DELETE NO ACTION
21        NOT VALID
22 )
23
24 TABLESPACE pg_default;
25
26 ALTER TABLE public."Session"
27     OWNER to postgres;
28 -- Index: fki_st_film_id
29
30 -- DROP INDEX public.fki_st_film_id;
31
32 CREATE INDEX fki_st_film_id
33     ON public."Session" USING btree
34     (st_film_id ASC NULLS LAST)
35     TABLESPACE pg_default;
36 -- Index: fki_st_room_id
37
38 -- DROP INDEX public.fki_st_room_id;
39
40 CREATE INDEX fki_st_room_id
41     ON public."Session" USING btree
42     (st_room_id ASC NULLS LAST)
43     TABLESPACE pg_default;

```

▼ Tables (/)

- > Client
- > Film
- > Room
- > Seat/Row
- > Session
- > Telephone

▼ Ticket

▼ Columns (4)

- id
- session_id
- place_id
- client_id

Data Output Explain Messages Notifications

	id [PK] integer	session_id integer	place_id integer	client_id integer	
1		0	0	11	0
2		1	0	12	0
3		2	0	13	1
4		3	1	11	0
5		4	1	12	1
6		5	1	15	0

```

1 -- Table: public.Ticket
2
3 -- DROP TABLE public."Ticket";
4
5 CREATE TABLE public."Ticket"
6 (
7     place_id integer NOT NULL,
8     session_id integer NOT NULL,
9     client_id integer NOT NULL,
10    CONSTRAINT "Ticket_pkey" PRIMARY KEY (place_id),
11    CONSTRAINT client_id FOREIGN KEY (client_id)
12        REFERENCES public."Client" (id) MATCH SIMPLE
13        ON UPDATE NO ACTION
14        ON DELETE NO ACTION
15        NOT VALID,
16    CONSTRAINT place_id FOREIGN KEY (place_id)
17        REFERENCES public."Seat/Row" (id) MATCH SIMPLE
18        ON UPDATE NO ACTION
19        ON DELETE NO ACTION
20        NOT VALID,
21    CONSTRAINT session_id FOREIGN KEY (session_id)
22        REFERENCES public."Session" (id) MATCH SIMPLE
23        ON UPDATE NO ACTION
24        ON DELETE NO ACTION
25        NOT VALID
26 )
27
28 TABLESPACE pg_default;
29
30 ALTER TABLE public."Ticket"
31     OWNER to postgres;
32 -- Index: fki_session_id
33
34 -- DROP INDEX public.fki_session_id;
35
36 CREATE INDEX fki_session_id
37     ON public."Ticket" USING btree
38     (session_id ASC NULLS LAST)
39     TABLESPACE pg_default;

```

▼	Tables (7)
▶	Client
▶	Film
▶	Room
▶	Seat/Row
▶	Session
▼	Telephone
▼	Columns (3)
	id
	client_id
	number

	Data Output	Explain	Messages	Notifications
	id [PK] integer	client_id integer	number integer	
1		0	1	931645372
2		1	1	931993038
3		2	0	977298400

```

1 -- Table: public.Telephone
2
3 -- DROP TABLE public."Telephone";
4
5 CREATE TABLE public."Telephone"
6 (
7     id integer NOT NULL,
8     client_id integer NOT NULL,
9     "number" integer NOT NULL,
10    CONSTRAINT "Telephone_pkey" PRIMARY KEY (id),
11    CONSTRAINT client_id FOREIGN KEY (client_id)
12        REFERENCES public."Client" (id) MATCH FULL
13        ON UPDATE NO ACTION
14        ON DELETE NO ACTION
15        NOT VALID
16 )
17
18 TABLESPACE pg_default;
19
20 ALTER TABLE public."Telephone"
21     OWNER to postgres;
22 -- Index: fki_client_id
23
24 -- DROP INDEX public.fki_client_id;
25
26 CREATE INDEX fki_client_id
27     ON public."Telephone" USING btree
28     (client_id ASC NULLS LAST)
29     TABLESPACE pg_default;

```

Passport

Columns (1)

id

Constraints (2)

Passport_pkey

id

Indexes

RLS Policies

Rules

Triggers

Room

Seat/Row

Data Output

Explain

Messages

Notifications

	id	
1	[PK] integer	0
2		1

```

1 -- Table: public.Passport
2
3 -- DROP TABLE public."Passport";
4
5 CREATE TABLE public."Passport"
6 (
7     id integer NOT NULL,
8     CONSTRAINT "Passport_pkey" PRIMARY KEY (id),
9     CONSTRAINT id FOREIGN KEY (id)
10         REFERENCES public."Client" (id) MATCH SIMPLE
11         ON UPDATE NO ACTION
12         ON DELETE NO ACTION
13         NOT VALID
14 )
15
16 TABLESPACE pg_default;
17
18 ALTER TABLE public."Passport"
19     OWNER to postgres;

```