



**Національний технічний університет України
“Київський політехнічний інститут”**

**Факультет прикладної математики
Кафедра системного програмування і спеціалізованих
комп’ютерних систем**

**Лабораторна робота №1
з дисципліни “Бази даних і засоби управління”**

*Тема: “Проектування бази даних та ознайомлення з базовими
операціями СУБД PostgreSQL”*

Виконав:
студент III курсу
групи KB-01
Таранич Артем
Перевірив: Павловський В. І.

Метою роботи є здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

Завдання роботи полягає у наступному:

1. Розробити модель «сутність-зв'язок» предметної галузі, обраної студентом самостійно, відповідно до пункту «Вимоги до ER-моделі».
2. Перетворити розроблену модель у схему бази даних (таблиці) PostgreSQL.
3. Виконати нормалізацію схеми бази даних до третьої нормальної форми (3НФ).
4. Ознайомитись із інструментарієм PostgreSQL та pgAdmin 4 та занести декілька рядків даних у кожен з таблиць засобами pgAdmin 4.

Вимоги до ER-моделі:

1. Сутності моделі предметної галузі мають містити зв'язки типу 1:N або N:M.
2. Кількість сутностей у моделі – 3-4. Кількість атрибутів у кожній сутності: від двох до п'яти.
3. Передбачити наявність зв'язку з атрибутом.

Для побудови ER-діаграм використовувати одну із нотацій: Чена, “Пташиної лапки (Crow's foot)”, UML.

Частина № 1

Опис предметної галузі

Для виконання лабораторної роботи я обрав тему – сайт для продажу авіаквитків.

Для розробки для даного сайту я видів наступні сутності: відомості про літак (Aircraft), відомості про аеропорт (Airport), відомості про білет (Ticket), відомості про акаунт користувача (User Account).

Атрибути заданих сутностей:

1. Aircraft: pilot, name, owner
2. Airport: name, location.
3. Ticket: price, class, date, flight time
4. User Account: name, mailbox, bank card

Опис зв'язків між сутностями предметної області

На літак може бути придбано якась кількість білетів, тому зв'язок між сутностями «Aircraft» і «Ticket» - 1:N.

На один білет може претендувати тільки один користувач й одночасно з цим користувач може придбати кілька білетів, тому зв'язок між сутностями «User Account» і «Ticket» - 1:N.

Сутність «Aircraft» має зв'язок M:N по відношенню до сутності «Airport», тому що один літак має потрапити в декілька аеропортів (з якого вилітає, куди має прилетіти), в той же на одному аеропорту можуть знаходитися декілька кілька літаків.

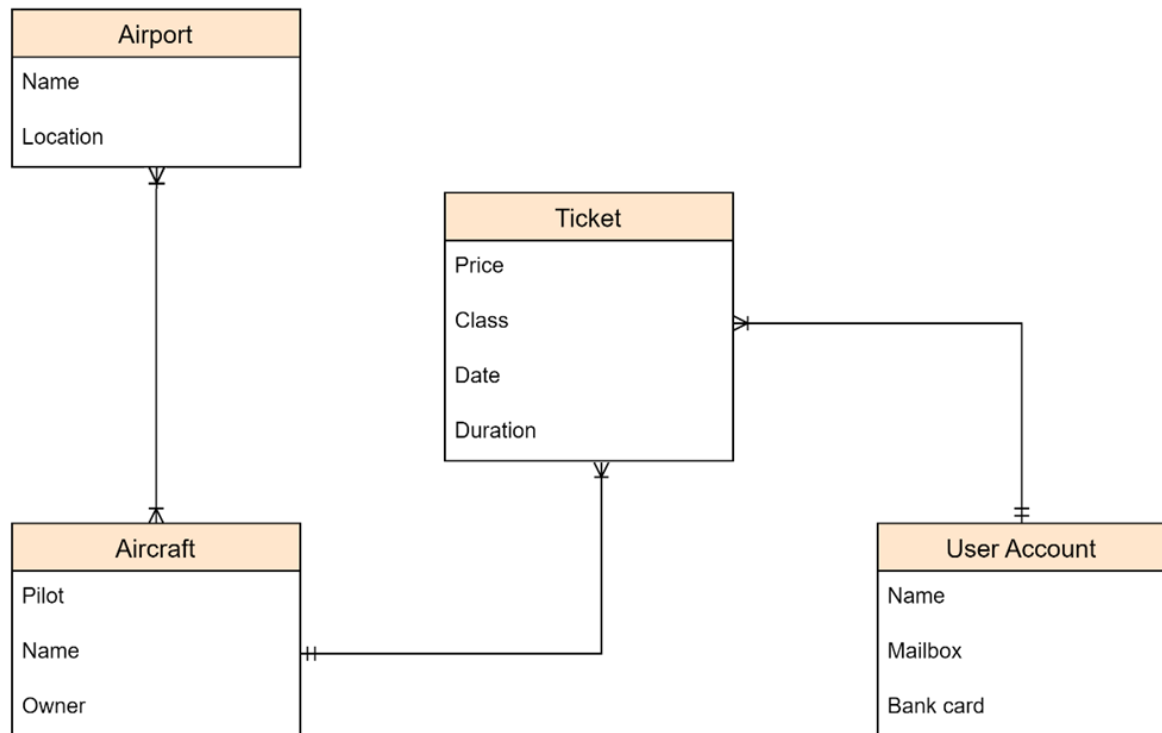


Рисунок 1 - ER-діаграма побудована за нотацією “Пташиної лапки”, побудована у додатку *draw.io*

Частина № 2

Перетворення концептуальної моделі у схему баз даних

Для кожної сутності створюється таблиця. Причому кожному атрибуту сутності відповідає стовпець таблиці. В даній моделі перетворення в схему баз даних відбувалося за такими правилами:

Якщо зв'язок типу 1: N і клас приналежності сутності на стороні N є обов'язковим, то необхідно побудувати таблицю для кожної сутності. Первинний ключ сутності повинен бути первинним ключем відповідної таблиці. Первинний ключ сутності на стороні 1 додається як атрибут в таблицю для сутності на стороні N.

Якщо зв'язок типу N: M, то необхідно побудувати три таблиці - по одній для кожної сутності і одну для зв'язку. Первинний ключ сутності повинен бути первинним ключем відповідної таблиці. Таблиця для зв'язку серед своїх атрибутів повинна мати ключі обох сутностей.

Сутність «User Account» перетворено у таблицю «user_account».

Сутність «Ticket» перетворено у таблицю «ticket».

Зв'язок 1:N між сутностями «User Account» та «Ticket» та зв'язок 1:N між сутностями «Aircraft» та «Ticket» зумовив появу 2 зовнішніх ключів: aircraft_id та user_acc_id у таблицю «ticket».

Сутність «Aircraft» перетворено у таблицю «aircraft».

Сутність «Airport» перетворено у таблицю «airport».

Зв'язок M:N між сутністю «Aircraft» та «Airport» зумовив появу додаткової таблицю «airport/aircraft» із зовнішніми ключами aircraft_id та airport_id.

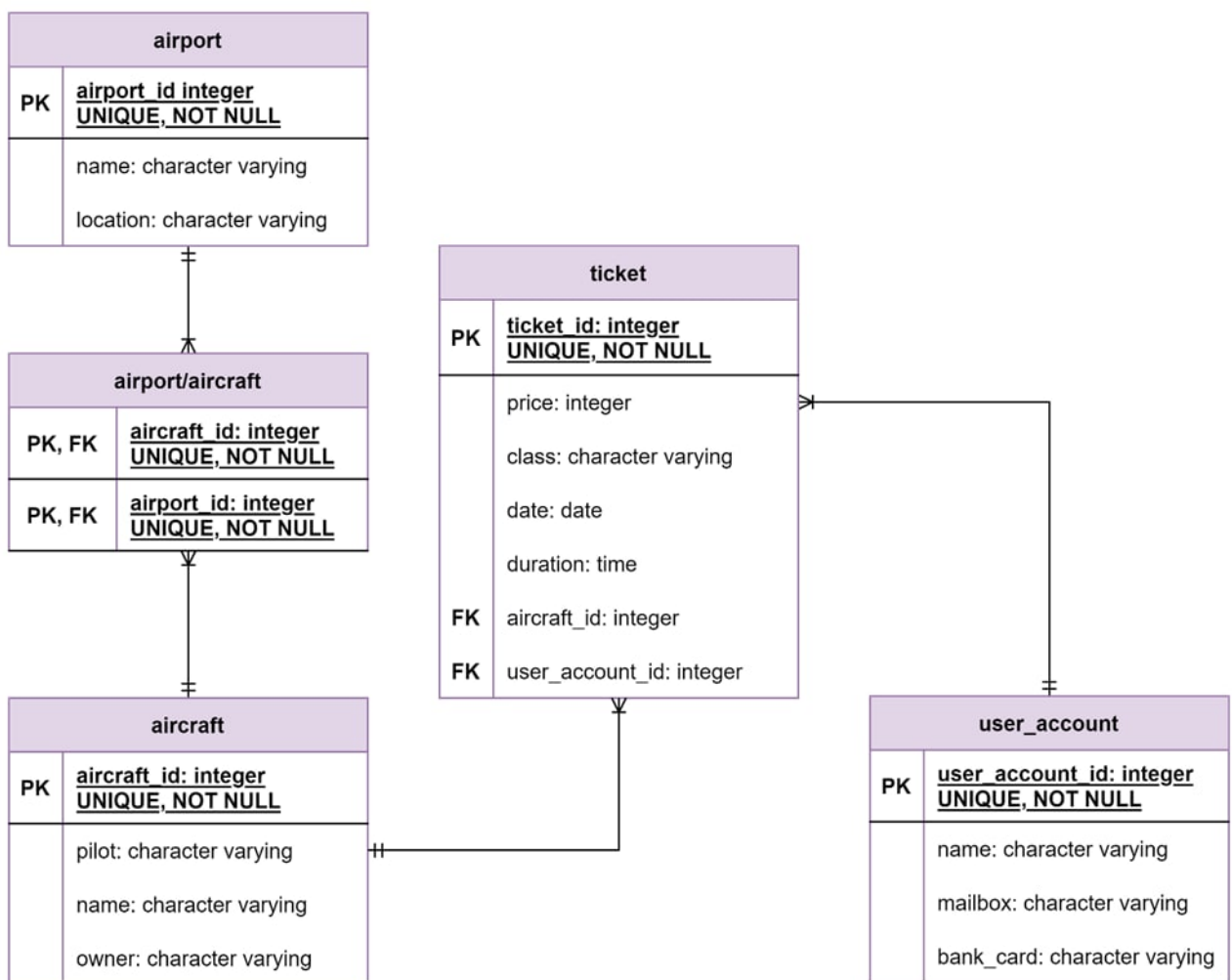


Рисунок 2 - Схема бази даних, побудована у додатку draw.io

Частина № 3

Відповідність схеми бази даних до третьої нормальної форми

Схема відповідає 1НФ, тому що:

- Значення в кожній комірці є атомарним;
- В таблиці немає дубльованих рядків.
- В кожному стовпці зберігаються дані одного типу.

Схема відповідає 2НФ, тому що:

- Вона відповідає 1НФ.
- Кожен неключовий атрибут функціонально залежить від всього ключа, а не від його частини.

Схема відповідає 3НФ, тому що:

- Вона відповідає 2НФ.
- Кожен не простий атрибут, є функціонально залежним від головних ключів.

Частина № 4

Фізична модель БД «Сайт для продажу авіаквитків» у pgAdmin4

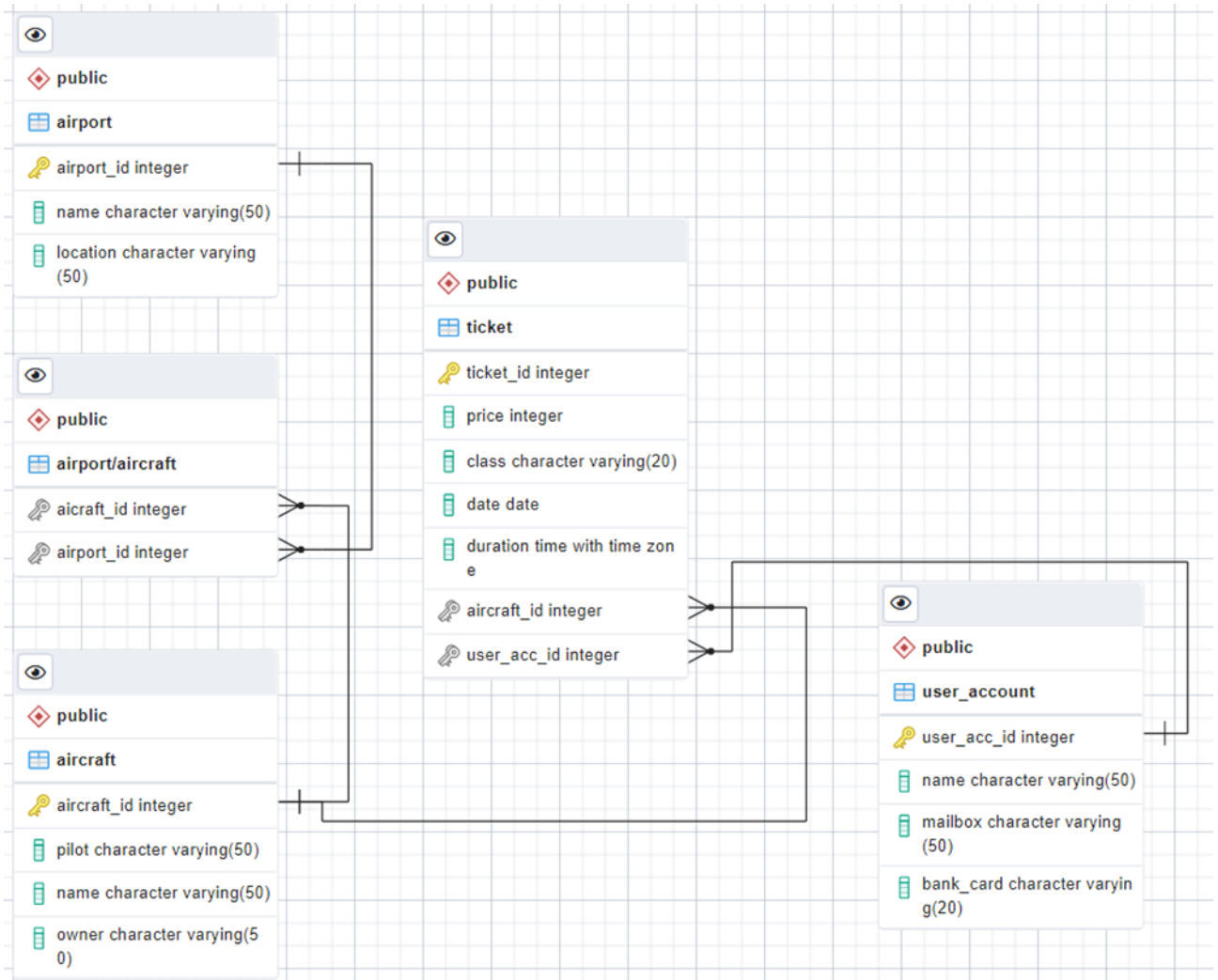


Рисунок 3 - Схема бази даних, побудована у pgAdmin 4

Airport

airport



General Columns Advanced Constraints Parameters Security SQL

Inherited from table(s)

Select to inherit from...



Columns



	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
	airport_id	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
	name	character varying	50		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	location	character varying	50		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

airport



General Columns Advanced Constraints Parameters Security SQL

Primary Key Foreign Key Check Unique Exclude

Name		Columns
	airport_pkey	airport_id

Query Query History



```
1 SELECT * FROM public.airport
2 ORDER BY airport_id ASC
```

Data output Messages Notifications



	airport_id [PK] integer	name character varying (50)	location character varying (50)
1	1	Vaclav Havel Airport Prague	Ruzyne, Prague
2	2	London City Airport	Newham, London
3	3	Boryspil International Airport	Boryspil, Ukraine
4	4	Dulles International Airport	Dulles, United States

airport/aircraft

airport/aircraft



General Columns Advanced Constraints Parameters Security SQL

Inherited from table(s)

Columns								+
		Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
		aircraft_id	integer			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="text"/>
		airport_id	integer			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="text"/>

airport/aircraft



General Columns Advanced Constraints Parameters Security SQL

Primary Key Foreign Key Check Unique Exclude

				+
		Name	Columns	Referenced Table
		aircraft_id	(aircraft_id) -> (aircraft_id)	public.aircraft
		airport_id	(airport_id) -> (airport_id)	public.airport

Query Query History



```
1 SELECT * FROM public."airport/aircraft"
2
```

Data output Messages Notifications



	aircraft_id integer	airport_id integer
1	1	2
2	1	3
3	2	4
4	2	1

Aircraft

aircraft

General

Columns

Advanced

Constraints

Parameters









Security

SQL

Inherited from table(s)

Select to inherit from...

Columns

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
 	aircraft_id	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
 	pilot	character varying	50		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 	name	character varying	50		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 	owner	character varying	50		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

aircraft

General

Columns

Advanced

Constraints

Parameters

Security

SQL

Primary Key

Foreign Key

Check

Unique

Exclude

Name	Columns
aircraft_pkey	aircraft_id

Query

Query History

1

SELECT * FROM public.aircraft









2

ORDER BY aircraft_id ASC

Data output

Messages

Notifications



	aircraft_id [PK] integer	pilot character varying (50)	name character varying (50)	owner character varying (50)
1	1	Donovan Russell	Boeing 707	Airbus
2	2	Hashim Parkes	Boeing 727	Lockheed Martin
3	3	Korben Shepherd	Boeing 777	Airbus
4	4	Eshan Thatcher	Boeing 747	Boeing

User account

user_account

General

Columns

Advanced

Constraints

Parameters









Security

SQL

Inherited from table(s)

Select to inherit from...

Columns

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
 	user_acc_id	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
 	name	character varying	50		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 	mailbox	character varying	50		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 	bank_card	character varying	20		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

user_account

General

Columns

Advanced

Constraints

Parameters

Security

SQL

Primary Key

Foreign Key

Check

Unique

Exclude

Name	Columns
user_account_pkey	user_acc_id

Query

Query History









1 SELECT * FROM public.user_account

2 ORDER BY user_acc_id ASC

Data output

Messages

Notifications



	user_acc_id [PK] integer	name character varying (50)	mailbox character varying (50)	bank_card character varying (16)
1	3	Anabella Li	openwhl@omdiaco.com	4441 1333 4324 3453
2	2	Latoya Cantu	alekskremnev@google.com	4441 2394 3905 3892
3	1	Aishah Matthams	melrocks1@skillion.org	4441 8943 8492 3812

Ticket

ticket

General

Columns

Advanced

Constraints

Parameters

Security

SQL

Inherited from table(s)

Select to inherit from...

Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
<div>ticket_id</div>	<div>integer</div>			<div></div>	<div></div>	<div></div>
<div>price</div>	<div>integer</div>			<div></div>	<div></div>	<div></div>
<div>class</div>	<div>character varying</div>	<div>20</div>	<div></div>	<div></div>	<div></div>	<div></div>
<div>date</div>	<div>date</div>			<div></div>	<div></div>	<div></div>
<div>duration</div>	<div>time with time zone</div>		<div></div>	<div></div>	<div></div>	<div></div>
<div>aircraft_id</div>	<div>integer</div>			<div></div>	<div></div>	<div></div>
<div>user_acc_id</div>	<div>integer</div>			<div></div>	<div></div>	<div></div>

ticket

General

Columns

Advanced

Constraints

Parameters

Security

SQL

Primary Key

Foreign Key

Check

Unique

Exclude

Name	Columns
<div>ticket_pkey</div>	<div>ticket_id</div>

ticket

General

Columns

Advanced

Constraints

Parameters

Security

SQL

Primary Key

Foreign Key

Check

Unique

Exclude

Name	Columns	Referenced Table
<div>aircraft_id</div>	<div>(aircraft_id) -> (aircraft_id)</div>	<div>public.aircraft</div>
<div>user_acc_id</div>	<div>(user_acc_id) -> (user_acc_id)</div>	<div>public.user_account</div>

Query

Query History

1

2

SELECT * FROM public.ticket

ORDER BY ticket_id ASC

Data output

Messages

Notifications

	ticket_id [PK] integer	price integer	class character varying (20)	date date	duration time with time zone	aircraft_id integer	user_acc_id integer
1	5	500	first	2022-10-15	06:00:00+03:00	1	2
2	4	400	business	2022-12-23	04:20:00+03:00	2	2
3	3	230	standart	2022-10-06	03:00:00+03:00	2	1
4	2	100	economy	2023-11-05	03:30:00+03:00	3	1
5	1	150	Premium economy	2022-10-05	02:00:00+03:00	4	3

SQL TEXT

```
-- Table: public.airport

-- DROP TABLE IF EXISTS public.airport;

CREATE TABLE IF NOT EXISTS public.airport
(
    airport_id integer NOT NULL,
    name character varying(50) COLLATE pg_catalog."default" NOT
NULL,
    location character varying(50) COLLATE pg_catalog."default"
NOT NULL,
    CONSTRAINT airport_pkey PRIMARY KEY (airport_id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.airport
    OWNER to postgres;

-- Table: public.aircraft

-- DROP TABLE IF EXISTS public.aircraft;

CREATE TABLE IF NOT EXISTS public.aircraft
(
    aircraft_id integer NOT NULL,
    pilot character varying(50) COLLATE pg_catalog."default" NOT
NULL,
    name character varying(50) COLLATE pg_catalog."default" NOT
NULL,
    owner character varying(50) COLLATE pg_catalog."default" NOT
NULL,
    CONSTRAINT aircraft_pkey PRIMARY KEY (aircraft_id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.aircraft
    OWNER to postgres;

-- Table: public.airport

-- DROP TABLE IF EXISTS public.airport;

CREATE TABLE IF NOT EXISTS public.airport
(
    airport_id integer NOT NULL,
    name character varying(50) COLLATE pg_catalog."default" NOT
NULL,
    location character varying(50) COLLATE pg_catalog."default"
NOT NULL,
    CONSTRAINT airport_pkey PRIMARY KEY (airport_id)
)
```

```

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.airport
    OWNER to postgres;

-- Table: public.airport/aircraft

-- DROP TABLE IF EXISTS public."airport/aircraft";

CREATE TABLE IF NOT EXISTS public."airport/aircraft"
(
    aircraft_id integer NOT NULL,
    airport_id integer NOT NULL,
    CONSTRAINT aircraft_id FOREIGN KEY (aircraft_id)
        REFERENCES public.aircraft (aircraft_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT airport_id FOREIGN KEY (airport_id)
        REFERENCES public.airport (airport_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."airport/aircraft"
    OWNER to postgres;

-- Table: public.ticket

-- DROP TABLE IF EXISTS public.ticket;

CREATE TABLE IF NOT EXISTS public.ticket
(
    ticket_id integer NOT NULL,
    price integer NOT NULL,
    class character varying(20) COLLATE pg_catalog."default" NOT
NULL,
    date date NOT NULL,
    duration time with time zone NOT NULL,
    aircraft_id integer NOT NULL,
    user_acc_id integer NOT NULL,
    CONSTRAINT ticket_pkey PRIMARY KEY (ticket_id),
    CONSTRAINT aircraft_id FOREIGN KEY (aircraft_id)
        REFERENCES public.aircraft (aircraft_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT user_acc_id FOREIGN KEY (user_acc_id)
        REFERENCES public.user_account (user_acc_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)

```

```
TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.ticket
    OWNER to postgres;

-- Table: public.user_account

-- DROP TABLE IF EXISTS public.user_account;

CREATE TABLE IF NOT EXISTS public.user_account
(
    user_acc_id integer NOT NULL,
    name character varying(50) COLLATE pg_catalog."default" NOT
NULL,
    mailbox character varying(50) COLLATE pg_catalog."default" NOT
NULL,
    bank_card character varying(20) COLLATE pg_catalog."default"
NOT NULL,
    CONSTRAINT user_account_pkey PRIMARY KEY (user_acc_id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.user_account
    OWNER to postgres;
```