Проект «Создание БД ПРОДАЖА БИЛЕТОВ»

12. БД "Продажа билетов".

Задача – информационная поддержка деятельности транспортных авиакасс

БД должна осуществлять:

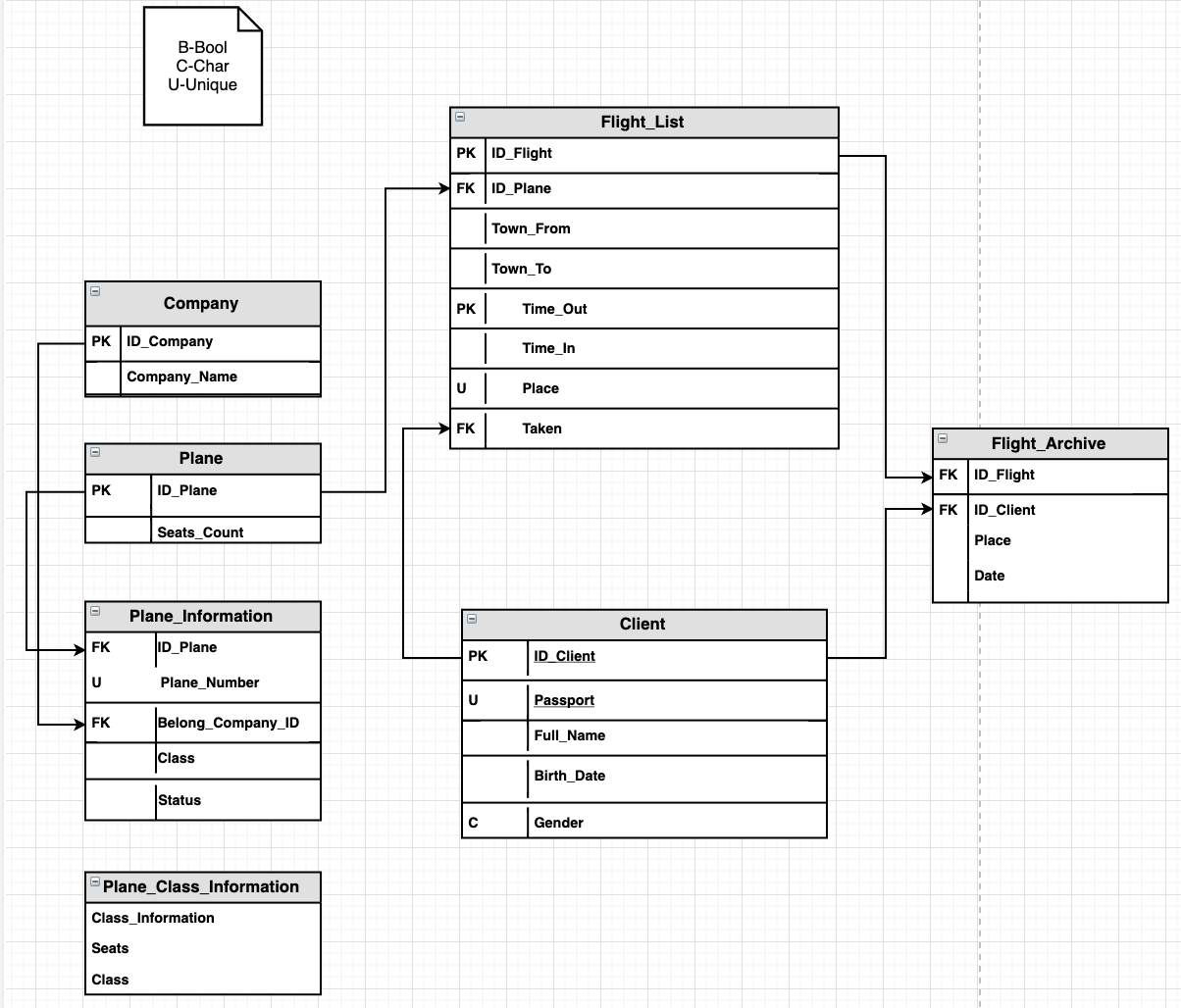
* ведение списка рейсов и билетов на них с указанием класса;
* учёт забронированных мест;
* ведение архива пассажиров за последний месяц.

Необходимо предусмотреть:

* продажу билетов в оба конца;
* поиск места на рейс в соответствии с требованиями заказчика;
* получение списка свободных мест на рейс;
* выдачу информации по конкретному рейсу;
* получение списка проданных мест;
* проверку наличия брони по имени клиента и/или названию организации.

РАЗРАБОТКА

**ER-модель**



ДОБАВИТЬ описание таблиц

**Код создании баз данных в СУБД SQLite (модуль Python):**

import pprint

import sqlite3

conn = sqlite3.connect('BD\_Air.sqlite')

c = conn.cursor()

pp = pprint.PrettyPrinter(width=120)

c.execute('''drop table if exists Company''')

c.execute('''drop table if exists Plane''')

c.execute('''drop table if exists Plane\_Information''')

c.execute('''drop table if exists Plane\_Class\_Information''')

c.execute('''drop table if exists Client''')

c.execute('''drop table if exists Flight\_List''')

c.execute('''drop table if exists Flight\_Archive''')

c.execute('''create table if not exists Company (

ID\_Company integer ,

Company\_Name char(40) not null,

Constraint PK\_Company primary key(ID\_Company)

);''')

c.execute ('''create table if not exists Plane\_Information(

ID\_Plane integer,

Plane\_Number Char(10) not null,

Belong\_Company\_ID integer not null,

Class char(30) not null,

Status char(1) not null,

Constraint PK\_ID\_Plane primary key(ID\_Plane),

Constraint FK\_Plane\_Information\_ID\_Plane FOREIGN KEY (ID\_Plane) REFERENCES Plane (ID\_Plane),

Constraint UQ\_Plane\_Information\_Plane\_Number Unique(Plane\_Number),

Constraint FK\_Plane\_Information\_Belong\_Company\_ID FOREIGN KEY (Belong\_Company\_ID) REFERENCES Company (ID\_Company),

Constraint CK\_Plane\_Information\_Status check(Status in('F','R'))

);''')

c.execute('''create table if not exists Plane\_Class\_Information(

Class\_Information char(30) not null,

Seats char(3),

Class char(1) not null,

Constraint FK\_Plane\_Class\_Information\_Class FOREIGN KEY (Class) REFERENCES Plane\_Information(Class)

);''')

c.execute ('''create table if not exists Client(

ID\_Client integer,

Passport integer(10) not null,

Full\_Name char(50) not null,

Brith\_Date date not null,

Gender char(1) not null,

Constraint PK\_Company primary key(ID\_Client),

Constraint UQ\_Client\_Passport Unique(Passport),

Constraint CK\_Client\_Gender CHECK(Gender in ('ж','м'))

);''')

c.execute('''create table if not exists Flight\_List(

ID\_Flight integer,

ID\_Plane integer not null,

Town\_From char(20) not null,

Town\_To char(20) not null,

Time\_Out date not null,

Time\_In date not null,

Constraint PK\_Flight\_List primary key(ID\_Flight),

Constraint FK\_Flight\_List\_ID\_Plane FOREIGN KEY (ID\_Plane) REFERENCES Plane\_Information (ID\_Plane)

);''')

c.execute('''create table if not exists Flight\_List\_Infrotmation(

ID\_Flight integer,

Place char(3),

Taken integer default null,

Constraint FK\_Flight\_List\_Infrotmation\_ID\_Flight foreign key (ID\_Flight) references plane(ID\_Flight),

Constraint UQ\_Flight\_List\_Infrotmation\_Place Unique (Place),

Constraint FK\_Flight\_List\_Infrotmation\_Taken FOREIGN KEY(Taken) REFERENCES Client(ID\_Client)

);''')

c.execute('''create table if not exists Flight\_Archive(

ID\_Flight integer,

ID\_Client integer,

Place char(3) not null,

Date date not null,

Constraint FK\_Flight\_Archive\_ID\_Flight foreign key (ID\_Flight) references Flight\_List(ID\_Flight),

Constraint FK\_Flight\_Archive\_ID\_Client foreign key (ID\_Client) references Client(ID\_Client)

);''')

Company = [

(1,'Fly Together'),

(2,'New Country'),

(3,'Back To The Pit')

]

Plane\_Information = [

(1,'1QDJ23Q8V0',1,'Туполев Ту-204','F'),

(2,'2QDJ23Q8V0',1,'Ильюшин ИЛ-86','F'),

(3,'3QDJ23Q8V0',2,'Туполев Ту-204','R'),

(4,'4QDJ23Q8V0',1,'Ильюшин ИЛ-86','F'),

(5,'5QDJ23Q8V0',2,'Туполев Ту-134','F'),

(6,'6QDJ23Q8V0',3,'Ильюшин ИЛ-86','R'),

(7,'7QDJ23Q8V0',2,'Туполев Ту-134','F')

]

Plane\_Class\_Information = [

('Туполев Ту-204','A01','B'),

('Туполев Ту-204','A02','B'),

('Туполев Ту-204','A03','B'),

('Туполев Ту-204','A04','B'),

('Туполев Ту-204','A05','B'),

('Туполев Ту-204','B01','E'),

('Туполев Ту-204','B02','E'),

('Туполев Ту-204','B03','E'),

('Туполев Ту-204','B04','E'),

('Туполев Ту-204','B05','E'),

('Ильюшин ИЛ-86','A01','B'),

('Ильюшин ИЛ-86','A02','B'),

('Ильюшин ИЛ-86','A03','B'),

('Ильюшин ИЛ-86','A04','B'),

('Ильюшин ИЛ-86','A05','B'),

('Ильюшин ИЛ-86','A06','B'),

('Ильюшин ИЛ-86','A07','B'),

('Ильюшин ИЛ-86','B01','E'),

('Ильюшин ИЛ-86','B02','E'),

('Ильюшин ИЛ-86','B03','E'),

('Ильюшин ИЛ-86','B04','E'),

('Ильюшин ИЛ-86','B05','E'),

('Ильюшин ИЛ-86','B06','E'),

('Ильюшин ИЛ-86','B07','E'),

('Туполев Ту-134','A01','B'),

('Туполев Ту-134','A02','B'),

('Туполев Ту-134','A03','B'),

('Туполев Ту-134','B01','E'),

('Туполев Ту-134','B02','E'),

('Туполев Ту-134','B03','E')

]

Client = [

(1,7518954612,'Иванов Иван Иванович','11.12.1990','м'),

(2,7518954613,'Курочкина Нина Ивановна','12.1.1991','ж'),

(3,7518954614,'Иванов Иван Иванович','13.2.1992','м'),

(4,7518954615,'Маликов Малик Маликович','14.3.1993','м'),

(5,7518954616,'Нативова Натива Нативовна','15.4.1994','ж'),

(6,7518954617,'Косяков Косяк Косякович','16.5.1995','м'),

(7,7518954618,'Старцев Старец Старецев','17.6.1996','м'),

(8,7518954619,'Мукина Мука Муковна','18.7.1997','ж'),

(9,7518954620,'Круосано Круасан Круасанов','19.8.1998','м'),

(10,7518954621,'Бедина Беда Бедовна','20.9.1999','ж')

]

c.executemany('''Insert Into Company VALUES(?,?)''',Company)

conn.commit()

c.executemany('''Insert Into Plane\_Information VALUES(?,?,?,?,?)''', Plane\_Information)

conn.commit()

c.executemany('''Insert Into Plane\_Class\_Information VALUES(?,?,?)''', Plane\_Class\_Information)

conn.commit()

c.executemany('''Insert Into Client VALUES(?,?,?,?,?)''', Client)

conn.commit()

conn.close()

**Код запросов:**

import pprint

import sqlite3

conn = sqlite3.connect('BD\_Air.sqlite')

c = conn.cursor()

pp = pprint.PrettyPrinter(width=120)

#Триггер на автозаполнение мест при добавлении полёта

c.execute(''' drop trigger IF EXISTS Create\_Fly ''')

c.execute('''create trigger if not exists Create\_Fly

after insert on Flight\_List

begin

insert into Flight\_List\_Infrotmation

select new.ID\_Flight,Seats,NULL

from Plane\_Class\_Information

where Plane\_Class\_Information = (

select Class

from Plane\_Information

where Plane\_Information.ID\_Plane = new.ID\_Plane

)

END;

''')

c.execute(''' insert into Flight\_List values(1,7,'Moscow','Cheluabinsk','07.11.2019')''')

conn.commit()

#Список рейсов с указанием мест и класса

c.execute(''' select Flight\_List.ID\_Flight, Town\_From, Town\_To, Time\_Out, Time\_In, Seats,Class

from Flight\_List join Plane\_Information

on Flight\_List.ID\_Plane=Plane\_Information.ID\_Plane

join Plane\_Class\_Information

on Plane\_Class\_Information.Class\_Information = Plane\_Information.Class

Where Taken = NULL''')

pp.pprint(c.fetchall())

#Учет всех забронированных мест

c.execute(''' select count(Taken) as Selled,

from Flight\_List join Flight\_List\_Infrotmation

on Flight\_List.ID\_Flight=Flight\_List\_Infrotmation.ID\_Flight

Where Taken <> NULL''')

pp.pprint(c.fetchall())

#Учет всех забронированных мест на рейсе

find =''' select count(Taken) as Selled,

from Flight\_List join Flight\_List\_Infrotmation

on Flight\_List.ID\_Flight=Flight\_List\_Infrotmation.ID\_Flight

Where Taken <> NULL and Flight\_List.ID\_Flight = ?'''

print('Где хотите посмотреть выкупленные билеты?')

id\_flight=int(input())

c.execute(find,[(id\_flight)])

pp.pprint(c.fetchall())

#Запрос информации о свободных местах на рейсе

find = '''select Flight\_List.ID\_Flight, Town\_From, Town\_To, Time\_Out, Time\_In, Place

from Flight\_List join Flight\_List\_Infrotmation

on Flight\_List.ID\_Flight=Flight\_List\_Infrotmation.ID\_Flight

Where Taken = NULL Flight\_List.ID\_Flight = ? '''

print('Где хотите посмотреть свободные места?')

id\_flight = int(input())

c.execute(find,[(id\_flight)])

pp.pprint(c.fetchall())

#Запрос места на рейс

find =''' select Flight\_List.ID\_Flight, Town\_From, Town\_To, Time\_Out, Time\_In, Place

from Flight\_List join Flight\_List\_Infrotmation

on Flight\_List.ID\_Flight=Flight\_List\_Infrotmation.ID\_Flight

where Town\_From = ? and Town\_To = ? and Time\_Out = ? and Taken = NULL'''

print('Введите от куда хотите улететь')

town\_from=input()

print('Введите куда хотите прилететь')

town\_to=input()

print('Введите когда хотите улететь')

time\_out=input()

c.execute(find,[(town\_from),(town\_to),(time\_out)])

pp.pprint(c.fetchall())

#Проверка наличии брони по паспорту(Имя не уникально)

find =''' select Flight\_List.ID\_Flight, Town\_From, Town\_To, Time\_Out, Time\_In, Place

from Flight\_List join Flight\_List\_Infrotmation

on Flight\_List.ID\_Flight=Flight\_List\_Infrotmation.ID\_Flight

Where Taken = (select ID\_Client

from CLient

where Passport = ?)'''

print('Введите паспорт клиента')

passport = int(input())

c.execute(find,[(passport)])

pp.pprint(c.fetchall())

conn.close()

ДОБАВИТЬ скрины исполнения запросов.