Первая нормальная форма (1NF) - это состояние, когда все столбцы в таблице содержат только атомарные значения, то есть значения, которые нельзя дальше разделить на более мелкие части. Каждая ячейка таблицы должна содержать только одно значение.

Вторая нормальная форма (2NF) - в таблице должен быть определен ключ (первичный ключ), и каждый неключевой атрибут должен быть полностью зависимым от ключа, а не от какой-либо его части. Если атрибуты зависят только от части ключа, то таблицу следует разделить на более мелкие таблицы.

Третья нормальная форма (3NF) - все неключевые атрибуты должны быть непосредственно зависимыми от первичного ключа, а не от других неключевых атрибутов. Если атрибуты зависят от других атрибутов, то таблицу следует разделить на отдельные таблицы.

**Типы связей в программировании:**

- Один к одному (One-to-One)

- Один ко многим (One-to-Many)

- Многие ко многим (Many-to-Many)

Реляционная модель данных - это подход к организации и хранению данных в базах данных, где данные представлены в виде таблиц, состоящих из строк и столбцов, а связи между данными устанавливаются с помощью ключей.

Организации защиты данных в хранилищах могут включать:

- Шифрование данных

- Управление доступом и аутентификация

- Резервное копирование и восстановление данных

- Обеспечение физической безопасности хранилища данных

- Мониторинг и аудит данных

- Использование защищенных протоколов связи

Регистрация

import re

from connect import connection

def user\_authentication(login, password):

with connection.cursor() as cursor:

user\_authentication\_query = f"SELECT email, login, password FROM Customer " \

f"WHERE (email = '{login}' AND password = '{password}') " \

f"OR login = '{login}' AND password = '{password}'"

cursor.execute(user\_authentication\_query)

user\_authentication\_result = cursor.fetchall()

return user\_authentication\_result

def user\_registration(first\_name, last\_name, email, phone, login, password, passport):

with connection.cursor() as cursor:

user\_registration\_query = f"INSERT INTO Customer (first\_name, last\_name, email, phone, login, password, passport) " \

f"VALUES ('{first\_name}', '{last\_name}', '{email}', '{phone}', '{login}', PASSWORD('{password}'), '{passport}')"

cursor.execute(user\_registration\_query)

connection.commit()

# Пример использования функций

login = input("Введите логин\email: ")

password = input("Введите пароль: ")

auth\_result = user\_authentication(login, password)

if auth\_result:

print("Вход выполнен успешно!")

else:

print("Неверный логин или пароль")

first\_name = input("Введите имя: ")

last\_name = input("Введите фамилию: ")

email = input("Введите email: ")

phone = input("Введите телефон: ")

login = input("Введите логин: ")

password = input("Введите пароль: ")

passport = input("Введите паспортные данные: ")

user\_registration(first\_name, last\_name, email, phone, login, password, passport)

print("Регистрация выполнена успешно!")

Авторизация

import re

from connect import connection

def user\_authentication(login, password):

with connection.cursor() as cursor:

user\_authentication\_query = f"SELECT email, login, password FROM Customer " \

f"WHERE (email = '{login}' AND password = '{password}') " \

f"OR login = '{login}' AND password = '{password}'"

cursor.execute(user\_authentication\_query)

user\_authentication\_result = cursor.fetchall()

return user\_authentication\_result

# Пример использования функции

login = input("Введите логин или email: ")

password = input("Введите пароль: ")

auth\_result = user\_authentication(login, password)

if auth\_result:

print("Вход выполнен успешно!")

else:

print("Неверный логин или пароль")

import re  
from connect import connection  
try:  
 with connection.cursor() as cursor:  
 while True:  
  
 DB = input("Авторизация (1), регистрация (2): ")  
 if DB == "1":  
 login = input("Введите логин\email: ")  
 password = input("Введите пароль: ")  
 user\_avtorization = f"select email, login, password from Customer " \  
 f"where (email = '{login}' and password = '{password}') " \  
 f"or login = '{login}' and password = '{password}'"  
 cursor.execute(user\_avtorization)  
 user\_avtorization = cursor.fetchall()  
 if user\_avtorization:  
 print("Вход выполнен успешно!")  
 break  
  
 else:  
 print("Неверный логин или пароль")  
  
 elif DB == "2":  
 print("Регистрация: ")  
 namereg = input("Введите имя:")  
 familreg = input("Введите фамилию:")  
 phonereg = input("Введите телефон:")  
 loginreg = input("Введите логин:")  
 while True:  
 passreg = input("Введите пароль: ")  
 if re.search("[@\_!#$%^&\*()<>?/\|}{~:]", passreg):  
 break  
 else:  
 print("Пароль должен содержать хотя бы один специальный символ: @, \_, !, #, $, %, ^, &, \*, (, ), <, >, ?, /, \, |, {, }, ~, :")  
 emailreg = input("Введите email: ")  
 paspreg = input()  
 user\_reg = f"insert into Customer(first\_name,last\_name,email,phone,login,password, passport) values " \  
 f"('{namereg}','{familreg}','{emailreg}','{phonereg}','{loginreg}',PASSWORD('{passreg}'),'{paspreg}')"  
 cursor.execute(user\_reg)  
 user\_reg = cursor.fetchall()  
 connection.commit()  
 if user\_reg:  
 print("Регестрация выполнен успешно!")  
 break  
finally:  
 connection.close()

import connect  
from connect import connection  
import sql\_queries  
  
try:  
 with connection.cursor() as cursor:  
 cursor.execute(sql\_queries.view\_List\_of\_pets\_by\_category)  
 view1 = cursor.fetchall()  
 print("Список животных по категориям:")  
 for i in view1:  
 print(f"Категория: {i['category']}, | кличка: {i['nickname']}, | возраст: {i['age']} , | Порода: {i['breed']}, | цена: {i['price']} ")  
  
 print("\_" \* 20)  
 cursor.execute(sql\_queries.view\_List\_of\_receipts\_and\_customers)  
 view2 = cursor.fetchall()  
 print("Продажи:")  
 for y in view2:  
 print(  
 f"Дата: {y['date']}, | Способ оплаты: {y['payment\_type']}, | Имя: {y['first\_name']},| Фамилия: {y['last\_name']},| Телефон: {y['phone']} ")  
  
 print("\_" \* 20)  
  
 cursor.execute(sql\_queries.view\_The\_most\_expensive\_pet)  
 view3 = cursor.fetchall()  
 print("Самые дороги:")  
 for e in view3:  
 print(f"Кличка: {e['nickname']}, | Порода: {e['breed']}, | Цена: {e['price']} ")  
  
 print("\_" \* 20)  
  
 cursor.execute(sql\_queries.view\_Category\_with\_the\_most\_pets)  
 view4 = cursor.fetchall()  
 print("Больше всего питомцев:")  
 for u in view4:  
 print(f"Количество животных : {u['num\_pets']}, | Категории : {u['category']}")  
  
  
finally:  
 connection.close()