

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**МИРЭА** – **Российский технологический университет**»

**РТУ МИРЭА**

**Институт комплексной безопасности и цифровых технологий (ИКБ) Кафедра КБ-14 «Цифровые технологии обработки данных»**

**Администрирование баз данных**

**Практическая работа №1**

Выполнили студенты 3 курса

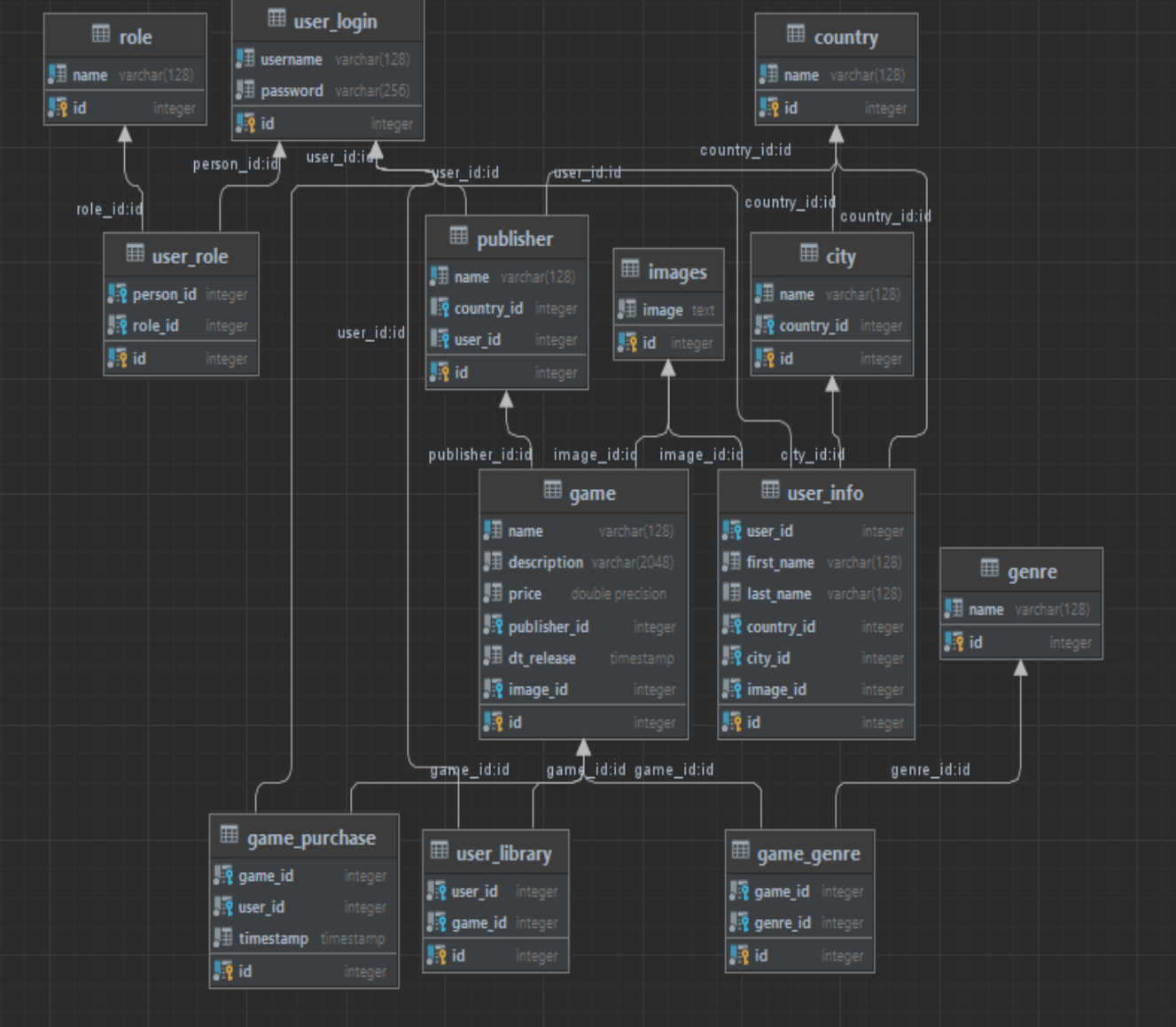
Зарин Н.Н.

Луговой И.И.

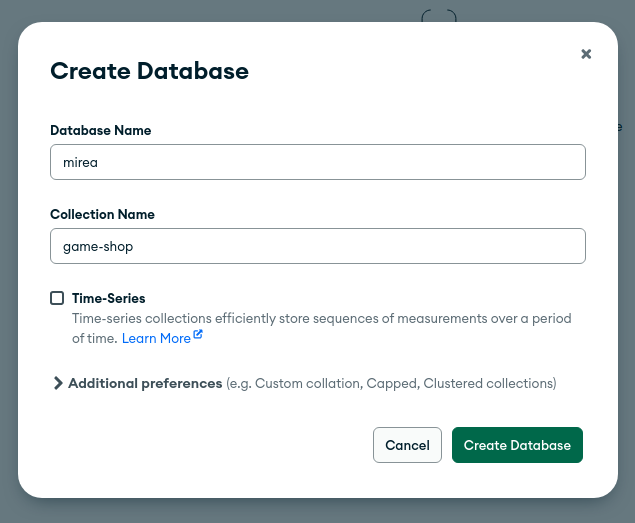
БСБО-05-20

Москва 2023

В качестве темы для БД был взят интернет-магазин игр, который использовался на прошлом курсе для CRM-системы.



Cоздание БД в MongoDB Compass



Создание коллекций

Категории

db.createCollection(  
 "categories", {  
 validator: {  
 $jsonSchema: {  
 bsonType: "object",  
 title: "Category validation",  
 required: [  
 "name"  
 ],  
 properties: {  
 name: {  
 bsonType: "string"  
 }  
 }  
 }  
 }  
 }  
)

Игры

db.createCollection(  
 "games", {  
 validator: {  
 $jsonSchema: {  
 bsonType:"object",  
 title: "Game validation",  
 required: [  
 "name",  
 "amount",  
 "price",  
 "publisher",  
 "category"  
 ],  
 properties: {  
 name: {  
 bsonType:"string"  
 },  
 amount: {  
 bsonType: "int"  
 },  
 price: {  
 bsonType: "int"  
 },  
 publisher: {  
 bsonType: "string"  
 },  
 category: {  
 bsonType: "objectId"  
 }  
 }  
 }  
 }  
 }  
)

Работники

db.createCollection(  
 "workers", {  
 validator: {  
 $jsonSchema: {  
 bsonType: "object",  
 title: "Worker validation",  
 required: [  
 "name",  
 "phone",  
 "email",  
 "position",  
 ],  
 properties: {  
 name: {  
 bsonType: "string"  
 },  
 position: {  
 bsonType: "string"  
 },  
 phone: {  
 bsonType: "string"  
 },  
 email: {  
 bsonType: "string"  
 }  
 }  
 }  
 }  
 }  
)

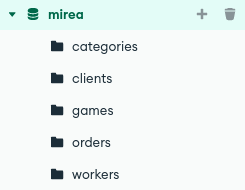
Клиенты

db.createCollection(  
 "clients", {  
 validator: {  
 $jsonSchema: {  
 bsonType: "object",  
 title: "Client validation",  
 required: [  
 "nickname",  
 "email"  
 ],  
 properties: {  
 nickname: {  
 bsonType: "string"  
 },  
 email: {  
 bsonType: "string"  
 }  
 }  
 }  
 }  
 }  
)

Заказы

db.createCollection(  
 "orders", {  
 validator: {  
 $jsonSchema: {  
 bsonType: "object",  
 title: "Order validation",  
 required: [  
 "client",  
 "games",  
 "date",  
 "status"  
 ],  
 properties: {  
 client: {  
 bsonType: "objectId"  
 },  
 games: {  
 bsonType: "array",  
 items: {  
 required: [  
 "game"  
 ],  
 properties: {  
 game: {  
 bsonType: "objectId"  
 }  
 }  
 }  
 },  
 date: {  
 bsonType: "date"  
 },  
 status: {  
 bsonType: "bool"  
 }  
 }  
 }  
 }  
 }  
)

БД



Заполнение данными

db.categories.insertMany([  
 {  
 "name": "Шутер"  
 },  
 {  
 "name": "Приключение"  
 },  
 {  
 "name": "Хоррор"  
 },  
 {  
 "name": "Файтинг"  
 },  
 {  
 "name": "Фентези"  
 },  
 {  
 "name": "Гонки"  
 },  
 {  
 "name": "Головоломки"  
 },  
 {  
 "name": "Детектив"  
 },  
 {  
 "name": "Аниме"  
 },  
 {  
 "name": "Симулятор"  
 },  
 {  
 "name": "Экшн"  
 }  
])  
db.clients.insertMany([  
 {  
 "nickname": "tawer228",  
 "email": "tawer228@ya.ru"  
 },  
 {  
 "nickname": "keka564",  
 "email": "fgh3@gmail.com"  
 },  
 {  
 "nickname": "gus",  
 "email": "gusein22@bk.com"  
 },  
 {  
 "nickname": "walter1",  
 "email": "ww563@gmail.com"  
 },  
 {  
 "nickname": "baltika7",  
 "email": "pivolublu@mail.ru"  
 }  
])  
db.workers.insertMany([  
 {  
 "name": "Жыксамонов Акылбек",  
 "phone": "+79037484847",  
 "email": "zhick\_a@gmail.com",  
 "position": "Старший модератор"  
 },  
 {  
 "name": "Иванов Иван",  
 "phone": "+79154862915",  
 "email": "ivaaaaaaaaaaan@ya.ru",  
 "position": "Модератор"  
 },  
 {  
 "name": "Зауров Владислав",  
 "phone": "+79037583916",  
 "email": "vlad1ck\_z@gmail.com",  
 "position": "Разработчик"  
 },  
])  
db.games.insertMany([  
 {  
 "name": "Rocket League",  
 "price": 0,  
 "publisher": "Epic Games",  
 "categories": [  
 {  
 "category": db.categories.findOne({"name": "Гонки"})["\_id"],  
 }  
 ]  
 },  
 {  
 "name": "Battlefield 5",  
 "price": 2499,  
 "publisher": "EA",  
 "categories": [  
 {  
 "category": db.categories.findOne({"name": "Шутер"})["\_id"],  
 }  
 ]  
 },  
 {  
 "name": "Uncharted 4",  
 "price": 2999,  
 "publisher": "Sony",  
 "categories": [  
 {  
 "category": db.categories.findOne({"name": "Экшн"})["\_id"],  
 },  
 {  
 "category": db.categories.findOne({"name": "Шутер"})["\_id"],  
 },  
 {  
 "category": db.categories.findOne({"name": "Приключение"})["\_id"],  
 }  
 ]  
 },  
 {  
 "name": "Resident Evil Village",  
 "price": 1499,  
 "publisher": "Capcom",  
 "categories": [  
 {  
 "category": db.categories.findOne({"name": "Хоррор"})["\_id"],  
 },  
 {  
 "category": db.categories.findOne({"name": "Приключение"})["\_id"],  
 }  
 ]  
 },  
 {  
 "name": "Assetto Corsa",  
 "price": 799,  
 "publisher": "Kunos Simulazioni",  
 "categories": [  
 {  
 "category": db.categories.findOne({"name": "Гонки"})["\_id"]  
 },  
 {  
 "category": db.categories.findOne({"name": "Симулятор"})["\_id"]  
 }  
 ]  
 }  
])  
db.orders.insertMany([  
 {  
 "client": db.clients.findOne({"email": "gusein22@bk.com"})["\_id"],  
 "games": [  
 {  
 "game": db.games.findOne({"name": "Resident Evil Village"})["\_id"]  
 }  
 ],  
 "date": new ISODate(),  
 "status": true,  
 },  
 {  
 "client": db.clients.findOne({"email": "tawer228@ya.ru"})["\_id"],  
 "games": [  
 {  
 "game": db.games.findOne({"name": "Rocket League"})["\_id"]  
 }  
 ],  
 "date": new ISODate(),  
 "status": true,  
 },  
 {  
 "client": db.clients.findOne({"email": "pivolublu@mail.ru"})["\_id"],  
 "games": [  
 {  
 "game": db.games.findOne({"name": "Battlefield 5"})["\_id"]  
 }  
 ],  
 "date": new ISODate(),  
 "status": false,  
 },  
 {  
 "client": db.clients.findOne({"email": "ww563@gmail.com"})["\_id"],  
 "games": [  
 {  
 "game": db.games.findOne({"name": "Assetto Corsa"})["\_id"]  
 }  
 ],  
 "date": new ISODate(),  
 "status": true,  
 },  
 {  
 "client": db.clients.findOne({"email": "fgh3@gmail.com"})["\_id"],  
 "games": [  
 {  
 "game": db.games.findOne({"name": "Uncharted 4"})["\_id"]  
 }  
 ],  
 "date": new ISODate(),  
 "status": true,  
 },  
])

Реализовать следующие запросы к созданной БД:

1. Получение списка всех категорий

db.categories.find()

1. Получение списка всех продуктов в категории

db.games.find({"categories.category": db.categories.findOne({name: "Гонки"})["\_id"]})

1. Поиск продукта по названию

db.games.findOne({"name": "Rocket League"})

1. Добавление продукта в корзину клиента

db.orders.insertOne(  
{  
 "client": db.clients.findOne({"email": "pivolublu@mail.ru"})["\_id"],  
 "games": [  
 {  
 "game": db.games.findOne({"name": "Assetto Corsa"})["\_id"]  
 }  
 ],  
 "date": new ISODate(),  
 "status": false,  
})

1. Получение списка всех заказов клиента

db.orders.find({"client": db.clients.findOne({"email": "pivolublu@mail.ru"})["\_id"]})

1. Обновление статуса заказа

db.orders.updateOne({"\_id": ObjectId("647b8163d9ca604b34de0474")}, {"$set": {"status": true}})

1. Получение списка топ-продаж за последние месяцы с учетом цены и количества проданных товаров

db.orders.aggregate([  
 {  
 $match: {  
 date: {  
 $gte: new ISODate('2023-06-03')  
 }  
 }  
 },  
 {  
 $unwind: "$games"  
 },  
 {  
 $lookup:  
 {  
 from: "games",  
 localField: "games.game",  
 foreignField: "\_id",  
 as: "games"  
 }  
 },  
 {  
 $unwind: "$games"  
 },  
 {  
 $group:  
 {  
 \_id: "$games.name",  
 total\_sum: {  
 $sum: "$games.price"  
 },  
 }  
 },  
 {  
 $sort: {  
 total\_sum: -1  
 }  
 }  
])

1. Получение списка клиентов, которые сделали более чем N покупок в последнее время

db.orders.aggregate([  
 {  
 $match: {  
 date: {  
 $gte: new ISODate('2023-06-01')  
 }  
 }  
 },  
 {  
 $lookup:  
 {  
 from: "clients",  
 localField: "client",  
 foreignField: "\_id",  
 as: "clients"  
 }  
 },  
 {  
 $group:  
 {  
 \_id: "$clients.\_id",  
 orders\_count: {  
 $count: {}  
 }  
 }  
 },  
 {  
 $match: {  
 orders\_count: {  
 $gt: 1  
 }  
 }  
 },  
 {  
 $sort: {  
 orders\_count: -1  
 }  
 }  
])

1. Получите какие категории товаров пользовались спросом в заданный срок

db.orders.aggregate(  
 {  
 $match: {  
 date: {  
 $gte: new ISODate('2023-06-01'),  
 $lte: new ISODate('2023-06-05')  
 }  
 }  
 },  
 {  
 $unwind: "$games"  
 },  
 {  
 $lookup: {  
 from: "games",  
 localField: "games.game",  
 foreignField: "\_id",  
 as: "game"  
 }  
 },  
 {  
 $unwind: "$games"  
 },  
 {  
 $lookup:  
 {  
 from: "categories",  
 localField: "game.category",  
 foreignField: "\_id",  
 as: "categories"  
 }  
 },  
 {  
 $group:  
 {  
 \_id: "$categories.name"  
 }  
 }  
)

1. Какие товары не были проданы в какую-то дату

db.orders.aggregate(  
 {  
 $match: {  
 date: {  
 $gte: new ISODate('2023-06-01'),  
 $lte: new ISODate('2023-06-05')  
 }  
 }  
 },  
 {  
 $unwind: "$games"  
 },  
 {  
 $lookup: {  
 from: "games",  
 localField: "game",  
 foreignField: "\_id",  
 as: "games"  
 }  
 },  
 {  
 $unwind: "$games"  
 },  
 {  
 $unset: "games"  
 }  
)

Создание пользователей для БД:

Роль администратора

db.createRole({  
 role: "admin",  
 privileges: [  
 {  
 resource: {db: "mirea", collection: "categories"},  
 actions: ["find", "insert", "update", "remove"]  
 },  
 {  
 resource: {db: "mirea", collection: "games"},  
 actions: ["find", "insert", "update", "remove"]  
 },  
 {  
 resource: {db: "mirea", collection: "workers"},  
 actions: ["find", "insert", "update", "remove"]  
 },  
 {  
 resource: {db: "mirea", collection: "clients"},  
 actions: ["find", "insert", "update", "remove"]  
 },  
 {  
 resource: {db: "mirea", collection: "orders"},  
 actions: ["find", "insert", "update", "remove"]  
 }  
 ],  
 roles: []  
})

Роль менеджера

db.createRole({  
 role: "manager",  
 privileges: [  
 {  
 resource: {db: "mirea", collection: "categories"},  
 actions: ["find", "insert", "update", "remove"]  
 },  
 {  
 resource: {db: "mirea", collection: "games"},  
 actions: ["find", "insert", "update", "remove"]  
 },  
 {  
 resource: {db: "mirea", collection: "orders"},  
 actions: ["find", "insert", "update", "remove"]  
 }  
 ],  
 roles: []  
})

Роль пользователя

db.createRole({  
 role: "user",  
 privileges: [  
 {  
 resource: {db: "mirea", collection: "games"},  
 actions: ["find"]  
 },  
 {  
 resource: {db: "mirea", collection: "orders"},  
 actions: ["find", "insert"]  
 }  
 ],  
 roles: []  
})

Роль гостя

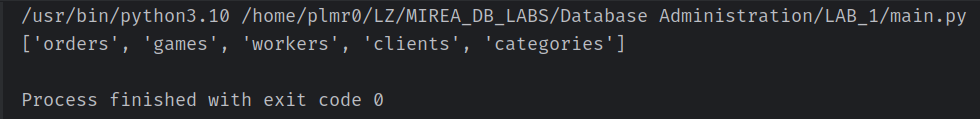
db.createRole({  
 role: "guest",  
 privileges: [  
 {  
 resource: {db: "mirea", collection: "games"},  
 actions: ["find"]  
 },  
 ],  
 roles: []  
})

Реализация программы на языке Python:

from pymongo import MongoClient  
from datetime import datetime  
  
  
class Client:  
 def \_\_init\_\_(self):  
 self.client = MongoClient('localhost', 27017)  
 self.db = self.client['mirea']  
  
 # Все коллекции  
 def show\_collections(self):  
 return self.db.list\_collection\_names()  
  
 # Все категории  
 def show\_categories(self):  
 categories\_collection = self.db['categories']  
 return [category['name'] for category in categories\_collection.find()]  
  
 # Поиск игры по названию  
 def find\_games\_by\_name(self, name):  
 games\_collection = self.db['games']  
 cursor = games\_collection.find({'name': {'$regex': name, '$options': 'i'}})  
 games = [game['name'] for game in cursor]  
 return games  
  
 # Поиск игр по категории  
 def find\_games\_by\_category(self, category):  
 category\_collection = self.db['categories']  
 game\_collection = self.db['games']  
 category\_games = game\_collection.find({'categories.category': category\_collection.find\_one({'name': category})["\_id"]})  
 games = [game['name'] for game in category\_games]  
 return games  
  
 # Поиск игр по издателю  
 def find\_games\_by\_publisher(self, publisher):  
 games\_collection = self.db['games']  
 cursor = games\_collection.find({'publisher': {'$regex': publisher, '$options': 'i'}})  
 games = [game['name'] for game in cursor]  
 return games  
  
 # Поиск игр в заданном ценовом диапазоне  
 def find\_games\_by\_price(self, min\_price, max\_price):  
 games\_collection = self.db['games']  
 cursor = games\_collection.find({'price': {'$gte': min\_price, '$lte': max\_price}})  
 games = [game['name'] for game in cursor]  
 return games  
  
 # Добавить игры в корзину  
 def add\_games\_to\_order(self, client\_email, \*args):  
 \_id = self.db.orders.insert\_one(  
 {  
 "client": self.db.clients.find\_one({"email": client\_email})["\_id"],  
 "games": [  
 {  
 "game": self.db.games.find\_one({"name": game})["\_id"]  
 }  
 for game in args  
 ],  
 "date": datetime.now(),  
 "status": False,  
 })  
 return \_id.inserted\_id  
  
 # Общая стоимость корзины  
 def calculate\_cart\_total(self, \*args):  
 total = 0  
 games\_collection = self.db['games']  
 for name in args:  
 game = games\_collection.find\_one({'name': name})  
 total += game['price']  
 return total

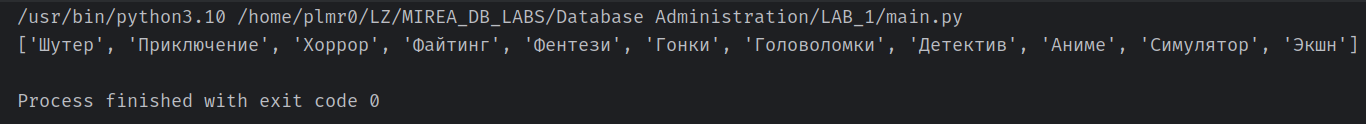
Вывод всех коллекций:

client = Client()  
print(client.show\_collections())



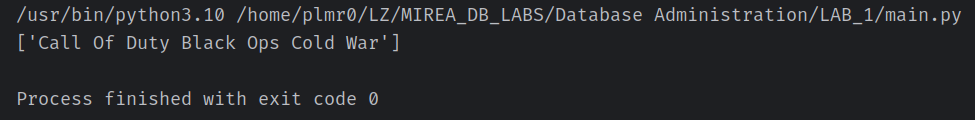
Вывод всех категорий:

client = Client()  
print(client.show\_categories())



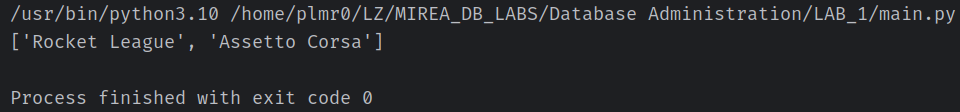
Поиск игр по названию:

client = Client()  
print(client.find\_games\_by\_name('Call Of Duty'))



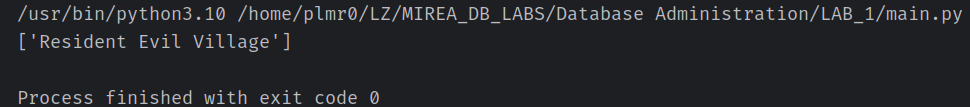
Поиск игр по категории:

client = Client()  
print(client.find\_games\_by\_category('Гонки'))



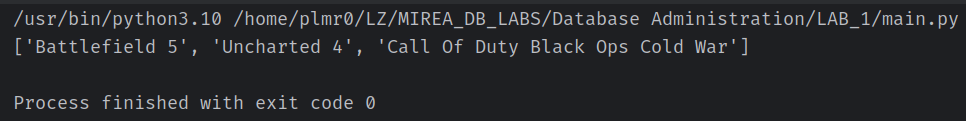
Поиск игр по издателю:

client = Client()  
print(client.find\_games\_by\_publisher('Capcom'))



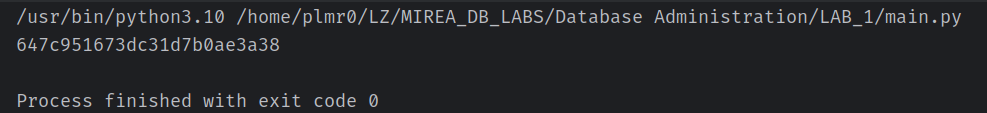
Поиск игр в заданном ценовом диапазоне:

client = Client()  
print(client.find\_games\_by\_price(1999, 4999))



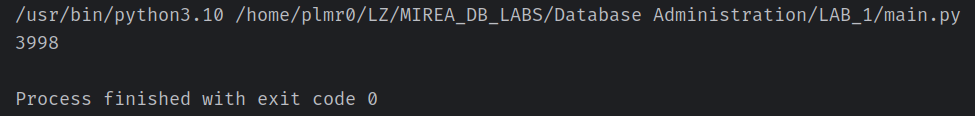
Добавление игр в корзину:

client = Client()  
print(client.add\_games\_to\_order('pivolublu@mail.ru', 'Rocket League', 'Call Of Duty Black Ops Cold War'))



Подсчет общей стоимости корзины:

client = Client()  
print(client.calculate\_cart\_total('Resident Evil Village', 'Battlefield 5'))



Тесты для проверки схемы базы данных:

import pymongo  
  
client = pymongo.MongoClient("mongodb://localhost:27017/")  
db = client["mirea"]  
  
  
def test\_games\_collection():  
 games = db["games"]  
  
 assert games is not None  
 assert "name" in games.find\_one()  
 assert "price" in games.find\_one()  
 assert "publisher" in games.find\_one()  
 assert "categories" in games.find\_one()  
  
  
def test\_categories\_collection():  
 categories = db["categories"]  
  
 assert categories is not None  
 assert "name" in categories.find\_one()  
  
  
def test\_clients\_collection():  
 clients = db["clients"]  
  
 assert clients is not None  
 assert "nickname" in clients.find\_one()  
 assert "email" in clients.find\_one()  
  
  
def test\_workers\_collection():  
 workers = db["workers"]  
  
 assert workers is not None  
 assert "name" in workers.find\_one()  
 assert "email" in workers.find\_one()  
 assert "phone" in workers.find\_one()  
 assert "position" in workers.find\_one()  
  
  
def test\_orders\_collection():  
 orders = db["orders"]  
  
 assert orders is not None  
 assert "client" in orders.find\_one()  
 assert "games" in orders.find\_one()  
 assert "date" in orders.find\_one()  
 assert "status" in orders.find\_one()

Результат тестов:

