Интерация – последовательное выполнение набора инструкций  
Итераторы – это объекты к которым можем применить for .. in ...

Генератор (yield) – это объект который мы можем перебрать в цикле. Он хранит в памяти только один последний объект.

генераторы применяются в тех случаях, когда нет необходимости сохранять всю последовательность и промежуточные значения в памяти. Так-же помогают обрабатывать большой объем данных.  
Генератор выполняет только один раз, он ленивый и его нужно пинать чтоб он работал.  
функцией next() можно извлечь объект генератора и перейти к следующей итерации

def gen (\*arg):

    for i in arg:

        yield i

gen = gen(1,2,3,4)

print(next(gen))

print(next(gen))

Инкапсуляция – ограничение доступа к полям и методам (\_\_tak), для безопасности и целостности состояния объекта, показывает программисту что это поле или метод нужно использовать только внутри класса.

Наследование –описание нового класса на базе существующего. Все методы и свойства у наследуются от родительского класса.

Полиморфизм – мы можем использовать один и тот же метод для разных типов данных.

Абстракция – это абстрактный класс, который содержит не менее одного абстрактного метода.  
 Абстрактным называется объявленный, но не реализованный метод.

Объектно-ориентированный подход программирования – парадигма программирования, один из видов структурного программирования, представлена на том что все можно представить как объект, который в свою очередь является экземпляром класса.

Концепции ООП: Наследование, абстракция, инкапсуляция, полиморфизм.

@staticmethod - похожи на методы класса, но полностью привязаны к классу, а не к конкретным объектам. Доступ к ним осуществляется с помощью имен классов, ему не нужен определенный первый аргумент не self не cls

@classmethod – Методы класса привязаны к самому классу, а не его экземпляру. Они могут менять состояние класса, что отразится на всех объектах этого класса, но не могут менять конкретный объект. методы класса отмечаются декоратором @classmethod

@property - декоратор метода, который позволяет делать метод атрибутом и вызывать его без скобок.

Объект – это набор данных(свойств) и функций(методов).  
Конструктор класса (\_\_init\_\_) – это функция для создания объектов.  
self – самый первый параметр во всех методах, нужен для того чтоб созданный объект мог обратиться к собственным методам или свойствам.  
Экземпляр класса – это объект который относится к определенному классу  
  
class <Имя класса>:

*# Конструктор класса — функция с именем \_\_init\_\_.*

def \_\_init\_\_(self, <параметры конструктора>):

*# Тело метода: тут объявляются свойства класса.*

Типизация в Python – Динамическая, переменные не привязаны жестко к определенному типу.

Изменяемые и не изменяемые типы - изменяемые можно изменить после его создания (list, set, dict), а не изменяемые нет (int, float, str, bool, tuple)

Функция id() – возвращает уникальный идентификатор объекта.

Функция hash()- возвращает хеш-значение объекта, если оно есть. Хэш-значения являются целыми числами. Они используются для быстрого сравнения ключей словаря во время поиска в словаре.

Почему ключом словаря должен быть неизменяемый объект? – Потому что в ключах словарей используются хеш-значения, и если поменять имя ключа то поменяется и хеш.

Может ли ключом словаря быть объект класса, объект модели ? -может, только если объект имеет хеш- значение (для этого нужен \_\_hash\_\_()метод)

Почему list.sort не вернет отсортированный список? – если в список содержит различные типы данных, то при сортировке выйдет исключение TypeError, для того чтоб сортировать различные типы данных нужно на аргумент key= указать lambda функцию.

Каким методом добавить элемент в кортеж? В кортеж нельзя добавить элемент.

Какие базовые структуры данных есть в python?  
 Структуры данных (TUPLE, LIST, DICT, SET) -   
 Кортеж () – не изменяемая упорядоченная коллекция произвольных типов элементов.  
 методы: index(n) – находит индекс элемента  
 count(n) – считает количество элементов n в кортеже.

Список [] – изменяемая упорядоченная коллекция произвольных типов элементов.  
 методы: list.append(element) – добавление элемента в конец списка.  
 list.insert(index, element) – добавление элемента по индексу.  
 list.extend([elem, elem, elem]) – добавляет элементы в конец списка.  
 list.clean() – очищает список.  
 list.count(n) – считает количество элементов n в списке.  
 list.copy() – Скопировать список в какую-нибудь переменную.  
 list.pop(index) – Извлекает элемент по индексу.  
 list.remove(element) – удаляет один element который быль самый первый.  
 list.reverse() – Переворачивает список.  
 list.sort(key=None, reverse=False) – Сортировка списка.

Словарь {‘1’:2} – изменяемая и упорядоченная коллекция.  
 методы: dict.get(key, [default]=None) – получение элемента словаря по ключу.  
 dict.pop(key, [default]) – извлекает элемент из словаря по ключу.  
 dict.values() – получает все значения словаря.  
 dict.keys() -получает все значения ключей.  
 dict.items() – получает все элементы ключей и значений словаря.  
 dict.update(other,\*\*kwargs) – обновляет значение словаря по ключу (key1=3),({key2’:2})

Множества {1,2} – хранят только уникальные значения, могут быть изменными set() и не изменными frozenset().  
 set.add(element) – добавить элемент.  
 set.difference(elements) – найдет разницу между между множеством и элементом и выведет не повторные элементы.  
 set.intersection(elements) – выведет только схожие элементы множества и введенных элементов.

Что такое comprehension? Какие виды существуют?

Это некие генераторы, при помощи которого можно визуально уменьшить код. бывают: генератор списка, генератор словаря, генератор множеств, генератор генераторов.

List comprehension – [I for I in range(5)]  
dect comprehension – {i: i\*\*2 for I in range(5)}  
set comprehension – {I for I in range(5)}  
generator comprehension – (i\*\*2 for I in range(5)}

Для каких базовых структур данных comprehension? Списки, словари, множества.

Как в python реализован switch? - Довольно распространённый способ организации конструкции switch-case в Python — это использование словаря.

def switch\_demo(argument):

    switcher = {

    1: "January",

    2: "February",

        3: "March",

        4: "April",

        5: "May",

        6: "June",

        7: "July",

        8: "August",

        9: "September",

        10: "October",

        11: "November",

        12: "December"

    }

    print(switcher.get(argument, "Invalid month"))

switch\_demo(101)

Функция в python - представляют блок кода, который выполняет определенную задачу и который можно повторно использовать в других частях программы. Объект, принимающий аргументы и возвращающий значение.

Аргументы функции. Передача аргументов в функцию – Аргументы это изменяемые и неизменяемые объекты. Для передачи аргументов в функцию нужно обратиться к функции и в круглых скобках указать что мы хотим передать в функцию. НО функция должна ждать эти аргументы, это должно быть заложено при создании этой функции.

Как проверить, что объект может быть вызван (callable)? - Функция callable() возвращает True , если указанный объект вызываемый, в противном случае она возвращает значение False.

Распаковка последовательностей, распаковка части последовательности - \*args и \*\*kwargs

Как проверить наличие атрибута у объекта? - hasattr (obj, attr\_name)  
if hasattr(sergey, 'name'):

    print(sergey.name)

Как проверить, что объект является экземпляром класса? isinstance(object, classinfo)

Как проверить, что класс А является подклассом В? - issubclass(class, classinfo)

Как применить функцию для каждого элемента последовательности? - map(function, iterable, ...)

Декоратор – это потерн проектирования, позволяет добавить новое поведение к существующему объекту без изменения структуры этого объекта.  
декораторы работают с декорируемыми объектами.

Вызвать декоратор можно явным образом либо при помощи синтаксического сахара (@)

При помощи @ лучше применять если функция без декоратора не используется.

Контекстный менеджер - это способ автоматического высвобождения ресурсов, не отвлекаясь на них. Допустим автоматически закрыть файл если забыли закрыть его.

Есть 2 способа написания контекстных менежеров:

- С помощью классов (внутри класса есть 2 волшебных метода \_\_enter\_\_, \_\_exit\_\_)

- С использованием contextlib

С помощью контекст менеджера:

from contextlib import contextmanager

@contextmanager

def context(\*args):

    resource = None

    try:

        resource = Resource()

        resource.open([1, 'ss', True])

        yield resource

    except:

        raise

    finally:

        if resource:

            resource.close()

if \_\_name\_\_ == '\_\_main\_\_':

    with context(1,2,3) as con:

        con.actions()

С помощью классов:

class ResourceWorker:

    def \_\_init\_\_(self, \*args):

        self.args = args

        self.resourse = None

    def \_\_enter\_\_(self):

        self.resourse = Resource()

        self.resourse.open(\*self.args)

        return self.resourse

    def \_\_exit\_\_(self, exc\_type, exc\_val, exc\_tb):

        if self.resourse:

            self.resourse.close()

if \_\_name\_\_ == '\_\_main\_\_':

    with ResourceWorker(1,2,3) as con:

        con.actions()

Можно ли одновременно использовать несколько менеджеров контекста? Можно?

Какой метод является связанным? - Если функция является атрибутом класса и к ней обращаются через экземпляры, они называются связанными методами. Привязанный метод - это метод, первым аргументом self Поскольку они зависят от экземпляра классов, они также известны как методы экземпляра.

Какой метод является несвязанным? - Методы, у которых нет экземпляра класса в качестве первого аргумента (self), называются несвязанными методами.

Какой метод является статическим? - Статические методы похожи на методы класса, но полностью привязаны к классу, а не к конкретным объектам. Доступ к ним осуществляется с помощью имен классов, ему не нужен определенный первый аргумент не self не cls

Какой метод является методом класса? - Методы класса привязаны к самому классу, а не его экземпляру. Они могут менять состояние класса, что отразится на всех объектах этого класса, но не могут менять конкретный объект. методы класса отмечаются декоратором @classmethod

Как определить публичный метод? - без одного или двух подчеркиваний вначале.

Как определить защищенный метод? – с одним подчеркиванием вначале.

Как определить приватный метод? – с двумя подчеркиваниями вначале.

Что такое исключение? - ещё один тип данных в python. Исключения необходимы для того, чтобы сообщать программисту об ошибках.

Возбуждение исключения? - Raise позволяет программисту принудительно вызвать указанное исключение.

Перехват и дальнейшая передача исключения –

Как обработать несколько исключений одновременно? - Python позволяет в одном блоке except обрабатывать сразу несколько типов исключений. В этом случае все типы исключения передаются в скобках except (ZeroDivisionError, ValueError)

назначение map() - функция, которая позволяет обрабатывать и преобразовывать все элементы в итерируемом объекте без использования явного цикла. полезен, когда вам нужно применить функцию преобразования к каждому элементу в коллекции или в массиве и преобразовать их в новый массив.

Назначение zip - создает итератор, который объединяет элементы из нескольких источников данных.

collections.namedtuple - Именованные кортеж, Именованные кортежи присваивают имя каждому значению элемента в кортеже для читаемости (import collections)

collections.Counter - Подсчет количества повторений элементов в последовательности (import collections) .

collections.defaultdict - Словарь со значениями по умолчанию.

Какие методы выполняют роль конструктора и деструктора? Def \_\_init\_\_(self), def \_\_del\_\_(self)

Какой пакет является Django-приложением? Django

Как подключить приложение к проекту? Приложение создается python manage.py startapp ice\_cream , после создания, в файле apps.py нужно скопировать название класса и добавить его в файл settings.py в список INSTALLED\_APPS.

Когда вычисляется Queryset? - это в основном список объектов, полученных из Model , путем компиляции запросов к базе данных.

Что значит, что QuerySets - ленивы? - запрос к базе данных выполняется не сразу, а только при необходимости - при оценке набора запросов. Это происходит, например, если вы вызываете его \_\_str\_\_метод при его печати, если вы вызываете list()его, или, что происходит в основном, вы выполняете итерацию по нему (for post in b..) . Эта ленивость должна избавить вас от выполнения ненужных запросов, а также позволяет, например, связывать наборы запросов и фильтры (вы можете фильтровать набор запросов так часто, как захотите).

Как получить количество строк в результате запроса?

>>> from django.db import connection

>>> print(connection.queries)

Для чего используются .values и .values\\_list -   
 values() Метод возвращает набор запросов, содержащий словари  
 values\_list()Метод возвращает набор запросов, содержащий кортежи

Создание миграций. Выполнение. Откат -

Makemigrations – создание  
migrate – запуск

migrate <приложение> \*\*\*\*\*\*\* - откат миграции

Как используются сигналы? -

Как собрать статику? - Задать настройку STATIC\_ROOT, собрать статику python manage.py collectstatic

Что такое Middleware? Примеры использования. - Middleware позволяет обрабатывать запросы из браузера, прежде чем они достигнут представления Django, а также ответы от представлений до того, как они возвращаются в браузер.

Настраивается в settings.py, можно описать в классе с магическим методом \_\_call\_\_

Примеры операций: HTTP аутентификация, управление сессиями, шифрование, валидация, рассылка оповещений, защита от межсайтовых запросо в.

Для чего используется HAVING? - Оператор HAVING определяет, какие группы будут включены в выходной результат, то есть выполняет фильтрацию групп. Применение HAVING во многом аналогично применению WHERE. Только есть WHERE применяется к фильтрации строк, то HAVING используется для фильтрации групп

Какие виды джоинов существуют? – оператор соединения таблиц

CROSS JOIN - перекрестное или декартово соединение

[INNER] JOIN - естественное или внутреннее соединение

LEFT [OUTER] JOIN - левое внешнее соединение

RIGHT [OUTER] JOIN - правое внешнее соединение

FULL [OUTER] JOIN - полное внешнее соединение

Что такое "нормализация"? – это процесс удаления избыточных данных. Дело в том, что избыточность данных создает предпосылки для появления различных аномалий, снижает производительность, и делает управление данными не гибким и не очень удобным.

Для чего нужны индексы и почему нельзя проиндексировать всё?   
 Индексы – дополнительная структура данных для ускорения работы запросов.  
 Они важны: для поиска, уникальности, сортировки, соединения таблиц.  
 Они не всегда полезны: малая селективность. Много весят, нужно время на расчет и долгая вставка, увеличивается время доступа к самому диску из за фрагментации.

Что делают данные конструкции? -  
 **CREATE INDEX** - создаёт индексы по указанному столбцу(ам) заданного отношения  
 **CREATE UNIQ INDEX** – создает индексы и контролирует уникальность индекса.  
 **CREATE CONCURRENTLY INDEX** – создаст индекс не устанавливая блокировок на чтение и запись.

***SELECT*** - указываем значение полей которые должны вернуться.  
 ***FROM*** – указываем название таблицы.  
 ***WHERE*** – указываем фильтрацию по строкам которые указали в select  
 ***BETWEEN***- поиск между. *(*WHERE release\_year BETWEEN 1980 AND 1990;)***IN*** – входит в список. (WHERE type IN ('Сериал', 'Фильм');)  
 ***LIKE*** – поиск строки по шаблону. (WHERE type LIKE 'Мульт%';)  
 -символы-«маски»: знак процента (**%**) заменяет любой набор символов;  
 -символ подчёркивания **\_** заменяет один любой символ (цифру, букву)  
 ***DISTINCT*** – для получения уникальных (неповторных) значений заданного столбца.  
 - SELECT DISTINCT type

**JOIN(INNER JOIN)** - Оператор INNER JOIN вернёт все записи из левой таблицы, которые имеют соответствующие записи в правой таблице.  
 *JOIN slogans ON movies.slogan\_id = slogans.id  
 JOIN types ON movies.type\_id = types.id;*  
**LEFT JOIN** - Данный запрос вернёт все записи из левой таблицы (FROM), независимо от того, имеются ли соответствующие им записи в правой таблице.  
**RIGHT JOIN** - В результирующей таблице полностью заполнены будут строки, для которых был найден ключ левой таблицы (FROM), соответствующий ключу правой таблицы (JPIN …). Также в результат будут добавлены все остальные строки правой таблицы. Для них поля, хранящие данные (FROM), будут иметь значение null  
**FULL JOIN** - выводятся все записи из объединяемых таблиц. Те записи, у которых запрошенные значения совпадают — выводятся парами, у остальных недостающее значение заменяется на NULL(None).  
 FULL JOIN slogans ON movies.slogan\_id = slogans.id;  
**CROSS JOIN** возвращает декартово произведение таблиц — каждая запись левой таблицы объединится с каждой записью правой. Параметр ON при запросах CROSS JOIN не применяется.

**UNION** - она объединяет данные из нескольких результирующих таблиц в одну:

**ORDER BY** – сортировка полученного результата  
 ORDER BY release\_year DESC – сортировка по убыванию  
 ORDER BY release\_year ASC; - сортировка по возврастанию(по умолчанию)

**GROUP BY** type; - объединяет в группы одинаковые объекты в заданной колонки.

**HAVING** – фильтрация групп

**LIMIT** 2; - ограничение количества строк

**LIMIT** 2 OFFSET 2; - ограничение количества строк и пропуск первых двух записей

SELECT ('столбцы (\* - для выбора всех столбцов); обязательно')

(могут применяться агрегатные функции COUNT, MIN, MAX, AVG и SUM; необязательно)

(и ключевое слово DISTINCT; необязательно)

FROM ('таблица; обязательно')

WHERE ('условие/фильтрация; необязательно')

GROUP BY ('столбец, по которому нужно сгруппировать данные; необязательно')

HAVING ('условие/фильтрация на уровне сгруппированных данных; необязательно')

ORDER BY ('столбец, по которому нужно ранжировать вывод; необязательно')

LIMIT ('сколько записей показывать; необязательно')

OFFSET ('сколько записей в выборке пропустить; необязательно')

Что такое VACUUM? - удаление мертвых строк и заморозка идентификаторов транзакций. Процесс можно запустить руками, но также он запускается и автоматически демоном AUTOVACUUM.

Перечислить и объяснить требования к транзакционной системе ACID - Требования ACID — набор требований, которые обеспечивают сохранность ваших данных.

1. Атомарность - гарантирует, что каждая транзакция будет выполнена полностью или не будет выполнена совсем. Не допускаются промежуточные состояния.
2. Согласованность - каждая успешная транзакция по определению фиксирует только допустимые результаты
3. Изолированность - Во время выполнения транзакции параллельные транзакции не должны оказывать влияния на её результат.
4. Надёжность - Если пользователь получил подтверждение от системы, что транзакция выполнена, он может быть уверен, что сделанные им изменения не будут отменены из-за какого-либо сбоя.

Типы данных(JS)-

-number для любых чисел: целочисленных или чисел с плавающей точкой; целочисленные значения ограничены диапазоном ±(253-1).

-bigint для целых чисел произвольной длины.

-string для строк. Строка может содержать ноль или больше символов, нет отдельного символьного типа.

-boolean для true/false.

-null для неизвестных значений – отдельный тип, имеющий одно значение null.

-undefined для неприсвоенных значений – отдельный тип, имеющий одно значение undefined.

-symbol для уникальных идентификаторов.

И один не является «примитивным» и стоит особняком:

-object для более сложных структур данных.

Как изменить последний коммит - git commit –amend

Как синхронизировать изменения с удаленной веткой

git remote

git fetch

Как синхронизировать изменения с удаленной веткой

git checkout

git pull

Как перенести коммит из одной ветки в другую?

git cherry-pick

Как откатить коммит?  
git reset \*\*\*\*\*\*\* - откат обновления (\*-первые 7 цифр)

Структурное программирование – парадигма программирования в основе которой лежит представление программы в виде иерархической структуры блоков. (Последовательность, цикл, ветвление)

Процедурный подход программирования – последовательный ввод инструкций (допустим “Переменных”), достоинство простота, недостаток дублирование кода.

Функциональный подход программирования – парадигма программирования в которой блок кода заключается в отдельную функцию, которую можно вызвать в любом месте программы. Достоинства – убирает повторение кода, вызов в любой части кода.

Композитный первичный ключ - PRIMARY KEY используется сочетание (director\_id, movie\_id) — эта пара уникальна в пределах таблицы. PRIMARY KEY (director\_id, movie\_id),