АКАДЕМИЯ МЕРКУРИ

.NET Backend Development

Вступительное задание

MERCURY DEVELOPMENT

2022

Основные моменты

Количество заданий

- Задания имеют разную сложность и направленность. Первые 2 обязательны к выполнению, два дополнительных не обязательны, их выполнение будет плюсом.

Качество выполнения

- Оценивается как корректность выполненного задания, так и качество кода (читаемость, расширяемость, следование основным принципам ООП).

Инструменты

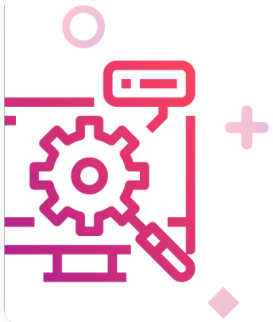
- Для выполнения задания по программированию вам понадобится интегрированная среда разработки Microsoft Visual Studio Community или иные инструменты на ваше усмотрение. Для работы с базами данных - MS SQL Server (любой редакции) и SQL Server Management Studio. Рекомендуем английские версии.

Структура

- Решение каждого задания (C# проект, SQL файл) должно лежать в соответствующем номеру задания каталоге (Solution1, Solution2, и т.д.).

Способ сдачи

- Каталоги с решениями следует запаковать в один архив (Имя Фамилия.zip) и отправить на электронную почту dotnet-internship@mercurydevelopment.com



Задание #1. ООП

Необходимо спроектировать и реализовать приложение для просмотра списка участников IT-мероприятия.

Заявки на участие в мероприятии независимо собирали несколько информационных сервисов. Каждый сервис экспортировал данные в свой формат. К заданию прикреплены файлы:

- Сервис №1 - participants.json
- Сервис №2 - participants.xml
- Сервис №3 - participants.csv

Требования:

1. Пользователь приложения должен иметь возможность:
    a. Постранично просматривать участников мероприятия. Размер страницы - 5 элементов.
    b. Искать участников по части фамилии или имени.

2. Любой результат выборки должен быть отсортирован по дате регистрации и представлен в виде следующей таблицы:

Имя	Фамилия	Дата регистрации	Поставщик
Владислав	Белов	20.10.2019 10:15	Сервис №1
...	...	...	...

3. Для отображения любых результатов данные необходимо объединить из всех вышеупомянутых источников.

4. Если один и тот же участник зарегистрировался несколько раз (имеется более чем в одном файле), то:

   a. Он не должен повторяться в списке.
   b. Из дубликатов должна быть выбрана запись с наиболее ранней датой регистрации.
   c. Дублированной считается запись, если фамилия и имя совпадает.

5. В разных источниках даты представлены в разных форматах. Необходимо привести к единому (указан в таблице).

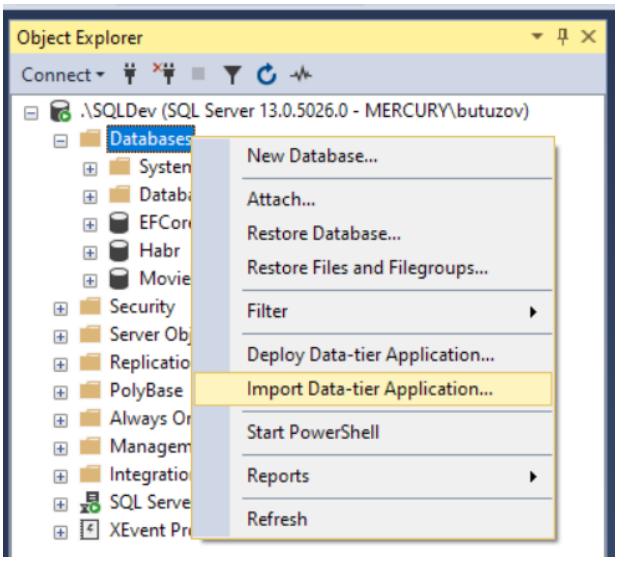
6. Задание должно быть реализовано в виде консольного приложения.

7. Список команд:

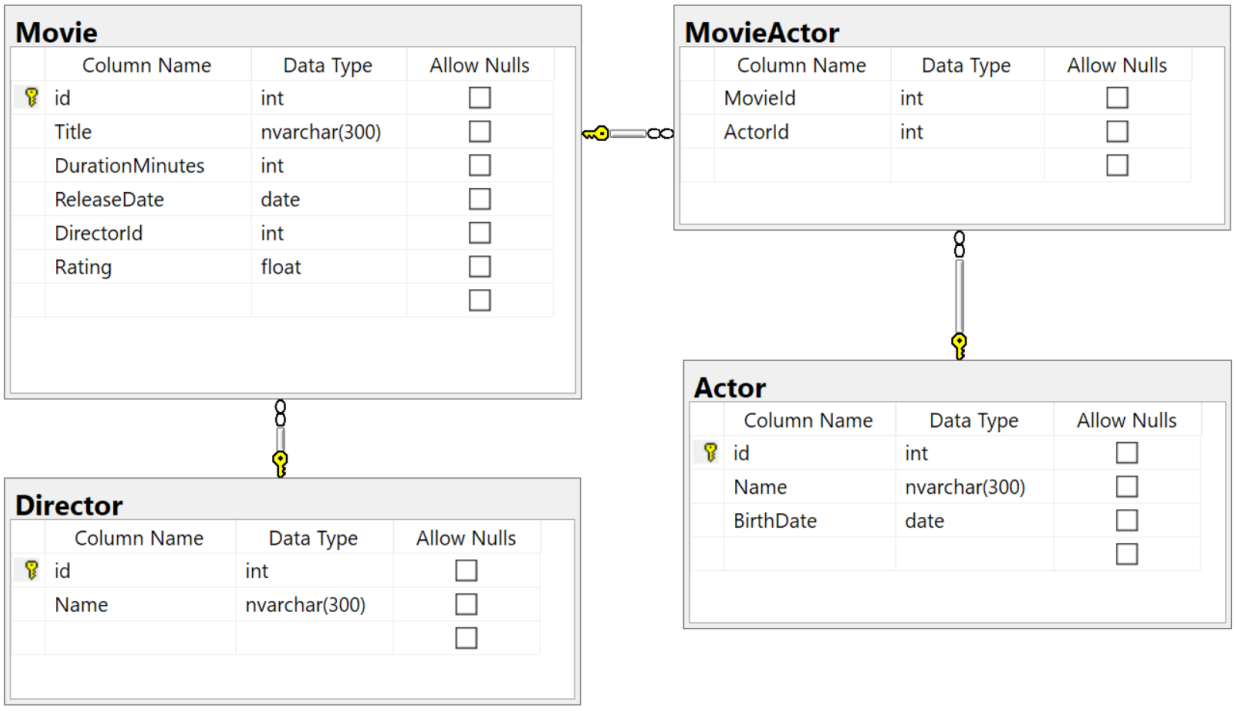
Действие	Пример команды
Получение указанной страницы списка	get-page 3
Поиск по списку	search “Ива”

Задание #2. SQL

К заданию прикреплен бэкап базы данных в формате bacpac. Используя SQL Server Management Studio необходимо импортировать его в вашу СУБД.



После импорта у вас появится база с именем MovieDB и следующей структурой:



Необходимо составить следующие SQL запросы:

1. Вывести имена режиссеров, для которых в БД забыли создать фильмы. Результат упорядочить по имени режиссера.

2. Вывести название фильма с наибольшим кол-вом актеров. В случае фильмов с одинаковым кол-вом актеров сортировка должна выполняться по названию фильма в алфавитном порядке.

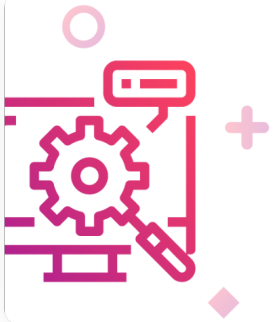
3. Вывести имена актеров, сыгравших в 9 эпизоде, но не сыгравших в 8 эпизоде Звездных Войн. Результат упорядочить по имени актера.

4. Вывести имя, дату рождения, число полных лет, знак зодиака актера и кол-во режиссеров, с которыми он работал. Условия для выборки:
    a. Актер хотя бы раз снимался в фильме с длительностью более 2 часов,
    b. Актер никогда не снимался в фильмах с рейтингом ниже 6,5.
    c. Актер работал как минимум с двумя разными режиссерами.

Результат упорядочить по дате рождения актера, начиная с самых молодых.
Для определения знака зодиака стоит использовать следующую таблицу:

Zodiac	Dates range
Aries	Mar. 21 – Apr. 19
Taurus	Apr. 20 – May 20
Gemini	May 21 – June 21
Cancer	June 22 – July 22
Leo	July 23 – Aug. 22
Virgo	Aug. 23 – Sept. 22
Libra	Sept. 23 – Oct. 23
Scorpio	Oct. 24 – Nov. 21
Sagittarius	Nov. 22 – Dec. 21

Capricorn	Dec. 22 – Jan. 19
Aquarius	Jan. 20 – Feb. 18
Pisces	Feb. 19 – Mar. 20



Дополнительное задание #1

Необходимо написать консольное приложение, удовлетворяющее следующим условиям.

1. Через консольный ввод принимаем положительное целочисленное значение.

2. Отображаем другое положительное целочисленное значение, которое составлено из тех же цифр, что и входное, но является следующем по величине.

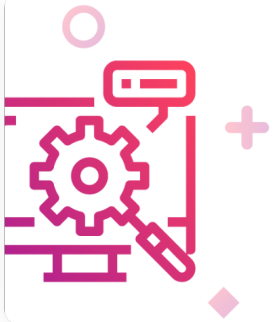
3. Если достигнуто максимальное значение, то возвращаем его же без изменения.

4. В Program.cs должна вызываться функция вида ulong GetNextNumber(ulong number).

5. В свою очередь при необходимости можно декомпозировать код в любые вспомогательные функции, которые будут вызываться из GetNextNumber.

Например:

Ввод	Вывод
127	172
172	217
217	271
271	712
712	721
721	721



Дополнительное задание #2

Необходимо написать консольное приложение, удовлетворяющее следующим условиям.

1. В качестве входных параметров пользователь вводит в консоль:
    a. Название города на английском языке, например - London;
    b. Единицу измерения температуры - Fahrenheit или Celsius;

```
The name of city:
London

The temperature scale (Fahrenheit or Celsius):
Celsius
```

2. В качестве результата приложение выводит следующую информацию:
    a. Название введенного города;
    b. Погодные условия (Rain, Snow, Extreme), с некоторым описанием;
    c. Температуру воздуха;
    d. Температура воздуха по критерию “ощущается как”:

```
The name of city:
London

The temperature scale (Fahrenheit or Celsius):
Celsius
========================================
City: London
Weather: Clouds(few clouds)
Temperature(Celsius): 37.96
Temperature feels like (Celsius): 29.62
```

3. В качестве источника данных необходимо использовать публичный API бесплатного сервиса прогноза погоды. Для интеграции с сервисом необходимо пройти регистрацию, после чего будет выдан API Key для вашего приложения. Ключ активируется в течение 2 часов.