# Задача 1

Имеется таблица Employees

CREATE TABLE Employees (

ID int NOT NULL IDENTITY(1,1) primary key,

TabNum int,

LastName nvarchar(50) NOT NULL,

FirstName nvarchar(50) NOT NULL,

Email nvarchar(250) NOT NULL UNIQUE,

DateOfBirth datetime2,

Gender char(1),

-- Еще двадцать столбцов

)

CREATE INDEX idx\_Employees\_TabNum ON Employees (TabNum)

Опишите, какие индексы объявлены в этой таблице.

## Ответ Задача 1:

ID – так как Primary Key

Email – так как UNIQUE

TabNum – так как целенаправленно создали

# Задача 2

В результате профилирования работы приложения в рабочей среде вы выяснили, что наиболее часто вызываются запросы вида

SELECT LastName, FirstName, Email

FROM Employees

WHERE TabNum = 1

Почему поиск с использованием индекса по полю TabNum для данного запроса – это хорошо?

Почему поиск с использованием индекса по полю TabNum для данного запроса – это плохо?

Что надо сделать, чтобы запрос с использованием индекса гарантированно давал преимущества по сравнению с неиспользованием индекса?

## Ответ Задача 2:

Хорошо: Поиск по индексу происходи быстрее

Плохо: Скорее всего, TabNum неуникальный индекс с низкой плотностью из-за чего смысла в нём мало.

Улучшения: В индексах по запрашиваемым аттрибутам смысла намного больше. Стоит создать либо 3 отдельных индекса для LastName, FirstName, Email. Либо составной индекс для всех трёх сразу.

# Задача 3

Мы добавили индекс

CREATE INDEX idx\_Employees\_DateOfBirth ON Employees (DateOfBirth)

Как он повлияет на скорость исполнения запроса, что можно изменить в индексе или запросе?

SELECT COUNT(\*)

FROM Employees

WHERE DATEPART(YEAR, DateOfBirth) = 1998

## Ответ задача 3:

Выборка будет происходить быстрее.

Возможно вынести Year в отдельный столбец, если where по году происходит довольно часто

# Задача 4

Мы добавили индекс

CREATE INDEX idx\_Employees\_LastName\_DateOfBirth

ON Employees (LastName, DateOfBirth)

Как он повлияет на скорость исполнения запроса, что можно изменить в индексе или запросе?

SELECT TOP 1 \*

FROM Employees

WHERE LastName = N'Иванов'

ORDER BY DateOfBirth DESC

## Ответ Задача 4:

По идее, исполнение запроса должно ускориться так как будет использоваться созданный нами композитный индекс по LastName и DateOfBirth соответственно.

Но мне кажется, что эффективнее было бы создать индексы отдельно для LastName и DateOfBirth. Как минимум потому что после создания нашего композитного индекса им сможет пользоваться только LastName, а DateOfBirth – нет.

То есть при запросе:

SELECT TOP 1 \*

FROM Employees

ORDER BY DateOfBirth DESC

DateOfBirth не сможет воспользоваться индексом.

# Задача 5

Мы добавили индекс

CREATE INDEX idx\_Employees\_LastName\_DateOfBirth

ON Employees (LastName, DateOfBirth)

Как он повлияет на скорость исполнения запроса, что можно изменить в индексе или запросе?

SELECT \*

FROM Employees

WHERE DateOfBirth > '2000-01-01'

## Ответ Задача 5:

Я не уверен, но либо никак, либо замедлит так как DateOfBirth не может самостоятельно пользоваться составным индексом.

Лучше сделать индекс только на DateOfBirth, остальные в данном запросе только увеличат нагрузку

# Задача 6

Мы добавили индекс

CREATE INDEX idx\_Employees\_Gender\_DateOfBirth

ON Employees (Gender, DateOfBirth)

Для ускорения запроса

SELECT DateOfBirth, COUNT(\*)

FROM Employees

WHERE Gender = 'M'

GROUP BY DateOfBirth

Затем запрос был изменен на

SELECT DateOfBirth, COUNT(\*)

FROM Employees

WHERE Gender = 'M'

AND LastName = N'Иванов'

GROUP BY DateOfBirth

Как это повлияет на скорость исполнения запроса, почему?

## Ответ Задача 6:

Должно ускорить так как в запросе появится больше уникальных значений.