# 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.

ID – primary key автоматически создаётся clustered индекс по этому полю

Email – создаётся unique индекс

# 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?

TabNum (табельный номер, как я понял) даёт высокую уникальность – то есть маленькая плотность поиска

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

При обычном сканировании идёт сканирование всей таблицы. TabNum = 1, скорее всего будет стоять в начале, следовательно поиск сравнение в этом случае займёт O(1), а не O(N).

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

Хранить большое кол-во значений, сохранять низкую плотность, не ставить индексация на атрибуты, значения которых часто меняются

# 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

Индексация повысит скорость поиска.

Что именно можно поменять, не совсем понимаю

# 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

Индексация однозначно повышает скорость исполнения запроса (низкая плотность).

Менять что-либо в запросе и индексе - не вижу смысла

# 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'

Скорость работы повысится, ибо указанные два атрибута для индексации обладают высокой селективностью и низкой плотностью.

Чтоб ещё сильнее повысить скорость работы, я бы поменял порядок для индекса:

ON Employees (DateOfBirth, LastName)

# 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?

Делать индексацию по полю Gender (очень высокая плотность) – лишнее нагромождение, которое создаст лишние структуры данных, к тому же замедлит работу программы.

Во втором случае мы делаем поиск по куче из-за атрибута Lastname (его бы я и индексировал).