Sumário

[Avaliacao 2](#_Toc144759391)

[QUERY PARA AVALIAR NOTAS DOS QUARTOS 2](#_Toc144759392)

[QUERY PARA ANALISAR OS PRINCIPAIS TIPOS DE AVALIAÇÃO 2](#_Toc144759393)

[ÍNDICE: Primary Key (tipo\_contato, id\_reserva) 3](#_Toc144759394)

[1º SELECT 3](#_Toc144759395)

[2º SELECT 3](#_Toc144759396)

[Resultado 4](#_Toc144759397)

[Quarto 4](#_Toc144759398)

[QUERY PARA ANALISAR O USO DO QUARTO 4](#_Toc144759399)

[ÍNDICE: Primary Key (id\_quarto) 5](#_Toc144759400)

[Resultado 5](#_Toc144759401)

[Reserva 6](#_Toc144759402)

[QUERY PARA ANALISAR O TIPO DE HÓSPEDE E SEUS PRINCIPAIS SERVIÇOS SOLICITADOS 6](#_Toc144759403)

[QUERY PARA DETERMINAR MOVIMENTO E LUCRO DE CADA ÉPOCA 7](#_Toc144759404)

[QUERY PARA DETERMINAR PRINCIPAIS CANCELAMENTOS, CONTATOS E PAGAMENTOS 8](#_Toc144759405)

[ÍNDICE: Primary Key (id\_reserva) 10](#_Toc144759406)

[1º SELECT 10](#_Toc144759407)

[2º SELECT 10](#_Toc144759408)

[3º SELECT 11](#_Toc144759409)

[Resultado 12](#_Toc144759410)

[Funcionario 12](#_Toc144759411)

[QUERY PARA MOSTRAR MÉDIAS DOS FUNCIONÁRIOS POR FUNÇÃO 12](#_Toc144759412)

[QUERY PARA IDENTIFICCAR A QUANTIDADE DE CHECKINS E CHEKOUTS REALIZADOS 13](#_Toc144759413)

[TABELAS DESCRITIVAS 14](#_Toc144759414)

Avaliacao

# QUERY PARA AVALIAR NOTAS DOS QUARTOS

Query para identificar a avaliação dada tanto para o quarto quanto para a estadia

SELECT

DISTINCT (qu.id\_quarto + 100) quarto,

ROUND(AVG(av.conforto), 2) conforto,

ROUND(AVG(av.custo\_beneficio), 2) custo\_beneficio,

ROUND(AVG(av.funcionarios), 2) 'funcionários',

ROUND(AVG(av.localizacao), 2) 'localização',

ROUND(AVG(av.limpeza), 2) limpeza,

ROUND((AVG(av.conforto) + AVG(av.custo\_beneficio) + AVG(av.localizacao) + AVG(av.limpeza) + AVG(av.funcionarios))/5, 2) total

FROM [scComodo].[Avaliacao] av

INNER JOIN [scReserva].[Reserva] rs

ON rs.id\_reserva = av.id\_reserva

INNER JOIN [scComodo].[Quarto] qu

ON qu.id\_quarto = rs.id\_quarto

GROUP BY qu.id\_quarto

ORDER BY quarto

GO

# QUERY PARA ANALISAR OS PRINCIPAIS TIPOS DE AVALIAÇÃO

Identifica os principais meios de contato usados pelos usuários ao dar uma avaliação

SELECT

ct.descricao tipo\_contato,

COUNT(ct.descricao) total\_avaliacao

FROM [scComodo].[Avaliacao] av

INNER JOIN [scReserva].[Tipo\_Contato] ct

ON ct.id\_tp\_contato = av.tipo\_contato

GROUP BY ct.descricao

ORDER BY total\_avaliacao DESC

## ÍNDICE: Primary Key (tipo\_contato, id\_reserva)

### 1º SELECT

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Reserva'. Scan count 1, logical reads 9, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Avaliacao'. Scan count 1, logical reads 11, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Quarto'. Scan count 1, logical reads 2, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Diagrama

Descrição gerada automaticamente

Principais custos: Sort e Inner Join

Tempo total: 0.016s

### 2º SELECT

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Tipo\_Contato'. Scan count 0, logical reads 12, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Avaliacao'. Scan count 1, logical reads 10, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Gráfico, Gráfico de dispersão

Descrição gerada automaticamente

Principais custos: Sort e seek no índice

Tempo total: menos de 0.001s

### Resultado

Resultado: foi decidido manter o formato atual de índice, pois atua perfeitamente tanto para o primeiro select quanto para o segundo, alterando apenas o fillfactor para 90

Na 1ª query, após a implementação do fillfactor e rebuild do índice, foi verificado a diminuição de logical reads na tabela “avaliação” de 11 para 10 e um custo de 0.001s a menos no inner join

Foi constado de que não há uma necessidade na adoção de um índice não clustered, o índice atual já supre todas as necessidades.

Quarto

# QUERY PARA ANALISAR O USO DO QUARTO

Seu objetivo é ver detalhes básicos do quarto e a necessidade de adição de novas camas

SELECT

DISTINCT (qu.id\_quarto + 100) quarto,

tq.descricao categoria,

qu.valor\_diaria diaria,

COUNT(rv.id\_quarto) reservas,

AVG(rs.id\_servico) media\_cama\_extra

FROM [scComodo].[Quarto] qu

INNER JOIN [scComodo].[Tipo\_Quarto] tq

ON tq.id\_tp\_quarto = qu.tipo\_quarto

INNER JOIN [scReserva].[Reserva] rv

ON rv.id\_quarto = qu.id\_quarto

INNER JOIN [scReserva].[Reserva\_Servico] rs

ON rs.id\_reserva = rv.id\_reserva

WHERE rs.id\_servico = 5

GROUP BY qu.id\_quarto, qu.valor\_diaria, tq.descricao

ORDER BY quarto

ÍNDICE: Primary Key (id\_quarto)

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Reserva'. Scan count 1, logical reads 9, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Avaliacao'. Scan count 1, logical reads 10, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Quarto'. Scan count 1, logical reads 2, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Gráfico

Descrição gerada automaticamente com confiança média

Principais custos: Inner joins

Tempo total: 0.056s

Resultado

Resultado: foi decidido manter o formato atual de índice, pois atua perfeitamente para o select desejado, alterando apenas o fillfactor para 90

Após a implementação do fillfactor e rebuild do índice, foi verificado a diminuição de logical reads e scan Count e um custo de 0.045s a menos na realização da query

Foi constado de que não há uma necessidade na adoção de um índice não clustered, o índice atual já supre todas as necessidades.

Reserva

# QUERY PARA ANALISAR O TIPO DE HÓSPEDE E SEUS PRINCIPAIS SERVIÇOS SOLICITADOS

A query mostra os tipos de hóspedes e quais foram os serviços mais e menos solicitados por eles

WITH HospedeCTE AS (

SELECT

rv.id\_reserva reserva,

th.descricao descricao

FROM [scReserva].[Reserva] rv

INNER JOIN [scHospede].[Hospede] hp

ON rv.hospede\_principal = hp.id\_hospede

INNER JOIN [scHospede].[Hospede\_Classe] hc

ON hc.id\_hospede = hp.id\_hospede

INNER JOIN [scHospede].[Tipo\_Hospede] th

ON th.id\_tp\_hospede = hc.id\_tp\_hospede

WHERE rv.cancelamento IS NULL

),

ServicoCTE AS(

SELECT

rv.id\_reserva reserva,

sv.descricao descricao

FROM [scReserva].[Reserva] rv

INNER JOIN [scReserva].[Reserva\_Servico] rs

ON rs.id\_reserva = rv.id\_reserva

INNER JOIN [scFuncionario].[Servico] sv

ON sv.id\_servico = rs.id\_servico

WHERE rv.cancelamento IS NULL

)

SELECT

hcte.descricao Tipo\_Hospede,

scte.descricao Tipo\_Servico,

count(scte.descricao) Solicitacoes

FROM HospedeCTE hcte

INNER JOIN ServicoCTE scte

ON scte.reserva = hcte.reserva

GROUP BY hcte.descricao, scte.descricao

ORDER BY hcte.descricao, Solicitacoes DESC

# QUERY PARA DETERMINAR MOVIMENTO E LUCRO DE CADA ÉPOCA

A query analisa as reservas e verifica a média de dias, a média e o total de hóspedes e a média e o total de lucro, agrupando-os em estações do ano e calculando apenas estadias não canceladas

WITH reservaCTE AS (

SELECT

id\_reserva,

num\_hospede,

valor\_total,

DATEDIFF(DAY, data\_checkin, data\_checkou) Dias,

CASE

WHEN MONTH(data\_checkin) <= 3 THEN 'Verão'

WHEN MONTH(data\_checkin) <= 6 THEN 'Primavera'

WHEN MONTH(data\_checkin) <= 9 THEN 'Inverno'

WHEN MONTH(data\_checkin) <= 12 THEN 'Outono'

END Periodo

FROM [scReserva].[Reserva]

WHERE cancelamento IS NULL)

SELECT

Periodo,

COUNT(id\_reserva) Reservas,

AVG(Dias) Media\_Dias,

AVG(num\_hospede) Media\_Hospedes,

SUM(num\_hospede) Total\_Hospedes,

CONCAT('R$ ', FORMAT(AVG(valor\_total), 'N2')) Media\_Lucro,

CONCAT('R$ ', FORMAT(SUM(valor\_total), 'N2')) Lucro\_Total

FROM reservaCTE

GROUP BY Periodo

ORDER BY Reservas DESC

# QUERY PARA DETERMINAR PRINCIPAIS CANCELAMENTOS, CONTATOS E PAGAMENTOS

WITH CancelamentoCTE AS(

SELECT

t\_can.id\_tp\_cancelamento,

t\_can.descricao Cancelamento,

COUNT(rv.cancelamento) Contagem\_Can

FROM [scReserva].[Tipo\_Cancelamento] t\_can

INNER JOIN [scReserva].[Reserva] rv

ON rv.cancelamento = t\_can.id\_tp\_cancelamento

GROUP BY t\_can.id\_tp\_cancelamento, t\_can.descricao

),

ContatoCTE AS(

SELECT

t\_con.id\_tp\_contato,

t\_con.descricao Contato,

COUNT(rv.tipo\_contato) Contagem\_Con

FROM [scReserva].[Tipo\_Contato] t\_con

INNER JOIN [scReserva].[Reserva] rv

ON rv.tipo\_contato = t\_con.id\_tp\_contato

GROUP BY t\_con.id\_tp\_contato, t\_con.descricao

),

PagamentoCTE AS(

SELECT

t\_pag.id\_tp\_pagamento,

t\_pag.descricao Pagamento,

COUNT(rv.tipo\_pagamento) Contagem\_Pag

FROM [scReserva].[Tipo\_Pagamento] t\_pag

INNER JOIN [scReserva].[Reserva] rv

ON rv.tipo\_pagamento = t\_pag.id\_tp\_pagamento

GROUP BY t\_pag.id\_tp\_pagamento, t\_pag.descricao

)

SELECT

Cancelamento,

Contagem\_Can,

COALESCE(Contato, '-') Contato,

COALESCE(Contagem\_Con, '-') Contagem\_Con,

COALESCE(Pagamento, '-') Pagamento,

COALESCE(Contagem\_Pag, '-') Contagem\_Pag

FROM CancelamentoCTE cte\_can

LEFT JOIN ContatoCTE cte\_con

ON cte\_con.id\_tp\_contato = cte\_can.id\_tp\_cancelamento

LEFT JOIN PagamentoCTE cte\_pag

ON cte\_pag.id\_tp\_pagamento = cte\_can.id\_tp\_cancelamento

ÍNDICE: Primary Key (id\_reserva)

1º SELECT

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Reserva\_Servico'. Scan count 1, logical reads 4, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Reserva'. Scan count 2, logical reads 18, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Hospede\_Classe'. Scan count 3, logical reads 9, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Tipo\_Hospede'. Scan count 1, logical reads 2, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Servico'. Scan count 1, logical reads 2, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Diagrama, Esquemático

Descrição gerada automaticamente

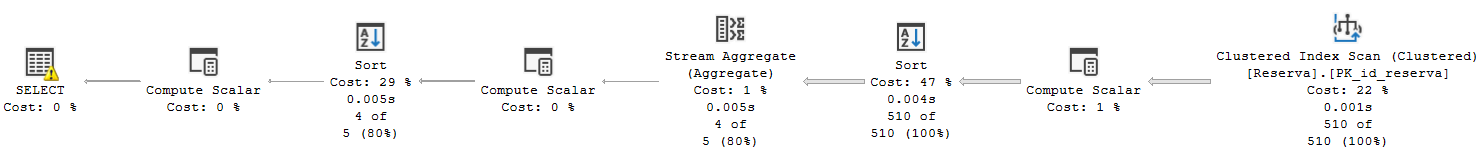
Principais custos: Hash Match e Inner joins

Tempo total: 0.029s

2º SELECT

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Reserva'. Scan count 1, logical reads 9, physical reads 1, page server reads 0, read-ahead reads 7, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.



Principais custos: Sort e Scan

Tempo total: 0.015s

3º SELECT

Table 'Reserva'. Scan count 11, logical reads 99, physical reads 1, page server reads 0, read-ahead reads 7, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Tipo\_Pagamento'. Scan count 0, logical reads 16, physical reads 1, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Tipo\_Contato'. Scan count 0, logical reads 16, physical reads 1, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Tipo\_Cancelamento'. Scan count 1, logical reads 2, physical reads 1, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Diagrama, Esquemático

Descrição gerada automaticamente

Principais custos: Hach Matches e Scans

Tempo total: 0.015s

Resultado

Resultado: foi decidido manter o formato atual de índice, pois atua perfeitamente para todos os selects, alterando apenas o fillfactor para 90

Foi realizado um ALTER INDEX nas tabelas associativas usando os seguintes códigos:

ALTER INDEX [PK\_hospede\_tipos] ON [scHospede].[Hospede\_Classe] REORGANIZE;

ALTER INDEX [PK\_reserva\_servico] ON [scReserva].[Reserva\_Servico] REORGANIZE;

ALTER INDEX [PK\_hospede\_reserva] ON [scHospede].[Estadia] REORGANIZE;

Após a implementação do fillfactor e rebuild do índice, foi verificado:

1º select: A diminuição de 2 ´logical reads´ em Hospede\_Clasee e uma diminuição de 0.007s no plano de execução

2º select: uma diminuição de 0.007s no plano de execução

3º select: uma diminuição de 0.008s no plano de execução

Foi constado de que não há uma necessidade na adoção de um índice não clustered, o índice atual já supre todas as necessidades, pois a única semelhança entre os 3 selects é o id\_reserva (Clustered)

Funcionario

# QUERY PARA MOSTRAR MÉDIAS DOS FUNCIONÁRIOS POR FUNÇÃO

Query informando, por profissão, o numero de funcionários, o numero de homens e mulheres e a quantidade de trabalhadores por faixa etária

SELECT

ff.descricao Profissao,

COUNT(fn.id\_funcionario) Num\_Funcionarios,

SUM(CASE WHEN fn.sexo = 'm' THEN 1 ELSE 0 END) Homens,

SUM(CASE WHEN fn.sexo = 'f' THEN 1 ELSE 0 END) Mulheres,

SUM(CASE WHEN DATEDIFF(YEAR, dt\_nascimento, '2023/09/04') BETWEEN 18 AND 30 THEN 1 ELSE 0 END) '18-30',

SUM(CASE WHEN DATEDIFF(YEAR, dt\_nascimento, '2023/09/04') BETWEEN 31 AND 45 THEN 1 ELSE 0 END) '31-45',

SUM(CASE WHEN DATEDIFF(YEAR, dt\_nascimento, '2023/09/04') BETWEEN 46 AND 60 THEN 1 ELSE 0 END) '46-60',

SUM(CASE WHEN DATEDIFF(YEAR, dt\_nascimento, '2023/09/04') > 60 THEN 1 ELSE 0 END) '60+'

FROM [scFuncionario].[Funcionario] fn

INNER JOIN [scFuncionario].[Funcao\_Funcionario] ff

ON ff.id\_funcao = fn.tipo\_funcao

GROUP BY ff.descricao

ORDER BY Profissao

# QUERY PARA IDENTIFICAR A QUANTIDADE DE CHECKINS E CHEKOUTS REALIZADOS POR FUNCIONÁRIO

SELECT

fn.id\_funcionario Registro,

ff.descricao Profissao,

CONCAT(fn.nome, ' ', fn.sobrenome) Funcionario,

SUM(CASE WHEN rv.funcion\_checkin = fn.id\_funcionario THEN 1 ELSE 0 END) Checkins,

SUM(CASE WHEN rv.funcion\_checkout = fn.id\_funcionario THEN 1 ELSE 0 END) Checkouts

FROM [scFuncionario].[Funcionario] fn

INNER JOIN [scFuncionario].[Funcao\_Funcionario] ff

ON ff.id\_funcao = fn.tipo\_funcao

LEFT JOIN [scReserva].[Reserva] rv

ON rv.funcion\_checkin = fn.id\_funcionario OR

rv.funcion\_checkout = fn.id\_funcionario

WHERE

fn.tipo\_funcao = 1 OR

fn.tipo\_funcao = 8

GROUP BY fn.id\_funcionario, ff.descricao, CONCAT(fn.nome, ' ', fn.sobrenome)

ORDER BY Profissao, Registro, Funcionario

ÍNDICE: Primary Key (id\_funcionario)

1º SELECT

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Funcao\_Funcionario'. Scan count 0, logical reads 22, physical reads 1, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Funcionario'. Scan count 1, logical reads 16, physical reads 1, page server reads 0, read-ahead reads 14, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.Gráfico, Gráfico de dispersão

Descrição gerada automaticamente

Principais custos: Hash Matche e Sort

Tempo total: 0.020s

2º SELECT

Table 'Reserva'. Scan count 1, logical reads 9, physical reads 1, page server reads 0, read-ahead reads 7, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Worktable'. Scan count 1, logical reads 2726, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Funcionario'. Scan count 1, logical reads 16, physical reads 1, page server reads 0, read-ahead reads 14, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Funcao\_Funcionario'. Scan count 1, logical reads 2, physical reads 1, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Diagrama

Descrição gerada automaticamente com confiança média

Principais custos: Nested Loops

Tempo total: 0.400s

Resultado

Resultado: foi decidido manter o formato atual de índice, pois atua perfeitamente para o select desejado, alterando apenas o fillfactor para 90

Após a implementação do fillfactor e rebuild do índice, foi verificado:

1º select: O aumento de 2 ´logical reads´ em Funcionario e uma diminuição de 0.004s no plano de execução

2º select: : O aumento de 2 ´logical reads´ em Funcionario e uma diminuição de 0.037s no plano de execução

Foi constado de que não há uma necessidade na adoção de mais um índice não clustered, a tabela possui o índice não clustered UK\_cpf\_func, além do índice clustered atual que já supre todas as necessidades.

Hospede

QUERY PARA MOSTRAR MÉDIAS DOS FUNCIONÁRIOS POR CLASSIFICAÇÃO

SELECT

ht.descricao,

SUM(CASE WHEN LOWER(ep.pais) = 'brasil' THEN 1 ELSE 0 END) Brasileiro,

SUM(CASE WHEN LOWER(ep.pais) <> 'brasil' THEN 1 ELSE 0 END) Estrangeiro,

SUM(CASE WHEN hs.sexo = 'm' THEN 1 ELSE 0 END) Homens,

SUM(CASE WHEN hs.sexo = 'f' THEN 1 ELSE 0 END) Mulheres,

SUM(CASE WHEN DATEDIFF(YEAR, hs.dt\_nascimento, '2023/09/04') < 18 THEN 1 ELSE 0 END) '-18',

SUM(CASE WHEN DATEDIFF(YEAR, hs.dt\_nascimento, '2023/09/04') BETWEEN 18 AND 30 THEN 1 ELSE 0 END) '18 - 30',

SUM(CASE WHEN DATEDIFF(YEAR, hs.dt\_nascimento, '2023/09/04') BETWEEN 31 AND 45 THEN 1 ELSE 0 END) '31 - 45',

SUM(CASE WHEN DATEDIFF(YEAR, hs.dt\_nascimento, '2023/09/04') BETWEEN 46 AND 60 THEN 1 ELSE 0 END) '46 - 60',

SUM(CASE WHEN DATEDIFF(YEAR, hs.dt\_nascimento, '2023/09/04') > 60 THEN 1 ELSE 0 END) '60+',

COUNT(hs.id\_hospede) Total

FROM [scHospede].[Hospede] hs

INNER JOIN [scPessoa].[Endereco\_Pessoa] ep

ON ep.id\_endereco = hs.id\_endereco

INNER JOIN [scHospede].[Hospede\_Classe] hc

ON hc.id\_hospede = hs.id\_hospede

INNER JOIN [scHospede].[Tipo\_Hospede] ht

ON ht.id\_tp\_hospede = hc.id\_tp\_hospede

GROUP BY ht.descricao

ÍNDICE: Primary Key (id\_hospede)

1º SELECT

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Endereco\_Pessoa'. Scan count 1, logical reads 13, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Hospede'. Scan count 1, logical reads 16, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Hospede\_Classe'. Scan count 3, logical reads 7, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Table 'Tipo\_Hospede'. Scan count 1, logical reads 2, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.

Diagrama, Esquemático

Descrição gerada automaticamente

Principais custos: Hash Matches

Tempo total: 0.038s

Resultado

Resultado: foi decidido manter o formato atual de índice, pois atua perfeitamente para o select desejado, alterando apenas o fillfactor para 90

Após a implementação do fillfactor e rebuild do índice, foi verificado o aumento de 2 ´logical reads´ em Hospede e uma diminuição de 0.004s no plano de execução

Foi constado de que não há uma necessidade na adoção de mais um índice não clustered, a tabela possui o índice não clustered UK\_cpf\_func, além do índice clustered atual que já supre todas as necessidades.

TABELAS DESCRITIVAS

Nas tabelas Tipo\_Hospede, Funcao\_Funcionario, Tipo\_Quarto, Tipo\_Pagamento, Tipo\_Cancelamento, Tipo\_Contato e Servico não foi visto uma necessidade de alteração e/ou manipulação. As tabelas citadas são descritivas, contendo apenas uma Primary Key (Identity e Índice Clustered) e uma descrição (com exceção de Servico, que conta com a coluna ‘valor’). Seus índices clustereds atuais já suprem as necessidades.

Diagrama, Esquemático

Descrição gerada automaticamente