

FACULDADES FACCAT

ADRIAN HIDEKI DOS SANTOS

SOFTWARE DE OTIMIZAÇÃO DE BANCO DE DADOS

**TUPÃ
2020**

ADRIAN HIDEKI DOS SANTOS

SOFTWARE DE OTIMIZAÇÃO DE BANCO DE DADOS

Trabalho de conclusão de curso apresentado ao curso de Sistemas de Informação das Faculdades FACCAT, como requisito para o exame de defesa e obtenção do título de Bacharel em Sistemas de Informação. **Orientador(a):** Esp. Ewerton Silva.

**TUPÃ
2020**

ADRIAN HIDEKI DOS SANTOS

SOFTWARE DE OTIMIZAÇÃO DE BANCO DE DADOS

Trabalho de Conclusão de Curso apresentado à Faculdades FACCAT, como parte dos requisitos necessários para obtenção do título de Bacharel em Sistemas de Informação.

BANCA EXAMINADORA

Prof.º Adriano de Oliveira Cipriano

Prof.º Ms. José Marcelo Pereira da Silva

Prof.ª Dr.ª Patrícia da Silva Moreno e Souza (Orientadora)

TUPÃ
2020

RESUMO

Atualmente um dos requisitos mais importantes na operação de um software é o tempo de execução de determinada ação no sistema, onde a atividade deve ocorrer no menor tempo possível. Porém nem sempre todos os processos em um sistema são assim, podendo até demorar horas para concluir uma ação no sistema, gerando insatisfação ao usuário. O objetivo deste trabalho é disponibilizar uma aplicação que realize otimizações no banco de dados. A aplicação foi desenvolvida em C#, utilizando como banco de dados o SQL Server. Inicialmente no desenvolvimento de sistemas já existiam boas práticas para banco de dados, começando pela normalização de dados, que define algumas regras de estruturação de dados, que permitem a otimização nas operações que envolvem as tabelas afetadas, melhorando operações de consulta e inserção de dados. Com um conjunto de boas práticas para banco de dados, foi possível criar uma aplicação que analise a estruturas do banco de dados e realize sugestões de melhorias, que devem otimizar as operações e evidenciar processos com problemas de performance. Com o desenvolvimento desta aplicação foi possível avaliar a importância da performance dos processos de software, de forma que possibilite as aplicações atuais retornem as informações no menor tempo possível.

Palavras-chave: Banco de dados; Performance; Normalização; SQL Server;

ABSTRACT

Currently, one of the most important requirements in the operation of a software is the execution time of a certain action in the system, where the activity must occur in the shortest possible time. However, not all processes in a system are always like this, and it may even take hours to complete an action in the system, generating user dissatisfaction. The objective of this work is to provide an application that performs optimizations in the database. The application was developed in C #, using SQL Server as database. Initially in the development of systems, there were already good practices for databases, starting with data normalization, which defines some data structuring rules, which allow optimization in the operations involving the affected tables, improving query and data insertion operations. With a set of good practices for the database, it was possible to create an application that analyzes the database structures and makes suggestions for improvements, which should optimize operations and highlight processes with performance problems. With the development of this application, it was possible to evaluate the importance of the performance of software processes, in a way that allows current applications to return information in the shortest possible time.

Keywords: Database; Performance; Normalization; SQL Server;

LISTA DE FIGURAS

Figura 1 - Tabelas de um Banco de Dados.....	13
Figura 2 - Edgar Frank Codd.....	14
Figura 3 - Página de dados SQL Server.....	22
Figura 4 - Extensão de páginas.....	23
Figura 5 - Sintaxe de criação de índices.....	24
Figura 6 - Modelo de entidade e relacionamento.....	26
Figura 7 - Diagrama de Classe.....	34
Figura 8 - Diagrama de caso de uso.....	35
Figura 9 - Tela de Cadastro de Usuário.....	36
Figura 10 - Tela de Login.....	37
Figura 11 - Tela de Configuração dos Ambientes.....	38
Figura 12 - Tela de Seleção de ambientes configurados.....	39
Figura 13 - Tela de seleção de banco de dados.....	40
Figura 14 - Tela de visualização de configurações.....	41
Figura 15 - Tela de desfragmentação de índices.....	42
Figura 16 - Tela de atualização de estatísticas.....	43
Figura 17 - Tela de índices ausentes.....	44
Figura 18 - Tela de Sessões.....	46
Figura 19 - Tela de Detalhes da Sessão.....	46
Figura 20 - Tela de Edição de Índices.....	47
Figura 21 - Tela de Consultas.....	48
Figura 22 - Tela de Detalhes da Consulta.....	49
Figura 23 - Tela de Edição de Usuários.....	50
Figura 24 - Tela de Índices Ausentes.....	55
Figura 25 - Relatório de Índices Ausentes.....	55
Figura 26 - Instruções SQL para gerar o relatório de índices ausentes.....	56
Figura 27 - Tela de Atualização de Estatísticas.....	57
Figura 28 - Relatório de Atualização de Estatísticas.....	57
Figura 29 - Instruções SQL para gerar o relatório de atualização de estatísticas.....	58
Figura 30 - Tela de Índices Fragmentados.....	59
Figura 31 - Relatório de Índices Fragmentados.....	59
Figura 32 - Instruções SQL para gerar o relatório de índices fragmentados.....	60

Figura 33 - Tela de Conexões ativas.....	61
Figura 34 - Relatório de conexões ativas.....	61
Figura 35 - Instruções SQL para gerar o relatório de conexões ativas.....	62
Figura 36 - Tela de Consultas.....	63
Figura 37 - Relatório de Consultas.....	63
Figura 38 - Instruções SQL para gerar o relatório de consultas.....	64

LISTA DE TABELAS

Tabela 1 - Exemplo de tabela antes da primeira forma.....	17
Tabela 2 - Exemplo de tabela com a primeira forma normal aplicada.....	17
Tabela 3 - Tabela de vendas antes da segunda forma.....	18
Tabela 4 - Tabela de Produtos segunda forma.....	19
Tabela 5 – Tabela de Venda x Produto.....	19
Tabela 6 - Tabela de Servidores.....	27
Tabela 7 - Tabela de configurações do servidor.....	27
Tabela 8 - Tabela de Configurações.....	27
Tabela 9 - Tabela de Banco de dados.....	28
Tabela 10 - Tabela de configurações do banco de dados.....	28
Tabela 11 - Tabela de Grupo de Arquivos.....	29
Tabela 12 - Tabela de arquivos.....	29
Tabela 13 - Tabela de tabelas.....	30
Tabela 14 - Tabela de restrições.....	31
Tabela 15 - Tabela de estatísticas.....	32
Tabela 16 - Tabela de índices.....	32
Tabela 17 - UC001 Autenticar.....	35
Tabela 18 - UC002 Autenticar.....	36
Tabela 19 - UC003 Configurar ambiente.....	38
Tabela 20 - UC004 Selecionar Ambiente.....	39
Tabela 21 - UC005 Selecionar Banco de dados.....	39
Tabela 22 - UC006 Visualizar Configurações.....	40
Tabela 23 - UC007 Gerar relatório de índices fragmentados.....	41
Tabela 24 - UC008 Atualizar estatísticas.....	42
Tabela 25 - UC009 Índices Ausentes.....	44
Tabela 26 - UC010 Monitorar Conexões.....	45
Tabela 27 - UC011 Editar Índices.....	47
Tabela 28 - UC012 Monitorar Consultas.....	47
Tabela 29 - UC013 Editar Usuários.....	49
Tabela 30 - Permissões de acesso.....	53

SUMÁRIO

1. Introdução.....	10
1.1. Definição do problema.....	10
1.2. Objetivos.....	11
1.2.1. Objetivo Geral.....	11
1.2.2. Objetivo Específico.....	11
1.3. Justificativa.....	11
1.4. Metodologia.....	12
2. Fundamentação teórica.....	12
2.1. Introdução ao banco de dados.....	12
2.2. SGBD.....	15
2.3. Melhores práticas de banco de dados.....	16
2.4. Normalização de dados.....	16
2.4.1. Primeira forma.....	17
2.4.2. Segunda forma.....	18
2.4.3. Terceira forma.....	19
2.4.4. Quarta forma.....	19
2.4.5. Quinta forma.....	20
2.5. SQL Server.....	21
2.6. Indexação de tabelas e estrutura de dados.....	22
3. Desenvolvendo o projeto proposto.....	26
3.1. Diagramas UML.....	26
3.1.1. MER - Modelo de Entidade e Relacionamento.....	26
3.1.2. Dicionário de dados.....	27
3.1.3. Diagrama de Classes.....	34
3.1.4. Diagrama de Casos de Uso.....	35
3.1.4.1. Descrição dos Casos de Uso.....	35

3.2. Requisitos não funcionais.....	50
3.2.1. Manutenção.....	51
3.2.2. Suporte.....	51
3.2.3. Infraestrutura.....	51
3.2.4. Segurança.....	52
3.3. Métodos para controle de segurança do sistema.....	52
3.3.1. Controle de Segurança Lógica.....	53
3.3.2. Plano de Contingência.....	54
3.4. Layout dos Relatórios.....	54
3.4.1. Relatório de índices ausentes.....	54
3.4.1.1. Tela do relatório.....	54
3.4.1.2. Exemplo do relatório.....	55
3.4.1.3. Instruções SQL.....	56
3.4.1.4. Ferramenta utilizada para desenvolver e apresentar o relatório.....	56
3.4.2. Relatório de estatísticas.....	56
3.4.2.1. Tela do relatório.....	56
3.4.2.2. Exemplo do relatório.....	57
3.4.2.3. Instruções SQL.....	58
3.4.2.4. Ferramenta utilizada para desenvolver e apresentar o relatório.....	58
3.4.3. Relatório de índices fragmentados.....	58
3.4.3.1. Tela do relatório.....	59
3.4.3.2. Exemplo do relatório.....	59
3.4.3.3. Instruções SQL.....	60
3.4.3.4. Ferramenta utilizada para desenvolver e apresentar o relatório.....	60
3.4.4. Relatório de conexões ativas.....	60
3.4.4.1. Tela do relatório.....	60
3.4.4.2. Exemplo do relatório.....	61

3.4.4.3. Instruções SQL.....	62
3.4.4.4. Ferramenta utilizada para desenvolver e apresentar o relatório.....	62
3.4.5. Relatório de consultas.....	62
3.4.5.1. Tela do relatório.....	63
3.4.5.2. Exemplo do relatório.....	63
3.4.5.3. Instruções SQL.....	64
3.4.5.4. Ferramenta utilizada para desenvolver e apresentar o relatório.....	64
4. Implementações Futuras.....	64
5. Conclusão.....	65
6. Referências Bibliográficas.....	65

1. Introdução

Atualmente muitas soluções possuem sua lógica de negócio concentrada no banco de dados, ou seja, em regras de negócio e tecnologia, torna-se cada vez mais importante prezar pelo controle e performance do que acontece no banco de dados. Para que os clientes tenham maior fluidez ao utilizar o sistema e os usuários consigam fazer as operações normalmente com eficiência e praticidade.

Com os padrões definidos pela tecnologia, de melhores práticas, seja para estruturação e codificação, torna-se importante cada vez mais adequar os programas existentes para este padrão, para sistemas antigos acaba sendo necessário revisar os códigos existentes, gerando atividades que não seriam necessárias se fosse utilizado um padrão de desenvolvimento eficiente.

O SQL Server possui uma linguagem chamada T-SQL (Transact SQL), onde é possível de modo transacional, interagir com o banco de dados, fazendo consultas e atualizações no mesmo (MICROSOFT, 2020). Quando necessitamos processar as informações no banco de dados, é necessário utilizar das funções disponíveis na linguagem T-SQL, todavia deve sempre atentar-se com relação às operações no banco de dados que possuem filtros ou ordenações pois quando não existem índices e suporte uma consulta pode ficar muito lenta, consumindo recursos desnecessários do servidor.

Para a Microsoft (2020), os índices são objetos no banco de dados criados para otimizar operações no banco de dados para determinadas tabelas e visões, basicamente os índices ordenam os registros de uma tabela, a partir dos campos, ou seja, quando um índice é criado precisamos informar as colunas chaves, criando o índice quando realizarmos uma consulta utilizando de filtros os campos do índices, onde temos um ganho de performance, pois não é necessário ler toda a tabela ou ordenar os dados para realizar a pesquisa.

1.1. Definição do problema

A lentidão no banco de dados pode ser causada por diversos fatores, seja uma tabela muito grande, comandos utilizados de forma incorreta, a ausência de indexação ou a má indexação de uma e várias tabelas, onde podem afetar as funcionalidades do sistema tornando-as lentas impedindo o fluxo do usuário. Ainda,

configurações indevidas, no banco, ou tabelas com uma grande volumetria, podem atrapalhar o funcionamento do ambiente, prejudicando todos os usuários do ambiente.

1.2. Objetivos

1.2.1. Objetivo Geral

Desenvolver um sistema de otimização de banco de dados, tornando possível gerar relatórios de pontos que possam ser melhorados no sistema, com relação ao banco de dados, sendo possível fazer análises de informações presentes na base e de rotinas sendo executadas em tempo real.

1.2.2. Objetivo Específico

Identificar problemas que degradam a performance do banco de dados, criar e alterar índices em tabelas no banco de dados, de forma que facilite o trabalho de identificação de problemas e customização de índices na base de dados.

- Estudar e apresentar as melhores práticas de banco de dados;
- Desenvolver um sistema que automatize verificações realizadas manualmente;
- Exemplificar situações que podem afetar a performance de um sistema;

1.3. Justificativa

Cada vez mais se torna necessário realizar análises nos servidores de banco de dados, devido à quantidade de itens que podem afetar o desempenho do servidor, por isso torna-se necessário economizar tempo e identificar o problema de forma automática e que o resultado seja compreendido, considerando sempre as melhores práticas de configuração e programação no banco de dados. Com isso, é possível evidenciar para os clientes os problemas que afetam o servidor e o sistema em si.

1.4. Metodologia

Será consultada a documentação do produto SQL Server atualizada, assim como sites de profissionais nesta tecnologia para que seja sempre levando em consideração os melhores padrões de configuração e desenvolvimento. O software será desenvolvido em C#, utilizando os recursos mais atualizados dotnet framework.

2. Fundamentação teórica

2.1. Introdução ao banco de dados

O banco de dados é um aplicativo que armazena os dados em formato de tabelas, contendo linhas e colunas. Quando se fala de banco de dados, temos que saber a diferença entre dados e informações, para Silva (2015):

Os dados são os fatos brutos, em sua forma primária, e podem não fazer nenhum sentido quando estão isolados; já as informações são o agrupamento de dados organizados, de forma que façam sentido e gerem algum conhecimento.

Um banco de dados pode ter diversas tabelas, assim como uma tabela pode ter várias linhas. Sempre que criamos uma tabela, temos o objetivo de armazenar um conjunto de dados sobre uma entidade específica, ou seja, se criarmos uma tabela de pessoas, criaremos campos que armazenam dados de uma pessoa, como nome, data de nascimento, peso. Segundo Silva (2015):

Uma das definições de banco de dados afirma que se trata de uma coleção de informações que se relacionam de modo que criem algum sentido, isto é, é uma estrutura bem organizada de dados que permite a extração de informações. Assim, são muito importantes para empresas e tornaram-se a principal peça dos sistemas de informação.

Com esta afirmação, conseguimos entender que através de um banco de dados é possível que sistemas processem informações, e que elas sejam utilizadas por uma organização ou indivíduo, por este motivo é importante ter uma estrutura muito bem organizada ao se criar um banco de dados, definindo as colunas, os tipos das colunas, as nomenclaturas e um dos itens mais importantes do banco de dados que é o relacionamento das tabelas (a figura 1 é uma representação visual da

estrutura de uma tabela e seus relacionamentos), que garante a integridade do banco de dados e da regra de negócio da aplicação em si.

Figura 1 - Tabelas de um Banco de Dados



Fonte: Site Estudo Pratico. Disponível em

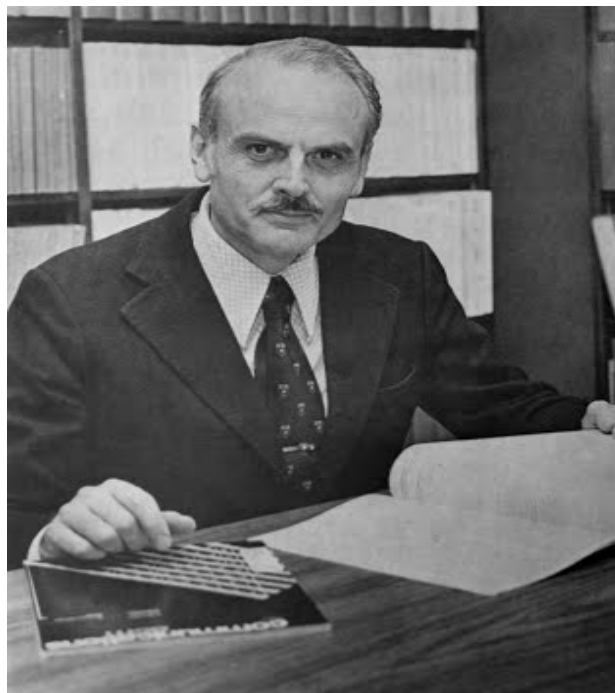
<<https://www.estudopratico.com.br/banco-de-dados/>>.

Atualmente os bancos de dados são disponibilizados com recursos modernos que facilitam a criação, manutenção e monitoramento do banco de dados, todavia os bancos de dados nem sempre foram aplicações tão robustas, mas sim aplicações que evoluíram conforme as necessidades do mercado e da comunidade.

A história do banco de dados, conforme explica Rezende (2020), tem seu início com os softwares de armazenamento de informações (aplicações como Clipper, Dbase 2, Fox Pro, COBOL), os mesmos disponibilizavam métodos e funcionalidades para armazenar informações, selecionar e editar as mesmas. Para Rezende (2020), durante os anos de 1970 e 1972 surgiu o modelo de banco de dados relacional, proposto por Edgar Frank Codd (figura 2), isso fez com que o modo em que os dados são armazenados não estivessem relacionados com a estrutura lógica do banco de dados, criando-se assim uma modelagem que buscava

relacionar entidades do banco de dados, evitando uma modelagem que possa conter erros ou dados duplicados em uma mesma tabela.

Figura 2 - Edgar Frank Codd



Fonte: Site Pioneiros. Disponível em
<<https://sites.google.com/site/pioneiroscomputacao/p-1970-89/p61>>.

O banco de dados relacional proporcionou muitas melhorias em projetos de banco de dados, de forma que não ocorram redundâncias e campos desnecessários nas tabelas, ganhando armazenamento e melhorando a performance. Em 1976 o Dr. Peter Chen propôs o modelo de entidades e seus relacionamentos (conhecido atualmente como MER), sendo uma técnica de criação de projetos de banco de dados, capaz de descrever a estrutura inteira do banco e seus respectivos relacionamentos, sendo fácil, intuitivo e muito prático em casos de dúvidas na fase de desenvolvimento e manutenção do produto. Em 1980 segundo Rezende (2020):

A Linguagem Estruturada de Consulta – SQL (Structured Query Language) se torna um padrão mundial. A IBM transforma o DB2 como carro chefe da empresa em produtos para BD. Os modelos em rede e hierárquico passam a ficar em segundo plano praticamente sem desenvolvimentos utilizando seus conceitos, porém vários sistemas legados continuam em uso. O desenvolvimento do IBM PC

desperta muitas empresas e produtos de BD como: RIM, RBASE 5000, PARADOX, OS/2 Database Manager, Dbase III e IV (mais tarde transformado em FoxBase e mais tarde ainda como Visual FoxPro), Watcom SQL, entre outros.

Em 1990, um modelo de aplicação começou a ganhar espaço no mercado, de acordo com Rezende (2020), “O modelo cliente-servidor (client-server) passa a ser uma regra para futuras decisões de negócio e vemos o desenvolvimento de ferramentas de produtividade como Excel/Access (Microsoft) e ODBC”.

Na metade dos anos 90, os processos de transação em tempo real atingem uma estabilidade com seu intenso uso em pontos de venda (REZENDE, 2020). A partir dos anos 2000, os sistemas gerenciadores de banco de dados evoluíram muito, tendo capacidade de armazenar grande quantidade de dados e aumentaram os recursos fornecidos para que se torne cada vez mais fácil trabalhar com as informações no banco de dados.

Atualmente os bancos de dados seguem com a tendência de não ser somente um repositório de dados, mas sim uma fonte de informações que torna-se possível extrair estatísticas e assim fornecer uma informação que seja possível utilizar em uma tomada de decisão, temos diversos bancos de dados que suportam capacidades superiores a Terabytes de informações, e tecnologias que conseguem processar as mesmas de acordo com as necessidades de negócio e níveis estratégicos da empresa (REZENDE, 2020). Recentemente as empresas têm investido muito em bancos de dados potentes, apostando em sistemas relacionais e não relacionais, visando sempre o melhor desempenho e praticidade em suas aplicações.

2.2. SGBD

Os SGBD nada mais são do que os Sistemas de gerenciamento de banco de dados, os mesmos são aplicações que tem o intuito de gerenciar o banco de dados, sendo responsável por fazer os processamentos, armazenar, atualizar, apagar e selecionar informações no banco de dados que está sendo gerenciado, segundo Sanches (2005):

Os sistemas de banco de dados são projetados para gerenciar grandes grupos de informações. O gerenciamento de dados envolve a definição de estruturas para armazenamento de informação e o fornecimento de mecanismos para manipulá-las. Além disso, o sistema de banco de dados precisa fornecer segurança das informações armazenadas, caso o sistema dê problema, ou contra tentativas de acesso não-autorizado. Se os dados devem ser divididos entre diversos usuários, o sistema precisa evitar possíveis resultados anômalos [...] A importância das informações na maioria das organizações e o consequente valor dos bancos de dados têm orientado o desenvolvimento de um grande corpo de conceitos e técnicas para o gerenciamento eficiente dos dados.

O sistema gerenciador do banco de dados deve ser uma aplicação completa, onde seja possível garantir a integridade dos dados e que seja possível também realizar atividades de contingência, onde caso ocorra algum imprevisto, seja possível recuperar as informações, sendo um mecanismo conhecido pelo termo de *backup*.

2.3. Melhores práticas de banco de dados

As boas práticas de programação existem também quando falamos de banco de dados, muitos programas de banco de dados possuem uma linguagem de programação onde é possível criar rotinas com validações lógicas, manipulação de dados e estruturas de repetição. Com todos estes recursos disponíveis em um banco de dados, deve-se sempre utilizar as melhores práticas de desenvolvimento voltado para banco de dados, principalmente na estruturação de tabelas, envolvendo princípios básicos de modelagem de banco de dados e indexação de dados.

2.4. Normalização de dados

A normalização do banco de dados é uma das técnicas mais importantes utilizadas na modelagem do banco de dados e na criação de suas tabelas, com a utilização da mesma, temos diversos aproveitamentos do banco de dados em si, observando o entendimento do sistema, custo de armazenamento e otimização do mesmo, segundo Machado (2015):

Normalização é o processo de modelar o banco de dados projetando a forma como as informações serão armazenadas a fim de eliminar, ou pelo menos minimizar, a redundância no banco. Tal procedimento

é feito a partir da identificação de uma anomalia em uma relação, decompondo-as em relações melhor estruturadas.

Normalmente precisamos remover uma ou mais colunas da tabela, dependendo da anomalia identificada e criar uma segunda tabela, obviamente com suas próprias chaves primárias e relacionarmos a primeira com a segunda para assim tentarmos evitar a redundância de informações.

Ao se utilizar a normalização de dados em um banco de dados, temos cinco formas para aplicar e assim normalizar um banco de dados, de forma simples, o objetivo da normalização do banco de dados é eliminar a duplicidade de informações e também evitar que informações não obrigatórias sejam armazenadas na mesma tabela, analisando todas as formas normais cada forma faz com que os dados sejam mais sintéticos e íntegros. Para considerar um banco de dados como normalizado não é necessário aplicar todas as formas normais, aplicando até a terceira forma já é possível considerar como normalizada (MARCHI, 2013).

2.4.1. Primeira forma

A primeira forma consiste em remover campos que possuem mais de um valor informado. Nesse caso é necessário separar os as informações de um único campo em vários campos. Outro quesito da primeira forma normal é identificar a chave primária da tabela, sendo uma chave natural ou artificial (MELLO, 2016).

Tabela 1 - Exemplo de tabela antes da primeira forma

Código	Nome	Endereço
10	Adrian Hideki	Rua Cherentes, 60, CEP: 17600-000
11	Alanis Mayumi	Rua Tapajós, 110, 17600-030
12	Luciane Mitiko	Rua Carijós, 180, 17601-611
14	Marcos Oliveira	Avenida Lélio Pizza, 180, 17605-655

Fonte: Autoria própria.

Tabela 2 - Exemplo de tabela com a primeira forma normal aplicada

Código (PK)	Nome	Endereço	Número	CEP
-------------	------	----------	--------	-----

10	Adrian Hideki	Rua Cherentes	60	17600-000
11	Alanis Mayumi	Rua Tapajós	110	17600-030
12	Luciane Mitiko	Rua Carijós	180	17601-611
14	Marcos Oliveira	Avenida Lélío Pizza	180	17605-655

Fonte: Autoria própria.

Analisando as tabelas apresentadas, é possível identificar que o campo endereço (da tabela 1) foi separado em um total de três campos, sendo eles Endereço, Número e CEP (na tabela 2), mostrando assim uma tabela antes da primeira forma normal e após.

2.4.2. Segunda forma

A segunda forma normal pode ser utilizada quando a tabela em si já está na primeira forma normal, tendo o objetivo de identificar campos que não dizem respeito a chave primária da tabela completamente, sendo necessário gerar uma nova tabela com esses dados (MELLO, 2016). Sendo assim podemos assumir um exemplo da tabela de vendas (tabela 3), onde temos o código da venda e do produto vendido, porém nem todos os campos dependem completamente da chave primária desta tabela:

Tabela 3 - Tabela de vendas antes da segunda forma

Código Venda	Código produto	Nome Produto	Quantidade	Valor Produto
1	1	Caneca	10	1,50
2	2	Lápis	30	0,80
3	3	Borracha	15	1,50

Fonte: Autoria própria.

Ao aplicar a segunda forma normal é gerada duas tabelas, uma de produtos (tabela 4) e outra de vendas x produtos (tabela 5):

Tabela 4 - Tabela de Produtos segunda forma

Código Produto (PK)	Nome Produto	Valor Produto
1	Caneca	1.50
2	Lápis	0,80
3	Borracha	1,50

Fonte: Autoria Própria.

Tabela 5 – Tabela de Venda x Produto

Código Venda (PK)	Código Produto (FK, PK)	Quantidade
1	1	10
2	2	30
3	3	15

Fonte: Autoria Própria.

Note que agora eu não tenho mais o nome do produto na tabela de vendas, fazendo com que todos os campos da tabela de vendas dependam da chave primaria nos campos Código Venda e Código Produto, sendo uma chave composta.

2.4.3. Terceira forma

Segundo Mello (2016), para a terceira forma, deve-se “Identificar todos os atributos que são funcionalmente dependentes de outros atributos não chave”, nesse caso devemos remover esses campos.

2.4.4. Quarta forma

Para Mello (2016), uma tabela estar na quarta forma normal, ela deve estar na terceira forma e não existir dependências multivaloradas. Nesse caso devemos dividir a tabela em várias, evitando assim redundância de dados. Abaixo conseguimos observar uma tabela antes (tabela 6) e depois da quarta forma normal (tabelas 7 e 8):

Tabela 6 – Tabela antes da quarta forma normal

Código Aula	Aluno	Sala
A001	E001	S001
A001	E002	S001
A001	E001	S002
A001	E002	S002

Fonte: Autoria Própria.

Tabela 7 - Tabela de Aula x Alunos

Código Aula	Aluno
A001	E001
A001	E002

Fonte: Autoria própria.

Tabela 8 - Tabela de Aula x Sala

Código Aula	Sala
A001	S001
A001	S002

Fonte: Autoria própria.

Nesta forma normal, foram geradas duas tabelas, sendo uma com a relação das aulas e alunos, e na outra a relação das aulas com as salas, evitando assim duplicidade nos dados das tabelas, como era possível observar na tabela 6.

2.4.5. Quinta forma

Para Marchi (2013), a quinta forma normal pode ser definida no seguinte conceito:

A 5FN ou Forma normal de Projeção-Junção baseia-se no conceito de dependência de junção. Dependência de junção expressa a capacidade de um tabela, que tenha sido decomposta em duas ou mais tabelas menores, de se reconstruir novamente a partir das

novas tabelas, obtendo-se a tabela original. Logo, uma tabela está na 5FN quando não é mais possível decompô-la. Há um consenso que são raras as exceções que necessitam da passagem para a 5FN, na maioria dos casos, quando uma tabela está na 4FN também estará na 5FN.

Conforme é explicado por Marchi (2013), a quinta forma normal é raramente utilizada, basicamente é utilizada quando existem informações dependentes decompostas em tabelas separadas que precisam estar juntas em uma tabela.

2.5. SQL Server

O SQL Server é um sistema gerenciador de banco de dados mais populares da atualmente, presente em diversas empresas ao redor do mundo, devido a sua grande comunidade técnica e facilidade de uso, é preferência em diversas empresas desenvolvedoras de software, porém sua primeira versão surgiu inicialmente em 1988, sendo disponibilizado junto a versão do Windows NT (uma das versões iniciais do Windows) e posteriormente foi evoluindo como um produto separado, evoluindo recursos e funcionalidades novas em cada versão (PACIEVITCH, 2020).

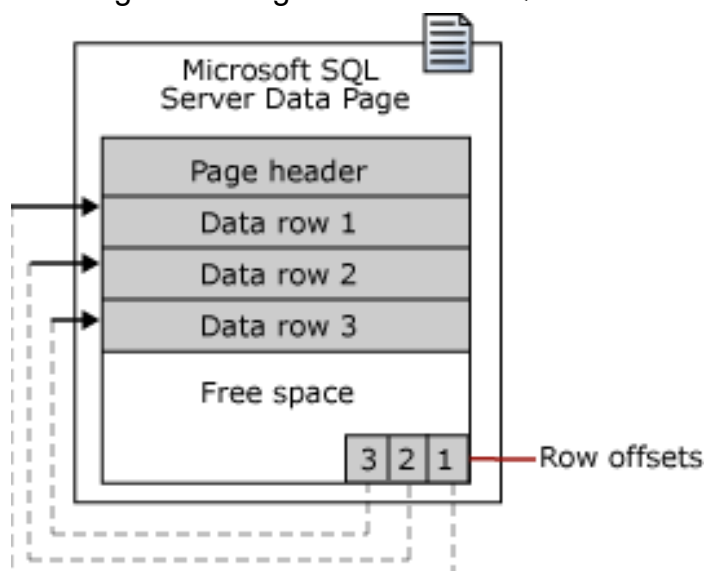
Os bancos de dados relacionais em geral, seguem um padrão de codificações que deve ser iguais para todos os bancos de dados, este é o padrão ANSI de programação, o SQL possui a linguagem de programação chamada de T-SQL (Transact SQL), que possui os comandos do padrão ANSI e além disso, funcionalidades que somente o SQL Server possui, como comandos de agregações e funções que não fazem parte do padrão do SQL Server.

Recentemente a Microsoft lançou a versão do SQL Server 2019, hoje o SQL Server além de ser um sistema gerenciador de banco de dados, o mesmo possui funcionalidades de atividades como processamento de massas de dados (conhecido pelo termo Big Data), suporte a Data Warehouse (conhecido por DW, uma forma de armazenamento e processamento de dados através de visões) e processos de BI (Business Intelligence), além de possuir sua versão online no Azure, uma plataforma digital de serviços, como máquinas virtuais e servidores de banco de dados SQL ou NoSQL (MICROSOFT, 2020).

2.6. Indexação de tabelas e estrutura de dados

O SQL Server possui um mecanismo de armazenamento de informações em páginas de dados, segundo a documentação do Microsoft SQL Server (MSSQL) a página de dados é a unidade de armazenamento fundamental no armazenamento dos dados, qualquer informação gravada no banco de dados é gravado em páginas de dados, cada página de dados possui um tamanho de 8 kilobytes (ou 8KB). A estrutura da página de dados (representado visualmente pela Figura 3) pode ser dividida entre o cabeçalho, que contém algumas informações de metadados e linhas de dados.

Figura 3 - Página de dados SQL Server

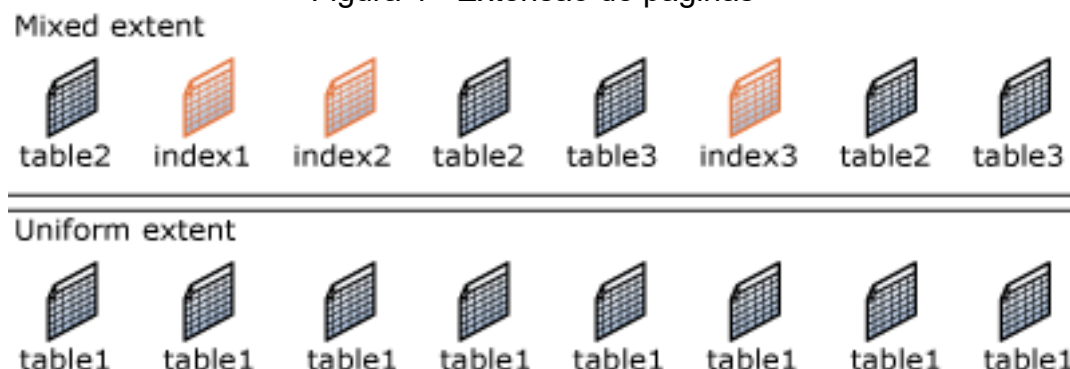


Fonte: Microsoft SQL Server Docs. Disponível em:

<<https://docs.microsoft.com/pt-br/sql/relational-databases/pages-and-extents-architecture-guide?view=sql-server-ver15>>.

Entendendo como funcionam as páginas, é possível compreender as extensões, que nada mais é do que um conjunto de oito páginas de dados, segundo as documentações da Microsoft, as extensões (representada pela figura 4) podem ser classificadas em duas, uniformes e mixadas, onde as extensões uniformes possuem dados de somente uma tabela e extensões mixadas possuem páginas de diversas tabelas.

Figura 4 - Extensão de páginas



Fonte: Microsoft SQL Server Docs. Disponível em:

<<https://docs.microsoft.com/pt-br/sql/relational-databases/pages-and-extends-architecture-guide?view=sql-server-ver15>>.

Quando vamos inserir alguma informação caso o SQL Server não tenha página de dados já criadas em branco é necessário no momento da inserção dos dados gerar estas páginas, isso envolve custos de processamento no servidor que o SQL Server está instalado, consumindo recursos de processador, disco e memória, por isso sempre é recomendado deixar páginas de dados já criadas no banco de dados, evitando assim lentidões em inserções massivas no sistema.

Os índices são objetos essenciais para o funcionamento de um banco de dados de sistemas online, este objetivo tem o objetivo de otimizar consultas realizadas no banco de dados, por isso a ausência de índices eficazes em um banco de dados pode causar gargalos no servidor e acabar afetando o tempo de resposta da consulta. Segundo a documentação da MSSQL (Microsoft SQL Server 2020), os índices devem ser criados de forma que ofereçam suporte a consultas complexa, contendo filtros e ordenação de dados na mesma, no SQL Server existem diversos tipos de índices:

- Clusterizado
- Não clusterizado
- Exclusivo
- Filtrado
- Columnstore
- Hash

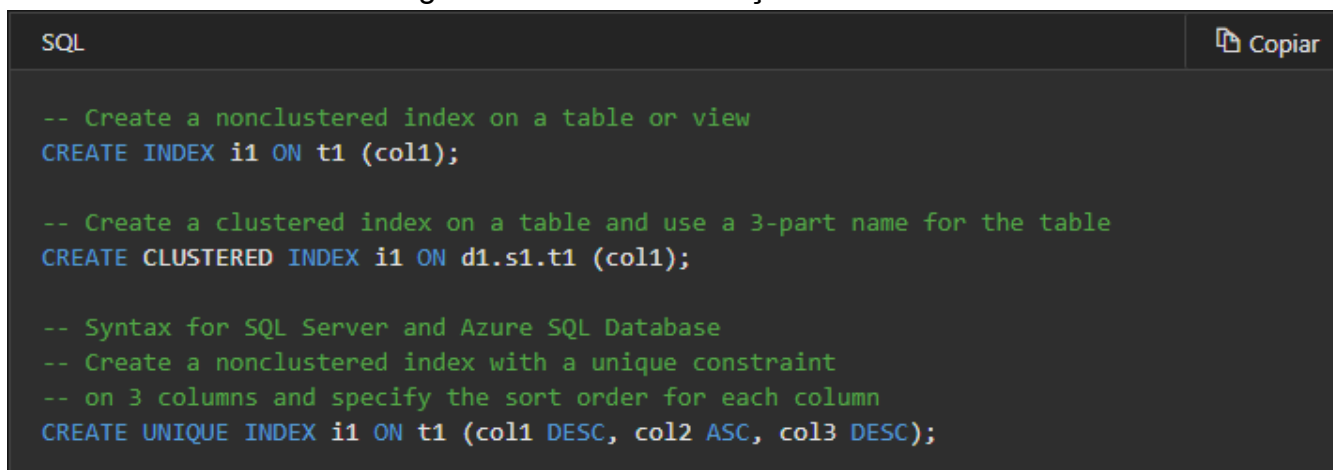
- Não clusterizado com otimização em memória

Os índices nada mais são do que cópias dos dados de uma tabela em um objeto interno do SQL Server, com o intuito de otimizar a leitura dos dados de uma determinada tabela, pense no índice como se fosse o sumário e a tabela um livro, para a Microsoft (2019) um índice pode ser definido como:

Um índice é uma estrutura em disco associada a uma tabela ou exibição, que agiliza a recuperação das linhas de uma tabela ou exibição. Um índice contém chaves criadas de uma ou mais colunas da tabela ou exibição. Essas chaves são armazenadas em uma estrutura (árvore B) que habilita o SQL Server a localizar a linha ou as linhas associadas aos valores de chave de forma rápida e eficaz.

Uma tabela pode ter vários índices, todavia deve-se tomar cuidado para não criar muitos, pois acaba aumentando o tamanho da tabela e aumenta o tempo das operações de inserção e atualização de dados, pois é necessário replicar as alterações nos índices criados, ou seja, um índice desnecessário pode prejudicar a performance de uma consulta (RODRIGUES, 2020). Para criar um índice utilizando o SQL Server, observe a sintaxe do SQL Server demonstrada na figura 5:

Figura 5 - Sintaxe de criação de índices

A screenshot of a SQL code editor with a dark background. The editor has a tab labeled 'SQL' and a 'Copiar' (Copy) button in the top right corner. The code is written in a light green font and includes several comments and SQL statements for creating different types of indexes. The comments are preceded by '--'. The SQL statements use 'CREATE INDEX' for nonclustered indexes and 'CREATE CLUSTERED INDEX' for clustered indexes. The last statement uses 'CREATE UNIQUE INDEX' for a unique index and specifies sort orders for multiple columns.

```
SQL                                                                    Copiar

-- Create a nonclustered index on a table or view
CREATE INDEX i1 ON t1 (col1);

-- Create a clustered index on a table and use a 3-part name for the table
CREATE CLUSTERED INDEX i1 ON d1.s1.t1 (col1);

-- Syntax for SQL Server and Azure SQL Database
-- Create a nonclustered index with a unique constraint
-- on 3 columns and specify the sort order for each column
CREATE UNIQUE INDEX i1 ON t1 (col1 DESC, col2 ASC, col3 DESC);
```

Fonte: Microsoft SQL Server Docs. Disponível em:

<<https://docs.microsoft.com/pt-br/sql/t-sql/statements/create-index-transact-sql?view=sql-server-ver15>>.

Note que após o comando “CREATE” é colocado o tipo do índice, quando o tipo do índice não é informado o SQL Server assume que o mesmo é não

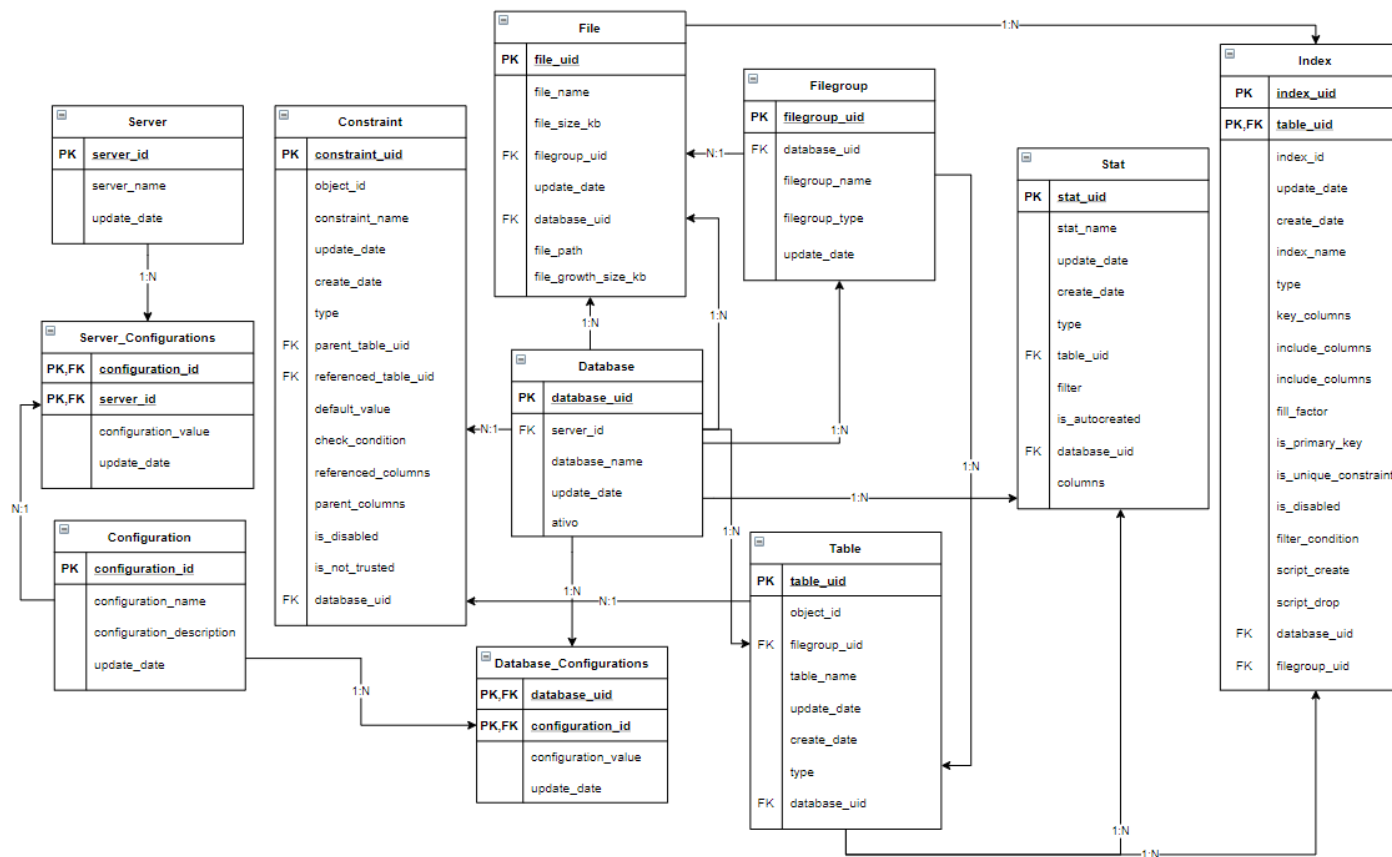
clusterizado (NONCLUSTERED), após o tipo do índice é necessário escrever “INDEX”, declarar o nome do índice, seguido por “ON” e o nome da tabela, observe que é necessário informar as colunas e a ordenação das mesmas (ordem ascendente ou descendente), normalmente as colunas informadas no índice são as que sofrem filtragem ou ordenação de dados.

3. Desenvolvendo o projeto proposto

3.1. Diagramas UML

3.1.1. MER - Modelo de Entidade e Relacionamento

Figura 6 - Modelo de entidade e relacionamento



Fonte: Autoria própria.

3.1.2. Dicionário de dados

Tabela 6 - Tabela de Servidores

Entidade: Server					
Descrição: Armazenar os dados do servidor					
Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição
01	server_id	Integer	X		Identificador do servidor
02	server_name	varchar(255)			Nome
03	update_date	datetime			Data de atualização
RELACIONAMENTOS:					
ID	Entidade		Atributo Origem	Atributo Destino	
Não há					

Fonte: Autoria própria.

Tabela 7 - Tabela de configurações do servidor

Entidade: Server_Configurations					
Descrição: Armazenar as configurações do servidor					
Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição
01	configuration_id	Integer	X	X	Identificador da configuração
02	server_id	Integer	X	X	Identificador do servidor
03	configuration_value	varchar(255)			Valor da configuração
04	update_date	datetime			Data de atualização
RELACIONAMENTOS:					
ID	Entidade		Atributo Origem	Atributo Destino	
01	Server		server_id	server_id	
02	Configuration		configuration_id	configuration_id	

Fonte: Autoria própria.

Tabela 8 - Tabela de Configurações

Entidade: Configuration	
Descrição: Armazenar as configurações	

Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição
01	configuratio n_id	Integer	X		Identificador da configuração
02	configuratio n_name	varchar(25 5)			Nome da configuração
03	configuratio n_descripti on	varchar(25 5)			Descrição da configuração
04	update_dat e	datetime			Data de atualização
RELACIONAMENTOS:					
ID	Entidade		Atributo Origem	Atributo Destino	
Não há					

Fonte: Autoria própria.

Tabela 9 - Tabela de Banco de dados

Entidade: Database					
Descrição: Armazenar os bancos de dados					
Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição
01	database_u id	Integer	X		Identificador da base
02	server_id	Integer		X	Identificador do servidor
03	database_n ame	varchar(25 5)			Nome da base
04	update_dat e	datetime			Data de atualização
05	ativo	bit			Ativo
RELACIONAMENTOS:					
ID	Entidade		Atributo Origem	Atributo Destino	
01	Server		server_id	server_id	

Fonte: Autoria própria.

Tabela 10 - Tabela de configurações do banco de dados

Entidade: Database_Configurations					
Descrição: Armazenar as configurações do banco de dados					
Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição

01	configuratio n_id	Integer	X	X	Identificador da configuração
02	configuratio n_value	varchar(25 5)			Valor da configuração
03	update_dat e	datetime			Data de atualização
04	database_u id	Integer	X	X	Identificador Único do Banco de dados
05	ativo	bit			Se o banco está ativo ou não
RELACIONAMENTOS:					
ID	Entidade	Atributo Origem	Atributo Destino		
01	Database	database_ui d	database_u id		
02	Configuration	configuratio n_id	configuratio n_id		

Fonte: Autoria própria.

Tabela 11 - Tabela de Grupo de Arquivos

Entidade: Filegroup					
Descrição: Armazenar as informações do grupo de arquivos					
Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição
01	filegroup_ui d	Integer	X		Identificador Único do Filegroup
02	filegroup_name	varchar(25 5)			Nome do grupo de arquivos
03	filegroup_ty pe	varchar(25 5)			Tipo do Filegroup
04	database_u id	Integer		X	Identificador Único do Banco de dados
05	update_dat e	datetime			Data de atualização
RELACIONAMENTOS:					
ID	Entidade	Atributo Origem	Atributo Destino		
01	Database	database_ui d	database_u id		

Fonte: Autoria própria.

Tabela 12 - Tabela de arquivos

Entidade: File

Descrição: Armazenar as informações dos arquivos do banco de dados					
Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição
01	file_uid	Integer	X		Identificador Único do File
02	file_name	varchar(255)			Nome do arquivo
03	file_size_kb	numeric(18,6)			Tamanho do Arquivo
04	filegroup_uid	Integer		X	Identificador do grupo de arquivos
05	update_date	datetime			Data de atualização
06	database_uid	Integer		X	Identificador Único do Banco de dados
07	file_path	varchar(500)			Caminho do arquivo
08	file_growth_size_kb	numeric(18,6)			Tamanho do fator de crescimento
RELACIONAMENTOS:					
ID	Entidade		Atributo Origem	Atributo Destino	
01	Database		database_uid	database_uid	
02	Filegroup		filegroup_uid	filegroup_uid	

Fonte: Autoria própria.

Tabela 13 - Tabela de tabelas

Entidade: Table					
Descrição: Armazenar as informações das tabelas					
Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição
01	table_uid	Integer	X		Identificador Único da Tabela
02	object_id	Integer			Identificador do objeto
03	table_name	varchar(255)			Nome da tabela
04	filegroup_uid	Integer		X	Identificador do grupo de arquivos
05	create_date	Datetime			Data de criação
06	type	varchar(10)			Tipo
07	update_date	Datetime			Data de atualização
08	database_uid	Integer		X	Identificador Único

	id				do Banco de dados
RELACIONAMENTOS:					
ID	Entidade	Atributo Origem	Atributo Destino		
01	Database	database_uid	database_uid		
02	Filegroup	filegroup_uid	filegroup_uid		

Fonte: Autoria própria.

Tabela 14 - Tabela de restrições

Entidade: Constraint					
Descrição: Armazenar as informações das constraints					
Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição
01	constraint_uid	Integer	X		Identificador da restrição
02	constraint_name	varchar(255)			Nome da restrição
03	update_date	DateTime			Data de atualização
04	create_date	DateTime			Data de criação
05	type	varchar(10)			Tipo
06	parent_table_uid	Integer		X	Identificador da tabela pai
07	referenced_table_uid	Integer		X	Identificador da tabela referenciada
08	default_value	varchar(255)			Valor padrão
09	check_condition	varchar(4000)			Condição
10	referenced_columns	varchar(4000)			Colunas referenciadas
11	parent_columns	varchar(4000)			Colunas pai
12	is_disabled	bit			Desativado
13	is_not_trusted	bit			Não confiável
14	constraint_uid	bit			Identificador Único da restrição
15	database_uid	Integer		X	Identificador Único do Banco de dados
16	object_id	Integer			Identificador do objeto
RELACIONAMENTOS:					
ID	Entidade	Atributo	Atributo		

		Origem	Destino	
01	Database	database_uid	database_uid	
02	Table	parent_table_uid	table_uid	
03	Table	referenced_table_uid	table_uid	

Fonte: Autoria própria.

Tabela 15 - Tabela de estatísticas

Entidade: Stat					
Descrição: Armazenar as informações das estatísticas					
Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição
01	stat_uid	Integer	X		Identificador da estatística
02	stat_name	varchar(255)			Nome da estatística
03	update_date	DateTime			Data de atualização
04	create_date	DateTime			Data de criação
05	type	varchar(10)			Tipo
06	filter	varchar(4000)			Filtro
07	is_autocreated	bit			Auto-criado
08	table_uid	integer		X	Identificador da tabela
10	database_uid	integer		X	Identificador Único do Banco de dados
11	columns	varchar(5000)			Colunas
RELACIONAMENTOS:					
ID	Entidade	Atributo Origem	Atributo Destino		
01	Database	database_uid	database_uid		
02	Table	table_uid	table_uid		

Fonte: Autoria própria.

Tabela 16 - Tabela de índices

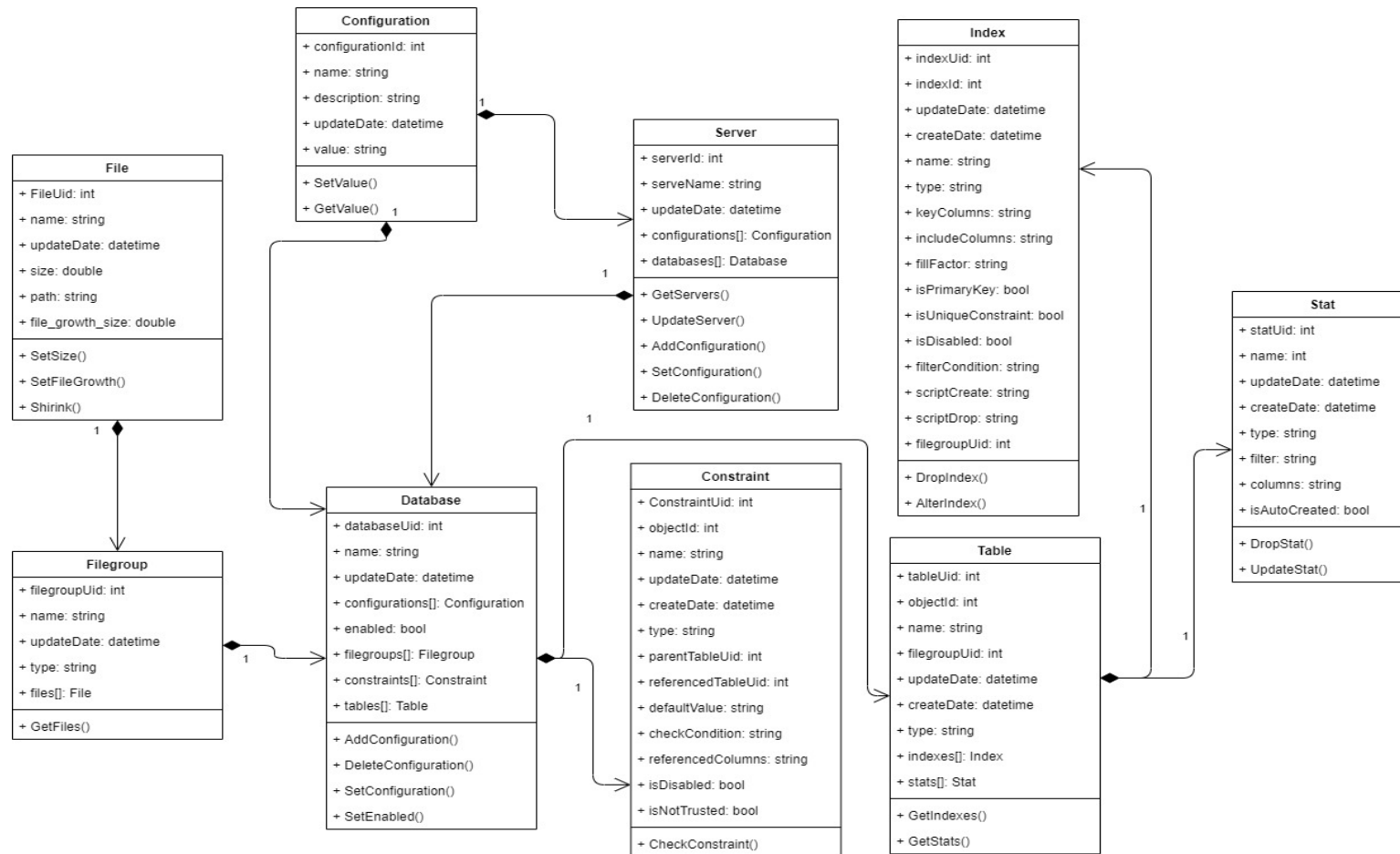
Entidade: Index	
Descrição: Armazenar as informações dos índices	

Nro do Atributo	Atributo	Tipo	Chave Primária	Chave Estrangeira	Descrição
01	table_uid	Integer	X	X	Identificador da tabela
02	filegroup_uid	Integer		X	Identificador do grupo de arquivos
03	index_id	Integer			Identificador do índice
04	create_date	DateTime			Data de criação
05	update_date	DateTime			Data de atualização
06	index_name	varchar(255)			Nome do índice
07	type	varchar(10)			Tipo
08	key_columns	varchar(8000)			Colunas chaves
09	include_columns	varchar(8000)			Colunas incluídas
10	fill_factor	integer			Fator de preenchimento
11	is_primary_key	bit			Chave primária
12	is_unique_constraint	bit			Restrição única
13	is_disabled	bit			Desabilitado
14	filter_condition	varchar(1000)			Condição de filtragem
15	script_create	varchar(8000)			Script de criação
16	script_drop	varchar(8000)			Script de apagar
17	index_uid	integer	X		Identificador Único do índice
18	database_uid	integer		X	Identificador Único do Banco de dados
RELACIONAMENTOS:					
ID	Entidade		Atributo Origem	Atributo Destino	
01	Database		database_uid	database_uid	
02	Table		table_uid	table_uid	
02	Filegroup		filegroup_uid	filegroup_uid	

Fonte: Autoria própria.

3.1.3. Diagrama de Classes

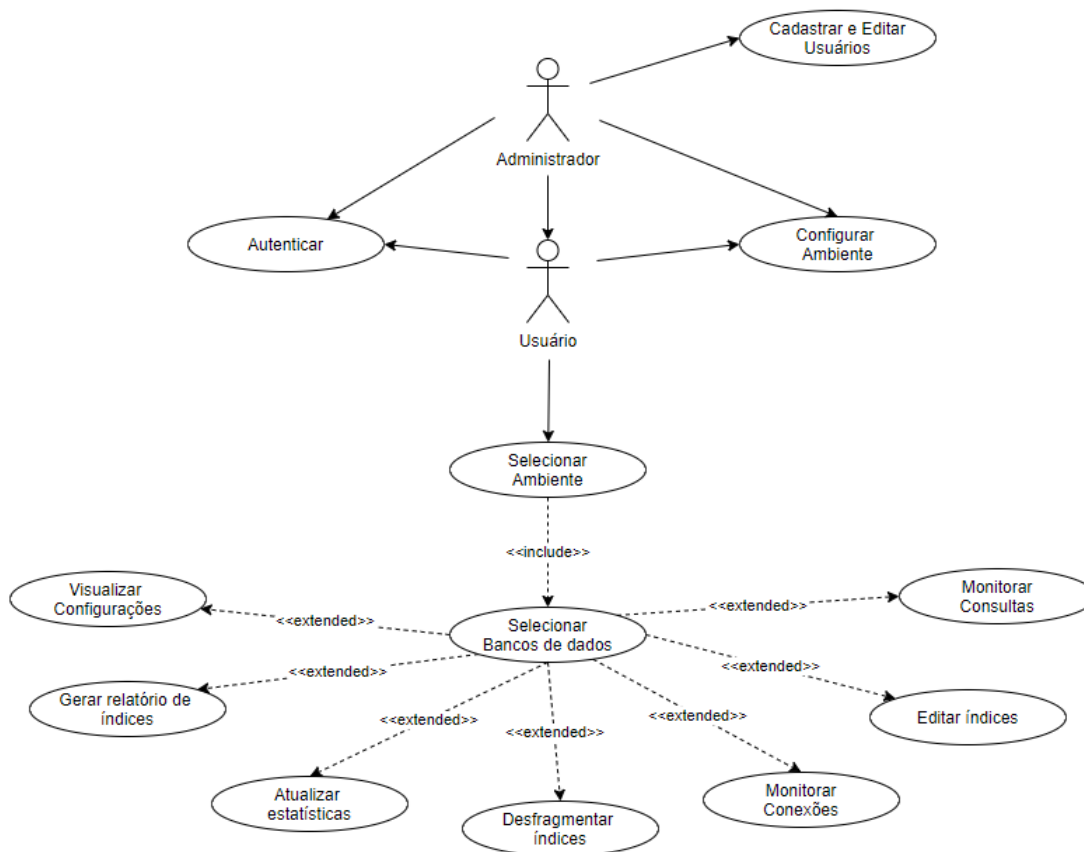
Figura 7 - Diagrama de Classe



Fonte: Autoria própria.

3.1.4. Diagrama de Casos de Uso

Figura 8 - Diagrama de caso de uso



Fonte: Autoria Própria.

3.1.4.1. Descrição dos Casos de Uso

Tabela 17 - UC001 Autenticar

UC001

Descrição

Atores

Pré-condições

Pós-condições

Fluxos de evento

Fluxos Alternativos

Cadastrar Usuário

O usuário deverá se cadastrar no sistema para utilizar a ferramenta.

Administrador e usuário

Ter acesso ao sistema.

O usuário deverá autenticar para ter acesso à aplicação.

1. O usuário deverá acessar a tela de cadastro;

2. Preencher o cadastro do usuário completamente;

3. Confirmar o cadastro;

O usuário poderá cancelar o cadastro

Requisitos Especiais
Ponto de relacionamento

Não há.

Não há.

Fonte: Autoria Própria.

Figura 9 - Tela de Cadastro de Usuário

sindex

Usuário:

Usuário

Senha:

Senha

Confirmar Senha:

Confirmar Senha

e-mail:

E-mail

Perfil:

Administrador

Voltar

Cadastrar

Fonte: Autoria Própria.

Tabela 18 - UC002 Autenticar

UC002
Descrição

Autenticar
O usuário deverá autenticar com o usuário

Atores	cadastrado previamente no sistema. 1. Administrador 2. Usuário
Pré-condições	Possuir um usuário cadastrado no sistema.
Pós-condições	Não há.
Fluxos de evento	O ator deverá informar as credenciais na tela de login do sistema e clicar em “Entrar”.
Fluxos Alternativos	Caso o ator não tenha uma conta cadastrada o mesmo poderá cadastrar uma clicando em “Cadastrar Conta”.
Requisitos Especiais	Possuir uma conta cadastrada no sistema sem bloqueio.
Ponto de relacionamento	Não há.

Fonte: Autoria Própria.

Figura 10 - Tela de Login

Fonte: Autoria Própria.

Tabela 19 - UC003 Configurar ambiente

UC003

Descrição

Configurar ambiente

O ator deverá cadastrar a instância de banco de dados que deseja monitorar utilizando a ferramenta, preenchendo as informações solicitadas na tela de configuração do ambiente.

Atores

Usuário e administrador.

Pré-condições

Estar autenticado no sistema.

Pós-condições

Configurar um ambiente válido.

Fluxos de evento

O ator deverá preencher as informações solicitadas na tela e clicar em “Adicionar”.

Fluxos Alternativos

Caso o ator possua ambientes já configurados:

1. Alterar as informações dos ambientes e clicar em “Atualizar” para salvar as configurações.

2. Testar a conexão do ambiente clicando em “Testar a conexão”.

3. Excluir um ambiente.

4. Selecionar um ambiente.

Requisitos Especiais

Não há.

Ponto de relacionamento

Não há.

Fonte: Autoria Própria.

Figura 11 - Tela de Configuração dos Ambientes

Ambiente:

Nome do Ambiente:

Servidor:

Usuário:

Senha:

Base:

Minutos para atualizar tabelas:

Segundos para atualizar gráficos:

Usuário: sa Ambiente:

Fonte: Autoria Própria.

Tabela 20 - UC004 Selecionar Ambiente

UC004	Selecionar Ambiente
Descrição	O ator deverá selecionar o ambiente que deseja realizar o monitoramento.
Atores	Usuário e administrador.
Pré-condições	Estar autenticado e possuir um ambiente configurado corretamente.
Pós-condições	Não há.
Fluxos de evento	Selecionar o ambiente na caixa de seleção superior e clicar em “Selecionar Ambiente”.
Fluxos Alternativos	Não há.
Requisitos Especiais	Não há.
Ponto de relacionamento	Não há.

Fonte: Autoria Própria.

Figura 12 - Tela de Seleção de ambientes configurados

Fonte: Autoria Própria.

Tabela 21 - UC005 Selecionar Banco de dados

UC005	Selecionar Banco de Dados
Descrição	O ator deve selecionar os bancos de dados que deseja realizar o monitoramento.
Atores	Usuário e administrador.
Pré-condições	Ter o ambiente pré selecionado.

Pós-condições
Fluxos de evento

Não há.

Selecionar os bancos de dados que o ator deseja monitorar, marcando os registros na grade de seleção, posteriormente clicando em “Salvar e Fechar”.

Fluxos Alternativos

É possível marcar todos os bancos de dados marcando a opção “Marcar Todos”.

Requisitos Especiais

Não há.

Ponto de relacionamento

Não há.

Fonte: Autoria Própria.

Figura 13 - Tela de seleção de banco de dados

Selecione o(s) banco(s) de dados que deseja monitorar:

Database ID	Banco de Dados	Última Atualização	Monitorar
1	master	20/10/2020 20:59	<input checked="" type="checkbox"/>
2	tempdb	20/10/2020 20:59	<input checked="" type="checkbox"/>
3	model	20/10/2020 20:59	<input checked="" type="checkbox"/>
4	msdb	20/10/2020 20:59	<input checked="" type="checkbox"/>
5	AdventureWorks2016	20/10/2020 20:59	<input checked="" type="checkbox"/>
6	Cooperativa	20/10/2020 20:59	<input checked="" type="checkbox"/>
7	bySPED	20/10/2020 20:59	<input checked="" type="checkbox"/>
8	AdventureWorks2017	20/10/2020 20:59	<input checked="" type="checkbox"/>
9	FACCAT	20/10/2020 20:59	<input checked="" type="checkbox"/>
10	sindex	20/10/2020 20:59	<input checked="" type="checkbox"/>

Pesquisar:

☐ Marcar Todos

Salvar e Fechar

Fonte: Autoria Própria.

Tabela 22 - UC006 Visualizar Configurações

UC006

Descrição

Visualizar Configurações

Caso o ator já esteja logado e deseje revisar as configurações do ambiente, basta acessar a tela de configurações do ambiente.

Atores

Usuário e administrador.

Pré-condições

Estar autenticado.

Pós-condições

Não há.

Fluxos de evento

Revisar as configurações necessárias e clicar

Fluxos Alternativos

em “Atualizar”.

Caso o ator possua um ambiente configurado:

1. Testar a conexão do ambiente clicando em “Testar Conexão”;
2. Excluir o ambiente clicando em “Excluir”;
3. Caso o usuário esteja com o ambiente selecionado, clicar em “Selecionar Ambiente” para prosseguir com o acesso ao sistema.

Requisitos Especiais

Não há.

Ponto de relacionamento

Não há.

Fonte: Autoria Própria.

Figura 14 - Tela de visualização de configurações

The screenshot displays a web application interface for configuration. On the left is a dark sidebar with a menu containing: 'Configurações' (with a yellow icon), 'Ambientes', 'Bancos de dados', 'Atualizar Dados', 'Usuários', 'Tuning', 'Monitoramento', and 'Sistema'. The main area has a light gray background and contains the following fields and buttons:

- Ambiente:** A dropdown menu currently showing 'homologação'.
- Nome do Ambiente:** A text input field containing 'homologação'.
- Servidor:** A text input field containing '\\sql2017'.
- Usuário:** A text input field containing 'sa'.
- Senha:** A password input field with masked characters '*****'.
- Base:** A text input field containing 'sindex'.
- Minutos para atualizar tabelas:** A text input field containing '1'.
- Segundos para atualizar gráficos:** A text input field containing '3'.
- Buttons:** A row of five buttons: 'Testar Conexão', 'Excluir', 'Atualizar', 'Adicionar', and 'Selecionar Ambiente'.
- Status Bar:** At the bottom, it shows 'Usuário: sa' and 'Ambiente: homologação'.

Fonte: Autoria Própria.

Tabela 23 - UC007 Gerar relatório de índices fragmentados

UC007

Descrição

Gerar Relatório de Índices Fragmentados

Nesta tela é possível visualizar os índices fragmentados para os bancos de dados selecionados, sendo possível desfragmentar os mesmos, assim como gerar relatórios.

Atores

Usuários com acesso ao perfil “Performance” e administradores.

Pré-condições

Ter acesso ao perfil de administrador ou ter o

Pós-condições

perfil de “Performance” no sistema.

Ter permissões para executar a desfragmentação dos índices no banco de dados.

Fluxos de evento

1. Selecionar os índices que deseja desfragmentar, clicar com o botão direito sobre a grade de seleção e clicar em “Desfragmentar índices selecionados”.

Fluxos Alternativos

1. Para imprimir relatórios, clique com o botão direito sobre a grade de seleção e seccione o menu “Gerar relatório” e escolha entre os modelos de relatórios disponíveis.

Requisitos Especiais

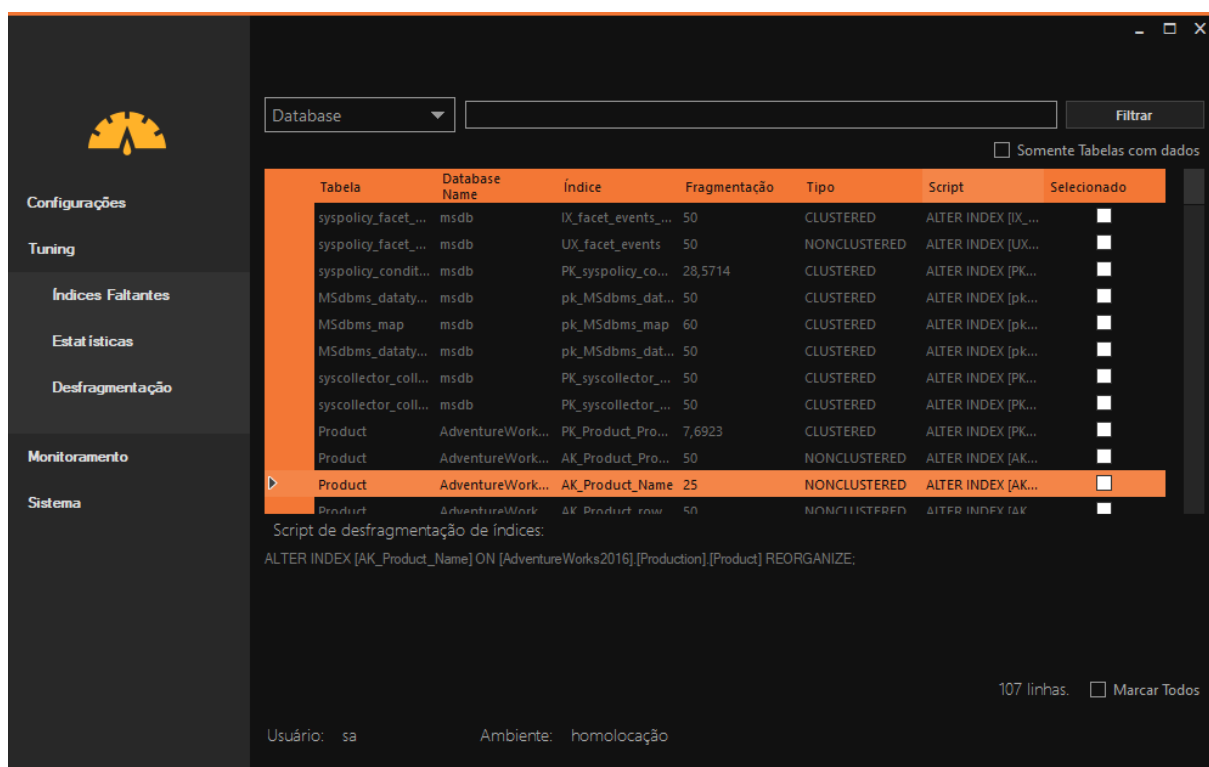
Não há.

Ponto de relacionamento

Não há.

Fonte: Autoria Própria.

Figura 15 - Tela de desfragmentação de índices



Fonte: Autoria Própria.

Tabela 24 - UC008 Atualizar estatísticas

UC008

Descrição

Atualizar estatísticas

Nesta tela é possível identificar as estatísticas do banco de dados que estão desatualizadas,

Atores

Pré-condições

Pós-condições

Fluxos de evento

Fluxos Alternativos

Requisitos Especiais

Ponto de relacionamento

de forma que seja possível gerar um relatório e atualizar as mesmas.

Usuários com o perfil de “Performance” e administrador.

Ter o ambiente selecionado.

Ter permissão para atualizar as estatísticas no banco de dados.

1. Marcar as estatísticas que deseja atualizar, clicar com o botão direito sobre a grade de seleção e clicar em “Atualizar estatísticas selecionadas”.

1. Para imprimir relatórios, clique com o botão direito sobre a grade de seleção e seccione o menu “Gerar relatório” e escolha entre os modelos de relatórios disponíveis.

Não há.

Não há.

Fonte: Autoria Própria.

Figura 16 - Tela de atualização de estatísticas

Database: [] Quantidade de dias desde a última atualização: [0] Filtrar

Opções de geração do script de atualização de estatística:

☒ Full Scan ☐ Sample Rows: [0]

☐ Resample ☐ Sample Percent: [0] ☐ Somente tabelas com dados

Estatística	Tabela	Criado Automático	Dias Última Atualização	Data de Atualização	Banco de Dados	Script	Selecionado
PK__sysutili_...	sysutility_mi_...	<input type="checkbox"/>	27	23/09/2020 1...	msdb	UPDATE STATI...	<input type="checkbox"/>
PK__sysutili_...	sysutility_mi_...	<input type="checkbox"/>	27	23/09/2020 1...	msdb	UPDATE STATI...	<input type="checkbox"/>
PK__syspoli_...	syspolicy_con...	<input type="checkbox"/>	27	23/09/2020 1...	msdb	UPDATE STATI...	<input type="checkbox"/>
PK__log_ship...	log_shipping_...	<input type="checkbox"/>	44122	01/01/1900	msdb	UPDATE STATI...	<input type="checkbox"/>
UQ__log_ship...	log_shipping_...	<input type="checkbox"/>	44122	01/01/1900	msdb	UPDATE STATI...	<input type="checkbox"/>
uc1sprimary_...	log_shipping_...	<input type="checkbox"/>	44122	01/01/1900	msdb	UPDATE STATI...	<input type="checkbox"/>
nc1sprimary_...	log_shipping_...	<input type="checkbox"/>	44122	01/01/1900	msdb	UPDATE STATI...	<input type="checkbox"/>
nc2sprimary_...	log_shipping_...	<input type="checkbox"/>	44122	01/01/1900	msdb	UPDATE STATI...	<input type="checkbox"/>
PK__syspoli_...	syspolicy_ma...	<input type="checkbox"/>	27	23/09/2020 1...	msdb	UPDATE STATI...	<input type="checkbox"/>

Script de atualização de estatísticas:

```
UPDATE STATISTICS [msdb].[dbo].[sysutility_mi_emo_objects_to_collect_internal]([PK__sysutili__077EA96588DD3D7D]) WITH FULLSCAN;
```

782 linhas. ☐ Marcar Todos

Usuário: sa Ambiente: homologação

Fonte: Autoria Própria.

Tabela 25 - UC009 Índices Ausentes

UC009

Descrição

Atores

Pré-condições

Pós-condições

Fluxos de evento

Fluxos Alternativos

Requisitos Especiais

Ponto de relacionamento

Índices Ausentes

Nesta tela o ator conseguirá identificar índices é preciso criar na base, por causa de determinada instrução SQL que necessita do mesmo.

Usuário com o perfil “Performance” ou administradores.

Ter o ambiente selecionado.

Ter permissão de criar índices no banco de dados.

1. Marcar os índices que deseja criar e clicar com o botão direito na grade de seleção, clicando na opção “Criar índices selecionados”.

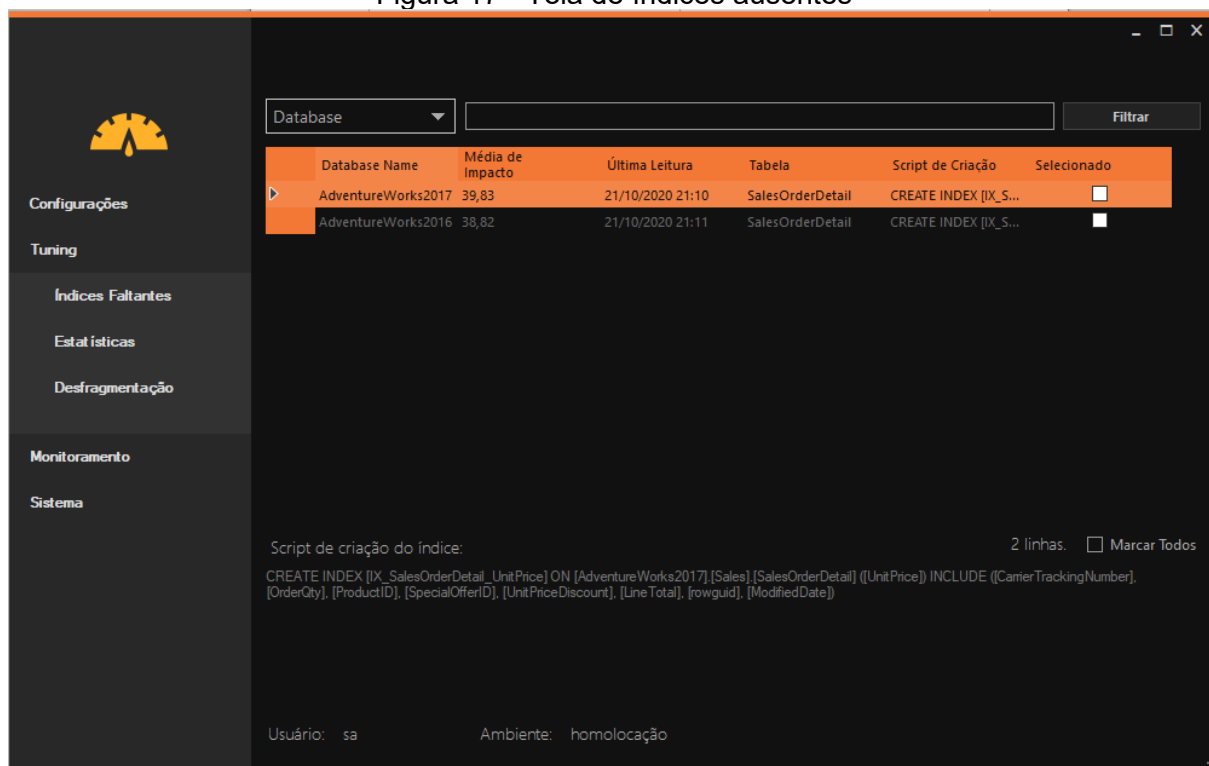
1. Para imprimir relatórios, clique com o botão direito sobre a grade de seleção e seccione o menu “Gerar relatório” e escolha entre os modelos de relatórios disponíveis.

Não há.

Não há.

Fonte: Autoria Própria.

Figura 17 - Tela de índices ausentes



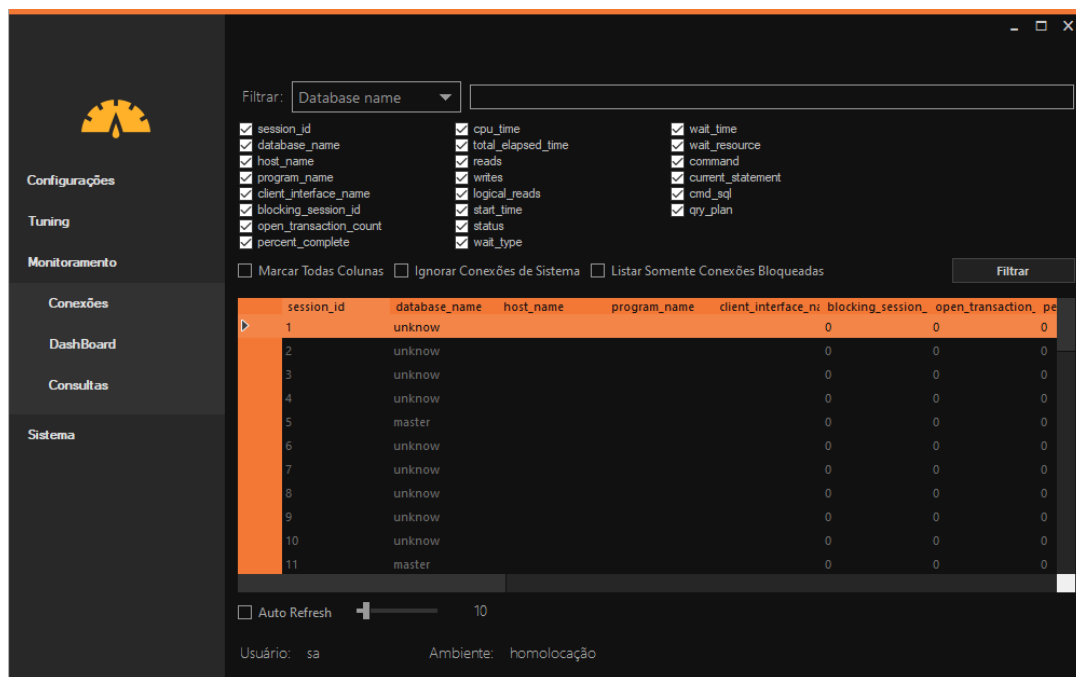
Fonte: Autoria Própria.

Tabela 26 - UC010 Monitorar Conexões

UC010	Monitorar Conexões
Descrição	Nesta tela o ator conseguirá monitorar as conexões ativas na instância de banco de dados, de forma que seja possível visualizar seu detalhe e caso necessário derrubar a sessão.
Atores	Usuário com o perfil “Monitoramento” ou administradores.
Pré-condições	Ter o ambiente selecionado.
Pós-condições	Ter permissão de visualizar o status do servidor e derrubar sessões.
Fluxos de evento	1. O usuário pode realizar o filtro das sessões ativas no banco de dados, variando entre as opções de “Banco de dados”, “Host”, “Status”, “Reads”, “Writes”, “CPU” e “Session ID”, de forma que seja possível identificar uma possível sessão.
Fluxos Alternativos	1. Para imprimir relatórios, clique com o botão direito sobre a grade de seleção e seccione o menu “Gerar relatório” e escolha entre os modelos de relatórios disponíveis. 2. Para visualizar os detalhes da sessão desejada, selecione a sessão na grade de seleção, clicando com o botão direito e selecionando a opção “Ver Detalhes”. 3. Para derrubar a sessão o usuário deverá selecionar a sessão na grade de seleção e clicar com o botão direito, escolhendo a opção “Matar Sessão”.
Requisitos Especiais	Não há.
Ponto de relacionamento	Não há.

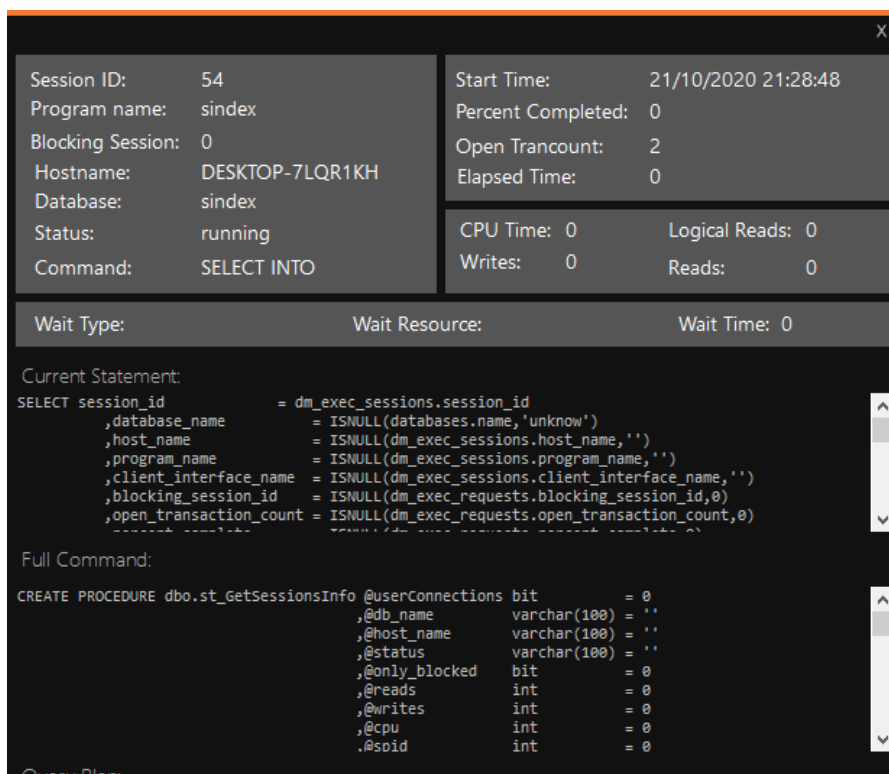
Fonte: Autoria Própria.

Figura 18 - Tela de Sessões



Fonte: Autoria Própria.

Figura 19 - Tela de Detalhes da Sessão



Fonte: Autoria Própria

Tabela 27 - UC011 Editar Índices

UC011

Descrição

Atores

Pré-condições

Pós-condições

Fluxos de evento

Fluxos Alternativos

Requisitos Especiais

Ponto de relacionamento

Editar Índices

Nesta tela o ator conseguirá pesquisar índices criados na base e conseguir editá-los.

Usuário e administrador.

Ter autenticado no sistema.

Não há.

1. Para apagar o índice, selecione o mesmo na grade de seleção e clique em “Apagar Índice”.

2. Para criar um índice, selecione o mesmo na grade de seleção e clique em “Criar Índice”.

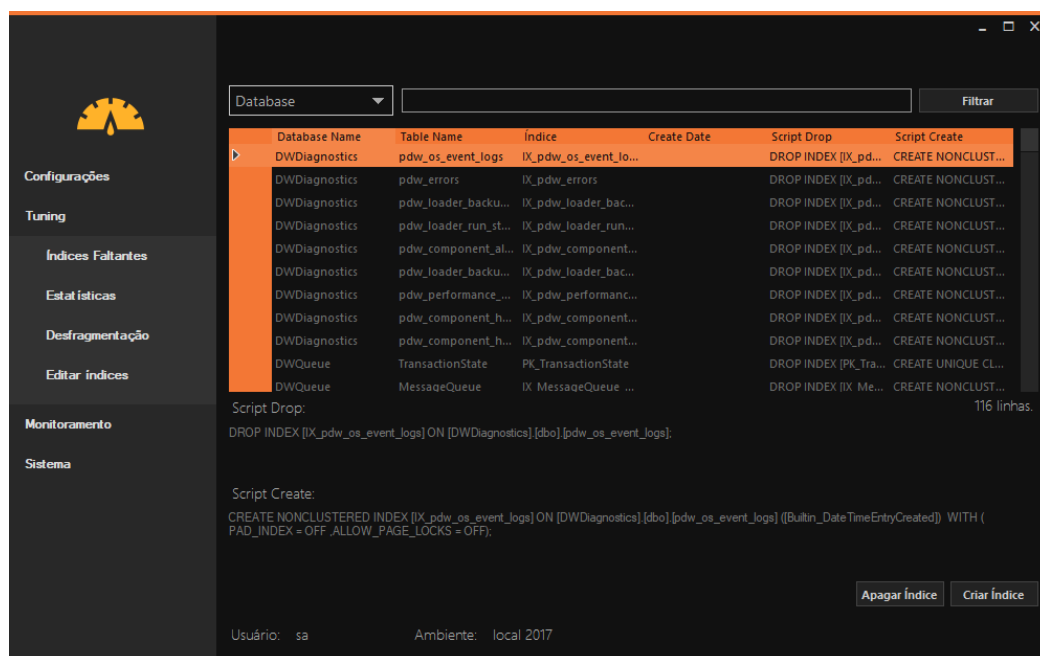
1. Para imprimir relatórios, clique com o botão direito sobre a grade de seleção e selecione o menu “Gerar relatório” e escolha entre os modelos de relatórios disponíveis.

O usuário deve ter permissão para apagar e criar índices.

Não há.

Fonte: Autoria Própria.

Figura 20 - Tela de Edição de Índices



Fonte: Autoria Própria,

Tabela 28 - UC012 Monitorar Consultas

UC012

Descrição

Monitorar Consultas

Nesta tela o ator conseguirá visualizar as

Atores

Pré-condições

Pós-condições

Fluxos de evento

Fluxos Alternativos

Requisitos Especiais

Ponto de relacionamento

consultas que consumiram mais recursos do servidor de banco de dados, de forma que seja possível identificar os comandos que estão lentos e gerando possíveis problemas de desempenho no ambiente.

Usuário com o perfil “Monitoramento” ou administradores.

Ter o ambiente selecionado.

Ter permissão de visualizar o status do servidor.

1. O usuário pode realizar das consultas mais pesadas na instância de dados, podendo realizar o filtro por banco de dados e limitar os resultados que podem retornar em número de linhas.

1. Para imprimir relatórios, clique com o botão direito sobre a grade de seleção e seccione o menu “Gerar relatório” e escolha entre os modelos de relatórios disponíveis.

2. Para visualizar os detalhes da consulta desejada, selecione a consulta na grade de seleção, clicando com o botão direito e selecionando a opção “Ver Detalhes”.

Não há.

Não há.

Fonte: Autoria Própria.

Figura 21 - Tela de Consultas

Data de Execução	Última Execução	Tempo Total CPU	Média de CPU	Média de Leituras	Média de Escritas	Tempo Médio de Execução	Quantidade de Execução	Banco de Dados	Média de Threads	Média MAXDOP	Declaração
21/10/2...	21/10/2...	3,9219	3,9219	0	8268	4,1729	1	index	0	1	SELECT...
21/10/2...	21/10/2...	0,3452	0,069	0	0	0,0851	5		0	1	SELECT...
21/10/2...	21/10/2...	0,17	0,17	1121	0	0,7047	1		0	1	SELECT...
21/10/2...	21/10/2...	0,1288	0,0644	0	2	0,0645	2		0	1	INSERT...
21/10/2...	21/10/2...	0,1257	0,0629	10	130	0,0637	2		0	1	INSERT...
21/10/2...	21/10/2...	0,0974	0,0974	744	0	0,4348	1		0	1	SELECT...
21/10/2...	21/10/2...	0,0933	0,0933	320	0	0,1771	1		0	1	SELECT...
21/10/2...	21/10/2...	0,0851	0,0851	0	0	0,111	1		0	1	SELECT...
21/10/2...	21/10/2...	0,0828	0,0828	36	758	0,0845	1		0	1	UPDAT...
21/10/2...	21/10/2...	0,0635	0,0635	16	0	0,0854	1		0	1	SELECT...
21/10/2...	21/10/2...	0,0611	0,0611	0	230	0,0611	1		0	1	UPDAT...
21/10/2...	21/10/2...	0,0594	0,0594	0	0	0,0738	1		0	1	SELECT...

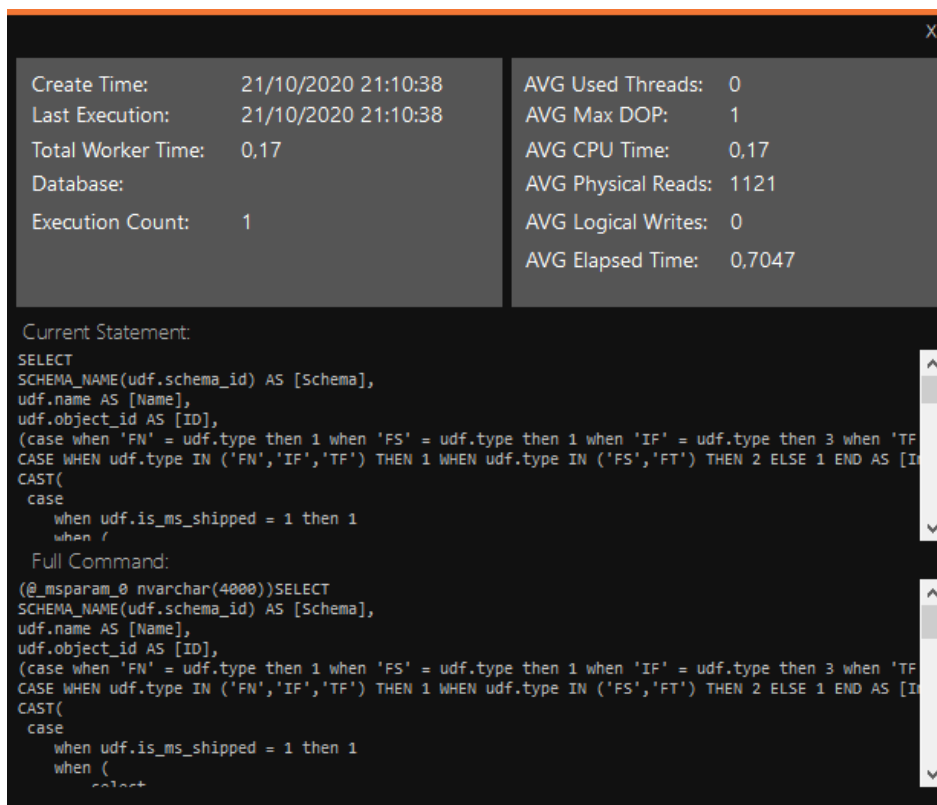
Comando:

```
SELECT
  SCHEMA_NAME(udf.schema_id) AS [Schema],
  udf.name AS [Name],
  udf.object_id AS [ID],
  (case when FN = udf.type then 1 when FS = udf.type then 1 when IF = udf.type then 3 when TF = udf.type then 2 when FT = udf.type then 2 else 0 end)
  AS [Function Type],
  CASE WHEN udf.type IN (FN,IF,TF) THEN 1 WHEN udf.type IN (FS,FT) THEN 2 ELSE 1 END AS [Implementation Type],
  CAST...
```

Usuário: sa Ambiente: homologação

Autoria: Própria.

Figura 22 - Tela de Detalhes da Consulta



Fonte: Autoria Própria.

Tabela 29 - UC013 Editar Usuários

UC013

Descrição

Atores

Pré-condições

Pós-condições

Fluxos de evento

Fluxos Alternativos

Requisitos Especiais

Ponto de relacionamento

Editar Usuários

Nesta tela o ator conseguirá gerenciar os usuários cadastrados na ferramenta, de forma que seja possível bloquear e desbloquear usuários,

Usuário e administrador.

Ter autenticado no sistema.

Não há.

1. Para editar o usuário o ator deverá atualizar as informações nos campos e clicar em atualizar.

2. Para excluir um usuário o administrador deverá selecionar o usuário que deseja excluir na grade de seleção e clicar em excluir.

Para bloquear um usuário, deverá ser marcada a opção bloqueado para o usuário selecionado e clicar em selecionar.

Para atualizar informações de outros usuários, o usuário autenticado deve estar como administrador.

Não há.

Fonte: Autoria Própria.

Figura 23 - Tela de Edição de Usuários

Usuário:

Usuário	e-mail	Bloqueado	Perfil
sa	adrian.hideki.br@gmail.com	<input type="checkbox"/>	administrador

1 linha.

Usuário: sa
 Senha:
 Confirmar Senha:
 Perfil:
 e-mail:
☐ Bloqueado

Usuário: sa Ambiente: homologação

Fonte: Autoria Própria.

3.2. Requisitos não funcionais

- O sistema operacional deve ser Windows 8.1 e superior;
- O banco de dados da aplicação deve ser SQL Server, preferencialmente a versão 2016 ou superior;
- Os bancos de dados que serão analisados e o servidor de banco deve ser SQL Server;
- O usuário do SQL Server que será responsável por gerar os relatórios deve ter permissão *dbowner*;
- O sistema deverá entregar as informações via relatório ou tela;
- O processamento será cliente-servidor;
- As informações coletadas podem ser armazenadas no banco de dados da aplicação;
- O sistema será desenvolvido em C#;

- O acesso ao sistema será através de um software instalado localmente no terminal;
- O armazenamento das informações será realizado de forma híbrida, onde parte das informações serão salvas no disco rígido e outra parte no banco de dados;

3.2.1. Manutenção

A manutenção dos computadores será realizada de forma periódica conforme o padrão organizacional do usuário, todavia é recomendado que seja feita uma manutenção nos equipamentos onde o software está instalado, de forma que seja otimizado o espaço em disco através da limpeza de arquivos temporários, assim como a limpeza física dos equipamentos, evitando a depreciação dos materiais. A manutenção no sistema será feita de forma de solicitações dos clientes e entregas, onde será feito o suporte tendo em busca melhorias e correções, a cada mês será feita uma atualização de release com todas as melhorias e correções realizadas, buscando sempre incentivar o usuário a manter-se atualizado.

3.2.2. Suporte

O suporte será fornecido através de interações por e-mails do cliente e pelo repositório da plataforma, onde as mensagens serão analisadas e respondidas, caso seja uma dúvida ainda não presente nas documentações, a mesma será adicionada e será indicado ao cliente onde consultar, caso seja um erro, o mesmo será corrigido e disponibilizado na próxima release (sendo possível gerar uma versão específica para um determinado erro, dependendo da gravidade do mesmo), caso seja enviado uma sugestão de melhoria, a implementação da mesma será analisada e disponibilizada na próxima release caso seja pertinente ao software.

3.2.3. Infraestrutura

Os dados do aplicativo serão salvos de forma híbrida, de forma que as informações sejam armazenadas no terminal onde o software foi instalado e parte das demais informações no banco de dados. Será necessário criar um banco de dados específico para a aplicação, no mesmo servidor dos bancos de dados que serão analisados pela ferramenta. O servidor que será analisado precisará ter o SQL

Server instalado, sendo necessário ter a versão 2008 ou superior. Os requisitos mínimos para a instalação do software é 4 GB de memória RAM, 1 GB de espaço livre em disco, o sistema operacional deve ser Windows 8.1 ou superior, sendo necessário a instalação dotnet framework 4.7.2 ou superior.

3.2.4. Segurança

Deverá ser programado uma rotina automática de backup no banco de dados, para salvar os dados da aplicação, a periodicidade deverá ser definida pelo padrão da organização, sendo o recomendado de no mínimo de uma semana e no máximo de um dia. O aplicativo deve ser instalado no diretório local da aplicação, evitando salvar o aplicativo na rede, que pode gerar acesso indevido ao aplicativo. O servidor de banco de dados pode ser tanto local quanto hospedado em algum servidor, pois o aplicativo está analisando as bases do servidor que a aplicação for configurada.

3.3. Métodos para controle de segurança do sistema

Com a popularização dos sistemas de informações, cada vez mais a segurança dos sistemas se tornam primordiais, sendo até pré-requisito e diferencial para que os clientes contratem determinado sistema. Os principais métodos de segurança envolvem desde segurança física até lógica, dentre as funcionalidades de login, perfis de acesso, rede, e acesso ao servidor da aplicação.

Segundo Tutida (2019) “a maioria dos outros processos de segurança são centrados no usuário”, isso mostra que um dos maiores riscos de segurança em um sistema pode ser os usuários, de forma que uma informação vazada ou ações indevidas podem gerar riscos que afetam o funcionamento do sistema.

Devido a importância da segurança das informações que o usuário tem acesso as aplicações devem ser intuitivas e restritivas com relação ao que o usuário pode e não pode fazer, com base no seu nível de acesso, outro ponto interessante é como a empresa em si faz a gestão de segurança da rede, onde é possível limitar o acesso as redes e controlar o acesso dos colaboradores aos terminais.

3.3.1. Controle de Segurança Lógica

A segurança lógica é um dos itens mais importantes, segundo Flach (2020) pode ser definida como:

Segurança Lógica é a forma como um sistema é protegido, seja por softwares ou regras de restrições de acesso. Normalmente é considerada como proteção contra ataques, mas também significa proteção de sistemas contra erros não intencionais, como remoção acidental de importantes arquivos de sistema ou aplicação. Fazendo com que o Emprego de recursos tecnológicos nos processos, como por exemplo, utilização de firewalls, antivírus, anti-spam entre outros, sejam extremamente necessários.

Um sistema deve ter tratamentos para que não tenha brechas que possam ser utilizadas pelos usuários para fazer algo não esperado que possam ocasionar problemas, por isso o controle de permissões no sistema é essencial para os softwares, assim como apresentar aos usuários a importância de determinados processos e como eles podem afetar o sistema se não utilizados corretamente. Um ponto importante também na estrutura onde o software é instalado, pois envolve a segurança lógica de rede e a responsabilidade em grande parte é da organização, onde a mesma deve controlar os acessos aos terminais e determinados caminhos de rede, um acesso liberado indevidamente pode gerar grandes transtornos.

Tabela 30 - Permissões de acesso

Usuário	Permissões de acesso
Administrador	Acesso a todas as funcionalidades e pode editar usuários
Monitoria	Acesso somente a rotinas de monitoramento
Performance	Tem acesso apenas as rotinas de melhoria de performance

Fonte: Autoria própria.

O sistema terá uma política de senhas, que não permite senhas menores que seis dígitos e também não pode conter somente números. A conta é bloqueada após três tentativas inválidas de acesso ao sistema e poderá ser desbloqueada somente pelo administrador.

3.3.2. Plano de Contingência

O plano de contingência de um sistema deve sempre abordar modos de recuperação e de alta disponibilidade, onde sejam minimizados o tempo de inatividade do sistema, para a maioria dos sistemas no mercado é implementado rotinas de backup dos dados, de forma que caso o sistema sofra algum problema seja possível restaurar as informações do último backup realizado. O recomendado é que os encarregados na organização pelo banco de dados, implementem rotinas programadas de backup (automatizado), de forma que ao menos uma vez por dia seja realizada esta operação, caso o sistema apresente algum problema que seja necessário voltar o backup, as informações estarão iguais ao último backup realizado, minimizando as perdas das informações em caso de alguma catástrofe.

3.4. Layout dos Relatórios

3.4.1. Relatório de índices ausentes

Neste relatório será apresentado os índices que podem ser criados para melhorar o desempenho de determinados comandos executados nos bancos de dados.

3.4.1.1. Tela do relatório

Ao clicar com o botão direito sobre os registros da tela, aparecerá a opção de imprimir (padrão, grid e excel).

Figura 24 - Tela de Índices Ausentes



Fonte: Autoria própria.

3.4.1.2. Exemplo do relatório

Figura 25 - Relatório de Índices Ausentes

Database	Table	Impacto	Última Leitura	Script
Adventure Works2017	SalesOrderDetail	79,66	12/10/2020 16:18:11	CREATE INDEX [IX_SalesOrderDetail_UnitPrice] ON [AdventureWorks2017].[Sales].[SalesOrderDetail] ([UnitPrice]) INCLUDE ([CarrierTrackingNumber], [OrderQty], [ProductID], [SpecialOfferID], [UnitPriceDiscount], [LineTotal], [rowguid], [ModifiedDate])

Data: 12/10/2020 16:23:40

1

Fonte: Autoria Própria.

3.4.1.3. Instruções SQL

Figura 26 - Instruções SQL para gerar o relatório de índices ausentes

```
CREATE PROCEDURE dbo.st_GetMissingIndexes @table varchar(100) = ''
,@database varchar(100) = ''
AS
BEGIN
IF OBJECT_ID('tempdb..#tb_indexes') IS NOT NULL
DROP TABLE #tb_indexes;

SELECT DatabaseName = db_name(dm_mid.database_id)
,Avg_Estimated_Impact = dm_migs.avg_user_impact*(dm_migs.user_seeks+dm_migs.user_scans)
,Last_User_Seek = dm_migs.last_user_seek
,TableName = OBJECT_NAME(dm_mid.OBJECT_ID,dm_mid.database_id)
,Create_Statement = 'CREATE INDEX [IX_' + OBJECT_NAME(dm_mid.OBJECT_ID,dm_mid.database_id) + '_] ' + REPLACE(REPLACE(REPLACE(ISNULL(dm_mid.equality_columns,''),',','_'),' ',''),',','')
+ CASE WHEN dm_mid.equality_columns IS NOT NULL AND dm_mid.inequality_columns IS NOT NULL THEN ' '
ELSE ''
END
+ REPLACE(REPLACE(REPLACE(ISNULL(dm_mid.inequality_columns,''),',','_'),' ',''),',','')
+ ' ON ' + dm_mid.statement
+ ' (' + ISNULL(dm_mid.equality_columns,'')
+ CASE WHEN dm_mid.equality_columns IS NOT NULL AND dm_mid.inequality_columns
IS NOT NULL THEN ',' ELSE '' END
+ ISNULL(dm_mid.inequality_columns, '')
+ ')'
+ ISNULL(' INCLUDE (' + dm_mid.included_columns + '), ''')
= CAST(0 AS bit)
,Selecionado
INTO #tb_indexes
FROM sys.dm_db_missing_index_groups dm_mig
INNER JOIN sys.dm_db_missing_index_group_stats dm_migs
ON dm_migs.group_handle = dm_mig.index_group_handle
INNER JOIN sys.dm_db_missing_index_details dm_mid
ON dm_mig.index_handle = dm_mid.index_handle
WHERE dm_mid.database_id IN (SELECT db_id([database_name]) FROM [dbo].[database] WHERE [database].ativo = 1)
ORDER BY Avg_Estimated_Impact DESC;

SELECT *
FROM #tb_indexes
WHERE EXISTS(SELECT 1
WHERE ISNULL(@table,'') = ''
UNION ALL
SELECT 1
WHERE TableName LIKE '%'+ISNULL(@table,'')+'%')
AND EXISTS(SELECT 1
WHERE ISNULL(@database,'') = ''
UNION ALL
SELECT 1
WHERE DatabaseName LIKE '%'+ISNULL(@database,'')+'%')
OPTION(RECOMPILE);
END
```

Fonte: Autoria Própria.

3.4.1.4. Ferramenta utilizada para desenvolver e apresentar o relatório

A ferramenta utilizada para gerar o relatório é o ReportViewer disponível para a linguagem de programação C#.

3.4.2. Relatório de estatísticas

Neste relatório será apresentado as estatísticas dos bancos de dados, de forma que seja possível identificar a última atualização realizada, assim como gerar os comandos para fazer a atualização da mesma.

3.4.2.1. Tela do relatório

Ao clicar com o botão direito sobre os registros da tela, aparecerá a opção de imprimir (padrão, grid e excel).

Figura 27 - Tela de Atualização de Estatísticas

Database: DWDiagnos... Quantidade de dias desde a última atualização: 0 Filtrar

Opções de geração do script de atualização de estatística:

☒ Full Scan
 ☐ Sample Rows: 0
☐ Resample
 ☐ Sample Percent: 0
☐ Somente tabelas com dados

Estatística	Tabela	Criado Automático	Dias Última Atualização	Data de Atualização	Banco de Dados	Script	Selecionado	
PK_pdw_dia...	pdw_diagnos...	<input type="checkbox"/>	44114	01/01/1900	DWDiagnos...	UPDATE STATI...	<input type="checkbox"/>	
PK_pdw_com...	pdw_compon...	<input type="checkbox"/>	26	16/09/2020 2...	DWDiagnos...	UPDATE STATI...	<input type="checkbox"/>	
IX_pdw_com...	pdw_compon...	<input type="checkbox"/>	26	16/09/2020 2...	DWDiagnos...	UPDATE STATI...	<input type="checkbox"/>	
PK_pdw_os_e...	pdw_t...	Atualizar estatísticas selecionadas			3/2020 2...	DWDiagnos...	UPDATE STATI...	<input type="checkbox"/>
IX_pdw_os_ev...	pdw_t...	Gerar relatório			Padrão	ostics UPDATE STATI...	<input type="checkbox"/>	
PK_pdw_perf...	pdw_perform...	<input type="checkbox"/>	44114	01/0	Grid	ostics UPDATE STATI...	<input type="checkbox"/>	
IX_pdw_perfo...	pdw_perform...	<input type="checkbox"/>	44114	01/0	Excel	ostics UPDATE STATI...	<input type="checkbox"/>	
PK_pdw_load...	pdw_loader...	<input type="checkbox"/>	44114	01/01/1900	DWDiagnos...	UPDATE STATI...	<input type="checkbox"/>	
IX_pdw_load...	pdw_loader...	<input type="checkbox"/>	44114	01/01/1900	DWDiagnos...	UPDATE STATI...	<input type="checkbox"/>	

Script de atualização de estatísticas:

```
UPDATE STATISTICS [DWDiagnosics].[dbo].[pdw_diagnostics_sessions]([PK_pdw_diag__08F643FBCCFD4111]) WITH FULLSCAN;
```

Fonte: Autoria Própria.

3.4.2.2. Exemplo do relatório

Figura 28 - Relatório de Atualização de Estatísticas

Database	Table	Estatística	Data de Atualização	Script
DWDiagno stics	pdw_diagnostics_ sessions	PK_pdw_di ag__08F643 FBCCFD411 1	01/01/1900 00:00:00	UPDATE STATISTICS [DWDiagnosics]. [dbo].[pdw_diagnostics_sessions] ([PK_pdw_diag__08F643FBCCFD4111]) WITH FULLSCAN;
DWDiagno stics	pdw_component_ alerts_data	PK_pdw_co mponent_al erts_data	16/09/2020 20:20:21	UPDATE STATISTICS [DWDiagnosics]. [dbo].[pdw_component_alerts_data] ([PK_pdw_component_alerts_data]) WITH FULLSCAN;
DWDiagno stics	pdw_component_ alerts_data	IX_pdw_com ponent_aler ts_data	16/09/2020 20:20:21	UPDATE STATISTICS [DWDiagnosics]. [dbo].[pdw_component_alerts_data] ([IX_pdw_component_alerts_data]) WITH FULLSCAN;
DWDiagno stics	pdw_os_event_lo gs	PK_pdw_os_ _event_logs	16/09/2020 20:20:21	UPDATE STATISTICS [DWDiagnosics]. [dbo].[pdw_os_event_logs] ([PK_pdw_os_event_logs]) WITH FULLSCAN;
DWDiagno stics	pdw_os_event_lo gs	IX_pdw_os_ event_logs	16/09/2020 20:20:21	UPDATE STATISTICS [DWDiagnosics]. [dbo].[pdw_os_event_logs] ([IX_pdw_os_event_logs]) WITH FULLSCAN;

Data: 12/10/2020 16:47:24

1

Fonte: Autoria Própria.

3.4.2.3. Instruções SQL

Figura 29 - Instruções SQL para gerar o relatório de atualização de estatísticas

```
CREATE PROCEDURE dbo.st_GetUpdateStatistics @fullscan bit = 0
, @sample bit = 0
, @resample bit = 0
, @percent int = 0
, @rows int = 0
, @tablesWithData bit = 0
, @database varchar(100) = ''
, @object varchar(100) = ''
, @stat varchar(100) = ''
, @minDays int = 0
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @db_name varchar(100) = ''
    , @cmd varchar(MAX) = ''
    , @db_uid integer = 0;

    IF OBJECT_ID('tempdb..#tb_main') IS NOT NULL
        DROP TABLE #tb_main;

    CREATE TABLE #tb_main(
        statName varchar(300) COLLATE Latin1_General_CI_AS
    , objectName varchar(300) COLLATE Latin1_General_CI_AS
    , schemaName varchar(300) COLLATE Latin1_General_CI_AS
    , autoCreated bit
    , statUpdateDate datetime
    , databaseName varchar(300) COLLATE Latin1_General_CI_AS
    , scriptUpdate varchar(MAX) COLLATE Latin1_General_CI_AS
    );

    SELECT @minDays = ISNULL(@minDays, 0);

    DECLARE curDatabases CURSOR LOCAL FAST_FORWARD
    FOR SELECT databases.name
    , [database].database_uid
    FROM sys.databases
    INNER JOIN dbo.[database]
    ON [database].database_name COLLATE Latin1_General_CI_AS = databases.name COLLATE Latin1_General_CI_AS
    WHERE [database].ativo = 1
    AND EXISTS(SELECT *
        FROM dbo.fn_GetServerId() fn
        WHERE fn.server_id = [database].server_id);
```

Fonte: Autoria Própria.

3.4.2.4. Ferramenta utilizada para desenvolver e apresentar o relatório

A ferramenta utilizada para gerar o relatório é o ReportViewer disponível para a linguagem de programação C#.

3.4.3. Relatório de índices fragmentados

Neste relatório será apresentado os índices dos bancos de dados, de forma que seja possível identificar a fragmentação dos mesmos.

3.4.3.1. Tela do relatório

Ao clicar com o botão direito sobre os registros da tela, aparecerá a opção de imprimir (padrão, grid e excel).

Figura 30 - Tela de Índices Fragmentados

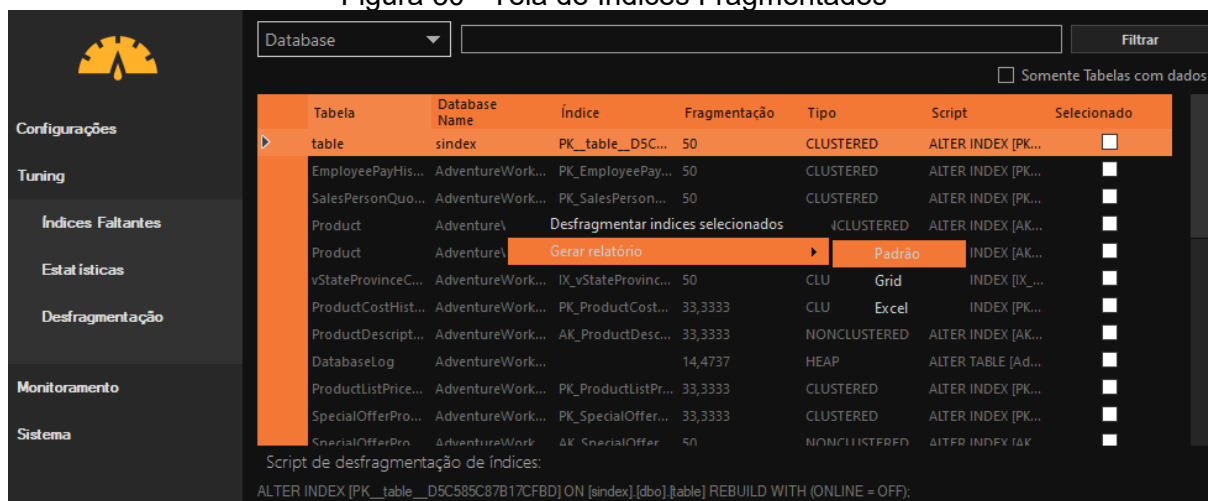


Tabela	Database Name	Índice	Fragmentação	Tipo	Script	Selecionado
table	sindex	PK_table_D5C...	50	CLUSTERED	ALTER INDEX [PK...	<input type="checkbox"/>
EmployeePayHis...	AdventureWork...	PK_EmployeePay...	50	CLUSTERED	ALTER INDEX [PK...	<input type="checkbox"/>
SalesPersonQuo...	AdventureWork...	PK_SalesPerson...	50	CLUSTERED	ALTER INDEX [PK...	<input type="checkbox"/>
Product	Adventure	Desfragmentar índices selecionados		CLUSTERED	ALTER INDEX [AK...	<input type="checkbox"/>
Product	Adventure	Gerar relatório		CLUSTERED	ALTER INDEX [AK...	<input type="checkbox"/>
Product	Adventure	Padrão		CLUSTERED	ALTER INDEX [AK...	<input type="checkbox"/>
vStateProvinceC...	AdventureWork...	IX_vStateProvinc...	50	CLU	INDEX [IX_...	<input type="checkbox"/>
ProductCostHist...	AdventureWork...	PK_ProductCost...	33,3333	CLU	INDEX [PK...	<input type="checkbox"/>
ProductDescript...	AdventureWork...	AK_ProductDesc...	33,3333	NONCLUSTERED	ALTER INDEX [AK...	<input type="checkbox"/>
DatabaseLog	AdventureWork...		14,4737	HEAP	ALTER TABLE [Ad...	<input type="checkbox"/>
ProductListPrice...	AdventureWork...	PK_ProductListPr...	33,3333	CLUSTERED	ALTER INDEX [PK...	<input type="checkbox"/>
SpecialOfferPro...	AdventureWork...	PK_SpecialOffer...	33,3333	CLUSTERED	ALTER INDEX [PK...	<input type="checkbox"/>
SerialOfferPro...	AdventureWork...	AK_SerialOffer...	50	NONCLUSTERED	ALTER INDEX [AK...	<input type="checkbox"/>

Script de desfragmentação de índices:
 ALTER INDEX [PK_table_D5C585C87B17CFBD] ON [sindex].[dbo].[table] REBUILD WITH (ONLINE = OFF);

Fonte: Autoria Própria.

3.4.3.2. Exemplo do relatório

Figura 31 - Relatório de Índices Fragmentados



Relatório de Índices Fragmentados

Database	Table	Índice	Fragmentação	Tipo	Script
sindex	table	PK_table_D5C585C87B17CFBD	50	CLUSTERED	ALTER INDEX [PK_table_D5C585C87B17CFBD] ON [sindex].[dbo].[table] REBUILD WITH (ONLINE = OFF);
Adventure Works2017	EmployeePayHistory	PK_EmployeePayHistory_BusinessEntityID_RateChangeDate	50	CLUSTERED	ALTER INDEX [PK_EmployeePayHistory_BusinessEntityID_RateChangeDate] ON [AdventureWorks2017].[HumanResources].[EmployeePayHistory] REBUILD WITH (ONLINE = OFF);
Adventure Works2017	SalesPersonQuotaHistory	PK_SalesPersonQuotaHistory_BusinessEntityID_QuotaDate	50	CLUSTERED	ALTER INDEX [PK_SalesPersonQuotaHistory_BusinessEntityID_QuotaDate] ON [AdventureWorks2017].[Sales].[SalesPersonQuotaHistory] REBUILD WITH (ONLINE = OFF);
Adventure Works2017	Product	AK_Product_ProductNumber	50	NONCLUSTERED	ALTER INDEX [AK_Product_ProductNumber] ON [AdventureWorks2017].[Production].[Product] REBUILD WITH (ONLINE = OFF);

Data: 12/10/2020 17:10:13

1

Fonte: Autoria própria.

3.4.3.3. Instruções SQL

Figura 32 - Instruções SQL para gerar o relatório de índices fragmentados

```

CREATE PROCEDURE dbo.st_GetFragmentedIndexes @database      varchar(100) = ''
                                             ,@table        varchar(100) = ''
                                             ,@type         varchar(50)  = ''
                                             ,@tablesWithData bit = 0
                                             ,@fragmentation numeric(8,4) = 0
AS
BEGIN
    DECLARE @db_name varchar(100) = ''
            ,@cmd      varchar(MAX) = ''
            ,@db_uid   integer      = 0;

    IF OBJECT_ID('tempdb..#tb_indexes') IS NOT NULL
        DROP TABLE #tb_indexes;

    CREATE TABLE #tb_indexes (
        tableName          varchar(300) COLLATE Latin1_General_CI_AS
        ,dbName             varchar(300) COLLATE Latin1_General_CI_AS
        ,indexName          varchar(300) COLLATE Latin1_General_CI_AS
        ,avg_fragmentation_in_percent numeric(8,4)
        ,indexType          varchar(100) COLLATE Latin1_General_CI_AS
        ,defrag_action       varchar(3000) COLLATE Latin1_General_CI_AS
        ,selecionado        bit
    );

    DECLARE curDatabases CURSOR LOCAL FAST_FORWARD
    FOR SELECT databases.name
           , [database].database uid
    FROM sys.databases
        INNER JOIN dbo.[database]
        ON [database].database_name COLLATE Latin1_General_CI_AS = databases.name COLLATE Latin1_General_CI_AS
    WHERE [database].ativo = 1
        AND EXISTS(SELECT *
                   FROM dbo.fn_GetServerId() fn
                   WHERE fn.server_id = [database].server_id);

    OPEN curDatabases;

```

Fonte: Autoria Própria.

3.4.3.4. Ferramenta utilizada para desenvolver e apresentar o relatório

A ferramenta utilizada para gerar o relatório é o ReportViewer disponível para a linguagem de programação C#.

3.4.4. Relatório de conexões ativas

Neste relatório será apresentado as conexões ativas a instância conectada, de forma que seja possível analisar possíveis conexões que estão prejudicando a performance do servidor.

3.4.4.1. Tela do relatório

Ao clicar com o botão direito sobre os registros da tela, aparecerá a opção de imprimir (padrão, grid e excel).

Figura 33 - Tela de Conexões ativas

Fonte: Autoria Própria.

3.4.4.2. Exemplo do relatório

Figura 34 - Relatório de conexões ativas



Relatório de Sessões

Session Id	Database	Status	Start Time	Reads	Writes	CPU
1	unknown	sleeping	12/10/2020 15:39:01	0	0	0
2	unknown	sleeping	12/10/2020 15:39:01	0	0	0
3	unknown	sleeping	12/10/2020 15:39:01	0	0	0
4	unknown	sleeping	12/10/2020 15:39:01	0	0	0
5	master	sleeping	12/10/2020 15:39:04	0	0	0
6	unknown	sleeping	12/10/2020 15:39:01	0	0	0
7	unknown	sleeping	12/10/2020 15:39:01	0	0	0
8	unknown	sleeping	12/10/2020 15:39:01	0	0	0
9	unknown	sleeping	12/10/2020 15:39:01	0	0	0
10	unknown	sleeping	12/10/2020 15:39:01	0	0	0
11	unknown	sleeping	12/10/2020 15:39:01	0	0	0
12	master	sleeping	12/10/2020 15:39:04	0	0	0
13	master	sleeping	12/10/2020 17:12:18	0	0	0
14	master	sleeping	12/10/2020 15:39:02	0	0	0
15	master	sleeping	12/10/2020 15:39:02	0	0	0

Data: 12/10/2020 17:24:04

1

Fonte: Autoria Própria.

3.4.4.3. Instruções SQL

Figura 35 - Instruções SQL para gerar o relatório de conexões ativas

```
CREATE PROCEDURE dbo.st_GetSessionsInfo @userConnections bit = 0
    ,@db_name          varchar(100) = ''
    ,@host_name        varchar(100) = ''
    ,@status           varchar(100) = ''
    ,@only_blocked     bit         = 0
    ,@reads            int          = 0
    ,@writes           int          = 0
    ,@cpu              int          = 0
    ,@spid             int          = 0
AS
BEGIN
    IF OBJECT_ID('tempdb..#tb_sessions') IS NOT NULL
        DROP TABLE #tb_sessions;

    SELECT session_id          = dm_exec_sessions.session_id
    ,database_name             = ISNULL(databases.name, 'unknown')
    ,host_name                  = ISNULL(dm_exec_sessions.host_name, '')
    ,program_name              = ISNULL(dm_exec_sessions.program_name, '')
    ,client_interface_name     = ISNULL(dm_exec_sessions.client_interface_name, '')
    ,blocking_session_id       = ISNULL(dm_exec_requests.blocking_session_id, 0)
    ,open_transaction_count    = ISNULL(dm_exec_requests.open_transaction_count, 0)
    ,percent_complete          = ISNULL(dm_exec_requests.percent_complete, 0)
    ,cpu_time                  = ISNULL(dm_exec_sessions.cpu_time, 0)
    ,total_elapsed_time        = ISNULL(dm_exec_sessions.total_elapsed_time, 0)
    ,reads                     = ISNULL(dm_exec_sessions.reads, 0)
    ,writes                    = ISNULL(dm_exec_sessions.writes, 0)
    ,logical_reads             = ISNULL(dm_exec_sessions.logical_reads, 0)
    ,start_time                = ISNULL(dm_exec_requests.start_time, '1900-01-01')
    ,status                    = ISNULL(dm_exec_sessions.status, '')
    ,wait_type                 = ISNULL(dm_exec_requests.wait_type, '')
    ,wait_time                 = ISNULL(dm_exec_requests.wait_time, 0)
    ,wait_resource              = ISNULL(dm_exec_requests.wait_resource, '')
    ,command                   = ISNULL(dm_exec_requests.command, '')
    ,current_statement         = ISNULL(SUBSTRING(dm_exec_sql_text.text, (dm_exec_requests.statement_start_offset)/2)+1,
        ((CASE dm_exec_requests.statement_end_offset
            WHEN -1 THEN DATALENGTH(dm_exec_sql_text.text)
            ELSE dm_exec_requests.statement_end_offset
        END - dm_exec_requests.statement_start_offset)/2) + 1), '')
    ,cmd_sql                   = ISNULL(dm_exec_sql_text.text, '')
    ,qry_plan                  = ISNULL(dm_exec_query_plan.query_plan, '')
    INTO #tb_sessions
    FROM sys.dm_exec_sessions WITH (NOLOCK)
    LEFT JOIN sys.dm_exec_requests WITH (NOLOCK)
        ON dm_exec_requests.session_id = dm_exec_sessions.session_id
    LEFT JOIN sys.databases WITH (NOLOCK)
        ON databases.database_id = dm_exec_sessions.database_id
    OUTER APPLY sys.dm_exec_sql_text(dm_exec_requests.sql_handle)
    OUTER APPLY sys.dm_exec_query_plan(dm_exec_requests.plan_handle);
```

Fonte: Autoria Própria.

3.4.4.4. Ferramenta utilizada para desenvolver e apresentar o relatório

A ferramenta utilizada para gerar o relatório é o ReportViewer disponível para a linguagem de programação C#.

3.4.5. Relatório de consultas

Neste relatório será apresentado as consultas que consumiram mais recursos do servidor, de forma que seja possível analisar um possível problema na instrução de consulta.

3.4.5.1. Tela do relatório

Ao clicar com o botão direito sobre os registros da tela, aparecerá a opção de imprimir (padrão, grid e excel).

Figura 36 - Tela de Consultas

	Data de Execução	Última Execução	Tempo Total CPU	Média de CPU	Média de Leituras	Média de Escritas	Tempo Médio de Execução	Quantidade de Execução	Banco de Dados	Média de Threads	Média MAXDOP	Declaração
▶	12/10/2...	12/10/2...	0,0932	0,0932	7	177	0,0932	1		0	1	INSERT INTO index...
	12/10/2...	12/10/2...	0,0263	0,0263	0	139	0,0263	1		0	1	UPDATE [stat] SE...
	12/10/2...	12/10/2...	0,0237	0,0237	30	0	0,0265	1		0	1	INSERT INTO [sinde...
	12/10/2...	12/10/2...	0,0232	0,0232	0	11	0,0232	1		0	1	UPDATE [index] S...
	12/10/2...	12/10/2...	0,0159	0,0159	0	42	0,0159	1		0	1	UPDATE [stat] SE...
	12/10/2...	12/10/2...	0,0148	0,0148	8	0	0,0155	1		0	1	INSERT INTO [sinde...
	12/10/2...	12/10/2...	0,0132	0,0044	7	4	0,0057	3	index	0	1	SELECT TOP(@Top) ...
	12/10/2...	12/10/2...	0,0123	0,0123	0	5	0,0123	1		0	1	UPDATE [index] S...
	12/10/2...	12/10/2...	0,0119	0,0119	112	16	0,0162	1		0	1	UPDATE [stat] SE...
	12/10/2...	12/10/2...	0,0116	0,0116	0	10	0,0116	1		0	1	UPDATE [index] S...
	12/10/2...	12/10/2...	0,0112	0,0112	0	11	0,0112	1		0	1	UPDATE [stat] SE...
	12/10/2...	12/10/2...	0,0112	0,0112	24	1	0,0152	1		0	1	INSERT INTO index...

Comando: INSERT INTO index.dbo [stat]

50 linhas.

Fonte: Autoria Própria.

3.4.5.2. Exemplo do relatório

Figura 37 - Relatório de Consultas

Database	AVG CPU Time	AVG Elapsed Time	AVG Reads	AVG Writes	Script
	0,0932	0,0932	7	177	<pre> INSERT INTO index.dbo.[stat](stat_name ,update_date ,create_date ,type ,table_uid ,filter ,is_autocreated ,database_uid ,columns) SELECT stat_name = stats.name ,update_date = GETDATE() ,create_date = GETDATE() ,type = " ,table_uid = [table].table_uid ,filter = ISNULL (stats.filter_definition,") ,is_autocreated = stats.auto_created ,database_uid = 15 ,ISNULL(STUFF </pre>

Data: 12/10/2020 17:38:40

Fonte: Autoria Própria.

3.4.5.3. Instruções SQL

Figura 38 - Instruções SQL para gerar o relatório de consultas

```
CREATE PROCEDURE dbo.st_GetTopQueries @Top int = 50
, @db_name varchar(100) = ''
AS
BEGIN
    SELECT TOP(@Top)
        creation_time = qs.creation_time
        , last_execution_time = qs.last_execution_time
        , total_worker_time = CAST((qs.total_worker_time+0.0)/1000000. AS numeric(18,4))
        , AvgCPUtime = CAST((qs.total_worker_time+0.0)/qs.execution_count / 1000000. AS numeric(18,4))
        , AvgPhysicalReads = CAST((qs.total_physical_reads + 0.0) / (qs.execution_count) AS bigint)
        , AvgLogicalWrites = CAST((qs.total_logical_writes + 0.0) / (qs.execution_count) AS bigint)
        , AvgElapsedTime = CAST(qs.total_elapsed_time / qs.execution_count / 1000000. AS numeric(18,4))
        , execution_count = qs.execution_count
        , query = st.text
        , query_plan = qp.query_plan
        , databaseName = ISNULL(db_name(st.dbid), '')
        , avgUsedThreads = CAST((qs.total_used_threads) / (qs.execution_count) AS bigint)
        , avgMaxDop = CAST((qs.total_dop) / (qs.execution_count) AS bigint)
        , statement_text = SUBSTRING(ST.text, (QS.statement_start_offset/2) + 1,
            ((CASE statement_end_offset
                WHEN -1 THEN DATALENGTH(ST.text)
                ELSE QS.statement_end_offset END
                - QS.statement_start_offset)/2) + 1)
    FROM sys.dm_exec_query_stats qs
    CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) st
    CROSS APPLY sys.dm_exec_query_plan(qs.plan_handle) qp
    WHERE qs.total_worker_time > 0
    AND EXISTS(SELECT 1
        WHERE @db_name = ''
        UNION ALL
        SELECT 1
        WHERE db_name(st.dbid) LIKE '%' + @db_name + '%')
    ORDER BY total_worker_time DESC;
END
```

Fonte: Autoria Própria.

3.4.5.4. Ferramenta utilizada para desenvolver e apresentar o relatório

A ferramenta utilizada para gerar o relatório é o ReportViewer disponível para a linguagem de programação C#.

4. Implementações Futuras

Com relações as implementações futuras, é possível melhorar o software em dois aspectos:

- Criar rotinas automatizadas e pré-agendadas, de forma que torne possível salvar os resultados em tabelas, criando rotinas de monitoramento para relatórios mais completos, contemplando períodos de tempos.
- Criar mais relatórios além dos já implementados na ferramenta, de forma que seja possível identificar situações mais específicas relacionadas a monitoramento e performance de banco de dados.

Com estas implementações futuras o software ficará completo, de forma que o mesmo atenda não só as validações básicas de performance mas sim validações mais detalhistas e específicas em determinadas situações.

5. Conclusão

Através deste estudo foi possível avaliar a importância da performance dos processos de software, de forma que as aplicações atuais devem retornar os dados sem apresentar lentidão, com isso foi possível desenvolver um software que realize validações de melhores práticas com relação aos objetos de banco de dados que afetam a performance de consultas e conseqüentemente processos, tendo como objetivo evitar o aumento dos tempos de execução dos processos.

O software facilitará a identificação de problemas que afetam a performance de uma consulta, consumindo metadados do produto SQL Server e exibindo amigavelmente ao usuário, com isso será disponibilizado como produto a entregar os relatórios que o aplicativo gera, de forma que seja possível tomar uma ação sobre os dados gerados.

6. Referências Bibliográficas

ALVES, G. O QUE É UM BANCO DE DADOS? Acessado em 13 de maio de 2020. Disponível em: <<https://dicasdeprogramacao.com.br/o-que-e-um-banco-de-dados/>>.

FLACH, C. SEGURANÇA LÓGICA: COMO POSSO ME PROTEGER? Acessado em 10 de outubro de 2020. Disponível em: <<https://micreiros.com/seguranca-logica-como-posso-proteger/>>.

MACHADO, D. NORMALIZAÇÃO EM BANCO DE DADOS. Acessado em 8 de abril de 2020. Disponível em: <<https://medium.com/@diegobmachado/normaliza%C3%A7%C3%A3o-em-banco-de-dados-5647cdf84a12>>.

MARCHI, K. NORMALIZAÇÃO. Acessado em 13 de maio de 2020. Disponível em: <<http://kessiamarchi.blogspot.com/2013/10/normalizacao.html>>.

MELLO, I. MODELAGEM DE DADOS: FORMAS NORMAIS. Acessado em 26 de maio de 2020. Disponível em: <<http://www.consultoriadba.com/post/2016/05/10/modelagem-de-dados-formas-normais>>.

MICROSOFT. CREATE INDEX (TRANSACT-SQL). Acessado em 22 de junho de 2020. Disponível em: < <https://docs.microsoft.com/pt-br/sql/t-sql/statements/create-index-transact-sql?view=sql-server-ver15>>.

MICROSOFT. GUIA DE ARQUITETURA E DESIGN DE ÍNDICES DO SQL SERVER. Acessado em 6 de fevereiro de 2020. Disponível em: <<https://docs.microsoft.com/pt-br/sql/relational-databases/sql-server-index-design-guide?view=sql-server-ver15>>.

MICROSOFT. Guia de arquitetura de página e extensões. Acessado em 15 de junho de 2020. Disponível em: <<https://docs.microsoft.com/pt-br/sql/relational-databases/pages-and-extents-architecture-guide?view=sql-server-ver15>>.

MICROSOFT. Índices clusterizados e não clusterizados descritos. Acessado em 7 de julho de 2020. Disponível em: < <https://docs.microsoft.com/pt-br/sql/relational-databases/indexes/clustered-and-nonclustered-indexes-described?view=sql-server-2017>>.

MICROSOFT. SQL SERVER 2019. Acessado em 13 de junho de 2020. Disponível em <<https://www.microsoft.com/pt-br/sql-server/sql-server-2019>>.

PACIEVITCH, Y. SQL SERVER. Acessado em 13 de junho de 2020. Disponível em: <<https://www.infoescola.com/informatica/sql-server/#:~:text=O%20SQL%20Server%20%C3%A9%20um,melhorar%20o%20programa%20ap%C3%B3s%20isto.>>>.

REZENDE, R. A HISTÓRIA DOS BANCO DE DADOS. Acessado em 3 de março de 2020. Disponível em: <<https://www.devmedia.com.br/a-historia-dos-banco-de-dados/1678>>.

RODRIGUES, J. Índices MySQL: Otimização de consultas. Acessado em 7 de julho de 2020. Disponível em: < <http://www.linhadecodigo.com.br/artigo/3620/indices-mysql-otimizacao-de-consultas.aspx>>.

SANCHES, A. R. DISCIPLINA: FUNDAMENTOS DE ARMAZENAMENTO E MANIPULAÇÃO DE DADOS. Acessado em 8 de abril de 2020. Disponível em: <<https://www.ime.usp.br/~andrers/aulas/bd2005-1/aula3.html>>.

SILVA, D. TECNOLOGIA - BANCO DE DADOS. Acessado em 20 de maio de 2020. Disponível em: <<https://www.estudopratico.com.br/banco-de-dados/>>.

TUTIDA, D. O QUE É SEGURANÇA DE DADOS: DEFINIÇÃO, APLICAÇÃO E TÉCNICAS. Acessado em 10 de outubro de 2020. Disponível em: <<https://encontreumnerd.com.br/blog/o-que-e-seguranca-de-dados>>.