Livro Eletrônico

Estratégia CONCURSOS

Aula 00

SGBDs para Concursos - Curso Regular 2018

Professor: Thiago Rodrigues Cavalcanti

AULA 00: Princípios sobre administração de bancos de dados.

Sumário

Apresentação do professor

Olá senhoras e senhores,

Sejam bem-vindos a mais um curso de banco de dados! Hoje começamos mais uma versão do mais completo curso de administração de banco de dados para concursos públicos. Esse curso deve cobrir um pouco do conhecimento necessário para administração e uso dos principais SGBDs.

Como eu sempre digo é um prazer imenso fazer parte desta equipe de professores do Estratégia Concursos e ter a oportunidade de apresentar um pouco do meu conhecimento e experiência em concursos públicos!

Gostaria, antes de começar de fato o conteúdo teórico desta aula, de me apresentar de forma rápida. Meu nome é Thiago, sou casado, pernambucano, tenho um filho de sete anos. Torço pelo Sport Clube do Recife. Sou cristão. Frequento a IPN – Igreja Presbiteriana Nacional. Me formei em Ciência da Computação pela UFPE. Tenho mestrado em engenharia de software na mesma instituição. Atualmente faço doutorado em economia na UnB.

Frequento academia para manter a forma, mas meu hobby mesmo é pedalar! Decidi vender o carro e viver num desafio intermodal de transporte. Ia para o trabalho de *bike* sempre que possível! Ultimamente tenho usado mais Uber/Cabify do que a magrela, mais isso é um detalhe! A pergunta é: onde eu trabalho? No Banco Central do Brasil!

Fruto de uma trajetória de dois anos de estudos diários. Aposentei as canetas em 2010. Hoje estou de licença do Banco Central para fazer doutorado que começou em março de 2017. Antes de me licenciar eu trabalhava com análise e modelagem de dados.

Minha mais recente experiência com dados, seja na administração ou modelagem, é parte de uma estratégia profissional de alinhar meu trabalho diário como servidor público com minha carreira paralela de professor e consultor de Banco de Dados (BD) e *Business Intelligence* (BI). A ideia é conseguir me especializar cada vez mais no tema, desta nova carreira dentro da TI, que o mercado está denominando de **cientista dos dados (*Data scientist*)**.

Entrei neste universo como professor de concurso há alguns anos. Desde 2012, tenho me dedicado especificamente ao conteúdo de BD e BI. Minhas experiências em cursos presencias aqui em Brasília e em diversas partes do Brasil, bem como na gravação sistemática de aulas on-line me ajudaram a desenvolver um conteúdo exclusivo para os alunos do Estratégia Concursos.

A ideia é desenvolver um material completo, recheado de questões e com diversas dicas para ajudar você no seu objetivo: **ser aprovado e nomeado**!

Para finalizar, não deixe de seguir minha página no Facebook® (profthiagocavalcanti), onde eu publico, sistematicamente, questões comentadas e dicas semanais. Tenho também uma conta no Instagram, lá eu posto motivações e dicas rápidas a respeito do conteúdo de banco de dados e análise de informações. Agora que você já me conhece! Vamos seguir em frente com o nosso curso!

Motivação para o curso

Esse é um curso regular de SGBDs. É difícil encontrar conteúdo de qualidade deste assunto focado em concursos públicos. Nosso objetivo é apresentar o conteúdo referente aos principais SGBDs que se sempre aparecem em provas de concursos públicos de TI. Esse curso vem suprir também a necessidade de diversos concurseiros que não tem o material para o seu concurso específico disponível no site do Estratégia.

Para atender a essas demandas resolvemos dividir os cursos de banco de dados em quatro módulo. Este que você está lendo trata do terceiro módulo. Um primeiro módulo trata da parte introdutória do assunto. O segundo módulo apresenta a parte avançada de banco de dados. No terceiro curso temos como ementa os assuntos relacionados aos SGBDs específicos. Por fim, o quarto e último vai tratar do conteúdo de Business Intelligence e Big Data.

Teremos muito trabalho pela frente. Por isso, montamos um **curso teórico em PDF,** baseado nas mais diversas bancas, apresentando o conteúdo observando as variadas formas de cobrança do mesmo pelas bancas examinadoras.

Teremos ainda videoaulas que apresentam o conteúdo teórico de forma detalhada para algumas partes da matéria. Existe uma força tarefa para gravação de todo o assunto, mas não temos como garantir o término deste trabalho até a data de publicação das aulas. Mas não se preocupe, nosso objetivo é garantir que você tenha capacidade e conhecimento para ser aprovado. Logo, todo conteúdo necessário para a prova estará presente nos PDFs.

Existe ainda a previsão de um curso extensivo com o conteúdo deste curso. Deveremos transmitir várias aulas ao vivo no canal do Estratégia Concursos no YouTube. Fiquem atentos! Eu sempre publico o cronograma no meu perfil do Instagram.

Vamos juntos?

Cronograma

Para proporcionar uma visão geral do assunto e fornecer uma linha de ação para o estudo da matéria dividimos o curso em **dez** aulas, sendo esta a aula 00. A aula engloba a parte introdutória da matéria de administração de dados e banco de dados. As demais aulas, seguindo a ementa do curso, são apresentadas abaixo e estão distribuídas como se segue:

EMENTA DO CURSO: Administração de banco de dados, MySQL, PostgreSQL, PL/pgSQL, Oracle, PL/SQL – Oracle, SQL Server, Microsoft Business Intelligence - SSIS SSAS SSRS.

Pois bem, e como serão distribuídas as nossas aulas?

Aula	Conteúdo
Aula 00	Administração de banco de dados
Aula 01	MySQL
Aula 02	PostgreSQL
Aula 03	PL/pgSQL
Aula 04	Oracle
Aula 05	PL/SQL – Oracle
Aula 06	SQL Server
Aula 07	Microsoft Business Intelligence - SSIS SSAS SSRS.Banco de dados Distribuídos

Definido o cronograma, vamos partir para o conteúdo da nossa aula demonstrativa.

Conceitos de Banco de Dados

1. Considerações iniciais

Quando comecei a escrever esse curso meu pensamento era o seguinte: como fornecer ao aluno segurança para fazer as questões relativas a Banco de Dados? Um *brainstorm* rápido me trouxe algumas ideias: trazer as questões mais recentes das mais diversas bancas e redigir um texto enxuto mais com todos os conceitos e explicações necessárias para levar você a marcar a alternativa correta.

Vamos tentar resolver algumas questões que vão além do escopo teórico do curso. Expandir o pensamento criando uma linha de raciocínio adequada vai facilitar a fixação do assunto. Começaremos pelos conceitos básicos relacionados a Banco de dados.

2. Conceitos básicos

Em qualquer ciência, o entendimento completo do seu conteúdo deve se basear nos conceitos fundamentais. Para usar de forma adequada um banco de dados precisamos entender aspectos teóricos do seu funcionamento. A primeira aula é focada nesses conceitos. Quando iniciamos o estudo sobre Banco de Dados, logo vem um questionamento: o que é banco de dados? Para definir esse termo podemos começar montado o conceito por suas partes.

Dados são fatos conhecidos que podem ser registrados e possuem significado implícito. Esse conceito é um pouco amplo para nosso intuito. Quando reduzimos o escopo à tecnologia da informação, temos um conceito mais enxuto para dado. Ele é a representação física de um evento no tempo e espaço que não agrega fundamento para quem o sente ou recebe. É basicamente um registro, por exemplo, 01, cinco, teste, Thiago, ...

Banco de dados é uma coleção de dados relacionados. Mas essa definição é considerada muito simplista para alguns autores. O Navathe, por exemplo, cita três propriedades implícitas que contribuem para o entendimento do termo banco de dados. Primeiramente, ele representa **algum aspecto do mundo real**, às vezes chamado de **minimundo** ou de **universo de discurso** (*UoD – Universe of Discourse*). As mudanças no minimundo devem ser refletidas no banco de dados.

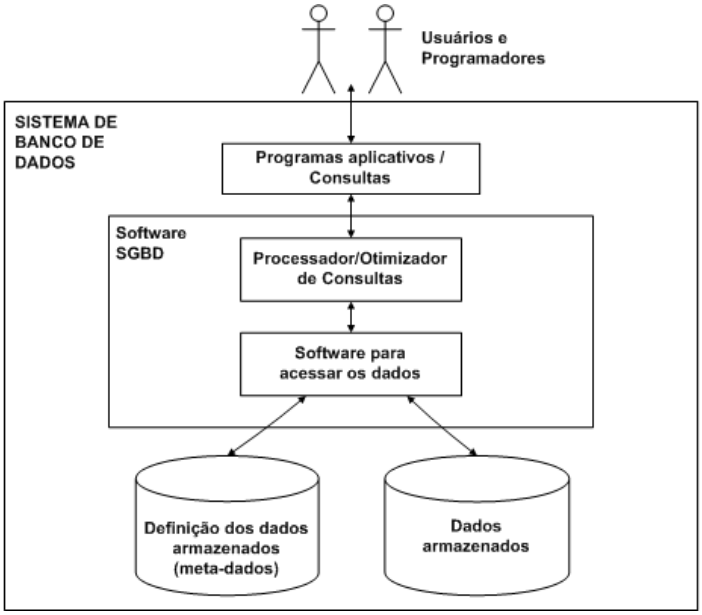
A segunda característica implícita diz que a **coleção de dados é logicamente coerente** com algum significado inerente. Uma variedade aleatória de dados **não** pode ser chamada de banco de dados. O terceiro ponto afirma que um banco de dados é construído e populado com dados para uma **finalidade específica**. Ele possui um grupo de **usuários** bem definido e

algumas aplicações, previamente concebidas, sobre as quais esses usuários estão interessados.

Outra definição de banco de dados que resume o que apresentamos até agora é um conjunto de dados **estruturados** que são confiáveis, coerentes e compartilhados por usuários que têm necessidades de informações diferentes.

Vamos agora entender a diferença entre **banco de dados**, **sistemas de gerenciamento de banco de dados (SGBD)** e **sistemas de banco de dados (SBD)**. São três conceitos diferentes para os autores dos livros teóricos sobre o assunto. Para entender essas diferenças peço que observem a figura a seguir:

Observa-se que o conjunto de usuários e programadores se comunica com o sistema de banco de dados que por sua vez faz acesso ao software do sistema de gerenciamento do banco de dados. Este, por sua vez, usa as informações presentes nos bancos de dados, representados pelos cilindros da figura, para ter acesso aos dados armazenados.



Um **Sistema de Gerenciamento de Banco de Dados (SGBD) é** um conjunto de programas que permitem armazenar, modificar e extrair informações de um banco de dados. Seu principal objetivo é proporcionar um ambiente tanto conveniente quanto eficiente para a **recuperação e armazenamento** das informações do banco de dados. Mas os SGBDs não se restringem apenas a manipulação dos dados no banco. Ele fornece variedades de programas com diferentes funcionalidades.

Podemos, por exemplo, definir um banco de dados. Essa tarefa envolve especificar os tipos, estruturas e restrições dos dados a serem armazenados. A definição ou informação descritiva do banco de dados também é armazenada pelo SGBD na forma de catálogo ou dicionário de dados, chamado de **metadados.**

É possível ainda fazer o compartilhamento que permite a diversos usuários e programas acessarem o banco de dados simultaneamente. Outras funções importantes também são providas como **proteção** do sistema contra defeitos de hardware e software, feitos por meio de redundância ou replicação, e **proteção** de segurança contra acesso não autorizados ou maliciosos.

Outros aspectos interessantes estão relacionados com o controle de transações, recuperação após falha, otimização de consultas ou do próprio SGBD, auditoria por meio e logs de sistema, enfim, são várias as funcionalidades providas pelos softwares presentes em um SGBD. Vamos agora definir o próximo conceito.

O **Sistema de banco de dados (SBD)** é considerado a união entre o banco de dados e o sistema de gerenciamento de banco de dados. Em outras palavras, consiste em uma coleção de dados inter-relacionados e de um conjunto de programas para acessá-los. Partindo da figura que apresentamos acima conseguimos construir a seguinte fórmula:

SBD = BD + SGBD + (Programa de aplicação/consulta)

01. BANCA: VUNESP ANO: 2013 ÓRGÃO: CETESB PROVA: ANALISTA DE TECNOLOGIA DA INFORMAÇÃO - ADMINISTRADOR DE BANCO DE DADOS

Sistemas de Gerenciamento de Bancos de Dados desempenham inúmeras funções. Dentre tais funções, é correto afirmar que esses sistemas

A contêm software que permite a comunicação em tempo real por meio da internet.

B implementam ferramentas de gestão de projetos como, por exemplo, gráficos de Gantt.

C mantêm a integridade dos dados inseridos no banco de dados, por exemplo, impedindo a duplicação do valor de chaves primárias.

D possuem dicionários de línguas, permitindo a tradução imediata do conteúdo de qualquer banco de dados.

E são capazes de fazer a compilação de todas as linguagens de programação orientadas a objetos.

Comentário. Essa questão trata de conceitos relacionados aos SGBDs. Mas analisaremos de forma rápida cada uma das alternativas. Na alternativa A tenta impor a necessidade de um software para conexão com a internet. Sabemos que o acesso a sua rede ocorrer por meio do provedor de Internet – utilizando a tecnologia TCP/IP, um modo de comunicação baseado no endereço de IP (Internet Protocol). Este IP é o endereço de cada um dos dispositivos conectados a rede. Para um banco de dados está disponível na internet basta que um host disponibilize o serviço por meio de uma porta TCP.

O diagrama de Gantt é um gráfico usado para ilustrar o avanço das diferentes etapas de um projeto. Os intervalos de tempo, representando o início e o fim de

cada fase, aparecem como barras coloridas sobre o eixo horizontal do gráfico. É muito usado em gerencia de projetos, mas não tem relação direta com as funcionalidades de um SGBD. Isso invalida a letra B.

Analisando tudo que foi exposto no curso até o momento podemos chegar a concluir que a alternativa C é a nossa resposta. Veremos em outra aula que manter a integridade é um dos motivos da existência das propriedades presentes em transações de bancos de dados relacionais. As propriedades são: Atomicidade, Consistência, Isolamento e Durabilidade. (ACID).

A palavra dicionário está relacionada aos metadados disponíveis no SGBD, conhecido como dicionário de dados. Quanto aos idiomas, todos os SGBDs dispõem da opção de LOCALE, através dela você define a linguagem utilizada. Agora, fazer tradução entre os diferentes idiomas, ainda não é uma funcionalidade presente.

Os SGBDs geralmente entendem apenas SQL e linguagem procedural. Para se comunicar com o servidor de banco de dados por meio de linguagens orientadas a objetos é necessário que você possua um driver que vai traduzir suas consultas para uma linguagem que seja entendida pelo SGBD.

Gabarito: C.

3. Características da abordagem de BD

Segundo Navathe, são quatro, as principais características da abordagem de banco de dados que a fazem sobressair em relação às abordagens de processamento de arquivo.

1. **Natureza de autodescrição** de um sistema de banco de dados
2. Isolamento entre programas e dados, **abstração de dados**
3. Suporte a **múltiplas visões** de dados
4. **Compartilhamento** de dados e processamento de transação multiusuário.

Esses esforços visam **reduzir a redundância** o que implica em reduzir o desperdício no espaço de armazenamento e os esforços para manter os dados comuns atualizados. Tudo realizado por meio de um único repositório!

Vejam que a lista acima pode ser caracterizada como uma enumeração e, como eu sempre digo, listas fazem parte do rol de questões de prova de concurso. Seja qual for a matéria, sempre gaste um pouco do seu tempo lendo, mais de uma vez, cada uma das listas pertencentes ao assunto. A verdade é: que não importa o grau de relevância dentro do assunto, um examinador preguiçoso sempre está propicio a utilizar deste artifício ao elaborar uma questão.

A primeira característica listada pelo Navathe é conhecida por nós como catálogo do SGBD, dicionário de dados ou metadados. Esta propriedade permite

ao SGBD gravar as definições das suas estruturas e restrições. E, quais são as descrições que podem ser gravadas? Tamanho do campo, tipo dos dados, propriedade de ser nulo ou não, valores default, entre outros. Para facilitar sua visualização pense numa definição de uma tabela em SQL. Veja o exemplo a seguir e observe algumas dessas descrições.

```
CREATE TABLE PRL_EMPLOYEE(
  ID_EMPLOYEE NUMBER,
  FK_ID_MANAGER NUMBER,
  EMPLOYEE_FIRST_NAME VARCHAR2(100) NOT NULL,
  EMPLOYEE_LAST_NAME VARCHAR2(100) NOT NULL,
  EMPLOYEE_EMAIL VARCHAR2(100) NOT NULL,
  EMPLOYEE_B_DATE DATE NOT NULL,
  EMPLOYEE_START_DATE DATE DEFAULT SYSDATE,
  EMPLOYEE_END_DATE DATE DEFAULT NULL,
  EMPLOYEE_LOCKED NUMBER(1) DEFAULT 1
)
```

A próxima característica é uma decorrência da anterior. A partir do momento em que temos um dicionário de dados, é possível excluir da estrutura dos programas a definição dos dados presentes nos mesmos. Agora isolados, dados e aplicações, criam um conceito chamado **independência de dados do programa**. Este só é possível por conta da **abstração de dados**. A abstração de dados permite a criação de diferentes níveis de modelos.

Suportar múltiplas visões parte do princípio que diferentes usuários têm diferentes necessidades sobre os dados. Se pensarmos em SQL, uma VIEW representa um subconjunto de informações referentes a uma ou mais tabelas (ou até a nenhuma tabela). Do ponto de vista mais abstrato, uma visão é a parte do banco de dados a qual um usuário ou grupo de usuário tem acesso. Porém, existe ainda a possibilidade dessa visão conter um **dado virtual** que é derivado das informações armazenadas. Imagine, por exemplo, a idade calculada a partir da data de nascimento.

Quando falamos de suporte a múltiplos usuários queremos, basicamente, permitir que diferentes usuários acessem o banco de dados ao mesmo tempo. Para garantir que isso ocorra é preciso que o SGBD forneça um mecanismo de controle de concorrência. As transações efetuadas devem levar o sistema a um estado válido, não ter conhecimento uma das outras, serem executadas sempre por completo (ou não serem executadas) e, uma vez gravadas na base, devem persistir ao longo do tempo.

Acabamos de tratar das características que o Navathe utiliza para diferenciar sistemas de arquivo de sistemas de banco de dados. Vamos agora listar as características descritas pelo Date e Silberchatz. Date chama de benefícios da abordagem de banco de dados. Quais sejam:

1. O dado pode ser compartilhado

2. A redundância pode ser reduzida
3. Inconsistências podem ser evitadas
4. Pode-se utilizar o suporte a transações
5. A integridade pode ser mantida
6. A segurança pode ser aperfeiçoada
7. Requisitos conflitantes podem ser balanceados
8. Padrões podem ser utilizados

Já Siberchatz trata das desvantagens de se utilizar um **sistema de arquivo**:

1. Redundância e inconsistência dos dados
2. Dificuldade de acesso a dados
3. Isolamento dos dados
4. Problemas de Integridade
5. Problemas de atomicidade
6. Anomalias de acesso concorrente
7. Problemas de Segurança

Lembrem-se, não precisamos decorar todas essas listas, apenas tomar conhecimento da sua existência, pois fazem parte do contexto. Elas procuram sempre expor as características que diferenciam os sistemas de arquivos dos sistemas de banco de dados.

CAIU na prova!

02. BANCA: CESPE ANO: 2013 ÓRGÃO: MC PROVA: ANALISTA DE NÍVEL SUPERIOR - TECNOLOGIA DA INFORMAÇÃO

Julgue os itens a seguir, acerca dos fundamentos e das finalidades do banco de dados.

51 Atualmente, os bancos de dados são utilizados para armazenar e processar dados de caracteres em geral, não apresentando recursos para tratar dados multimídias, como filmes e fotografias.

52 Uma característica fundamental do banco de dados e dos antigos sistemas de arquivos é o inter-relacionamento dos dados, sem redundâncias ou duplicação de dados.

53 Para definir e manter os dados em um banco é necessário o uso de sistemas de aplicação, o que caracteriza a dependência de dados, que é um fundamento do banco de dados.

Comentário. Vamos analisar as alternativas acima. Começando pelo item 51. Veja que a questão sugere que existe uma limitação nos tipos de dados armazenados em bancos de dados. Sabemos que todos os SGBDs comerciais que implementam SQL possuem o tipo de dados BLOB – Binary Large Object. Nele é possível gravar qualquer informação em formato binário como arquivos multimídias.

Observem que a alternativa 52 tentar comparar os sistemas de arquivos com os bancos de dados colocando uma das suas principais diferenças como uma similaridade entre eles. A diminuição da redundância e da duplicação ocorre primordialmente com a evolução dos sistemas de arquivo para os bancos de dados. Sendo assim, alternativa também está incorreta.

Vimos no corpo do texto teórico por estarem isolados, dados e aplicações, criam um conceito chamado independência de dados do programa. Este só é possível por conta da **abstração de dados**. A abstração de dados permite a criação de diferentes níveis de modelos. Falaremos mais sobre os níveis de abstração quando apresentarmos os modelos de dados. Mas, por enquanto, o nosso conhecimento já é suficiente para analisarmos a questão 53 como errada.

Gabarito: E E E.

4. Personagem do ecossistema de BD

Quando tratamos de grandes organizações, as atividades relacionadas a banco de dados devem ser compartilhadas entre diferentes pessoas. Trataremos agora dos dois principais papeis dentro desse processo: o administrador de banco de dados (DBA) e o administrador de dados (AD).

Só para termos uma ideia, algumas empresas do setor bancário chegam a ter algumas dezenas de ADs dentro da organização. O Bradesco tem por volta de 40 ADs. Vamos então começar falando um pouco sobre esse perfil de trabalhador especializado em Banco de dados.

O AD é a pessoa que toma as decisões estratégicas e de normas com relação aos dados da empresa. Os **administradores de dados** também podem ser conhecidos por projetista de dados. Suas tarefas são realizadas principalmente antes do banco de dados ser realmente implementado e/ou populado.

Eles são responsáveis por identificar os dados a serem armazenados e escolher estruturas apropriadas para representar esses dados. Para isso precisam se comunicar com todos os potenciais usuários a fim de entender suas necessidades e criar um projeto que as atenda suas necessidades. Eles definem então visões para cada grupo de usuários. Podemos listar ainda como atribuições do AD:

- Padronizar os nomes dos objetos criados no BD

- Gerenciar e auxiliar na definição das regras de integridade
- Controlar a existência de informações redundantes
- Trabalhar de forma corporativa nos modelos de dados da organização

Falaremos agora do BDA, ou, da pessoa que fornece o **suporte técnico** necessário para implementar as decisões. Assim, o DBA é responsável pelo controle geral do sistema em um nível técnico. Tem como **recurso primário** o banco de dados e como recursos secundários o SGBD e os softwares relacionados.

O DBA é o responsável por autorizar o acesso ao banco de dados, coordenar e monitorar seu uso, adquirir recursos de software e hardware conforme a necessidade e por resolver problemas tais como falha na segurança e demora no tempo de resposta do sistema. Segundo o Date uma lista de atividades associadas ao DBA cotem as tarefas de:

- Definir o esquema conceitual (às vezes conhecido como lógico)
- Definir o esquema interno
- Contatar com os usuários
- Definir restrições de segurança e integridade
- Monitorar o desempenho e responder a requisitos de mudanças.
- Definir normas de descarga e recarga (dumping)

03. BANCA: UERJ ANO: 2015 ÓRGÃO: UERJ PROVA: PROGRAMADOR - JAVA

O DBA, como um dos usuários do ambiente de banco de dados, interage com as seguintes interfaces:

A consulta interativa e instruções DDL

B instruções DDL e comandos privilegiados

C programas de aplicação e consulta interativa

D comandos privilegiados e programas de aplicação

Comentário. Na linha de "uma imagem vale mais do que mil palavras" uma figura que mostra os componentes do ambiente de um sistema de banco de dados. Vejam que o DBA se encontra na parte superior esquerda e que suas funções permitem entre outras coisas emitir comandos DDL e comandos privilegiados contra o banco de dados.

As instruções DDL definem o banco de dados ou realizam ajustes, alterando sua definição. O compilador DDL processa as definições de esquema especificadas e armazena as descrições dos esquemas (metadados) no catálogo do SGBD.

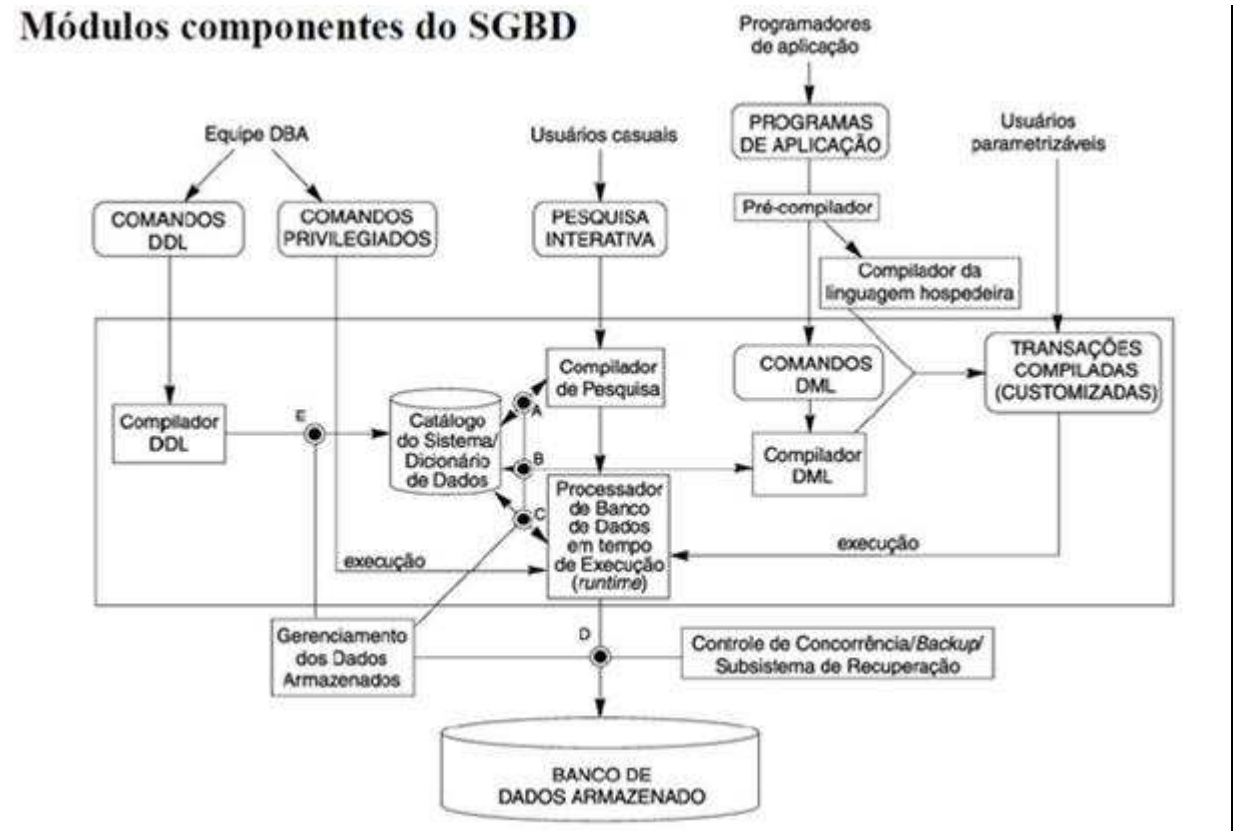
O catálogo de dados inclui informações como o nome e os tamanhos dos arquivos, nome e tipo de dados dos itens de dados, detalhes de armazenamento de cada arquivo, informações do mapeamento entre esquemas e restrições. Além disso, o catálogo armazena outros tipos de informações essenciais para os módulos do SGBD, como por exemplo: estatísticas sobre os dados armazenados que são usados pelo otimizador de consultas.

Vejam que a questão se baseou nesta figura, mas existem várias outras operações que o DBA pode executar atuando dentro das suas funções. Optamos por colocar abaixo algumas dessas operações e suas descrições:

Backup – Cria uma cópia de segurança do banco de dados, normalmente copiando o banco de dados inteiro para fita ou outro meio de armazenamento em massa. A cópia backup pode ser usada para restaurar o banco de dados no caso de uma falha catastrófica no disco. Os backups incrementais também costumam ser utilizados, e registram apenas as mudanças ocorridas após o backup anterior. O backup incremental é mais complexo, mas economiza espaço de armazenamento.

Reorganização do armazenamento do banco de dados – Executado utilizando um utilitário é usado para reorganizar um conjunto de arquivos do banco de dados em diferentes organizações de arquivo, e cria novos caminhos de acesso para melhorar o desempenho.

Monitoramento do desempenho – Monitora o uso do banco de dados e oferece estatísticas ao DBA. O BDA usa as estatísticas para tomar decisões se deve ou não reorganizar arquivos ou se deve incluir ou remover índices para melhorar o desempenho.



Gabarito: B.

5. Evolução histórica dos SGBDs

É interessante conhecer a evolução dos modelos até o NoSQL. Afinal, quando começamos a tratar as informações em sistemas como elas eram armazenadas?

Os primeiros sistemas de gerenciamento de banco de dados são implementados no final da década de 1960. Charles Bachmann desenvolveu o primeiro SGBD chamado Integrated Data Store (IDS) enquanto trabalhava na Honeywell. Esse sistema usava o **modelo de rede** onde as relações de dados são representadas como um **grafo bidirecional**.

Contudo, o primeiro SGBD que obteve sucesso comercial foi desenvolvido pela IBM chamado Information Management System (IMS). Ele usava o **modelo hierárquico** no qual as relações entre os dados são representadas como **uma árvore**. Por incrível que pareça, ainda está em uso hoje no sistema de reservas SABRE da IBM na American Airlines. Nesta época a Conference On Data Systems Languages (CODASYL) definiu um modelo de rede mais padronizado.

Esses dois modelos, em rede e hierárquico, apresentavam problemas sérios, entre eles:

- O acesso ao banco de dados feito através de operações com o ponteiro de baixo nível

- Detalhes de armazenamento dependiam do tipo de dados a serem armazenados

- Para adicionar um campo no banco era necessário reescrever o esquema subjacente de acesso/modificação, em outras palavras o modelo de dados físico.

- Ênfase nos registros a serem processados, não na estrutura global.

- O usuário tinha que conhecer a estrutura física da BD, para fim de consulta das informações.

No geral os primeiros SGBDs eram muito complexos e inflexíveis o que tornou cada vez mais difícil o trabalho quando era necessária a adição de novos aplicativos ou a reorganização dos dados. Para resolver esses e outros problemas Edgar (Ted) Codd, conhecido com o pai do modelo relacional, trabalhando no laboratório da IBM em San Jose propôs no artigo "A Relational Model of Data for Large Shared Data Banks." a definição do modelo Relacional.

Segundo Codd, o modelo fornece um meio de descrição de dados apresentando apenas a sua estrutura natural - isto é, sem sobreposição de qualquer estrutura adicional para efeitos de representação física dos dados. Assim, ele forneceu uma base para uma linguagem de dados de alto nível que permite obter a independência máxima entre dados e programas com a representação de um lado e a estrutura física da máquina do outro.

Em outras palavras, o modelo relacional consistiu na independência de dados e, na forma de acesso aos dados definida por uma linguagem. Em vez de processar um registro de cada vez, um programador pode usar o idioma para especificar operações individuais que seriam realizados em todo o conjunto de dados.

Devido à natureza técnica do artigo e a relativa complicação matemática presente no texto, o significado e proposição do artigo não foram prontamente identificados. Entretanto ele levou a IBM a montar um grupo de pesquisa conhecido como System R (Sistema R).

O projeto do Sistema R era criar um sistema de banco de dados relacional o qual eventualmente se tornaria um produto. Os primeiros protótipos foram utilizados por muitas organizações, tais como na Sloan School of Management (renomada escola de negócios norte-americana). Novas versões foram testadas com empresas de aviação para rastreamento de manufaturas em estoque.

Eventualmente o Sistema R evoluiu para SQL/DS, o qual posteriormente tornou-se o DB2. A linguagem criada pelo grupo do Sistema R foi a Structured Query Language (SQL) - Linguagem de Consulta Estruturada. Esta linguagem tornou-se um padrão na indústria para bancos de dados relacionais e, hoje em dia, é um padrão ISO (International Organization for Standardization). A

linguagem SQL era originalmente conhecida como SEQUEL (Structured English QUEry Language). Depois teve seu nome modificado para SQL por problemas de patentes.

Em meados da década de 80 tornou-se óbvio que existiam várias áreas onde bancos de dados relacionais não eram aplicáveis, por causa dos tipos de dados envolvidos. Estas áreas incluíam medicina, multimídia e física nuclear, todas com necessidades de flexibilidade para definir como os dados seriam representados e acessados.

Este fato levou ao início de pesquisas em bancos de dados orientados a objetos, nos quais os usuários poderiam definir seus próprios métodos de acesso aos dados e como estes seriam representados e acessados. Ao mesmo tempo, linguagens de programação orientadas a objetos (Object Oriented Programming - POO) tais como C++ começaram a surgir na indústria.

No início de 1990, temos a aparição do primeiro Sistema de Gerenciamento de Banco de Dados Orientado a Objetos, através da companhia Objectivity. Isso permitiu que usuários criassem sistemas de banco de dados para armazenar resultados de pesquisas como o CERN (maior laboratório que trabalha com partículas em pesquisas de física nuclear - europeu) e SLAC (Centro de Aceleração Nuclear - norte-americano), para mapeamento de rede de provedores de telecomunicações e para armazenar registros médicos de pacientes em hospitais, consultórios e laboratórios.

Fomos aos SGBDs orientados a objetos, mas voltamos para o objeto relacional, pois a grande maioria das empresas continuou utilizando o modelo relacional. Contudo esse modelo começou a apresentar outra lista de problemas ou desafios:

1. Dados na ordem de dezenas ou centenas de TB – abordagem de cluster é cara

2. Poder de crescimento elástico horizontal – controle de transação ACID torna inviável com a elasticidade

3. Fácil distribuição dos dados e/ou processamento – SGBD paralelos são caros

4. Tipos de dados variados, complexos e/ou semiestruturados – modelo de dados objeto-relacional não resolve todos os requisitos.

Tivemos então o surgimento de um novo movimento no mercado em busca de uma solução que superasse tais problemas: o movimento NoSQL. Este teve sua origem em junho de 2009, para nomear um encontro promovido por Johan Oskarsson e Eric Evans, que teve como objetivo discutir o surgimento crescente de soluções open source de armazenamento de dados distribuídos não relacionais.

Podemos considerar NoSQL uma nova onda de SGBDs, pois propõe algumas alternativas ao modelo relacional, porém com uma grande diferença histórica: o movimento NoSQL não tem como objetivo invalidar ou promover a total substituição do modelo relacional, e sim o fim do modelo relacional como bala de prata, como a única solução correta ou válida. Inclusive, é importante entender que NoSQL não significa "no SQL" (não ao SQL), mas sim "not only SQL" (não só SQL).

Curiosidade: Ao que tudo indica o termo NoSQL foi criado em 1998 por Carlo Strozzi para nomear seu projeto open source, que tinha como objetivo ser uma implementação mais leve de um banco de dados relacional, porém sua principal característica era não expor a interface SQL. Portanto é bem irônico usar o termo NoSQL, criado para nomear um banco de dados relacional, para classificar soluções de armazenamento de dados não relacionais.

Juntamente com NoSQL surge o conceito de BigData. A definição mais tradicional usa a equação dos cinco Vs. Big Data = volume + variedade + velocidade + veracidade + valor, de dados. Volume porque além dos dados gerados pelos sistemas transacionais, temos a imensidão de dados gerados pelos objetos na Internet das Coisas, como sensores e câmeras, e os dados gerados nas mídias sociais por meio de PCs, smartphones e tablets. Variedade porque estamos tratando tanto de dados textuais estruturados como não estruturados como fotos, vídeos, e-mails e tuites. E velocidade, porque muitas vezes precisamos responder aos eventos quase que em tempo real, ou seja, estamos falando de criação e tratamento de dados em volumes massivos.

O ponto de vista da veracidade também deve ser considerado, pois não adianta muita coisa lidar com a combinação "volume + velocidade + variedade" se não houver dados confiáveis. É necessário que haja processos que garantam a consistência dos dados. O último V considera que informação é poder, informação é patrimônio. A combinação "volume + velocidade + variedade + veracidade", além de todo e qualquer outro aspecto que caracteriza uma solução de Big Data, se mostrará inviável se o resultado não trouxer benefícios significativos e que compensem o investimento. Este é o aspecto do valor.

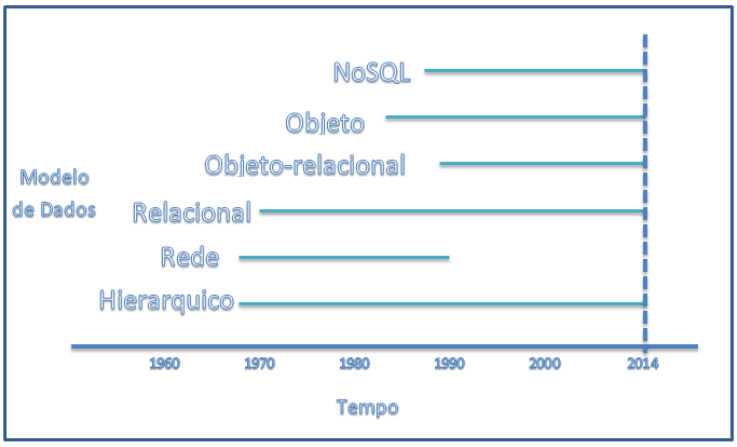
O que acontece agora? Diante destas definições é importante à implementação de SGBDs que suporte a estratégia definida pelo Big Data. A solução, portanto, foi criar soluções NoSQL. Bancos do tipo NoSQL são mais flexíveis, sendo inclusive compatíveis com um grupo de premissas que "compete" com as propriedades ACID dos SGBDs tradicionais: a BASE (*Basically Available, Soft state, Eventually consistency* – Básico disponível, Estado Leve, eventualmente consistente).

Exemplos de bancos de dado NoSQL são o Cassandra, o MongoDB, o HBase, o CouchDB e o Redis. Mas, quando o assunto é Big Data, apenas um banco de dados do tipo não basta. É necessário também contar com ferramentas

que permitam o tratamento dos volumes. Neste ponto, o Hadoop é, de longe, a principal referência.

O Hadoop é uma plataforma open source desenvolvida especialmente para processamento e análise de grandes volumes de dados, sejam eles estruturados ou não estruturados. Pode-se dizer que o projeto teve início em meados de 2003, quando o Google criou um modelo de programação que distribui o processamento a ser realizado entre vários computadores para ajudar o seu mecanismo de busca a ficar mais rápido e livre das necessidades de servidores poderosos (e caros). Esta tecnologia recebeu o nome de MapReduce.

Vamos ficando por aqui, isso é o que nos interessa para o contexto histórico. Apresentamos abaixo uma figura com uma evolução dos modelos de dados ao longo do tempo.



CAIU na prova!

04. BANCA: CESPE ANO: 2014 ÓRGÃO: TJ-SE PROVA: ANALISTA JUDICIÁRIO – BANCO DE DADOS

Acerca de bancos de dados semiestruturados e bancos de dados NOSQL, julgue os itens subsecutivos.

[86] Bancos de dados NOSQL orientados a documentos são apropriados para o armazenamento de dados semiestruturados.

[87] Para garantir a eficiência das consultas a bancos de dados semiestruturados, é fundamental a adoção de técnica de indexação que leve em consideração, além das informações, as propriedades estruturais dos dados.

[88] Devido à escalabilidade esperada para os bancos de dados NOSQL, a implementação desses bancos utiliza modelos de armazenamento de dados totalmente distintos dos utilizados em sistemas relacionais.

Comentário. Vimos que um dos desafios que os banco de dados NoSQL tenta resolver tem relação com os tipos de dados variados, complexos e/ou semiestruturados. Assim podemos considerar a alternativa 86 como **correta**.

A questão 87 envolve alguns conceitos interessantes. Começa falando sobre dados semiestruturados, por exemplo, XML ou JSON. Consultas em bancos de dados semiestruturados consideram tanto a estrutura quanto os valores. Outra questão é a criação de índice sobre um conjunto de dados semiestruturados. Para avaliar se um índice deve ou não ser criado é importante usar as informações sobre a estrutura dos dados e os valores armazenados. Neste caso, considerando a necessidade de um espaço maior para armazenamento e do custo de manutenção, a criação do índice deve melhorar a performance para ser de fato implementado. Logo, a assertiva está **correta**.

A alternativa 88 vai exigir conhecimento sobre os modelos de armazenamento utilizados por bancos de dados NoSQL. Quando tratamos de bases de dados NoSQL podemos classifica-las em quatro diferentes tipos, são eles:

Chave/valor (Key/Value): conhecidos como tabelas de hash distribuídas. Armazenam objetos indexados por chaves, e facilita a busca por esses objetos a partir de suas chaves.

Orientados a Documentos: os documentos dos bancos são coleções de atributos e valores onde um atributo pode ser multivalorado. Em geral, os bancos de dados orientados a documento não possuem esquema, ou seja, os documentos armazenados não precisam possuir uma estrutura em comum. Essa característica faz deles boas opções para o armazenamento de dados semiestruturados.

Colunar: Bancos relacionais normalmente guardam os registros das tabelas contiguamente no disco. Por exemplo, caso se queira guardar id, nome e endereço de usuários em um banco de dados relacional, os registros seriam:

Id1, Nome1, Endereço1;

Id2, Nome2, Endereço2.

Essa estrutura torna a escrita muito rápida, pois todos os dados de um registro são colocados no disco com uma única escrita no banco. Também é eficiente caso se queira ler registros inteiros. Mas para situações onde se quer ler algumas poucas colunas de muitos registros, essa estrutura é pouco eficiente, pois muitos blocos do disco terão de ser lidos.

Para esses casos onde se quer otimizar a leitura de dados estruturados, bancos de dados de famílias de colunas são mais interessantes, pois eles guardam os dados contiguamente por coluna.

O exemplo anterior em um banco de dados dessa categoria ficaria:

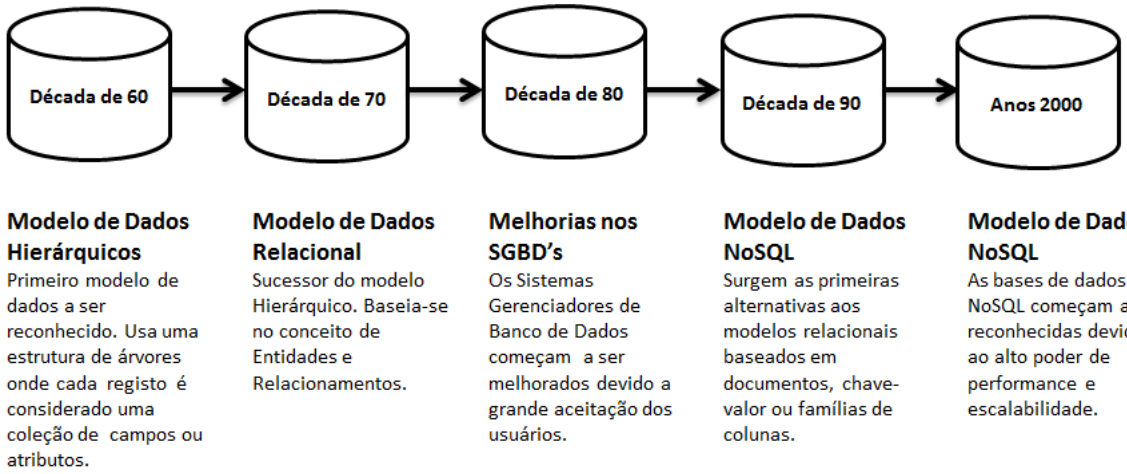
Id1, Id2; Nome1, Nome2; Endereço1, Endereço2.

Os bancos de dados de famílias de colunas são mais interessantes para processamento analítico online (OLAP). Bigtable é uma implementação da Google dessa categoria de bancos de dados.

Orientado a Grafos: diferente de outros bancos de dados NoSQL, esse está diretamente relacionado a um modelo de dados estabelecido, o modelo de grafos. A ideia desse modelo é representar os dados e/ou o esquema dos dados como grafos dirigidos, ou como estruturas que generalizem a noção de grafos. O modelo de grafos é aplicável quando "informações sobre a interconectividade ou a topologia dos dados são mais importantes, ou tão importante quanto os dados propriamente ditos". Possui três componentes básicos: os nós (são os vértices

do grafo), os relacionamentos (são as arestas) e as propriedades (ou atributos) dos nós e relacionamentos.

Agora vamos voltar a questão, precisamos responder a seguinte pergunta: nenhum dos modelos acima tem relação com bancos de dados relacionais? Ou ainda, não é possível criar estruturas em SGBDs relacionais que representem esses conceitos? Veja a figura abaixo:



Para finalizar vamos fazer apenas um comentário sobre escalabilidade: A escalabilidade em um banco de dados relacional pode ocorrer de duas formas: horizontal e vertical. A forma horizontal ocorre pela utilização de mais equipamentos e particiona a estrutura de dados de acordo com critérios estabelecidos. A forma vertical ocorre pelo aumento da capacidade do equipamento em que o sistema gerenciador de banco de dados está instalado. Bases de dados NoSQL têm como um de seus motivadores o baixo custo para realizar uma escalabilidade horizontal, o que torna possível o uso de equipamentos mais acessíveis. Além disso, proporciona um modelo de particionamento nativo (Sharding).

Gabarito C C E.

Aspectos da administração de Banco de Dados

6. Considerações Iniciais

Está parte do assunto tem o objetivo de detalhar um pouco mais as atividades elencadas no rol de tarefas do administrador de banco de dados. Estas tarefas vão garantir o funcionamento a contento do banco de dados com o nível de serviço adequado. Ao descrever cada uma das responsabilidades queremos que você entenda o que está por trás de cada atividade do ponto de vista teórico, nas aulas subsequentes vamos usar os SGBDs específicos para reforçar cada um destes conceitos.

7. Visão Geral das responsabilidades

O DBA deve ser capaz de realizar muitas tarefas para garantir que os dados da organização e o banco de dados sejam úteis, utilizáveis, disponíveis e corretos. Essas tarefas incluem design, monitoramento de performance e tuning, garantia da disponibilidade, autorizações de segurança, backup e recuperação, garantia da integridade dos dados, e, qualquer coisa que faça parte da interface com as bases de dados da empresa. Vamos examinar cada um destes tópicos.

7.1. Design do banco de dados

A primeira tarefa que a maioria das pessoas pensa de quando imaginam o trabalho de um DBA é a capacidade de criar bancos de dados. Esses bancos devem ser especialmente bem desenhados. Para projetar e criar corretamente bancos de dados relacionais, os DBAs devem entender e fazer uso das boas práticas de design relacional.

Eles devem entender a **teoria relacional** e a **implementação específica** do SGBD que está sendo utilizado para criação do banco de dados. O design de banco de dados requer uma sólida compreensão das técnicas de **modelagem de dados conceitual e lógica**. Sendo assim, a capacidade de criar e interpretar diagramas entidade-relacionamento (ER) é essencial para a concepção de um banco de dados relacional.

Além disso, o DBA deve ser capaz de **transformar um modelo de dados lógicos em um modelo de dados físicos**. O DBA deve assegurar que a concepção e implementação de banco de dados permitirão a criação de um banco de dados útil para as aplicações e os clientes. Em resumo a primeira tarefa importante do DBA:

O DBA deve ser capaz de transformar um modelo de dados numa base de dados lógica física.

Na verdade, o projeto de um banco de dados é uma habilidade importante para o DBA. No entanto, o trabalho do DBA é desproporcionalmente associado com o projeto de banco de dados. Apesar da atividade de projetar bancos de dados ser importante, é uma parte relativamente pequena do trabalho do DBA. O DBA provavelmente vai gastar mais tempo **administrando e ajustando os bancos de dados** do que originalmente na concepção e construção.

De maneira nenhuma, porém, você deve interpretar isto dizendo que o design de banco de dados não é importante. Um projeto relacional pobre pode resultar num desempenho fraco, ou em um banco de dados que não atende as necessidades da organização e com dados potencialmente imprecisos.

7.2. Monitoramento de performance e *tuning*

O segundo papel importante associado com o DBA é o **monitoramento e ajuste** de desempenho, também conhecido como *tuning*. Mas o que queremos dizer com desempenho do banco de dados? Pense, por um momento, em desempenho do banco de dados usando os conceitos familiares de oferta e demanda.

Usuários exigem informações do banco de dados. O SGBD fornece as informações para aqueles que as solicitam. A taxa de entrega (informação por alguma medida de tempo) à qual o SGBD fornece as informações pode ser denominada de desempenho do banco de dados. Lógico que não é tão simples assim, mas essa é a ideia inicial que você precisa ter em mente.

Cinco fatores influenciam o desempenho do banco de dados: **carga de trabalho (*workload*)**, **rendimento (*throughput*)**, **recursos**, **otimização** e **contenção**.

A **carga de trabalho** que é solicitado do SGBD define a demanda. É uma combinação de transações on-line, batch jobs, consultas ad hoc, consultas analíticas e comandos feitos diretamente ao sistema a qualquer momento. A carga de trabalho pode variar drasticamente ao longo do tempo.

Às vezes a carga de trabalho pode ser prevista (como processamento pesado no final do mês de folha de pagamento, ou um acesso mais leve após as 20h, quando a maioria dos usuários já não está nas estações de trabalho), mas em outros momentos, é imprevisível. A carga de trabalho global tem um grande impacto no desempenho do banco de dados.

A taxa de transferência ou ***throughtput*** define a capacidade total de processamento do hardware e software. É um composto pela velocidade de I/O, velocidade da CPU, capacidades paralelas da máquina, e pela eficiência do sistema operacional e do próprio SGBD. As ferramentas de hardware e software à disposição do sistema são conhecidas como os **recursos do sistema**.

Exemplos incluem o *kernel* do banco de dados, espaços em disco, controladores de cache, e o microcódigo.

O quarto elemento definidor do desempenho do banco de dados é a **otimização**. Todos os tipos de sistemas podem ser otimizados, mas as consultas relacionais são as únicas tarefas em que a otimização é realizada internamente ao SGBD. No entanto, existem muitos outros fatores que precisam ser otimizados (definições SQL, parâmetros de banco de dados, programação mais eficiente, e assim por diante) para permitir que o otimizador crie os caminhos de acesso mais eficientes.

Quando a demanda (carga de trabalho) para um determinado recurso é alta, a **contenção** pode acontecer. Contenção é a condição em que dois ou mais componentes da carga de trabalho estão tentando usar um único recurso de uma forma conflitante (por exemplo, duas atualizações para o mesmo dado). Quando a contenção aumenta, o rendimento diminui.

---

*Por conseguinte, o desempenho do banco de dados pode ser definido como a otimização da utilização dos **recursos** para aumentar o **rendimento** e minimizar a **contenção**, que permita a maior **carga de trabalho** possível ser processada.*

Sempre que os problemas de desempenho são encontrados por um aplicativo que usa um banco de dados, o DBA é geralmente o primeiro a ser chamado para resolver o problema. Claro que, o DBA não pode controlar o desempenho do banco de dados no vácuo. Aplicações se comunicam regularmente com outras aplicações, sistemas e componentes da infraestrutura de TI.

Um efetivo monitoramento de desempenho e estratégia de ajuste não requer apenas uma perícia sobre o SGBD, mas o conhecimento fora do âmbito da administração de banco de dados. Muitas tarefas de gerenciamento de desempenho devem ser compartilhadas entre o DBA e outros técnicos. Em outras palavras, lidar com problemas de desempenho é realmente um esforço de toda a empresa.

O DBA deve estar vigilante no monitoramento do desempenho do sistema, do banco de dados e das aplicações. Tanto quanto possível, isto deve ser conseguido utilizando software e scripts automatizados. *Polling* de tabelas do sistema e construção de alertas baseados em limites podem ser usados para identificar problemas de forma proativa.

Os alertas podem ser configurados para disparar um e-mail ao DBA quando as métricas de desempenho não estiverem dentro dos limites aceitos. Muitas

tarefas e habilidades são necessárias ao DBA para assegurar o acesso às bases de dados eficientes.

Algumas destas habilidades incluem a construção de índices apropriados, especificação de buffers e caches, alinhando da implementação de banco de dados com a infraestrutura de TI, monitoramento contínuo das bases de dados e aplicações, recuperação de banco de dados e adaptação às mudanças de negócios. Alguns fatores que podem afetar são mais dados, mais usuários, processamento adicional, e mudanças de requisitos ou regulamentação.

7.3. Garantido a disponibilidade

Disponibilidade de dados e bancos de dados é muitas vezes estreitamente alinhada com o desempenho, mas na verdade é uma preocupação separada. Claro, se o SGBD está *off-line*, o desempenho vai ser horrível, porque os dados não podem ser acessados. Mas, garantir a disponibilidade do banco de dados é um processo multifacetado.

O primeiro componente de disponibilidade é manter o SGBDs instalado e funcionando. Monitoramento e alertas automáticos podem ser usados para alertar sobre falhas do SGBD e executar alguma ação corretiva.

Bancos de dados individuais também devem ser mantidos de modo que os dados neles contidos estejam disponíveis sempre que as aplicações e os clientes exigirem. Isso requer do DBA um projeto do banco de dados para que ele possa ser mantido com interrupções mínimas, mas também para ajudar aplicativos para minimizar os conflitos quando o acesso simultâneo é necessário.

Um componente adicional de disponibilidade é minimizar a quantidade de tempo de inatividade necessário para executar tarefas administrativas. Quanto mais rápido o DBA executar tarefas administrativas que exigem que os bancos de dados sejam desligados, mais disponíveis os dados se tornam.

Cada vez mais, vendedores de SGBD e fornecedores de software independentes (ISVs) estão fornecendo utilitários que podem ser executadas em bancos de dados enquanto as aplicações leem e escrevem, sem interrupções. Mas estes geralmente requerem mais habilidade e planejamento por parte do DBA.

O DBA deve compreender todos esses **aspectos de disponibilidade** e garantir que cada aplicativo esteja recebendo o nível correto de qualidade de serviço, ou seja, adequado as suas necessidades.

7.4. Segurança e autorização

Uma vez que o banco de dados foi concebido e implementado, programadores e usuários precisarão acessar e modificar os dados no banco de

dados. Mas só os programadores e usuários autorizados devem ter acesso para evitar violações de segurança e modificação de dados inadequada. É da responsabilidade do DBA garantir que os dados estão disponíveis apenas para usuários autorizados.

Normalmente, embora nem sempre, o DBA trabalha com as características internas de segurança dos SGBD sob a forma dos comandos de SQL GRANT e REVOKE, bem como quaisquer características do DBMS para autorização grupo. A segurança deve ser administrada por muitas ações exigidas pelo ambiente de banco de dados:

- Criação de objetos de banco de dados, incluindo bancos de dados, tabelas, *views* e estruturas de programa
- A alteração da estrutura de objetos de banco
- Acessar o catálogo do sistema
- Leitura e modificação de dados em tabelas
- Criar e acessar funções definidas pelo usuário e tipos de dados
- Executar procedimentos armazenados
- Iniciar e param os bancos de dados
- Definir parâmetros e especificações do ambiente SGBD
- Execução de utilitários de BD como LOAD, RECOVER e REORG

A **segurança de banco de dados** pode ser aplicada de outras maneiras também. Por exemplo, podem ser criadas visões para bloquear colunas ou linhas sensíveis, não permitindo que os dados sejam visualizados por usuários finais e programadores. O DBA também interage frequentemente com métodos de segurança externos quando eles afetam a segurança do banco de dados (ex.: LDAP).

O DBA deve compreender e ser capaz de implementar qualquer aspecto de segurança que afeta o acesso aos bancos de dados. Uma área que deve ser de interesse particular, levando em consideração as violações de dados, é ataques de injeção SQL e como evitá-los. Enfim, o DBA deve compreender todos os aspectos da segurança que afetam o acesso a bancos de dados.

7.5. Backup e recuperação

O DBA deve estar preparado para **recuperar os dados** em caso de falhas. Uma falha pode significar diferentes coisas, desde um problema de sistema ou erro de programa a um desastre natural que aconteça em uma determinada

organização. A maioria das recuperações ocorre como um resultado de erro do software de aplicação ou de erro humano.

As falhas de hardware não são tão relevantes como costumavam ser. De fato, as estimativas dos analistas indicam que 80 por cento dos erros de aplicação são devido a falhas de software ou erro humano. O DBA deve estar preparado para recuperar o banco de dados a um estado consistente, não importa qual a causa, e a fazer tal operação o mais rapidamente possível.

O primeiro tipo de recuperação de dados que normalmente vem à mente é uma **recuperação ao estado atual,** geralmente acontece quando uma falha tira o sistema do ar. O resultado da recuperação é que o banco de dados é trazido de volta ao seu estado atual no momento da falha. As aplicações ficam completamente indisponíveis até que a recuperação esteja completa.

Outro tipo de recuperação tradicional é uma **recuperação point-in-time**. A recuperação deste tipo normalmente é realizada para lidar com um problema em nível de aplicação. As técnicas convencionais para executar uma recuperação point-in-time irão remover os efeitos de todas as transações desde um ponto específico no tempo. Isso pode trazer problemas se aconteceu alguma transação válida durante esse período que ainda precisam ser aplicadas.

Recuperação de transações é um terceiro tipo de recuperação que aborda as deficiências dos tipos tradicionais de recuperação: o **tempo de inatividade** e **perda de dados de boa qualidade**. Assim, a recuperação de transação é uma recuperação de aplicativos em que os efeitos das transações específicas durante um período especificado são removidos do banco de dados.

Portanto, a recuperação de transação é por vezes referida como a recuperação de aplicações. Perceba que esse tipo de recuperação serve para ajustar a base de dados depois de modificações inconsistentes feitas por aplicações.

A maioria dos DBAs pensa em recuperação como meio apenas para resolver os desastres, como falhas de hardware. Apesar das falhas de hardware ainda ocorrem, e dos gestores de banco de dados precisarem estar preparados para se recuperar de tais falhas, a maioria das recuperações são necessárias, como já falamos, devido a erro humano ou erros nos programas.

O DBA deve estar preparado para lidar com todos esses tipos de recuperação. Isto envolve o desenvolvimento de uma estratégia de **cópia de segurança** para assegurar que os dados não sejam perdidos no caso de um erro em software, hardware, ou um processo manual.

A estratégia deve ser aplicável ao processamento de banco de dados, por isso deve incluir cópias de imagens de arquivos de banco de dados, bem como um plano de backup/recuperação para os logs de banco de dados. Ele precisa

dar conta de qualquer atividade de arquivos que possam impactar as aplicações de banco de dados também.

7.6. Garantia das regras de governança

Assegurar o cumprimento das **regulamentações da indústria e governamentais** é uma tarefa adicional da administração de banco de dados, pelo menos em termos de implementação de controles adequados (auditoria). O DBA deve trabalhar com gestores, auditores e especialistas de negócio para compreender a normatização que se aplica a sua área e a maneira pela qual os dados devem ser tratados.

Certos aspectos da conformidade com os padrões regulatórios afetam diretamente o trabalho do DBA. Por exemplo, os **regulamentos podem conter procedimentos de segurança e autorização específicos**, requisitos de auditoria, especificações de backup de dados e alteração nos procedimentos de gestão. Para garantir o cumprimento, no entanto, uma documentação mais rigorosa pode ser exigida, ou talvez um maior grau de diligência ou de automação (tais como rastreamento de auditoria mais detalhado).

Outros aspectos de conformidade regulatória podem exigir que o DBA adote diferentes técnicas, táticas e habilidades. Por exemplo, os regulamentos de retenção de dados podem necessitar que as informações fossem mantidas por muito tempo além do necessário numa base de dados de produção, o que requer capacidades de arquivamento do banco de dados. Ou determinados dados que podem necessitar de proteção adicional, necessitando de mascaramento de dados ou, em alguns casos, de criptografia.

Os DBAs não devem ser incumbidos de entender os regulamentos em qualquer profundidade, nem devem estabelecer padrões pelos quais a organização está em conformidade com os regulamentos. No entanto, os DBAs irão se envolver para ajudar a definir os controles e procedimentos para projetos de conformidade, especificamente e especialmente no que diz respeito ao tratamento de dados.

7.7. Garantia da integridade dos dados

Um banco de dados deve ser projetado para armazenar os dados corretos no caminho correto sem que os dados se tornem danificados ou corrompidos. Para assegurar esse processo, o DBA implementa as regras de integridade usando recursos do SGBD. Três aspectos da integridade são relevantes para nossa discussão de bases de dados: **físico, semântico** e **interno**.

Problemas físicos podem ser manipulados usando recursos do SGBD como domínios e tipos de dados. O DBA escolhe o tipo de dados apropriado para cada coluna de cada tabela. Esta ação assegura que apenas os dados desse tipo são

armazenados no banco de dados. Isto é, o SGBD impõe a integridade dos dados em relação ao seu tipo.

Uma coluna definida como "integer" pode conter apenas números inteiros. As tentativas de armazenar valores não numéricos ou não inteiros em uma coluna definida como inteiro falhará. O DBA também pode usar restrições para delinear com maior profundidade do tipo de dados que pode ser armazenado no banco de dados para cada coluna. A maioria dos produtos de SGBDs relacionais fornecem os seguintes tipos de restrições:

- **Restrições referenciais** são usadas para especificar as colunas que definem quaisquer relações entre tabelas. Restrições referenciais são usados para implementar a integridade referencial, o que garante que todas as referências de dados em uma coluna (ou conjunto de colunas) de uma tabela são válidos em relação aos dados em outra coluna da mesma ou de uma tabela diferente.

- As **restrições de unicidade** garantem que o valor para uma coluna ou um conjunto de colunas ocorra apenas uma vez na tabela.

- As restrições ***check*** são usadas para colocar as regras de integridade mais complexas em uma coluna ou conjunto de colunas de uma tabela. As restrições de verificação são tipicamente definidas utilizando o SQL e podem ser utilizadas para definir os valores de dados que são permitidas para uma coluna ou um conjunto de colunas.

Integridade semântica é mais difícil de controlar e menos facilmente definida. Os DBAs devem estar preparados para aplicar políticas e práticas para garantir que os dados armazenados em seus bancos de dados sejam precisos, apropriados e utilizáveis. Um exemplo de uma questão semântica é a qualidade dos dados na base de dados.

Armazenar todos os dados que reúne as definições de integridade físicas especificadas para o banco de dados não é suficiente. Procedimentos e práticas precisam estar no local para garantir a qualidade dos dados. Por exemplo, uma base de dados de clientes que contém um endereço errado ou número de telefone para 25 por cento dos clientes nela armazenada é um exemplo de uma base de dados com fraca qualidade.

Não existe um método sistemático ou físico de assegurar a precisão dos dados. A qualidade dos dados é incentivada por meio de código adequado da aplicação, práticas empresariais sólidas e políticas de dados específicas. Redundância é outra questão semântica. Se elementos de dados são armazenados de forma redundante em todo o banco de dados, o DBA deve documentar esse fato e trabalhar para garantir que os procedimentos estão no lugar para manter os dados redundantes sincronizadas e precisas.

O aspecto final da integridade é uma questão interna do SGBD. O SGBDS depende de estruturas internas e código para manter as ligações, ponteiros e identificadores. Na maioria dos casos o SGBD vai fazer um bom trabalho de manter estas estruturas, mas o DBA precisa estar ciente da sua existência e como lidar quando o SGBD falhar. Integridade interna é essencial nas seguintes áreas:

- **Consistência de índices**. Um índice é realmente nada além de uma lista ordenada de ponteiros para dados em tabelas de banco de dados. Se por algum motivo o índice fica fora de sincronia com os dados, o acesso indexado pode falhar ao retornar os dados incorretos. O DBA tem ferramentas à sua disposição para procurar e corrigir esses tipos de erros.

- **Consistência de ponteiros**. Às vezes, grandes objetos multimídia não são armazenados nos mesmos arquivos físicos como outros dados. Portanto, o SGBD requer estruturas de ponteiro para manter os dados de multimídia sincronizados com os dados da tabela base. Mais uma vez, esses ponteiros podem ficar fora de sincronia se os procedimentos de administração adequados não forem seguidos.

- **Consistência de backup**. Alguns produtos de SGBD, ocasionalmente, levam cópias de segurança impróprias que efetivamente não podem ser usados para a recuperação. É essencial para identificar esses cenários e tomar ações corretivas.

No geral, garantir a integridade é uma habilidade essencial DBA. Vejam que passamos por várias habilidades usadas pelos DBAs no gerenciamento de um banco de dados. Nas próximas aulas vamos observar como cada uma destas técnicas são implementadas em cada um dos SGBDs.

Considerações finais

Chegamos, pois, ao final da aula demonstrativa!

A continuação deste assunto e uma bateria de exercícios correspondente encontram-se na próxima aula. Espero reencontrar você como um aluno efetivo.

Espero que tenha gostado! E até breve!

Thiago Cavalcanti

Referências

Fiz uma lista com alguns links de referências caso você queria se aprofundar um pouco.

i. Fundamentals of Database Systems - Ramez Elmasri, Sham Navathe Addison-Wesley, 2011 - Computers - 1172 pages
ii. Introdução a sistemas de bancos de dados - By C. J. Date - Elsevier Brasil, 2004 - 865 pages
iii. Sistema de Banco de Dados - Abraham Silberschatz, Henry F. Korth, S. Sudarshan - Editora: ELSEVIER BRASIL

ESSA LEI TODO MUNDO CONHECE:
PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.

1. 

Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.

2. 

Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).

3. 

Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).

4. 

Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).

5. 

Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).

6. 

Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).

7. 

Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.

8. 

O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.

Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.