Banco de Dados Relacional

A tecnologia de banco de dados tem evoluído rapidamente nas últimas três décadas desde a ascensão e eventual domínio dos sistemas de banco de dados relacionais (SGBDR – sistema gerenciador de banco de dados relacional, RDBMS – relational database management system). Enquanto muitos sistemas de bancos de dados especializados (orientado a objeto, espacial, multimídia, etc.) têm encontrado usuários substanciais nas comunidades científicas e engenharia, os sistemas relacionais continuam dominando a tecnologia de banco de dados para negócios empresariais.

O banco de dados relacional gerou uma indústria multi-bilionária, é o tipo mais utilizado de banco de dados no mundo de hoje, e é uma parte essencial de nossas vidas cotidianas. É muito provável que você está usando um banco de dados relacional toda vez que compra mercadorias em uma loja, faz planos de viagem com seu agente de viagens, confere um livro na biblioteca, ou faz uma compra na Internet.

O desenho de um banco de dados relacional tem evoluído de uma arte para uma ciência que tem sido parcialmente implementado como uma ferramenta de ajuda de desenvolvimento de software. Muitas dessas ferramentas têm surgido como um componente do banco de dados – CASE, computer-aided software engineering. Muitas delas oferecem a capacidade de modelagem interativa, usando um modelo de dados simplificado aproximado.

O desenho lógico que é a estrutura de relacionamento de dados básica e suas definições em um sistema particular de banco de dados é em grande parte o domínio das aplicações de *designers*. O trabalho desses designers podem ser efetivamente feito com ferramentas como o ERwim Data Modeler ou Rational Rose com Unified Modeling Language (UML – linguagem de modelagem unificada), assim como um manual simplesmente aproximado.

O desenho físico, a criação eficiente do armazenamento e mecanismos de recuperação de dados na plataforma de computação que você está usando, é tipicamente o campo de conhecimento do administrador de banco de dados (DBA – database administrator).

Hoje em dia os DBAs têm um variedade de ferramentas à venda disponível para ajudar a desenhar o banco de dados mais eficiente.

Tipos de banco de dados

Um banco de dados é uma coleção organizada de dados utilizados para a finalidade de modelar algum tipo de organização ou processo organizacional. Realmente não importa se você estiver usando papel ou um programa de software de computador para coletar e armazenar os dados. Enquanto você está reunindo dados de alguma forma organizada para uma finalidade específica, você tem um banco de dados.

Existem dois tipos de banco de dados encontrados na gestão de banco de dados, banco de dados operacionais e bancos de dados analíticos.

Bancos de dados operacionais é a espinha dorsal de muitas empresas, organizações, e instituições em todo o mundo de hoje. Este tipo de banco de dados é usado principalmente em cenários de processamento de transações on-line (OLTP), ou seja, em situações onde há necessidade de se recolher, modificar e manter os dados em uma base diária. O tipo de dados armazenados em um banco de dados operacional é

dinâmico, significando que ele muda constantemente e sempre reflete informações up-to-the-minute. Organizações, tais como lojas de varejo, companhias de manufatura, hospitais e clínicas, e editoras, usam bancos de dados operacionais, pois seus dados estão em um constante estado de fluxo.

Em contraste, bancos de dados analíticos são usados principalmente em cenários de processamento analítico on-line (OLAP), onde há a necessidade de armazenar e controlar dados históricos e dependente de tempo. Um banco de dados analítico é um recurso valioso quando há uma necessidade de acompanhar as tendências, ver dados estatísticos durante um longo período de tempo, e fazer projeções de negócios táticas ou estratégicas. Este tipo de banco de dados armazena dados estáticos, o que significa que os dados nunca (ou muito raramente) é modificado. A informação recolhida a partir de um banco de dados analítico reflete dados em um ponto instantâneo no tempo. Laboratórios químicos, geológicos, empresas de marketing e análise são exemplos de organizações que usam bancos de dados analíticos.

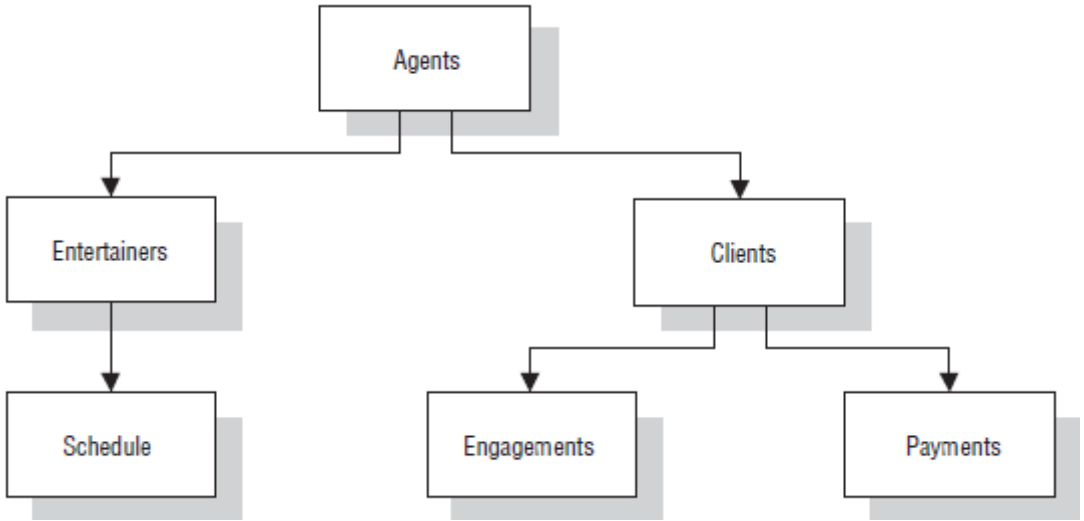
Bancos de dados analíticos costumam usar dados de bancos de dados operacionais como sua principal fonte de dados, assim pode haver certa quantidade de associações entre eles; no entanto, bases de dados operacionais e analíticos cumprem tipos muito específicos de processamento de dados, necessidades, e a criação de suas estruturas requerem metodologias de projeto radicalmente diferentes.

Os primeiros modelos de banco de dados

Nos dias que antecederam o modelo de banco de dados relacional, dois modelos de dados foram usados para manter e manipular dados – o modelo de banco de dados hierárquico e o modelo de banco de dados de rede.

Modelo de banco de dados hierárquico

Dados neste tipo de banco de dados estão hierarquicamente estruturados e é tipicamente diagramado como uma árvore invertida. Uma única tabela no banco de dados funciona como a "raiz" da árvore invertida e as outras tabelas atuam como os ramos que fluem a partir da raiz. A Figura abaixo mostra um diagrama de uma estrutura de um típico banco de dados hierárquico.



Um agente reserva diversos artistas, e cada artista tem sua própria programação. Um agente também mantém um número de clientes cujas necessidades de entretenimento são atendidas pelo agente. Um cliente reserva compromissos por meio do agente e faz os pagamentos ao agente por seus serviços.

Uma relação num banco de dados hierárquico é representado pelo termo pai/filho. Neste tipo de relação, uma tabela pai pode ser associado com um ou mais tabelas filho, mas uma única tabela filho pode ser associado com apenas uma tabela pai. Estas tabelas são explicitamente ligadas através de um ponteiro ou pela disposição física dos registros dentro das tabelas. Um usuário acessa os dados dentro deste modelo, iniciando da tabela de raiz e trabalhando para baixo através da árvore para os dados de destino. Este método de acesso requer que o utilizador esteja muito familiarizado com a estrutura do banco de dados.

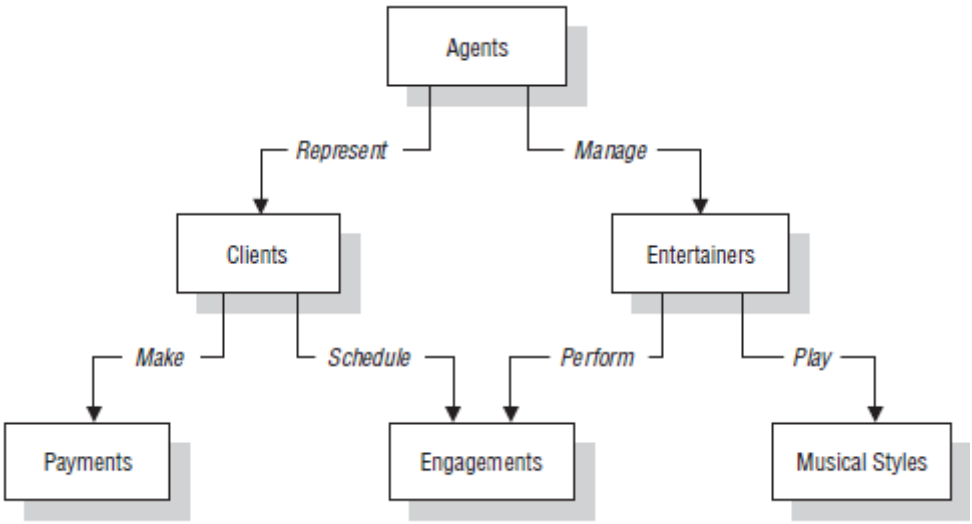
Uma vantagem de usar um banco de dados hierárquico é que um usuário pode obter dados muito rapidamente, porque existem ligações explícitas entre as estruturas de tabelas. Outra vantagem é que a integridade referencial é construída e automaticamente executadas. Isso garante que um registro em uma tabela filho deve ser ligado a um registro existente em uma tabela pai, e que um registro excluído na tabela pai fará com que todos os registros associados na tabela filho para ser excluído.

Em um banco de dados hierárquico ocorre problema quando um usuário precisa armazenar um registro em uma tabela filho que está alheio a qualquer registro em uma tabela pai. Por exemplo, um usuário não pode cadastrar um novo artista na tabela Artistas até que o artista seja atribuído a um agente na tabela AGENTES. No entanto, na vida real, artistas comumente se inscrevem com a agência muito antes deles serem atribuídos a agentes específicos. Este cenário é difícil de modelar em um banco de dados hierárquico.

Este tipo de banco de dados não pode suportar relacionamentos complexos, e há muitas vezes problema com dados redundantes.

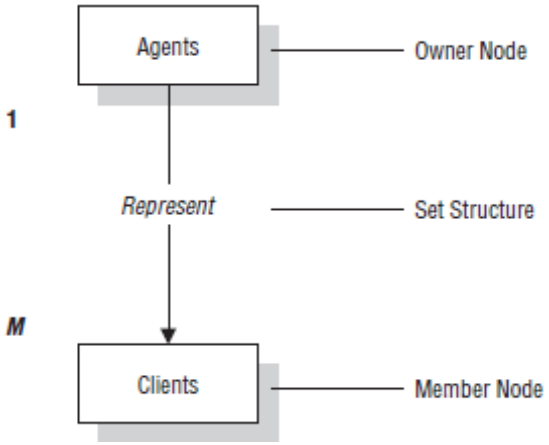
O modelo de banco de dados de rede

O banco de dados de rede foi, na sua maior parte, desenvolvido como uma tentativa para resolver alguns dos problemas da base de dados hierárquica. A estrutura de um banco de dados de rede está representado em termos de nós e estruturas definidas. A figura abaixo mostra um diagrama de um banco de dados de rede típica.



Um nó representa uma coleção de registros e uma estrutura definida estabelece e representa um relacionamento em um banco de dados da rede. É uma construção transparente que se relaciona com um par de nós em conjunto usando um nó como um proprietário e outro nó como um membro. A estrutura suporta um conjunto de relação um-para-muitos, o que significa que um registro no nó proprietário pode estar relacionado a um ou mais registros no nó do membro, mas uma único registro no nó do membro está relacionado com apenas uma gravação no nó proprietário. Além disso, um

registro no nó membro não pode existir sem estar relacionada a um registro existente no nó proprietário. Por exemplo, um cliente deve ser atribuído a um agente, mas um agente com nenhum cliente ainda pode ser listado na base de dados.



Uma vantagem que o banco de dados de rede fornece é um acesso rápido aos dados. Ele também permite aos usuários criar consultas que são mais complexas do que aquelas criadas usando um banco de dados hierárquico. A principal desvantagem do banco de dados da rede é que o usuário tem de estar muito familiarizado com a estrutura do banco de dados, a fim de trabalhar com o conjunto de estruturas. Outra desvantagem é que não é fácil mudar a estrutura do banco de dados sem afetar os programas aplicativos que interagem com ele. Você não pode mudar um conjunto de estrutura sem afetar os programas de aplicativos que usam essa estrutura para navegar pelos dados. Se você alterar uma conjunto de estrutura, você também deve modificar todas as referências feitas a partir do programa de aplicação para essa estrutura.

Embora o banco de dados de rede foi claramente um passo a partir do banco de dados hierárquico, algumas pessoas na comunidade de banco de dados acreditavam que devia haver uma maneira melhor de gerenciar e manter grandes quantidades de dados. Como cada modelo de dados surgiram, os usuários descobriram que podiam fazer perguntas mais complexas, aumentando assim as exigências feitas sobre a base de dados. E assim, chegamos ao modelo de banco de dados relacional.

O modelo de banco de dados relacional

Um banco de dados relacional armazena dados em relações, que o usuário entende como tabelas. Cada relação é composta de tuplas, ou registros e atributos, ou campos. A ordem física dos registros ou campos em uma tabela é completamente irrelevante, e cada registro na tabela é identificada por um campo que contém um valor único. Estas são as duas características de um banco de dados relacional que permitem aos dados existirem independentemente da forma como ele está fisicamente armazenado no computador. Como tal, um usuário não é obrigado a conhecer a localização física de um registro, a fim de recuperar seus dados. Isso é diferente dos modelos de banco de dados de rede e hierárquico, em que conhecer o layout das estruturas é crucial para a recuperação de dados.

A relação entre um par de tabelas é estabelecido implicitamente através de valores correspondentes de um campo compartilhado.

Agents

Agent ID	Agent First Name	Agent Last Name	Date of Hire	Agent Home Phone
100	Mike	Hernandez	05/16/95	553-3992
101	Greg	Piercy	10/15/95	790-3992
102	Katherine	Ehrlich	03/01/96	551-4993

Clients

Client ID	Agent ID	Client First Name	Client Last Name	Client Home Phone	.....
9001	100	Stewart	Jameson	553-3992	.....
9002	101	Shannon	McLain	790-3992	.....
9003	102	Estela	Pundt	551-4993	.....

Entertainers

Entertainer ID	Agent ID	Entertainer First Name	Entertainer Last Name	.....
3000	100	John	Slade	.....
3001	101	Mark	Jebavy	.....
3002	102	Teresa	Weiss	.....

Engagements

Client ID	Entertainer ID	Engagement Date	Start Time	Stop Time
9003	3001	04/01/96	1:00 PM	3:30 PM
9009	3000	04/13/96	9:00 PM	1:30 AM
9001	3002	05/02/96	3:00 PM	6:00 PM

A base de dados relacional fornece um número de vantagens sobre os modelos anteriores, tais como o seguinte:

- *Construído em multinível de integridade:* integridade dos dados é construído no modelo em nível de campo para garantir a precisão dos dados; no nível de tabela para garantir que os registros não sejam duplicados e para detectar falta de valores de chave primária; no nível de relacionamento para garantir que a relação entre um par de tabelas é válido; e no nível de negócios para assegurar que os dados sejam precisos em termos de negócio em si.
- *Independência lógica e física de dados de aplicativos de banco de dados:* Nem muda um usuário fazer o projeto lógico do banco de dados, nem tampouco muda um fornecedor de software de banco de dados fazer a implementação física do banco de dados, afetará negativamente as aplicações construídas em cima dele.
- *Garantida a consistência dos dados e precisão:* Os dados são consistentes e precisas devido aos vários níveis de integridade que você pode impor dentro do banco de dados.

- *A fácil recuperação de dados:* Ao comando do usuário, os dados podem ser recuperados quer a partir de uma tabela em particular ou a partir de qualquer número de tabelas relacionados dentro do banco de dados. Isso permite que um usuário visualize as informações em um número quase ilimitado de maneiras.

Dados e gerenciamento de banco de dados

O componente básico de um arquivo em um sistema de arquivo é um *item de dado*, que é chamada a menor unidade de um dado que tem significado no mundo real – por exemplo, último nome, primeiro nome, rua, número de identificação, e partido político.

Um grupo de itens de dados relacionados tratados como uma unidade por uma aplicação é chamada de *registro*. Exemplos de tipos de registros são pedidos, vendedores, clientes, produtos e departamentos.

Um *arquivo* é uma coleção de registros de um único tipo.

Sistemas de banco de dados são construídos e expandidos sobre essas definições. Em um banco de dados relacional, um item de dado é chamado *coluna* ou *atributo*, um registro é chamado *linha* ou *tupla*, e um arquivo é chamado *tabela*.

Uma base de dados é um objeto mais complexo; é uma coleção de dados armazenados interrelacionados que serve as necessidades de múltiplos usuários dentro de uma ou mais organizações – é uma coleção interrelacionada de muitos tipos diferentes de tabelas.

A motivação para a utilização de banco de dados ao invés de arquivos tem sido pela maior disponibilidade de um conjunto diversificado de usuários, integração de dados para facilitar o acesso e atualização para transações complexas e menos redundância de dados.

Um *sistema gerenciador de banco de dados* (SGBD) é um sistema de software generalizado para manipulação de banco de dados. Um SGBD suporta uma visão lógica (esquema, subesquema); visão física (métodos de acesso, agrupamento de dados); linguagem de definição de dados; linguagem de manipulação de dados; e importantes utilidades como gereciamento de transação e controle de concorrência, integridade de dados, recuperação de travamento e segurança.

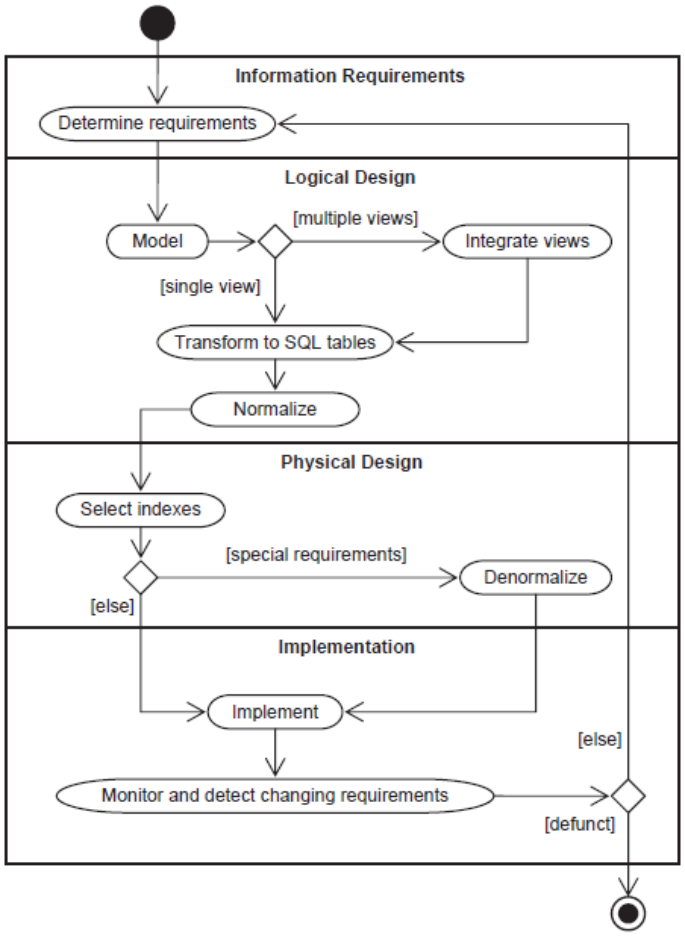
Sistemas de banco de dados relacionais, o tipo dominante de sistemas para negócios de banco de dados bem formatados, também provê um maior grau de independência de dados que os primeiros sistemas gerenciadores de banco de dados hierárquico e de rede.

Independência de dados é a habilidade de fazer mudanças na estrutura lógica ou física de um banco de dados sem precisar reprogramar os programas da aplicação. É também fazer a conversão e reorganização do banco de dados muito mais fácil. SGBD relacional provê um grau maior de independência de dados que os sistemas anteriores.

Ciclo de vida do banco de dados

O ciclo de vida de um banco de dados incorpora os passos básicos envolvidos na concepção de um esquema global do banco de dados lógico, alocando dados através de uma rede de computadores, e definindo um esquema local para um SGBD específico. Uma vez que o projeto esteja completo, o ciclo de vida continua com a implementação e manutenção do banco de dados.

O ciclo de vida de um banco de dados:



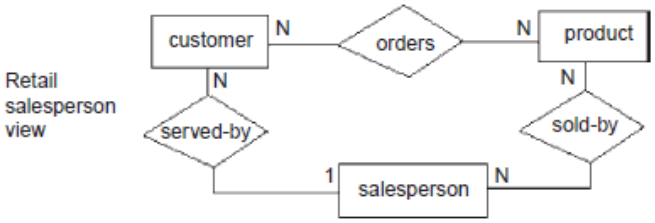
1. Análise de requisitos: os requisitos de um banco de dados são determinados por meio de entrevistas de produtores e usuários dos dados, usando as informações para produzir uma especificação formal de requisitos. Essa especificação inclui os dados necessários para o processamento, os relacionamentos de dados naturais, e da plataforma de software para a implementação de banco de dados. Como exemplo, a figura abaixo mostra o conceito de produtos, clientes, vendedores e outros sendo formulados a partir da opinião do usuário final durante o processo de entrevistas.
2. Projeto lógico: o esquema global, um diagrama de modelo conceitual de dados que mostra todos os dados e seus relacionamentos, é desenvolvido usando técnicas como entidade-relacionamento (ER) ou UML. A construção do modelo de dados deve ser, no final, transformado em tabelas.
   a. Modelagem conceitual de dados: os requisitos de dados são analisados e modelados usando um ER ou diagrama UML que inclui muitas características, por exemplo, a semântica para relacionamentos opcionais, relacionamentos ternários, supertipos e subtipos (categorias). Requisitos de processamento normalmente são especificados usando expressões de linguagem natural ou comandos SQL algo que ocorre com frequencia.
   b. Visão integrada: normalmente quando o projeto é grande e mais de uma pessoa está envolvida na análise de requisitos, múltiplas visões de dados e relacionamentos ocorrem, resultando em inconsistências devido à variação de taxonomia, a percepção de contexto. Para eliminar redundância e inconsistência do modelo, essas visões devem ser racionalizadas e consolidadas em uma única visão global. A visão integrada requer o uso de ferramentas semânticas ER, como identificação de sinônimos, agregação e generalização. Visão integrada

também é importante quando as aplicações tem que ser integradas, e cada pode ser escrita com sua própria visão de banco de dados.

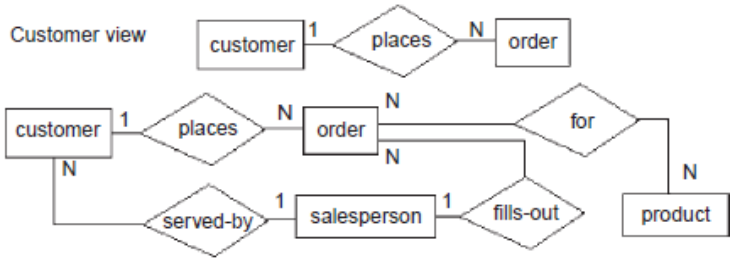
Step I Information Requirements (reality)

Step II Logical design

Step II.a Conceptual data modeling



Step II.b View integration



c. Transformação do modelo conceitual de dados para tabelas SQL: baseado na categorização da construção do modelo de dados e um conjunto de regras de mapeamento, cada relacionamento e suas entidades associadas são transformados em um conjunto de tabelas relacionais candidatas de um SGBD específico. Tabelas redundantes são eliminadas como parte desse processo.

Step II.c Transformation of the conceptual data model to SQL tables

Customer

cust-no	cust-name	.....

Product

prod-no	prod-name	qty-in-stock

Salesperson

sales-name	addr	dept	job-level	vacation-days

Order

order-no	sales-name	cust-no

Order-product

order-no	prod-no

```
create table customer
    (cust_no integer,
    cust_name char(15),
    cust_addr char(30),
    sales_name char(15),
    prod_no integer,
    primary key (cust_no),
    foreign key (sales_name)
        references salesperson,
    foreign key (prod_no)
        references product):
```

d. Normalização de tabelas: dada uma tabela (R), um conjunto de atributos (B) é funcionalmente dependente de outro conjunto de atributos (A) se, em cada instante de tempo, cada valor de A é associado com exatamente um valor de B.

Dependências funcionais são derivadas do diagrama conceitual do modelo de dados e da semântica dos relacionamentos de dados na análise de requisitos. Eles representam as dependências entre os elementos de dados que são identificadores únicos (chaves) de entidades. Dependências funcionais adicionais, que representam as dependências entre chave e atributos não chave com entidades, podem ser derivados a partir da especificação de requisitos. Tabelas relacionais candidatas associadas com todas as dependência funcionais são normalizados (modificadas por decomposição ou divisão de tabelas em tabelas menores) usando técnicas padrão de normalização.

Finalmente, redundâncias de dados que ocorrem nas tabelas candidatas normalizadas são analisadas para mais eliminações possíveis, com a restrição que a integridade de dados deve ser preservada.

Step II.c Transformation of the conceptual data model to SQL tables

Customer

cust-no	cust-name	----------

Product

prod-no	prod-name	qty-in-stock

```
create table customer
    (cust_no integer,
    cust_name char(15),
    cust_addr char(30),
    sales_name char(15),
    prod_no integer,
    primary key (cust_no),
    foreign key (sales_name)
        references salesperson,
    foreign key (prod_no)
        references product):
```

Salesperson

sales-name	addr	dept	job-level	vacation-days

Order

order-no	sales-name	cust-no

Order-product

order-no	prod-no

Step II.d Normalization of SQL tables

Decomposition of tables and removal of update anomalies.

Salesperson

sales-name	addr	dept	job-level

SalesVacations

job-level	vacation-days

3. Projeto físico: envolve a seleção de índices (métodos de acesso), particionamento e agrupamento de dados. A metodologia do projeto lógico no passo 2 simplifica a abordagem para a concepção de grandes bandos de dados relacionais por redução do número de dependências de dados que precisam ser analisados. Isso é realizado por inserção da modelagem conceitual de dados e medidas de integração na abordagem tradicional do projeto relacional. O objetivo dessas medidas é uma representação exata da realidade. A integridade de dados é preservada através da normalização das tabelas candidatas criadas quando o modelo conceitual de dados é transformado em um modelo relacional. A proposta do projeto físico é para otimizar o desempenho. Como parte do projeto físico, o esquema global pode às vezes ser refinado de forma limitada para refletir o processamento (consultas e transações) de requisitos se houverem obviamente grandes ganhos para ser feito com eficiência. Isto é chamado desnormalização. Consiste na seleção de processos dominantes na base de alta frequência, alto volume ou prioridade explícita; definir extensões simples para tabelas que irão melhorar o desempenho da consulta; avaliar o custo total por consulta, atualização e armazenamento; e considerando os efeitos colaterais, como

possível perda de integridade. Isso é particularmente importante para aplicações de processos analíticos online (OLAP – online analytical processing).

4. Implementação, monitoramento e modificação no banco de dados: uma vez que o projeto esteja concluído, o banco de dados pode ser criado através da implementação do esquema formal, usando a linguagem de definição de dados (DDL – data definition language) de um SGBD. Então, a linguagem de manipulação de dados (DML – data manipulation language) pode ser usada para consultas e atualizações no banco de dados, assim como a criação de indicies e estabelecer restrições, como integridade referencial. A linguagem SQL contém ambas construções DDL e DML; por exemplo o *create table* representa DDL, e o comando *select* representa DML.

   À medida que a base de dados começa a monitorizar as operções, indica se os requisitos de desempenho estão sendo satisfeitas. Se elas não estiverem sendo satisfeitas, modificações deverão ser feitas para melhorar o desempenho. Outras modificações podem ser necessárias quando os requisitos mudam ou as expectativas do usuário final aumentam com o bom desempenho. Desta maneira, o ciclo de vida continua com monitoramento, reprojeto, e modificações.

Modelagem conceitual de dados

A modelagem conceitual de dados é o componente inicial de um projeto lógico de banco de dados.

Diagramas de esquemas foram formalizados em 1960 por Charles Bachman. Ele usou retângulos para indicar tipos de registros e setas dirigidas a partir de um registro para outro para indicar um relacionamento um-para-muitos entre os casos de registros dos dois tipos.

A entidade-relacionamento (ER) aborda a modelagem conceitual de dados. A forma de Peter Chen (1976) representar modelos ER utiliza retângulos para especificar entidades, que são de forma análoga aos registros. Também usa losango para representar os vários tipos de relacionamentos, que são diferenciados por números ou letras colocados nas linhas de conexão dos losangos para os retângulos.

A linguagem de modelagem unificada (UML – United Modeling Language) foi introduzida em 1997 por Grady Booch e James Rumbaugh a tornou-se uma linguagem gráfica padrão para especificar e documentar sistemas de software em larga escala. O componente de modelagem de dados de UML (agora UML-2) tem muita similaridade com o modelo ER.

Na modelagem conceitual de dados, a ênfase principal é a simplicidade e legibilidade. A meta do projeto do esquema conceitual, onde as abordagens ER e UML são muito úteis, é capturar os requisitos de dados do mundo real de uma maneira simples e significativa que seja compreensível tanto para banco de dados como para usuário final. O usuário final é a pessoa responsável por acessar o banco de dados e executar consultas e atualizações através do uso de um software SGBD, e, portanto, tem interesse no processo de projeto do banco de dados.

Exercícios

1. Nomeie os dois principais tipos de bancos de dados em uso hoje.
2. Que tipo de dados armazena um banco de dados analítico?
3. A afirmação abaixo é _____.

Um banco de dados operacional é usado principalmente em cenários de processamento de transações on-line (OLTP).

4. Quais os dois modelos de dados que antecederam o modelo de banco de dados relacional?
5. Descreva uma relação pai/filho.
6. O que é um conjunto de estrutura?
7. Como o modelo de banco de dados relacional armazenada dados?
8. Como você recupera dados em um banco de dados relacional?
9. Cite duas vantagens de um banco de dados relacional.
10. O que é um sistema de gerenciamento de banco de dados relacional?
11. Quais as quatro etapas do ciclo de vida de um banco de dados?
12. Cite as fases do projeto lógico.
13. Qual a diferença entre as linguagens DDL e DML?
14. Qual o objetivo da modelagem conceitual de dados?
15. Quais os produtos apresentados na fase de modelagem conceitual de dados?
16. Assinale V para verdadeiro e F para falso.
    - ( ) Quanto maior o número de usuários que utilizam um sistema, mais aberto ele deve ser.
    - ( ) A existência de um sistema sempre pressupõe o uso de um computador, independente do tamanho do mesmo.
    - ( ) Todo sistema tem que fazer uso de pelo menos um computador.
    - ( ) É possível construir um sistema de informação integrado sem a utilização de computador.
    - ( ) Os dados nascem no nível estratégico da organização e caminham no sentido do nível operacional.
    - ( ) Os dados de um sistema de informações caminham do nível operacional para o nível estratégico.
    - ( ) As respostas de um sistema de planejamento estratégico são tão precisas quanto as de um sistema operativo.
    - ( ) Os sistemas feitos para o nível estratégico da organização, são, via de regra, mais inteligentes que os sistemas do nível operacional.
    - ( ) As atualizações nos dados no nível tático são rigorosas, já que os mesmos são formalmente estruturados.
    - ( ) Os sistemas de apoio a decisão não tem inteligência suficiente para decidir.
    - ( ) Os objetivos que o sistema deve cumprir são cuidadosamente levantados na fase de projeto.
    - ( ) O dado é a informação no nível atômico.
    - ( ) Todos os dados que entram em um processo devem ter algum uso dentro do mesmo.
    - ( ) Um processo produz dados derivados, mas nunca receberá um dado desse tipo.
    - ( ) Os dados que o sistema recebe do ambiente chegam através dos fluxos de dados.
    - ( ) O dado é independente do processo.
    - ( ) O processo é o elemento mais estável de um sistema.
    - ( ) Pode-se fazer o estudo dos dados de um sistema, independente do estudo dos processos.
    - ( ) Não se consegue estruturar os dados de um sistema sem conhecer profundamente os processos que os manipulam.
    - ( ) Uma consulta a um dado, efetuada por um processo, é suficiente para indicar

que o dado é útil ao sistema.

( ) Desde que bem descrito, um dado pode aparecer até três vezes no modelo conceitual de dados.
( ) Os três tipos de banco de dados comerciais são: hierárquico, rede e relacional.
( ) Durante a construção do modelo conceitual de dados não há participação de usuários.
( ) O modelo conceitual de dados pode apresentar redundância de dados quando o banco de dados a ser desenvolvido for do tipo hierárquico.
( ) Um modelo lógico de dados desenvolvido para um banco de dados do tipo hierárquico não será adequado para um banco de dados do tipo relacional.
( ) O modelo lógico de dados não se preocupada com o tipo de banco de dados que será instalado no sistema.
( ) O modelo conceitual de dados é desenvolvido sem se preocupar com a tecnologia a ser utilizada na instalação do sistema.
( ) O detalhamento dos processos durante a fase de análise podem trazer ajustes no modelo de dados.
( ) Tupla é a mesma coisa que atributo.
( ) O elemento de dado faz parte do fluxo de dados, enquanto o atributo forma a tabela.

17. Relacione.

(A) Dado	( ) coluna ou atributo
(B) Registro	( ) linha ou tupla
(C) Arquivo	( ) tabela
(D) SGBD	( ) programa para manipulação do banco de dados

Bibliografia

Database Modeling and Design: Logical Design
Toby Teorey, Sam Lightstone, Tom Nadeau, H. V. Jagadish
USA: Elsevier, 2011

Database Design for Mere Mortals – 2ª edição
Michal J. Hernandez
USA : Addison-Wesley, 2003