BANCOS DE DADOS

Profº Me. Jones Artur Gonçalves

PROJETO DE BANCO DE DADOS

Imagine que você foi contratado para criar um banco de dados de uma empresa recém-criada. Qual é o primeiro passo para o projeto de um banco de dados?

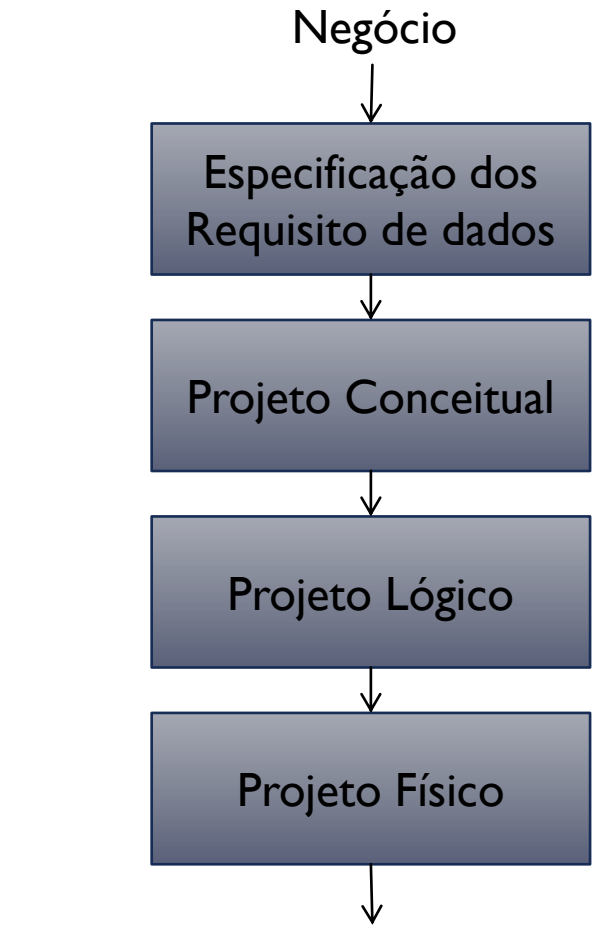
Projeto BD

- **Principais Etapas**

1. Especificação e Análise de Requisitos
   a. Os requisitos são documentados
2. Projeto Conceitual
   a. Baseado nos requisitos
3. Projeto Lógico
   a. Expresso em um modelo de dados, como o relacional
4. Projeto Físico
   a. Especificações para armazenar e acessar o banco de dados
   b. Implementação do BD, inserção de dados reais e manutenção.

Projeto BD

➢ Principais Etapas



Definição dos dados a serem modelados

Modelo de Entidade e Relacionamento

Scripts para criação da estrutura dos dados

LINGUAGEM SQL

Structured Query Language - Linguagem de Consulta Estruturada - Criada no início da década de 70 pela IBM e padronizada pela American National Standards Institute em 1986.

Cada SGBD possui uma interface específica para que comandos SQL possam ser executados nas suas bases de dados.

SQL

➢ Definição

- ✓ Linguagem declarativa;
- ✓ Script utilizados em bancos de dados relacionais;
  - ▪ colunas – conjunto ordenado de atributos da entidade que está sendo armazenada;
  - ▪ linhas – são registros individuais (ou tuplas) da entidade representada pela tabela em questão. Cada linha de uma tabela possui as mesmas colunas.

SQL

- **Subconjuntos**

DDL - Linguagem de Definição de Dados

Data Definition Language (DDL): permite criar e gerenciar a estrutura do banco de dados, como por exemplo, criar ou eliminar tabelas. É possível também a especificação de alguns itens de segurança adicionais, como por exemplo, a definição de quais informações é obrigatória dentro de uma dada tabela, que procedimentos devem ser associados a uma operação em particular sobre o banco de dados, etc.

SQL

➢ **Subconjuntos**

DML - Linguagem de Manipulação de Dado

Data Manipulation Language (DML): permite consultar, adicionar, alterar e excluir informações do banco de dados. Dentro da sintaxe de cada comando são permitidas algumas cláusulas adicionais que viabilizam operações relacionais tais como seleção, junção, união, etc e a utilização de algumas funções de uso mais comum, como de cálculo de média, soma, contagem de registros e determinação de máximo e mínimo entre valores.

SQL

- **Subconjuntos**

DCL - Linguagem de Controle de Dados

Data Control Language (DCL): permite definir as contas dos usuários autorizados a acessar o banco de dados, bem como impor restrições de uso a esses usuários.

SQL

- **Subconjuntos**

Mais recentemente, a difusão da SQL ganhou um significativo impulso com o advento da tecnologia Cliente / Servidor, tendo consolidado-se como o elemento chave na conexão entre SGBDs de fornecedores diferentes. É importante ressaltar que existem diferenças entre as implementações da sintaxe dos comandos conforme o fabricante, mas essas diferenças geralmente são secundárias e existe sempre uma sintaxe mínima que é suportada por todos os produtos.

SQL

- **Principais operadores**

Operador	*Significado*
+	Aritmético: Adição.
-	Aritmético: Subtração.
*	Aritmético: Multiplicação.
/	Aritmético: Divisão.
^	Aritmético: Exponenciação.
=	Comparação: Igualdade.
<	Comparação: Menor que ...
<=	Comparação: Menor ou igual a ...
<>	Comparação: Diferente de ...
>	Comparação: Maior que ...
>=	Comparação: Maior ou igual a ...
IN	Comparação: Verifica se um valor está contido em uma lista.
BETWEEN	Comparação: Verifica se um valor está contido numa faixa contínua de valores.
LIKE	Comparação: Verifica se o formato de um valor atende a um determinado padrão.
%	Curinga: Qualquer combinação de caracteres (usado com LIKE).
_	Curinga: Qualquer caracter na posição (usado com LIKE).
AND	Lógico: E (Conjunção).
OR	Lógico: OU não exclusivo.
NOT	Lógico: Negação.

SQL

➢ Tipo de Dados

Tipo de dado	Descrição
INTEGER OU INT	Número positivo ou negativo inteiro.
SMALLINT	Mesma função do INT, mas ocupa a metade do espaço.
NUMERIC	Número positivo ou negativo. Deve-se informar o tamanho do campo e casas decimais.
DECIMAL	Semelhante a NUMERIC, em alguns casos tem maior precisão em casas decimais.
REAL	Número de ponto flutuante de simples precisão(Exponencial).
DOUBLE PRECISION	Número de ponto flutuante de dupla precisão.
FLOAT	Número de ponto flutuante em que você define o nível de precisão (número de dígitos significativos).
BIT	Armazenamento de um número fixo de bits.

SQL

➢ Tipo de Dados (Cont.)

Tipo de dado	Descrição
BIT VARYING	Igual a BIT, permitindo armazenar valores maiores. Normalmente utilizado para armazenar imagens.
DATE	Permite armazenar datas.
TIME	Permite armazenar horários.
TIMESTAMP	Permite armazenar uma combinação de data e hora.
CHAR	Permite armazenar cadeia de caracteres. Tamanho informado será fixo.
VARCHAR	Permite armazenar cadeia de caracteres, mas de tamanho variável.

SQL

➢ Tipo de Dados (Cont.)

Tipo de Dado	**Tipo de Dado Suprido pelo Sistema**
Binário	Binary[(n)], varbinary[(n)]
Caracter	Char[(n)], varchar[(n)]
Data e hora	Datetime, smalldatetime
Numérico exato	Decimal[(p[,s)], numeric[(p[,s])]
Numérico aproximado	Float[(n)], real
Inteiro	Int, smallint, tinyint
Monetário	Money, smallmoney
Especial	Bit, timestamp, tipos definidos pelo usuário
Texto e imagem	Text, image

SQL

➢ Tipo de Dados (Cont.)

Tipo de dado	Descrição
Binary	Contém até 8.000 bytes de dados binários de tamanho fixa. Tamanho fixo significa que o campo irá sempre utilizar o espaço que você definiu.
Bit	Cria um campo true/false (verdadeiro/falso). Este tipo de dado armazena apenas os valores de 0 e 1 e não pode conter valores nulos.
Char	Contém até 8.000 bytes de uma string de tamanho fixo. Use este campo para dados this que irão sempre ser do mesmo tamanho.
Datetime	Armazena a data e hora com precisão de milissegundos em 8 bytes.
Decimal	Contém dados numéricos com um inteiro à esquerda do ponto decimal e uma parte fracionária a direita. Este campo irá ter um número variável de bytes, baseado na escala e precisão que você especificar. Escala é o número máximo de dígitos à direita do ponto decimal e precisão é o número total de dígitos em ambos os lados do the ponto decimal.
Float	Um tipo de dados numérico aproximado para armazenamento de longos números em ponto-flutuante, com a precisão de até 15 dígitos em 8 bytes. O SQL Server feralmente arredonda esses números para cima.
Image	Contém até 2.147.483.647 bytes (aproximadamente 2Gbytes) de dados binários e costuma ser utilizado para armazenar grandes quantidades de dados como imagens ou arquivos de som. Campos definidos com este tipo de dado não podem ser indexadas, pesquisadas, agrupadas ou ordenadas.
Int	Armazena um dado numérico preciso para todo número armazenado em 4 bytes. Pode conter números de -2^1031 a 2^1031.
Money	Contém dados monetários definidos com precisão de 4 dígitos. Ele é representado como um inteiro de precisão dupla e consome 8 bytes de armazenamento. Valores podem estar entre -992.337.293.685.477,5808 e 922.337.203.685.447,5807.
Nchar	Contém até 4.000 caracteres Unicode. Tipo de dado de largura fixa, sendo armazenado no dobro de bytes (caracteres Unicode precisam de 2 bytes por caracter).
Ntext	Armazena caracteres Unicode até 1.073.741.823 posições, armazenados no dobro dos bytes declarados (caracteres Unicode precisam de 2 bytes por caracter).
Numeric	Veja Decimal.
Nvarchar	Contém até 4.000 caracteres Unicode. Tipo de dado de largura variável, sendo armazenado no dobro de bytes (caracteres Unicode precisam de 2 bytes por caracter).

SQL

➢ Tipo de Dados (Cont.)

Real	Similar ao tipo de dado float, usando apenas 4 bytes de armazenamento com precisão de até 7 dígitos.
Smalldatetime	Campo data menos preciso, que armazena data e hora em precisão de minutos, utilizando 4 bytes.
Smallint	Armazena dados numéricos preciso até a quantidade de números armazenados em 2 bytes. Menor que o int, ele pode conter números de -32.768 até 32.767.
Smallmoney	Contém dados monetários decimais precisos até o 4 dígito. Usa 4 bytes de armazenamento e pode conter valores de -214.748,3648 até 214.748,3647.
Text	Contém uma string de caracteres não-Unicode com tamanho de até 2.147.483.647 caracteres. Similiar ao campo memo encontrado em outros bancos de dados, este tipo de dado armazena grandes quantidades de texto em páginas de 2kb. Colunas definidas com este tipo de dado não podem ser indexadas, pesquisadas, agrupadas ou ordenadas.
Timestamp	Um valor binário automaticamente atualizado cada vez que uma tabela com uma coluna deste tipo é alterada. Ele usa 8 bytes de armazenamento e é único dentro do banco de dados. Uma tabela só pode possui uma coluna deste tipo.
Tinyint	O menor tipo de dado inteiro, ele consome apenas 1 byte de armazenamento e pode conter números de 0 a 255.
Uniqueidentifier	Um número hexadecimal de 16 bytes, também conhecido como GUID (globally unique identification number). Use a função do SQL Server NEWID() para gerar novos identificadores únicos para alimentar uma variável ou uma coluna deste tipo de dado.
Varbinary	Similar ao tipo binário, este pode conter até 8.000 bytes de dados binários de tamanho variável. Tamanho variável significa que o campo irá consumir apenas o espaço necessário para armazenar os dados contidos no mesmo.
Varchar	Contém até 8.000 bytes de tamanho variável para strings. Use este tipo para colunas que irão armazenar valores nulos de dados que variam em tamanho.

SQL

➢ Restrições importantes

Restrição	**Descrição**
Primary Key	Define chave primária
Foreign key	Define chave estrangeira
Unique	Define chaves candidatas
Null, not null	Define se o campo permite ou não valores nulos
Identity (SqlServer)	Colunas preenchidas automaticamente

SQL

➢ Regras (SQL Server)

✓ Nomes de colunas devem ser únicos dentro de uma determinada tabela, mas um mesmo nome pode ser usado em diferentes tabelas dentro de um mesmo database.

✓ Toda coluna deve possuir um tipo de dado

✓ Se a coluna não tem a especificação de NULL ou NOT NULL o *default* é NULL

SQL

➢ Regras (SQL Server)

- Convenções de Nomeação
  - Pode ter um máximo de 128 caracteres
  - Iniciar com uma letra
  - Os caracteres subsequentes podem incluir:
    - Letras, como definido no Unicode Standard 3.2
    - Números decimais do latim básico ou outros scripts nacionais
    - Arroba (@), cifrão ($), sinal de número (#) ou sublinhado
  - Não duplicar o nome de qualquer outro objeto de propriedade do mesmo BD
  - O nome não deve ser uma palavra reservada do Transact-SQL
  - Não são permitidos espaços ou caracteres especiais

SQL

➢ Regras (SQL Server)

- Declarações SQL não são case-sensitive.
- Declarações SQL podem ser escritas em uma ou mais linhas.
- Palavras-chave não podem ser abreviadas ou cortadas em linhas seguintes.
- Cláusulas são normalmente colocadas em linhas separadas.
- Tabs e indentações são usados para a facilitar a leitura.

SQL

➢ Banco de Dados - Exemplo

- Cria um novo banco de dados e os arquivos usados para armazená-lo
- Sintaxe

  CREATE DATABASE database_name;

- Para selecionar o database:
  - USE database_name;

SQL

➢ Banco de Dados

- ✓ CREATE DATABASE *databasename*;
- ✓ DROP DATABASE *databasename*;
- ✓ USE *databasename;*

SQL

➢ Banco de Dados - Exemplo

```
CREATE DATABASE  programacao_bd
GO
USE programacao_bd
GO
DROP DATABASE programacao_bd
GO
```

SQL

➢ Criação de Tabelas

O comando CREATE TABLE é usado para criar uma tabela. A sua forma geral é:

CREATE TABLE <nome_tabela>
(
<nome_coluna1> <tipo_dado> <restrições_c>,
<restrições_t>
)

CREATE TABLE <nome-da-tabela>
<nome-da-coluna>, <tipo-do-dado> [not null]
PRIMARY KEY (nome-coluna-chave)
FOREIGN KEY (nome-coluna-chave=estrangeira)
REFERENCES (nome tabela pai)

➢ Criação de Tabelas

Argumento	*Significado*
<nome_tabela>	Especifica o nome da tabela a ser criada.
<nome_coluna >	Especifica o nome da coluna a ser criada.
<tipo_dado>	Tipo de dado da coluna, que deverá ser escolhido dentro do conjunto de tipos de dados válidos. Se necessário, especificar também o tamanho do dado.
<restrições_c>	Restrições de coluna a serem aplicadas para o dado.
<restrições_T>	Restrições de tabela a serem aplicadas.

Onde:

a) Nome-da-tabela: Representa o nome da tabela que será criada.

b) Nome-da-coluna: Representa o nome da coluna que será criada. A definição das colunas de uma tabela é feita relacionando-as uma após a outra.

c) Tipo-do-dado: Cláusula que define o tipo de tamanho dos campos definidos para a tabela. Os tipos de dados mais comuns serão definidos à frente.

d) NOT NULL: Exige o preenchimento do campo, ou seja, no momento da inclusão é obrigatório que possua um conteúdo.

e) PRIMARY KEY (nome-da-coluna-chave): define para o banco de dados a coluna que será a chave-primária. Caso tenha mais de uma coluna como chave, elas deverão ser relacionadas entre parênteses

f) FOREIGN KEY (nome-da-coluna-chave) REFERENCES (nome-tabela-pai): define para o banco de dados as colunas que são chave-estrangeiras, ou seja, campos que são chave-primária de outras tabelas. Na opção REFERENCES deve ser especificado a tabela na qual a coluna é chave-primária.

SQL

➢ **Criação de Tabelas**

Constraints (Restrições de Integridade e de domínio):

Integridade de Chave:

PRIMARY KEY(atributos_chave)

Integridade Referencial:

FOREIGN KEY (atributos) REFERENCES tabela_base(atributos)

Restrição de Integridade:

CHECK(condição)

SQL

➢ Criação de Tabelas - Exemplo

```
Create table Cliente
(
Cod_Cli          int not null,
Nome_Cli              varchar(40) not null,
End_Cli          varchar(30) not null,
Bai_Cli       varchar(20) not null,
Cid_Cli           varchar(20) not null,
Uf_Cli       char(3) not null,
Tel_Cli           varchar(15) null,
Constraint PK_Cliente  Primary Key(Cod_Cli)
)
```

SQL

➢ Criação de Tabelas - Exemplo

```
Create Table NotaFiscal
(
Num_Nota      int not null,
Cod_Cli       int not null,
Serie_Nota    varchar(10) not null,
Emissao_Nota      smalldatetime null,
Vtot_Nota         SmallMoney not null,

Constraint PK_NotaFiscal Primary Key(Num_Nota),
Constraint FK_Cliente Foreign Key(Cod_Cli) References cliente(Cod_Cli)
)
```

SQL

➢ Criação de Tabelas - Exemplo

```
Create table Cidade
(Codcidade   varchar(2) not null,
Nomecidade  varchar(40))

Create table Empregado
(Nrmatricula int,
Nome         varchar(50),
Data_demissao datetime,
Salario      float)

Create table estado
(cdestado   varchar(2) not null,
Nomeestado  varchar(30))
```

➢ Alteração de Tabela

O comando ALTER TABLE realiza alteração em tabelas existentes:

ALTER TABLE <nome-da-tabela>

DROP COLUMN <nome-coluna>

ADD <nome-coluna>, <tipo-do-dado> [not null]

ALTER COLUMN <nome-coluna> <tipo-do-dado> [NULL]

ADD CONSTRAINT <nome-constraint> PRIMARY KEY (nome-coluna)

ADD FOREIGN KEY (nome-coluna-chave-estrangeira)
REFERENCES (nome tabela pai)

DROP CONSTRAINT <nome-constraint>

SQL

➢ Alteração de Tabela

Onde:

a) Nome-da-tabela: Representa o nome da tabela que será criada.
b) Nome-da-coluna: Representa o nome da coluna que será criada. A definição das colunas de uma tabela é feita relacionando-as uma após a outra.
c) Tipo-do-dado: Cláusula que define o tipo de tamanho dos campos definidos para a tabela. Os tipos de dados mais comuns serão definidos à frente.
d) DROP <nome-coluna>: realiza a retirada da coluna especificada na estrutura da tabela.
e) ADD <nome-coluna. <tipo-do-dado>: Realiza a inclusão da coluna especificada na estrutura da tabela.
f) ALTER COLUMN: modifica o tipo e tamanho do dado de uma coluna já criada na tabela.
g) Nome-constraint: nome da constraint aplicada à tabela.

➢ Alteração de Tabela

```
Alter table cidade
Add primary key (codcidade);

Alter table cidade
Add cdestado char(2) not null,
    teste    varchar(1) null;

Alter table cidade
Drop column teste;

Alter table cidade
Alter column cdestado varchar(2);
```

➢ Alteração de Tabela

```
Alter table cidade
Drop constraint pk__cidade__014935cb;

Alter table cidade
Add constraint pk_codcidade primary key (codcidade);

Alter table estado
Add primary key (cdestado);


Alter table cidade
Add foreign key (cdestado) references estado (cdestado);
```

➢ Alteração de Tabela

O comando **Drop Table** deleta a estrutura e os dados existentes em uma tabela. Após a execução deste comando estarão deletados todos os dados, estrutura, índices de acessos que estejam associados a ela.

Sintaxe:

DROP TABLE <nome-tabela>

Onde:

Nome-tabela: Nome da tabela que será deletada.

Exemplo:

DROP TABLE TESTE

BIBLIOGRAFIA

BÁSICA:

DATE, C. J. PROJETO DE BANCO DE DADOS E TEORIA RELACIONAL: FORMAS NORMAIS E TUDO O MAIS. SÃO PAULO: NOVATEC, 2015.

ELMASRI, R.; NAVATHE, S. B. SISTEMAS DE BANCO DE DADOS: FUNDAMENTOS E APLICAÇÕES. 7 ED. SÃO PAULO: PEARSON, 2019.

HEUSER, C. A. PROJETO DE BANCO DE DADOS. 6 ED. PORTO ALEGRE: BOOKMAN, 2010.

COMPLEMENTAR:

HARRINGTON, J. L. Projeto de Bancos de Dados Relacionais: Teoria e Prática. São Paulo: Campus, 2002.

MACHADO, F. N. R., Banco de dados: projeto e implementação. 2 ed. São Paulo: Érica, 2008.

NADEAU, Tom et al. Projeto e Modelagem de Banco de Dados. 5 ed. Rio de Janeiro: Elsevier Brasil, 2013.

SILBERSCHATZ, Abraham; SUNDARSHAN, S.; KORTH, Henry F. Sistema de banco de dados. Rio de Janeiro: Elsevier Brasil, 2016.

Referências

ALVES, W. P. FUNDAMENTOS DE BANCOS DE DADOS. ÉRICA, 2004

HEUSER, CARLOS ALBERTO. PROJETO DE BANCO DE DADOS. SAGRA LUZZATTO, 2004.

TEOREY, TOBY J. PROJETO E MODELAGEM DE BANCO DE DADOS. ELSEVIER, 2007.

O.K. TAKAI; I.C.ITALIANO; J.E. FERREIRA, INTRODUÇÃO A BANCO DE DADOS

OSVALDO KOTARO, APOSTILA, DCC-IME-USP – FEVEREIRO - 2005

MATTOSO, MARTA, INTRODUÇÃO À BANCO DE DADOS – AULA

GILLENSON, MARK L. FUNDAMENTOS DE SISTEMAS DE GERÊNCIA DE BANCO DE DADOS. LTC, 2006.

BANCO DE DADOS BÁSICO, UNICAMP, CENTRO DE COMPUTAÇÃO, SLIDES.

BOGORNY VANIA, MODELO ENTIDADE-RELACIONAMENTO, SLIDES.

WWW.JOINVILLE.UDESC.BR/PORTAL/PROFESSORES/MAIA/.../6___MODELO_ER.PPT DATA DE ACESSO: 01/07/2015

ABREU, FELIPE MACHADO; ABREU, MAURÍCIO – PROJETO DE BANCO DE DADOS – UMA VISÃO PRÁTICA - ED. ÉRICA – SÃO PAULO

HEUSER, CARLOS ALBERTO. PROJETO DE BANCO DE DADOS – UMA VISÃO PRÁTICA. PORTO ALEGRE: SAGRA LUZATTO, 2004.

KORTH, H. F.; SUDARSHAN, S; SILBERSCHATZ, A. SISTEMA DE BANCO DE DADOS. 5A ED. EDITORA CAMPUS, 2006. - CAPÍTULO 6

HTTP://WWW.PROFTONINHO.COM/DOCS/MODELAGEM_AULA_6_ENTID_ASSOC.PDF DATA DE ACESSO: 01/07/2015

HTTPS://MATERIALPUBLIC.IMD.UFRN.BR/CURSO/DISCIPLINA/4/56/1/6 DATA DE ACESSO: 01/02/2023

ELMASRI, R.; NAVATHE S. B. SISTEMAS DE BANCO DE DADOS. 4 ED. EDITORA ADDISON-WESLEY. 2005. - CAPÍTULO 3

DAVENPORT, THOMAS H.; PRUSAK, LAURENCE. CONHECIMENTO EMPRESARIAL: COMO AS ORGANIZAÇÕES GERENCIAM O SEU CAPITAL INTELECTUAL. RIO DE JANEIRO: CAMPUS, 1998.

HTTP://WWW.IME.UNICAMP.BR/~HILDETE/DADOS.PDF ACESSO EM: 12 MAIO 2016.

OBRIGADO