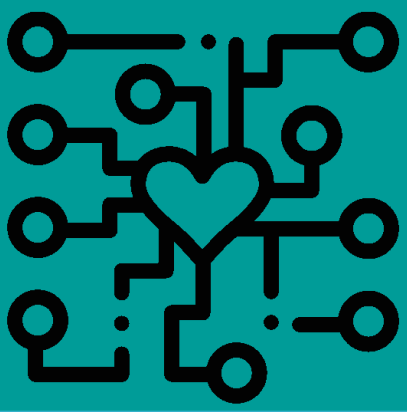
Banco de Dados - Relacionais e Não Relacionais

V3/2021

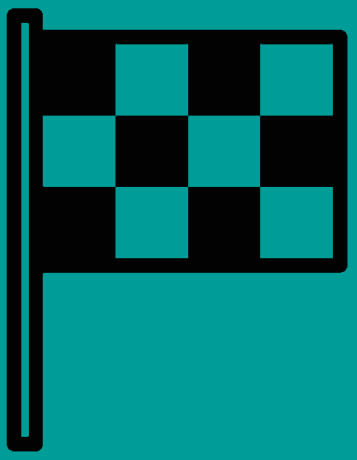
APOSTILA DE REDES, UNIVERSIDADE DO VALE DO PARAÍBA

HTTPS://WWW.UNIVAP.BR/COLEGIOS

Versão 3, Janeiro 2021



Sumário



1. Introdução

1.1 Quem está me ensinando?

Ao se iniciar um curso, seja ele remoto ou presencial é sempre importante conhecer a pessoa que irá te ensinar, para analisar a qualidade do curso que será ministrado. Mesmo se tratando de uma apostila virtual é no mínimo, interessante, conhecer a pessoa que a escreveu, por isso um breve resumo do autor deste livro virtual.

Meu nome é Bruno Michel Pera, sou formado em Engenharia da Computação pela Universidade do Vale do Paraíba e curso mestrado em Inovação Tecnológica na Universidade Federal de São Paulo. Abaixo está disposto algumas de minhas redes socias das quais vocês poderão manter contato caso seja necessário.

- https://www.univap.br/universidade.html
- http://lattes.cnpq.br/4209017189513990
- https://www.linkedin.com/in/bruno-michel-565b3a184/
- bruno.pera@univap.br

Caso haja algum problema com o conteúdo do curso ou queira deixar alguma dica ou sugestão, ficará bem mais facil entrar em contato.

Também vou deixar disponível algumas publicações já realizadas, caso tenham interesse em saber como trabalho.

- PERA, B. M.; LEMES, D. C. M. ; DOMINGOS, J. M. ; MARTINS, R. S. . VIDAINTELIGENTE: MONITORAMENTO REMOTO, PRONTUÁRIO ELETRÔNICO E E-HEALTH. 2020. (Apresentação de Trabalho/Congresso).
- João Victor Pereira Santos. Aplicativo que utiliza tecnologia híbrida para o aprendizado da língua inglesa. 2018. Iniciação Científica. (Graduando em Técnico em Informática) - Universidade do Vale do Paraíba. Orientador: Bruno Michel Pera.
- Carolina de Oliveira Rodrigues. Sitema de Automação Residencial/Empresarial Internet of Things. 2019. Iniciação Científica. (Graduando em Técnico Eletrônica) - Universidade do Vale do Paraíba. Orientador: Bruno Michel Pera.
- Gabriel Cunha Olopes. Desenvolvimento de Drones de Comunicação Híbra para Reconhe-

cimento com Visão Computacional. 2020. Iniciação Científica. (Graduando em Técnico Eletrônica) - Universidade do Vale do Paraíba. Orientador: Bruno Michel Pera.

1.2 O que estou recebendo?

Esta apostila não substitui as aulas presenciais, funciona apenas como um guia do conteúdo que irá ser visto durante o ano. Reforço que **não há necessidade de imprimir** a versão virtual ficará disposta 24h por dia, sete dias por semana durante o ano de 2021. Todo conteúdo até o final do ano está documentado aqui, lembrando que funcionará como um norte a ser seguido.

1.3 Ementa do curso de Banco de Dados

Abaixo segue a ementa do curso para o ano de 2021, será destacado todo o conteúdo que irá ser passado, todas as referências ficarão dispostas no capítulo final chamado **Referências**. Lembrando que os livros seguidos não são para livre distribuição e caso deseje utilizar algo deverá pedir permissão ao referido autor.

EMENTA

- Introdução à Banco de Dados Relacionais.
- DML, DDL, DCL. Comandos da DML: (Insert / Delete / Update / Select).
- Normalizações: 1fn/2fn/3fn.
- Anomalias: Inclusão / Alteração / Exclusão / Inconsistências. Modelo Conceitual, Lógico e físico.
- . Apresentação do ambiente de desenvolvimento MySQL.
- Componentes de um banco de dados SQL: Tabelas / Diagramas . Criação de bancos de dados através de assistentes.
- Criação de tabelas com assistentes. Chaves primárias simples e compostas.
- Tipos de relacionamentos de tabelas. Relacionando tabelas com diagramas. Inserção de registros (Tuplas) em tabelas pelo assistente. Testes de integridades de dados.
- Criando e executando Query no MySQL
- Executando query no MySQL. Painel de resultados. Gravando registros do painel de resultados.
- Manipulando Visões com linhas de comandos; Manipulando Banco de dados, Tabelas, Relacionamentos entre tabelas, através da DDL.

- Criação de campos auto-incrementáreis através da DDL.Criação de Subconsulta.

- STORED PROCEDURES: Manipulando Strored Procedures com o Object Explorer. Manipulando Stored Procedures através de linhas de comandos. Stored Procedures sem passagem de parâmetros. Stored Procedures com passagem de parâmetros. Stored Procedures com passagem de parâmetros e valores padrões para variáveis. Inner Join, Right Join, Left Join e Union. Trigger.

- Inner Join, Right Join, Left Join e Union. Trigger.

- Introdução ao MongoDB. Modelo de Armazenamento Orientado a Documentos.Tipagem de Dados. Manipulação de Dados.

- Inserção de Dados. Atualização e Exclusão de Dados.

- Consultando Dados.Backup/Restore/Importação de Dados. Fundamentos de Backup e Restore no MongoDB.

- Clusters MongoDB na Nuvem com o Serviço MongoDB Atlas.

2. Introdução a banco de dados relacional

Bem-vindo a apostila de Banco de Dados, aqui serão apresentados os principais pontos para um entendimento claro de um banco, de um SGBD e como ele se comunica com uma linguagem de programação. Iremos abordar também a linguagem SQL – que é a linguagem padrão para qualquer SGBD. Vamos ao que interessa! Por que aprender banco de dados? Se você deseja fazer uma aplicação, que tenha o mínimo de chance de fazer sucesso, com toda certeza você deverá ter um banco de dados, o registro do seu usuário é o que irá te permitir enviar e-mails, desenvolver machine learning e por assim em diante. Segue abaixo a lista das startups bilionárias e seus sistemas de gerenciamento bancos de dados:

- UBER - MySQL
- Facebook - Presto/MySQL
- Moneto - PostgreSQL
- Instagram - Presto/MySQL
- Whatsapp - Presto/MySQL

2.1 O que é um banco de dados?

Banco de dados ou base de dados pode simplesmente ser definida como um local – físico ou virtual – onde se armazenam dados de forma organizada e indexada. Percebam que não estamos nem na parte de informática e sim na parte física e analógica. Com isso em mente, podemos citar como exemplo de banco de dados, o bloco de notas, o documento no word, uma planilha, um caderno e até mesmo um baú. A internet também pode ser considerada um banco de dados

Quando consultamos qualquer bibliografia de banco de dados, o vocabulário utilizado pelos autores são os seguintes:

- ENTIDADES - São as tabelas do banco de dados;
- ATRIBUTOS - São os campos das entidades;
- DOMÍNIOS - São todos os valores possíveis de dados armazenados em um atributo. Exemplo: Todos os códigos de peças armazenados no atributo CODPEÇA.

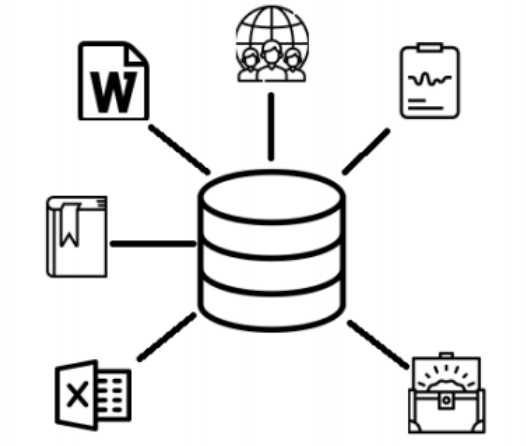


Figura 2.1: Exemplo de Bancos de Dados Fonte: O autor

- TUPLA - Cada registro armazenado numa entidade; lembre-se que um registro refere-se ao conjunto de dados armazenados em todas os atributos de uma tabela.
- CHAVE PRIMÁRIA/ PRIMARY KEY PK - um ou mais atributos, usados para identificar e unificar as tuplas armazenadas;
- CHAVE ESTRANGEIRA/ FOREIGN KEY FK - atributo existente em uma entidade filho, ou seja, entidade que recebe dados de outra entidade em um relacionamento. A entidade FILHO, no relacionamento, deve possuir os mesmos atributos da chave primária da tabela PAI, e estes devem ser do mesmo tipo e tamanho, a fim de tornar possível o relacionamento entre as tabelas

Todos os itens aqui citados, terão seu momento mais específico para definição, estas são apenas algumas palavras que irão ter que se familiarizar.

2.1.1 Sistema de Gerenciamento de Banco de Dados (SGBD)

Um SGBD é um software responsável por tornar o banco de dados gerenciável permitir que ele seja manipulado, ele ainda não é o software que vai ao usuário final e sim o que o DBA ou Database Admin irá utilizar. Sua principal função é facilitar a interface com o banco e do DBA.

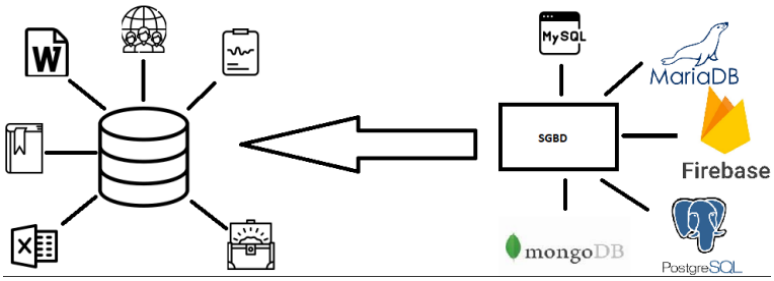


Figura 2.2: BD e SGBD Fonte: O autor

Segue uma lista de SGBDs mais utilizados:

- Oracle
- MySQL
- SQL Server

- PostregeSQL
- MongoDB

2.1.2 A linguagem SQL (*Structed Query Language*)

A linguagem SQL ou Linguagem de Consulta Estruturada é a linguagem responsável por fazer a comunicação entre o SGBD e o BD. A SQL se divide em quatro grupos estes são a DML (Data Manipulation Language), a DDL (Data Definition Language), a DCL (Data Control Language) e a DQL (Data Query Language).

DML

São os comandos que irão manipular o conteúdo das tabelas estes comandos são:

- SELECT
- UPDATE
- DELETE
- INSERT

SELECT – Seleciona dados de uma tabela:

SELECT nomeAlunos FROM alunos;

UPDATE – Atualiza os dados de uma tabela:

UPDATE paciente SET nome =’joão’ WHERE código = 1 AND idade =30;

Atualize (UPDATE) o campo nome **para (SET)** João **quando (WHERE)** o código for igual a 1 **e (AND)** a idade for igual a 30

UPDATE – Comando para atualizar os dados

SET – o que será alterado

WHERE – condição

AND – acrescenta outras condições

INSERT – Insere linhas de dados em uma coluna.

INSERT INTO cliente VALUES (1,’Jose’,30,’(12)5555-1234’);
INSERT – Insere linhas de dados em uma coluna.

INSERT INTO cliente VALUES (1,’Jose’,30,’(12)5555-1234’);

INSERT INTO – Comando para inserir os dados clientes – nome da tabela

VALUES – Valores que serão inseridos

1,'Jose',30, '(11)5555-1234 – Digamos que na tabela clientes contenha as seguintes colunas: codigo, nome, idade, telefone. Os dados (1,'Jose',30, '(12)5555- 5555') correspondes aos dados de cada coluna.

DELETE- Exclui dados de uma tabela.

DELETE FROM paciente WHERE nome='joão';

DELETE – comando para deletar

FROM – em, indica a tabela

WHERE – condição que indica o que será excluído

DDL São os comandos que irão definir a criação do conteúdo das tabelas estes comandos são:
- CREATE
- ALTER
- DROP

CREATE - Comando que irá criar nossa base de dados e nossas tabelas
create database colegio;
CREATE - Comando para criar

Database - o que será criado

ALTER – Comando que nos permite fazer ajustes nas tabelas criadas, adicionando uma coluna ou removendo-a.

ALTER TABLE alunos ALTER COLUMN sobrenome VARCHAR(75) NOT NULL;

DROP - DROP- É um comando para deletar de uma vez e sem chances de recuperação, a menos que haja um backup, uma tabela ou um banco de dados

DROP TABLE alunos;

DCL

Servem para controlar a parte de segurança do banco de dados – dar ou retirar permissões- estas são:
- GRANT
- REVOKE
- DENY

DQL

Aqui cabe um parênteses, a DQL em alguns livros listam apenas como o SELECT, em outros listam o SELECT na DML, neste caso vamos considerar DQL como SELECT e tudo que me permite selecionar ou filtrar dados, tais como:
- WHERE
- WHEN
- AND

- IF

2.2 Componentes de uma tabela

2.2.1 Entidade

Identifica o objeto de interesse do sistema e tem "vida" própria, ou seja, a representação abstrata de um objeto do mundo real sobre o qual desejamos guardar informações. Uma entidade é caracterizada pelos seguintes adjetivos:

- Algo do mundo real com existência independente
- Possui atributos
- Possui um valor para cada atributo
- Os atributos são propriedades particulares que descrevem as entidades

2.2.2 Atributos

São propriedades (características) que identificam as entidades. Uma entidade é representada por um conjunto de atributos. Os atributos podem ser simples, composto, multivalorado ou determinante. Nome, endereço, telefone e cidade, por exemplo, são atributos da entidade Clientes. Enquanto que salário, cargo e departamento são atributos da entidade funcionários. Existem quatro tipos de atributos: simples, composto, multivalorado e determinante.

Atributo Simples Não possui qualquer característica especial. A maioria dos atributos serão simples. Quando um atributo não é composto, recebe um valor único como nome, por exemplo e não é um atributo chave, então ele será atributo simples. A maioria dos atributos são considerados simples. Em uma entidade cliente, por exemplo, poderemos considerar como atributo simples: nome, sexo, data de nascimento, dentre outros.

Atributo Composto O seu conteúdo é formado por vários itens menores. Exemplo: Endereço. Seu conteúdo poderá ser dividido em vários outros atributos, como: Rua, Número, Complemento, Bairro, Cep e Cidade. Este tipo de atributo é chamado de atributo composto. Veremos mais de sua aplicação no post sobre normalização de dados.

É importante considerar que na aplicação do banco de dados um atributo composto geralmente é desmembrado, ou seja, para o caso do endereço, podemos desmembrá-lo em vários atributos simples, como: Rua, número, complemento, bairro, cidade e cep. Conceitualmente é aceito o endereço como um único atributo, mas na prática geralmente é feito este desmembramento para permitir a organização dos dados inseridos e facilitar a busca e indexação dos mesmos.

Atributo Multivalorado O seu conteúdo é formado por mais de um valor.

Exemplo: Telefone. Uma pessoa poderá ter mais de um número de telefone. É indicado colocando-se um asterisco precedendo o nome do atributo. O atributo multivalorado serão tratados com mais detalhes na normalização de dados.

Este tipo de atributo é aceito conceitualmente, mas ele pode ser um problema no banco de dados. Há duas possibilidades para tratar com ele. A primeira é mantê-lo como multivalorado e permitir que mais de um dado seja inserido no mesmo campo, como por exemplo: dois números de telefone. A segunda alternativa é aplicar o processo de normalização de dados e transformá-lo em uma entidade a parte ou uma tabela no banco de dados e relacioná-la com a tabela principal.

A primeira alternativa é mais simples, mas teríamos o problema da consulta de dados, caso precisássemos fazer uma consulta pelo número de um dos telefones apenas. A segunda é mais trabalhosa, porém é mais eficaz.

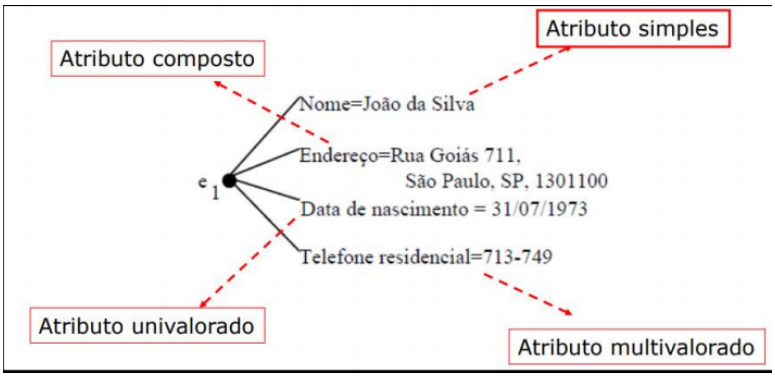


Figura 2.3: Tipos de Atributos Fonte: O autor

2.2.3 Tipos de Atributos e valores que aceitam

Na hora de armazenar datas, há que ter em conta que MySQL não verifica de uma maneira estrita se uma data é válida ou não. Simplesmente comprova que o mês está compreendido entre 0 e 12 e que o dia está compreendido entre 0 e 31.

- **Date**: tipo data, armazena uma data. A margem de valores vai desde o 1 de Janeiro de 1001 ao 31 de dezembro de 9999. O formato de armazenamento é de ano-mesdia.

- **DateTime**: Combinação de data e hora. A margem de valores vai desde o 1 ed Janeiro de 1001 às 0 horas, 0 minutos e 0 segundos ao 31 de Dezembro de 9999 às 23 horas, 59 minutos e 59 segundos. O formato de armazenamento é de ano-mes-dia horas:minutos:segundos

- **TimeStamp**: Combinação de data e hora. A margem vai desde o 1 de Janeiro de 1970 ao ano 2037. O formato de armazenamento depende do tamanho do campo:

- **Time**: armazena uma hora. A margem de horas vai desde -838 horas, 59 minutos e 59 segundos. O formato de armazenamento é 'HH:MM:SS'.

- **Year**: armazena um ano. A margem de valores permitidos vai desde o ano 1901 ao ano 2155. O campo pode ter tamanho dois ou tamanho 4 dependendo de se queremos armazenar o ano com dois ou quatro algarismos.

- **Char(n)**: armazena uma cadeia de longitude fixa. A cadeia poderá conter desde 0 até 255 caracteres.

- **VarChar(n)**: armazena uma cadeia de longitude variável. A cadeia poderá conter desde 0 até 255 caracteres. Dentro dos tipos de cadeia pode-se distinguir dois subtipos, os tipo Test e os tipo Blob (Binary Large Object) A diferença entre um tipo e outro é o tratamento que recebem na hora de ordená-los e compará-los. No tipo test ordena-se sem ter importância as maiúsculas e as minúsculas e no tipo blob ordenase tendo em conta as maiúsculas e minúsculas.

- **TinyText e TinyBlob**: Coluna com uma longitude máxima de 255 caracteres.

- **TinyInt**: é um número inteiro com ou sem signo. Com signo a margem de valores válidos é desde -128 até 127. Sem signo, a margem de valores é de 0 até 255.

- **Integer, Int**: número inteiro com ou sem signo. Com signo a margem de valores válidos é desde -2147483648 até 2147483647. Sem signo, a margem de valores é de 0 até 429.496.295

- **Real, Double**: número em vírgula flutuante de dupla precisão. Os valores permitidos vão desde -1.7976931348623157E+308 até -2.2250738585072014E-308, 0 e desde 2.2250738585072014E-308 até 1.7976931348623157E+308

- **Float**: número pequeno em vírgula flutuante de precisão simples. Os valores válidos vão desde -3.402823466E+38 até -1.175494351E-38,0 eté desde 175494351E-38 até 3.402823466E+38.

- **Decimal, Dec, Numeric**: Número em vírgula flutuante desempacotado. O número armazena-se como uma cadeia.

2.2.4 Chave Primária

A chave de um arquivo é um atributo ou um conjunto de atributos que identifica, de forma única, cada registro do arquivo. Todo arquivo deve possuir uma chave. A função da chave é garantir a unicidade dos registros. Por exemplo, o cadastro de motoristas no Detran tem como chave o número da carteira de motoristas. Não há dois motoristas com o mesmo número de carteira.

A chave de um arquivo deve ser:

- Única
- Universal
- Imutável

Uma chave deve ser única. Não podem existir dois registros com o mesmo valor para a chave. Às vezes, nós precisamos utilizar uma combinação de atributos para garantir a unicidade da chave. No exemplo dado anteriormente, do arquivo de pedidos, o número do pedido é atribuído em sequência em cada filial. Isto significa que em duas filiais diferentes podem existir pedidos com o mesmo número. Por isso, a chave, para ser única, precisa ser uma combinação do número da filial com o número do pedido

2.2.5 Chave Estrangeira

Chave estrangeira, ou Foreign Key (FK), ou ainda chave externa é a chave que permite a referência a registros oriundos de outras tabelas. Ou seja, é o campo ou conjunto de campos que compõem a chave primária de uma outra tabela. A utilização da chave estrangeira possibilita a implementação da integridade de dados diretamente no banco de dados, conhecida como integridade referencial. Uma chave estrangeira é a representação de um relacionamento entre tabelas. Isso significa que as tabelas são capazes de enxergar uma as outras. Este assuntos será abordado mais a frente quando falarmos sobre Entidade e Relacionamento, aonde também abordaremos mais profundamente a questão de Entidades tais como Entidades FOrtes e Fracas.

2.3 Lista de Exercícios

1- O que é um banco de dados?

2- Explique o que é um SGBD e cite exemplos.

3- O que são DML, DCL, DDL e DQL? Cite exemplos.

4- A internet pode ser considerada um banco de dados? Por quê?

5- Descreva o que é uma chave primária.

6- Descreva o que é chave estrangeira.

7- Explique o que é atributo composto.

8- Explique o que é atributo multivalorado.

9- Cite a diferença entre chave primaria e chave estrangeira.

10- Uma variável do tipo char armazena quais valores?

11- Um atributo do tipo TIME é capaz de armazenar qual range de valor? E qual sua utilização?

12- Qual o limite de valor de uma variável do tipo REAL?

13- Qual tipo de variável em BD é capaz de armazenar um valor mair DOUBLE ou REAL?

14- Qual a diferença entre Time, Year, DateTime e Date?

15- O que é uma Entidade?

16- Crie, no seu banco de dados, a Figura 2.4, insira os valores apresentados e em seguida escreva as consultas solicitadas abaixo. OBS: Os valores em branco devem ser nulos no banco de dados.

17- Utilizando o banco de dados que criou usando a Figura 2.4 como base, responda as questões 17, 18, 19, 20, 21 e 22. Pesquise os itens que foram vendidos sem desconto. As colunas presentes no resultado da consulta são: ID_NF, ID_ITEM, COD_PROD E VALOR_UNIT

18 - Pesquise os itens que foram vendidos com desconto. As colunas presentes no resultado da consulta são: ID_NF, ID_ITEM, COD_PROD, VALOR_UNIT E O VALOR VENDIDO. OBS: O valor vendido é igual ao VALOR_UNIT - (VALOR_UNIT*(DESCONTO/100)).

19 - Altere o valor do desconto (para zero) de todos os registros onde este campo é nulo.

21 - Pesquise os itens que foram vendidos. As colunas presentes no resultado da consulta são:

ID_NF, ID_ITEM, COD_PROD, VALOR_UNIT, VALOR_TOTAL, DESCONTO, VALOR_VENDIDO. OBS: O VALOR_TOTAL é obtido pela fórmula: QUANTIDADE * VALOR_UNIT. O VALOR_VENDIDO é igual a VALOR_UNIT - (VALOR_UNIT*(DESCONTO/100))

22 - Pesquise o valor total das NF e ordene o resultado do maior valor para o menor. As colunas presentes no resultado da consulta são: ID_NF, VALOR_TOTAL. OBS: O VALOR_TOTAL é obtido pela fórmula: QUANTIDADE * VALOR_UNIT. Agrupe o resultado da consulta por ID_NF

ID_NF	ID_ITEM	COD_PROD	VALOR_UNIT	QUANTIDADE	DESCONTO (%)
1	1	1	25,00	10	5
1	2	2	13,50	3	
1	3	3	15,00	2	
1	4	4	10,00	1	
1	5	5	30,00	1	
2	1	3	15,00	4	
2	2	4	10,00	5	
2	3	5	30,00	7	
3	1	1	25,00	5	
3	2	4	10,00	4	
3	3	5	30,00	5	
3	4	2	13,50	7	
4	1	5	30,00	10	15
4	2	4	10,00	12	5
4	3	1	25,00	13	5
4	4	2	13,50	15	5
5	1	3	15,00	3	
5	2	5	30,00	6	
6	1	1	25,00	22	15
6	2	3	15,00	25	20
7	1	1	25,00	10	3
7	2	2	13,50	10	4
7	3	3	15,00	10	4
7	4	5	30,00	10	1

Figura 2.4:

3. Formas Normais

O processo de normalização compreende o uso de um conjunto de regras, chamados de formas normais. Ao analisarmos o banco de dados e verificarmos que ele respeita as regras da primeira forma normal, então podemos dizer que o banco está na "primeira forma normal". Caso o banco respeite as primeiras três regras, então ele está na "terceira forma normal". Mesmo existindo mais conjuntos de regras para outros níveis de normalização, a terceira forma normal é considerada o nível mínimo necessário para grande parte das aplicações. [Microsoft 2007]

3.1 Primeira Forma Normal (1FN)

Um banco de dados dentro dos padrões de normalização reduz o trabalho de manutenção e ajuda a evitar o desperdício do espaço de armazenamento. Se tivermos cadastrado no banco um cliente e tivermos o seu telefone registrado em mais de uma tabela, havendo uma alteração no seu número de telefone, teremos que fazer essa atualização em cada tabela. A tarefa se torna muito mais eficiente se tivermos seu telefone registrado em apenas uma tabela.

3.1.1 Primeira Forma Normal (1FN)

A primeira forma normal serve ara reprovar atributos multivalorados, compostos e suas combinações. O domínio de um atributo deve incluir apenas valores atômicos (indivisíveis), e o valor de qualquer atributo em uma tupla deve ser únicos valor domínio desse atributo.

Uma tabela esta na primeira forma normal quando:

- Somente possui valores atômicos
- Não há grupos de atributos repetidos (há apenas um dado por coluna nas linhas)
- Existe uma chave primária
- Relação não possui atributos multivalorados ou relações aninhadas

A figura 3.1 mostra uma tabela **não** normalizada.

Perceba que nessa tabela, os campos "tel_cliente" e "endereço_cliente" estão poluídos ou seja, com mais de um valor no atributo, no telefone os valores se repetem e no endereço é um atributo multivalorado. A figura 3.2 mostra uma tabela na sua primeira normalização.

```
mysql> select * from cliente;
+-------------+--------------+---------------------+-----------------------------------------+
| cod_cliente | nome_cliente | tel_cliente         | endereco_cliente                        |
+-------------+--------------+---------------------+-----------------------------------------+
|        2453 | Ana          | 4213-6532/975635632 | Rua Min. Alberto Jorge 1492 - Vila Primavera |
|        2532 | José         | 99653-2145/2865-3212 | Rua das Giestas, 234, ap.45- Vila Bela |
|        2536 | Marcos       | 2643-5321           | Av. Carlos de Almeida, 459 - Vila das Rosas |
+-------------+--------------+---------------------+-----------------------------------------+
```

Figura 3.1: Tabela não normalizada Fonte: O autor

```
+-------------+--------------+-------------+-------------+--------------------------+------------+---------------+
| cod_cliente | nome_cliente | tel1_cliente | tel2_cliente | endereco_cliente         | complemento | bairro        |
+-------------+--------------+-------------+-------------+--------------------------+------------+---------------+
|        2453 | Ana          | 4213-6532   | 975635632   | Rua Min. Alberto Jorge 1492 | S/C        | Vila Primavera |
|        2532 | Jose         | 99653-2145  | 2865-3212   | Rua das Giestas, 234     | Ap.45      | Vila Bela     |
|        2536 | Marcos       | 2643-5321   |             | Av. Carlos de Almeida, 459 | S/C        | Vila das Rosas |
+-------------+--------------+-------------+-------------+--------------------------+------------+---------------+
```

Figura 3.2: Tabela na 1FN Fonte: O autor

3.1.2 Segunda Forma Normal (2FN)

Uma tabela esta na 2FN se:

tbl_Peças					
Cod_Peça	Cod_Fornec	Local_Fornecedor	Qtde_Estoque	Tel_Fornecedor	Qtde_Caixas
0009	121	São Paulo	512	2365-6532	52
0023	122	Manaus	263	4465-8632	27
0065	121	São Paulo	196	2365-6532	20
0071	123	Porto Alegre	89	2956-8653	9
0073	122	Manaus	296	4465-8632	30

Figura 3.3: Tabela apenas na 1FN Fonte: O autor

- Esta na 1FN
- Todos os atributos que não-chave são funcionalmente dependentes de todas as partes da chave primária.
- Não existem dependências parciais.
- Caso contrário, deve-se gerar uma nova tabela com os dados.

Deve-se criar uma nova relação para cada chave PK ou combinação de atributos que forem determinantes em uma dependência funcional. Esse atributo será a PK na nova tabela. Mova os

atributos não-chave dependentes desta PK para a nova tabela.

Após aplicar a segunda forma normal, teremos:

tbl_Peça			
Cod_Peça	Cod_Fornec	Qtde_Estoque	Qtde_Caixas
0009	121	512	52
0023	122	263	27
0065	121	196	20
0071	123	89	9
0073	122	296	30

Figura 3.4: Tabela na 2FN Fonte: O autor

tbl_Fornecedor		
Cod_Fornec	Local_Fornecedor	Tel_Fornecedor
121	São Paulo	2365-6532
122	Manaus	4465-8632
PK 123	Porto Alegre	2956-8653

Figura 3.5: Segunda tabela na 2FN Fonte: O autor

3.1.3 Terceira Forma Normal (3FN)

Uma tabela está na 3FN se ela estiver na segunda forma normal e se nenhuma coluna não-chave depender de outra coluna não-chave. Para cada atributo não—chave que for um determinante na relação, crie uma nova tabela. Esse atributo será a PK na nova relação. Mova então todos os atributos que são dependentes funcionalmente do atributo chave para a nova tabela. O Atributo (PK na nova relação) fica também na tabela original, e servirá como uma chave estrangeira para associaras duas relações

tbl_Venda				
Nota_Fiscal	Cod_Vendedor	Nome_Vendedor	Cod_Produto	Qtde_vendida
15326	002	Leila	132	10
15327	006	Ana	153	12
15328	002	Leila	143	11
15329	009	Fábio	132	9
15330	007	Renato	153	12

Figura 3.6: Tabela fora da 3FN Fonte: O autor

tbl_Venda			
Nota_Fiscal	Cod_Vendedor	Cod_Produto	Qtde_vendida
15326	002	132	10
15327	006	153	12
15328	002	143	11
15329	009	132	9
15330	007	153	12

Figura 3.7: Tabela na 3FN Fonte: O autor

tbl_Vendedor	
Cod_Vendedor	Nome_Vendedor
002	Leila
006	Ana
007	Renato
009	Fábio

Figura 3.8: Tabela na 3FN Fonte: O autor

3.2 Lista de Exercícios

1- Descreva o que é 1FN.

2- Descreva o que é 2FN.

3- Descreva o que é 3FN.

4- Normalize a tabela a seguir para a 1FN.

Código_cliente	Nome	Telefone	Endereço
C001	José	9563-6352 9847-2501	Rua Seis, 85 Morumbi 12536-965
C002	Maria	3265-8596	Rua Onze, 64 Moema 65985-963
C003	Janio	8545-8956 9598-6301	Praça ramos Liberdade 68858-633

Figura 3.9:

5- Normalize a tabela do exercício 4 para a 2FN.

6- Normalize a tabela do exercício 4 para a 3FN.

7- Normalize a ficha médica abaixo seguindo os conceitos de normalização de dados.

Ficha Médica			
Número paciente:		Nome:	
Data de Nasc.:	Sexo:		Convênio:
Est. Civil:	RG:		Telefone:
Endereço:			
Consultas			
Número Consulta	Data	Médico	Diagnóstico
Exames			
Número Consulta	Exame		Data

Figura 3.10:

8- Coloque a tabela (3.11) abaixo na 3FN

9- Coloque a tabela (3.12) abaixo na 3FN

10- A normalização é um processo obrigatório?

Filial

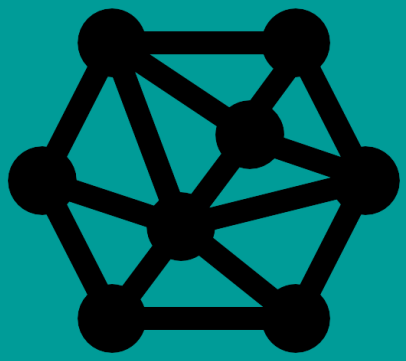
numFilial	enderecoFilial	telefones	numGerente	nomeGerente
B001	Rua Jefferson	503-555-3618, 503-555-2727, 503-555-6534	1	Tomas
B002	City Center Plaza	206-555-6756, 206-555-8836	2	Ana
B003	8th Avenue	212-371-3000	3	Maria
B004	14th Avenue	206-555-3131, 206-555-4112	4	Carlos

Figura 3.11:

Pedido

idPedido	dataPeddo	codProduto	nomeProduto	qtde	valorUnitario	valorTotal
1	01/07/09	1234	HD 250GB	2	R$ 100	R$ 200
2	01/07/09	1235	HD 180GB	1	R$ 80	R$ 80
3	03/07/09	1235	HD 180GB	4	R$ 80	R$ 320
4	05/07/09	1234	HD 250GB	6	R$ 100	R$ 600

Figura 3.12:



4. Modelagem de Dados e MER

4.1 Introdução ao MER

Modelagem de dados é o ato de explorar estruturas orientadas a dados. Como outros artefatos de modelagem, modelos de dados podem ser usados para uma variedade de propósitos, desde modelos conceituais de alto nível até modelos físicos de dados. Do ponto de vista de um desenvolvedor atuando no paradigma orientado a objetos, modelagem de dados é conceitualmente similar à modelagem de classes. Com a modelagem de dados identificamos tipos de entidades da mesma forma que na modelagem de classes identificamos classes. Atributos de dados são associados a tipos de entidades exatamente como associados atributos e operações às classes. Existem associações entre entidades, similar às associações entre classes – relacionamento, herança, composição e agregação são todos conceitos aplicáveis em modelagem de dados

4.1.1 Modelo Conceitual

Nesta etapa da modelagem de dados o DBA deve:

- Conhecer o negócio
- Rascunhar as principais entidades do BD. e seus principais atributos.
- Não se preocupar com os relacionamentos n : n.

4.1.2 Modelo Lógico

Nesta etapa da modelagem de dados o DBA deve:

- Representar o negócio.
- Criar as entidades de relacionamentos para substituir os relacionamentos n:n
- Definir as chaves primárias de cada entidade.
- Normalizar as tabelas.
- Adequar aos padrões do banco de dados escolhido.
- Documentar as entidades e seus atributos.

4.1.3 Modelo Fisico

Nesta etapa da modelagem de dados o DBA deve:

- Tomar ciência das limitações do banco de dados.
- Considerar os requisitos dos softwares que farão acesso ao banco de dados.
- Criar fisicamente as entidades, relacionamentos, chaves, índices, definir níveis de acessos entre outros, ou seja, criar fisicamente o banco de dados projetado nas etapas anteriores.

4.1.4 Modelo Entidade Relacionamento(MER)

É um modelo conceitual de alto-nível, ou seja, é projetado para ser compreensível aos usuários comuns.

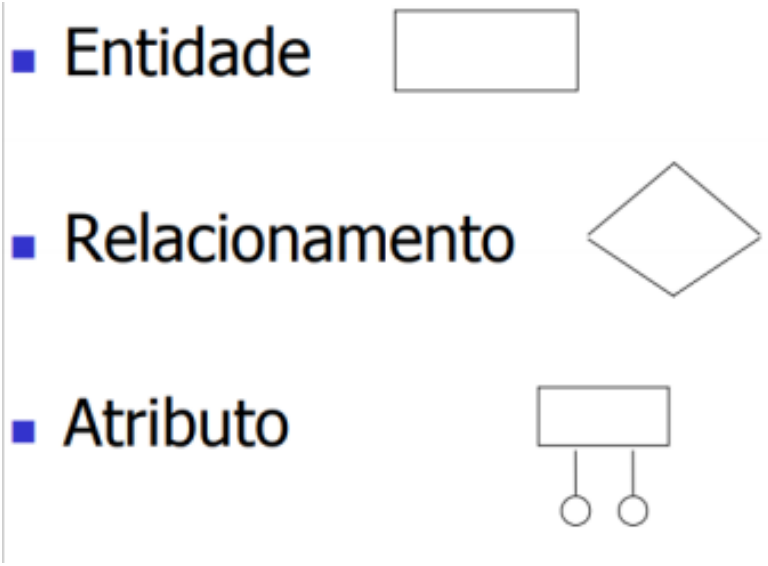


Figura 4.1: Representação gráfica MER Fonte: O autor

4.1.5 Relacionamento

Representa a associação entre os elementos do conjunto de uma entidade com outra entidade. Por exemplo:

"Joao está matriculado na disciplina banco de dados"

Figura 4.2: Exemplo relacionamento Fonte: O autor

4.1.6 Cardinalidade de Relacionamentos

Corresponde ao número de entidades com que um determinado conjunto de entidades pode se relacionar através de um determinando relacionamento.

Relacionamento 1:1 - O João é casado com a Maria

- João - Elemento do conjunto de valores do atributo Nome da entidade Homem.
- Maria - Elemento do conjunto de valores do atributo Nome da entidade Mulher.
- Casado - Ligação entre um João e Maria, sendo que João pode ser casado com uma e apenas uma Maria, assim como Maria pode ser casada com João.

Relacionamento 1:n ou n:1 - O Pedro trabalha no Departamento Pessoal

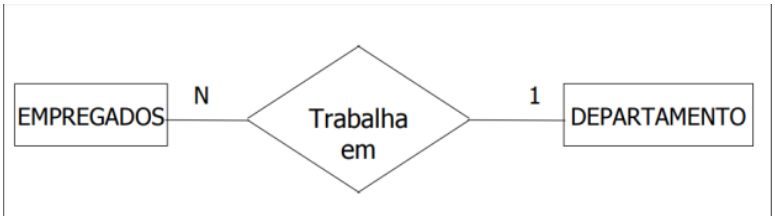


Figura 4.3: Relacionamento 1:N Fonte: O autor

Relacionamento n:n ou n:m ou *:* O Antônio está matriculado na disciplina Banco de Dados. Ligação existente entre um aluno e uma disciplina, onde um aluno pode estar matriculado em várias disciplinas e cada disciplina pode ter vários alunos matriculados.

Figura 4.4: Relacionamento N:N Fonte: O autor

4.2 Lista de Exercícios

1- Descreva o diagrama abaixo:

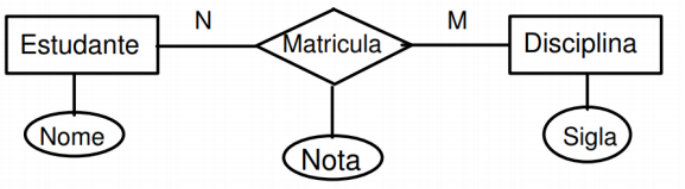


Figura 4.5:

2- Monte o modelo Entidade Relacionamento do texto abaixo

Um professor leciona várias matérias. Cada matéria tem um professor. Um professor trabalha em várias turmas. Cada escola tem mais de um professor.

3- Monte o modelo Entidade Relacionamento do texto abaixo:

Alunos fazem diversas disciplinas em uma escola. Cada disciplina possui vários alunos e um professor. Cada professor ministra uma ou mais disciplinas. Cada disciplina ministrada por um professor para uma turma de alunos ocorre em uma sala e horário específicos. Alunos tem nome, prontuário, endereço, data de nascimento e nome da mãe. Disciplinas possuem sigla, carga horária e descrição. Professores tem número de registro, nome e área de especialização conjunto de alunos cursando uma disciplina é, às vezes, chamado de "turma".

4- Monte o modelo Entidade Relacionamento do texto abaixo:

Um soldado, que possui nome, registro militar (RM) e data de nascimento, tem a si atribuídas as armas que estão em seu nome. As armas possuem número de série, tipo, calibre e tipos de munição. Munições de um determinado tipo podem ser utilizadas nas armas de mesmo calibre. Armas são limpas por diversos soldados e um soldado pode ser responsável pela limpeza de diversas armas.

5- Monte o modelo Entidade Relacionamento do texto abaixo: *Um médico, o qual se tem registrados CRM, nome e especialidade, atende pacientes. Um paciente, para o qual é necessário sabermos seu nome, endereço, idade e código de registro no hospital, pode ser tratado por vários médicos. Um médico receita, para um determinado paciente em uma determinada consulta, um ou mais medicamentos e pode requisitar exames. Um exame é identificado por seu número de série e tem anotados data de realização e descrição*

6- Monte o modelo Entidade Relacionamento do texto abaixo:

Um reino possui um rei, cada rei tem nome e numeração, por exemplo (Dom João VI). Vários vassalos possuem um rei e vivem um reino, os vassalos possuem nomes e animais que os representam. Cada animal representa uma casa.

7- Monte o modelo Entidade Relacionamento do texto abaixo:

A Biblioteca Multimídia mantém um acervo de itens que podem ser livros, cds e dvds. Todos os itens possuem um código, título e descrição. Além disso, os livros possuem nome dos autores, assuntos, ano de publicação e editora; os cds possuem o nome da produtora, ano, músicas (nome e cantor); e os dvds possuem uma produtora, diretor e o nome dos atores principais. A biblioteca mantém o cadastro dos fornecedores de seus itens por código, nome, endereço e telefones. Os associados da biblioteca são identificados por um número, nome, endereço e fone. Os associados fazem empréstimos dos itens em determinada data e devolvem em uma data específica também. Todo empréstimo é verificado por um funcionário da biblioteca em determinada data e horário, a fim de detectar se o item que está sendo devolvido não se encontra danificado. Os funcionários possuem um número de cadastro, nome, endereço e sexo. Os funcionários são responsáveis pelas estantes onde são armazenados os itens do acervo. As estantes possuem um número, andar e sala. Um item fica armazenado em uma única estante, mas uma estante pode armazenar vários itens.

8- Monte o modelo Entidade Relacionamento do texto abaixo:

A empresa aérea mantém o cadastro dos aeroportos nos quais ela tem autorização de pouso, sendo mantido o código, nome, cidade, estado e país do aeroporto. Os voos partem de um aeroporto de origem e vão até um aeroporto de destino, e possuem um código, duração e hora de partida. Além

disso, os voos podem ter escalas outros aeroportos. Os passageiros são cadastrados por código, RG, CPF, nome, telefone. Eles compram passagens aéreas (número, data emissão, validade, valor). As passagens estão associadas aos voos, sendo registrada nessa associação a data de embarque em cada voo.

9- Monte o modelo Entidade Relacionamento do texto abaixo:

No Hospital Modelo, cada paciente que é atendido é cadastrado por um Código de Registro Hospitalar (CRH), nome, endereço, cidade, fone, data de nascimento, sexo e responsável. Os pacientes são consultados pelos médicos em uma determinada data e hora. Os médicos possuem um CRM, nome, endereço, cidade, fones e especialidade. Ao consultar um paciente, o médico pode solicitar a realização de exames. Os exames são cadastrados por tipo, descrição, data da realização e resultado. Os pacientes podem ser internados em um leito do hospital, dependendo do encaminhamento do médico que o consultou. Os leitos do hospital são cadastrados por ala, número do quarto,número do leito e tipo. Na internação são registradas a data de entrada e a data prevista de saída. A cada paciente são associadas ocorrências de seu prontuário, informando problema, data de constatação, receita, cuidados especiais.

5. Trigger, View, Stored Procedure e Join

5.1 Comandos Armazenados SQL

5.1.1 Trigger

Uma TRIGGER ou gatilho é um objeto de banco de dados, associado a uma tabela, definido paraser disparado, respondendo a um evento em particular. Tais eventos são os comandos da DML(Data Manipulation Language): INSERT, REPLACE, DELETE ou UPDATE. Podemos definirinúmeros TRIGGERS em uma base de dados baseados diretamente em qual dos comandos acimairá disparálo, sendo que, para cada um, podemos definir apenas um TRIGGER. Os TRIGGERSpoderão ser disparados para trabalharem antes ou depois do evento.

5.1.2 View

Uma View (Exibição / Visão) é uma tabela virtual (estrutura de dados) baseada no conjunto de resultados de uma consulta SQL, criada a partir de um conjunto de tabelas (ou outras views) presentes no banco, que servem com tabelas-base. Contém linhas e colunas como uma tabela real, e pode receber comandos como declarações JOIN, WHERE e funções como uma tabela normal. A view fica armazenada no banco de dados. Mostra sempre resultados de dados atualizados, pois o motor do banco de dados recria os dados toda vez que um usuário consulta a visão.

5.1.3 Stored Procedure

Uma Stored Procedure nada mais é do que um recurso nativo do banco de dados que lhe permite armazenar no servidor um conjunto de instruções com as quais você pode processar um conjunto de valores ou ações.

5.1.4 Claúsulas Join

O Inner Join é o método de junção mais conhecido e, como ilustra a Figura 2, retorna os registros que são comuns às duas tabelas.

O Left Join, cujo funcinamento é ilustrao na Figura 4, tem como resultado todos os registros que estão na tabela A (mesmo que não estejam na tabela B) e os registros da tabela B que são comuns

à tabela A. Usando o Right Join, conforme mostra a Figura 6, teremos como resultado todos os

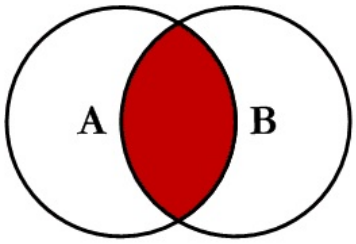


Figura 5.1: Inner Join Fonte: Devmedia

registros que estão na tabela B (mesmo que não estejam na tabela A) e os registros da tabela A que são comuns à tabela B.

O Outer Join (também conhecido por Full Outer Join ou Full Join), conforme mostra a Figura 8, tem como resultado todos os registros que estão na tabela A e todos os registros da tabela B.

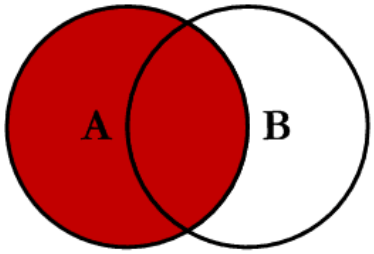


Figura 5.2: Left Join Fonte: Devmedia

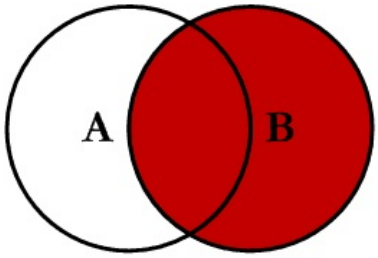


Figura 5.3: Right Join Fonte: Devmedia

5.2 Lista de Exercícios

Dado o banco de dados abaixo, faça o que se pede:

CREATE DATABASE BancoItau; use BancoItau; CREATE TABLE Clientes (ClienteCodigo int PRIMARY KEY NOT NULL AUTO_INCREMENT, ClienteNome varchar(20), ClienteEnd varchar(50), ClienteEmail varchar(50), ClienteIdade int, ClienteProfissao varchar(50),)Engine='InnoDB';

insert into Clientes values (1, ‘Robervaldo’, ‘Rua dos Alecrins’, ‘robervaldo@gmail.com’, 35, ‘Gerente de loja’);

insert into Clientes values (2, ‘Luciana’, ‘Rua Voluntarios da Patria‘, ‘luci_ana@gmail.com’, 29, ‘Confeiteira’);

insert into Clientes values (3, ‘Astolfo, ‘Avenida Brasil’,‘Astolf@yahoo.com’, 27, ‘Autonomo’);

insert into Clientes values (4, ‘Cecilia, ‘Av. Heitor Vila Lobos’, ‘ceci@hotmail.com’, 25, ‘Desenvolvedora Web’);

insert into Clientes values (5, ‘Enzo’, ‘Rua Manoel Bosco’, ‘enzo2343@outlook.com’, 33, ‘Ator’);

insert into Clientes values (6, ‘Maria Lucimara, ‘Rua Marte,‘marilu@gmail.com’, 44, ‘Cabeleireira’);

insert into Clientes values (7, ‘Joel’, ‘Rua Omega’, ‘joelslc@ymail.com’, 35, ‘Carteiro’);

insert into Clientes values (8, ‘Luana, ‘Rua dos Alecrins2’,luh_luh@gmail.com’, 26, ‘Professora’);

1- Crie as seguintes Triggers: - Quando fizer o insert, automaticamente me traga a data em que houve esse insert. Nesse caso, insira mais 2 pessoas. - Ao Deletar um cliente me traga todos os clientes que ainda restam.

2- Crie uma View que traga a média das idades de todos os clientes.

3- Crie uma Procedure que busque por todos os clientes que tenham mais de 24 anos

6. BDs NoSQL

Quando as pessoas usam o termo “banco de dados NoSQL”, geralmente o usam para se referir a qualquer banco de dados não relacional. Alguns dizem que o termo “NoSQL” significa “não SQL”, enquanto outros dizem que significa “não apenas SQL”.

6.1 Banco de dados NOSQL

Um erro muito comum é quando dizem que os bancos de dados não relacionais não armazenam bem dados de relacionamento. Eles podem armazenar dados de relacionamento, mas apenas os armazenam de maneira diferente dos bancos de dados relacionais. De fato, muitos consideram a modelagem relacionamentos nos bancos de dados NoSQL mais fácil do que nos bancos de dados SQL, porque os dados relacionados não precisam ser divididos entre as tabelas.

Os modelos de dados NoSQL permitem por exemplo, que os dados relacionados sejam feitos em uma única estrutura de dados. Diferentemente dos bancos relacionais, a estrutura de dados não precisa ser definida previamente, portanto, em uma mesma “tabela” você pode ter dados com propriedades diferentes.

Os bancos de dados NoSQL surgiram no final dos anos 2000, à medida que o custo do armazenamento diminuiu drasticamente. Já se foram os dias em que era necessário criar um modelo de dados complexo e difícil de gerenciar, simplesmente com o objetivo de reduzir a duplicação de dados.

Existem diversos tipos de banco de dados não relacional, eles são categorizadas pela sua maneira de armazenamento de dados. Os dois tipos mais utilizados de bancos NoSQL são:

- Banco de Documentos: Armazena seus dados em documentos semelhantes aos objetos JSON (JavaScript Object Notation). Possuem normalmente poderosas linguagens de consulta, esses bancos de dados de documentos são ótimos para usos gerais. Eles podem ser facilmente escalados horizontalmente para acomodar grandes volumes de dados. O MongoDB é constantemente classificado como o banco de dados NoSQL mais popular no mundo, e é um exemplo de banco de dados de documentos. Confira abaixo um exemplo de uma collection (“tabela”) do MongoDB:
- Chave-Valor: São um tipo mais “simples” de banco de dados, em que cada item contém chaves

e valores. Esses valores podem ser qualquer tipo de dado, um texto, um número, um JSON e eles podem ser recuperados fazendo referência a sua chave, fazendo com que sua consulta seja muito simples. Esses bancos são ótimos para quando você precisa armazenar grandes quantidades de dados, mas não precisa executar consultas complexas neles. Os usos mais comuns são para armazenamento de dados em cache. Redis e DynanoDB são provavelmente os bancos mais populares desse tipo

- Armazenando grandes volumes de dados sem estrutura definida. Um banco de dados NoSQL não limita os campos, diferente das colunas no SQL. Além disso, você pode adicionar novas propriedades conforme as necessidades dos negócios mudam, sem se preocupar com o impacto nas demais informações armazenadas.
- Usando computação e armazenamento em nuvem. Com o avanço e barateamento dos serviços clouds, é possível usar bancos de dados NoSQL inicialmente pequenos, como eles são projetados para escalar horizontalmente você consegue facilmente escalar-los conforme sua necessidade aumenta.
- Desenvolvimento rápido. Se você estiver desenvolvendo usando metodologias ágeis modernas, um banco de dados relacional provavelmente o atrasará. Um banco de dados NoSQL não requer o nível de preparação normalmente necessário para bancos de dados relacionais.

6.1.1 Principais bancos de dados NoSQL

REDIS O Redis é o banco de dados NoSQL de chave-valor mais utilizado em todo o mundo. Ele vincula um valor a uma chave na sua estrutura, o que facilita o armazenamento e a busca desses dados. Por isso, é muito utilizado pelos desenvolvedores.

MEMCACHED Este banco de dados não relacional também faz o armazenamento com chave-valor e usa um cache de memória distribuída. Geralmente, é utilizado para criar sites dinâmicos, pois acelera a abertura das páginas e diminui as buscas de dados de fontes externas.

CASSANDRA Este banco de dados NoSQL foi desenvolvido no Facebook. Ele usa um banco de dados descentralizado, em que os dados são armazenados em vários datacenters. Ele é otimizado para cluster e fornece baixa latência em suas atualizações.

HBASE O Hbase é um banco de dados que utiliza conjunto de linhas e colunas para armazenar as informações. Ele é utilizado em diferentes plataformas como o LinkedIn, Facebook e Spotify

AMAZON DYNAMODB Este é um banco de dados NoSQL em nuvem, disponibilizado pela Amazon Web Service. Ele tem baixa latência, é rápido e flexível, sendo o modelo ideal para aplicações móveis, jogos na web e soluções com internet das coisas.

Ele ainda apresenta alto desempenho e escalabilidade automática, características imprescindíveis para negócios que precisam crescer com eficiência.

NEO4J O Neo4j é um banco de dados não-relacional que se baseia em grafos (arestas que se relacionam aos nodes). Ele é uma implementação de código aberto e pode ser útil para casos de mineração de dados e reconhecimento de padrões.

MONGODB Este também é um banco de dados de código aberto com alta performance. Ele é aceito em diferentes sistemas operacionais e tem como característica ser orientado a documentos.

Sendo assim, ele armazena todas as informações relevantes em um documento e utiliza sistemas avançados de agrupamento e filtragem. Diferentes plataformas e linguagens possuem suporte ao MongoDB, entre elas estão o Java, JavaScript, PHP, Python e Ruby.

Os principais exemplos de empresas que usam o MongoDB são: o site Globo.com, MailBox, MTV e Pearson Education.

Esses são os principais exemplos de bancos de dados NoSQL ou não-relacionais. O uso entre eles pode se diferenciar de acordo com as necessidades de cada negócio. Portanto, o mais indicado é buscar o auxílio de uma empresa especializada em soluções de armazenamento de bancos de dados.

7. Referências

Ramalho, José Antônio Alves. SQL – A linguagem dos Bancos de Dados (Série Ramalho). Ed. Berkley.

ELMASRI, Ramez; NAVATHE, Sham. Sistemas de Banco de Dados. 4ª Ed. Editora Pearson

CARAVALHO, Vinícius. MySQL Comece com o principal banco de dados open source do mercado. Editora Casa do Código.

DATE, C. J. Introdução a sistemas de bancos de dados. Rio de Janeiro: Campus, c2004. 865 p.

GARFINKEL, Simson; SPAFFORD, Gene. Comércio segurança na web: riscos, tecnologias e estratégias. São Paulo: Market Press, c1999. 378 p.

BOAGLIO, Fernando. MongoDB. Construa novas aplicações com novas tecnologias. Editora Casa do Código 2016.

ALMEIDA, Flávio. Mean Full Stack JS para aplicações web com MongoDB. Editora Casa do Código. 2017

RIBEIRO, Caio. Meteor. Criando aplicações web realtime com JS. Editora Casa do Código. 2017

Sucesso é ir de fracasso em fracasso sem perder o entusiasmo. Winston Churchill