Ola amigos (as),

Nessa reta final, nada como um resumão, não acham? Então vamos a ele.

RESUMO

Um Banco de dados é um conjunto de informações que guardam uma correlação de forma que criem um sentido. Os Bancos de Dados representam determinada realidade, o que chamamos de minimundo. Essa realidade pode ser uma empresa, um setor, um processo, enfim, deve ser algo delimitado. Assim, as principais características dos Bancos de Dados são:

- Um BD representa alguns aspectos do mundo real (minimundo), sendo que as mudanças que ocorrem no mundo real devem ser refletidas no Banco de Dados;
- Um BD é uma coleção lógica e coerente de dados, com algum significado. Um grande conjunto de dados reunido ao acaso, sem correlação, não é um Banco de Dados;
- Um BD é projetado, construído e povoado por dados, atendendo a uma proposta específica. Isso significa que os dados que estão lá obedecem a certas regras da minha aplicação.

O termo Banco de Dados não é sinônimo de software. Existe um software chamado Sistema Gerenciador a Banco de Dados (SGBD, ou DBMS em inglês) que serve para implementar Bancos de Dados Informatizados em um Sistema de Computação.

Na arquitetura geral dos Sistemas de Informação, um Banco de dados fica na camada de persistência, pois é lá que os dados são gravados e devem estar disponíveis para acesso futuro.

As principais características de um SGBD são:

- Natureza autodescritiva: significa que os SGBDs não armazenam apenas os dados, mas também as descrições desses dados. Essas informações sobre os dados são chamados de metadados;
- Isolamento entre Programas e dados e Abstração de dados: Significa que com os SGBDs o trabalho de desenvolvimento de aplicações fica focado nas regras de negócio, sem preocupações em como armazenar e recuperar os dados. Quem se preocupa com essa parte é o SGBD. A abstração é a característica dos SGBDs que permitem que criemos abstrações (tabelas, campos, índices) para representar objetos e eventos do mundo real (pessoas, notas fiscais, processos etc).
- Compartilhamento dos dados e processamento de transações multi-usuário: Com os SGBDs os dados podem ser acessados simultaneamente por diversos usuários, e quando lidamos com transações, que são conjuntos de instruções que devem ser executadas totalmente para ser válidas, os SGBDs têm mecanismos para tratar essas transações, facilitando o trabalho de desenvolvimento de software.

Entre os principais atores do Banco de Dados, temos analistas de sistemas, projetistas de software, e o principal ator no campo de Banco de Dados, que é o Administrador de Banco de Dados (DBA). Entre suas tarefas típicas, temos:

- Manutenção de gerenciamento do BD;
- Garantia de recuperabilidade (backups);
- Garantia da integridade dos dados;
- Concessão de permissões de acesso;
- Definição da plataforma necessária para suportar o SGBD;
- Garantia de disponibilidade (normalmente 24 x 7).

A modelagem de Banco de dados é o processo de levantar a realidade de determinado usuário, sistema ou aplicação, e criar diversos modelos para a representação dessa realidade em um Banco de Dados.

O modelo conceitual é uma representação de alto nível do minimundo, bem próximo da realidade encontrada. A principal ferramenta utilizada para a modelagem conceitual é o Modelo de Entidades e Relacionamentos. Esse modelo utiliza um diagrama chamado de Diagrama de Entidades e Relacionamentos

Os principais elementos do DER são:

- Entidade: Algo do mundo real com uma existência independente. Uma entidade pode ser um objeto com existência física (por exemplo, uma pessoa, um carro, um DVD, um funcionário) ou um objeto com existência conceitual (um projeto, um curso universitário, uma disciplina).
- Atributo: Cada entidade tem atributos, que são propriedades particulares que a descrevem. Por exemplo, uma entidade Carro pode ter como atributos: marca, modelo, cor, fabricante, ano de fabricação, chassi e assim sucessivamente. Uma entidade Empregado pode ter como atributos nome, setor, data de nascimento, RG, salário etc. Então, os atributos são as características de uma entidade. São dados que dizem respeito à determinada entidade.
- Relacionamento: São relações entre duas ou mais entidades, determinando uma associação entre as mesmas. Representam interações entre duas ou mais entidades. Por exemplo, imaginem que existe no minimundo de uma Universidade a entidade Aluno e a entidade Disciplina. Essas duas entidades têm um relacionamento, uma vez que os alunos cursam disciplinas.

Todo relacionamento tem um grau, que é a quantidade de entidades que fazem parte do relacionamento. O mais comum é termos relacionamentos de grau dois (binários)

Outro atributo dos relacionamentos é a sua cardinalidade, que expressa, para cada entidade de um grupo, o número máximo de entidades do outro grupo que se relacionam com ela. Atenção: é muito comum as bancas considerarem que a cardinalidade define tanto o mínimo quanto o máximo de elementos que se relacionam. Assim, se a ESAF colocar essa informação, devemos aceitar, a não ser que a questão seja mais específica e cite a restrição de participação.

As cardinalidade mais comuns são 1:1, 1:N e N:M.

A restrição de participação determina de a existência de uma entidade depende ou não do fato de ela participar de um relacionamento. Se a participação for total, todas as entidades devem estar ligados a pelo menos um elemento da outra entidade que faz parte do relacionamento. Se for parcial, uma entidade pode existir mesmo não estando ligado a nenhum elemento da outra entidade em um relacionamento. Restrição de participação também é conhecida como restrição de cardinalidade mínima.

Um atributo descreve as características ou propriedades de um entidade ou relacionamento. O domínio de um atributo é o conjunto de possíveis valores para o mesmo. Existe um valor especial, chamado de NULL, que significa ausência de valor no atributo.

Os atributos podem ser:

- Monovalorados ou multivalorados, conforme armazenem um ou mais valores, respectivamente;
- Simples ou compostos, conforme sejam atômicos ou possam ser divididos (exemplo típico de composto é endereço);
- Armazenados ou derivados, conforme sejam guardados diretamente no BD, ou sejam calculados a partir de outros atributos;

- Atributo chave: identifica de forma única uma entidade no conjunto.

Um relacionamento não pode se ligar diretamente a outro relacionamento. Para pular essa limitação, foram criadas entidades associativas, que são uma forma de transformar um relacionamento em uma entidade, para ligar dessa forma a outro relacionamento.

Existe um modelo chamado MER estendido, que foi uma forma de expandir um pouco o MER para incorporar ideias do paradigma de Orientação a Objetos, que é um paradigma de desenvolvimento de Sistemas.

O principal elemento incorporado é a herança, que é uma forma de representar um objeto mais genérico (superclasse), e especializar esse elemento em elementos mais específicos (subclasses). As subclasses herdam todas as características da superclasse, e incorporam as suas próprias. Existem duas restrições nesse modelo. Restrição de integralidade, ou vertical, e restrição de disjunção, ou horizontal. Ver FAK para exemplos mais elaborados.

Uma entidade fraca é aquela que depende da existência de outra entidade para existir. O exemplo mais típico são os dependentes, que só podem existir se estiverem ligadas a sua entidade forte (Empregado).

O Modelo Relacional representa os dados em um Banco de Dados como um conjunto de tabelas, que no modelo são denominadas Relações. Cada Relação terá um nome, que será único, e um conjunto de atributos, com seus respectivos nomes e domínios.

O grau da Relação é simplesmente o número de atributos que a mesma possui. Já O instante (snapshot) da Relação é a fotografia de um determinado momento da Relação.

As relações têm as seguintes características:

- Ordenação: a ordem das tuplas e dos atributos não importam para a Relação.

- Atomicidade: todos os atributos devem ter valores atômicos, ou seja, únicos.
- Identidade: cada atributo de uma Relação tem um nome que é único para aquela Relação.
- Unicidade: todas as tuplas devem ser únicas.

Chave: conjunto de um ou mais atributos que determinam a unicidade de cada tupla (registro). Vamos ver que chave é um gênero que comporta muitas espécies diferentes.

- Superchave é um conjunto de um ou mais atributos que, tomados coletivamente, nos permitem identificar de maneira unívoca uma entidade em um conjunto de entidades.
- Chaves candidatas. São superchaves de tamanho mínimo, candidatas a serem chaves primárias da Relação. Ou seja, são um atributo ou conjunto de atributos que permitem identificar de forma inequívoca qualquer tupla dessa relação, mas que não pode ser reduzido sem perder qualidade. Toda superchave que tem apenas um atributo é uma chave candidata
- Entre as chaves candidatas possíveis, devemos escolher para cada relação uma, e somente uma, chave primária. Chave primária nada mais é um conjunto de um ou mais campos, cujos valores, considerando a combinação de valores de todos os campos da tupla, nunca se repetem e que podem ser usadas como um índice para os demais atributos da relação, escolhida entre as chaves candidatas. Em chaves primárias, não pode haver valores nulos nem repetição de tuplas.
- Chave estrangeira é um conjunto de um ou mais atributos de uma relação que fazem referência à chave primária de outra relação, ou até mesmo à própria (nos auto-relacionamentos)

As restrições de integridade são regras aplicadas pelo Banco de Dados para garantir que o Banco permaneça íntegro, exato e consistente.

- Integridade de domínio (ou restrição de domínio): visa garantir que os valores que cada atributo receba esteja dentro do seu domínio. Por exemplo, um campo sexo deve receber M ou F, um campo data deve receber uma data válida, e assim sucessivamente.
- Integridade de chave (ou restrição de chave): Impede que uma chave primária se repita.
- Integridade de Entidade (ou restrição de entidade): Impede que uma chave primária receba em qualquer de seus campos o valor NULL.
- Integridade referencial: garante que chaves estrangeiras estejam relacionadas a chaves existentes nas tabelas de origem.

A álgebra relacional é uma forma de cálculo sobre conjuntos ou relações.

- A seleção (particionamento horizontal) é utilizada para selecionar um conjunto de tuplas (linhas) de uma relação. Assim, ela funciona como um filtro para selecionar tuplas que atendem a determinado critério.
- A operação de projeção (particionamento vertical) seleciona certas colunas da tabela (relação) e descarta outras.
- Produto cartesiano é uma operação binária (atua em duas relações) que une cada elemento de uma relação a um elemento da outra relação.

DDL (Data Definition Language): Quando você cria um banco de dados, ele inicialmente está "vazio"; antes de começar a consultar e alterar dados é preciso definir onde e como as informações serão gravadas dentro do novo banco; então você cria diversas tabelas explicitando o tipo de dados de cada campo, as chaves estrangeiras, os índices, as regras e etc. Estes comandos de criação e alteração de estrutura são os comandos de DEFINIÇÃO dos dados, pois definem como

os dados serão armazenados; em inglês são chamados de: Data Definition Language (DDL).

DML (Data Manipulation Language): Depois que você criou suas tabelas, definiu relacionamentos, índices e etc., e hora de manipular seus dados. Quer dizer que você ira fazer operações típicas nas tabelas, como inserção de informações, alterações, exclusões e consultas. Para isso temos a linguagem de manipulação dos dados, chamada de DML.

DCL (Data Control Language): Com o banco de dados pronto e rodando é importante definir quem poderá acessá-lo, enfim, precisamos definir a segurança do seu banco. Em inglês, os comandos responsáveis pelo controle dos dados são chamados de Data Control Language (DCL).

O projeto físico de um Banco de Dados é a definição em DDL de seus objetos (tabelas, índices, views), pronto para ser gerado em um SGBD específico.

Structured Query Language (SQL), ou Linguagem de Consulta Estruturada, é uma linguagem de pesquisa declarativa para Banco de Dados relacionais. Ela tem diversos propósitos, sendo os dois principais servir como Linguagem de Definição de Dados (DDL), ou seja, uma linguagem que serve para informar ao SGBD qual a estrutura do meu Banco de Dados, dando uma descrição completa dos meus metadados, e também serve como Linguagem de Manipulação de Dados (DML), ou seja, uma linguagem que diz ao SGBD para gravar, alterar, excluir ou atualizar os dados propriamente ditos no meu Banco de Dados.

O comando CREATE é usado para a criação de objetos, especialmente a criação de tabelas no Banco de Dados. Como exemplo de seu uso temos:

```
CREATE TABLE AtorParticipa (  → definição do objeto criado
idDVD INTEGER UNSIGNED NOT NULL,  → definição de um campo
idAtor INTEGER UNSIGNED NOT NULL,
PRIMARY KEY(idDVD, idAtor),  → definição da chave primária
FOREIGN KEY(idDVD)    →  definição de chave estrangeira
```

REFERENCES DVD(idDVD) → campo referenciado na tabela origem
ON DELETE RESTRICT → o que acontece se eu excluir a chave prim
ON UPDATE CASCADE, → o que acontece se eu atualizar a cha pri.
FOREIGN KEY(idAtor)
REFERENCES Ator(idAtor)
ON DELETE RESTRICT
ON UPDATE CASCADE
);

Obs: Duvido demais que a ESAF cobre essa sintaxe. Se cobrar sintaxe, deve ser da DML.

O comando ALTER serve para alterar determinado objeto, por exemplo, alterar a estrutura de uma tabela acrescentando um campo.

O comando DROP apaga um objeto da Banco de dados.

A Linguagem de manipulação de dados (DML) é a linguagem usada para que possamos fazer as operações básicas com os dados, que é o que realmente nos interessa nos Banco de Dados. As principais operações feitas nos Bancos de Dados são: inclusão de dados, exclusão de dados, alteração de dados e consulta a dados. Tudo isso fazemos com os comandos da DML.

O INSERT é o comando SQL que serve para inserirmos dados em determinada tabela do Banco de Dados. Ex:

INSERT INTO Amigo VALUES ('12345678911','ZE','RUA Z NRO 15','2198765432',NULL,NULL);

O DELETE serve para apagar registro de uma tabela. Ex:

DELETE FROM Amigo WHERE cpf = '12345678911';

O UPDATE é usado para atualizar um ou mais campos de uma ou mais linhas de determinada tabela. Ex:

UPDATE Amigo SET tel2 = '6198789876', tel3 = '9298709862' WHERE cpf = '12345678911';

O SELECT serve para recupera as informações do Banco de Dados. Ele tem diversos uso e implementa tanto as operações de projeção, seleção como produto cartesiano.

Exemplo de uso do SELECT com projeção e seleção:

SELECT tituloPort, genero, ano FROM DVD WHERE ano > 2000;

Podem ser usados operadores lógicos no predicado (condição que vem depois do Where), como AND, OR, NOT, IN BETWEEN, LIKE etc.

Exemplos:

SELECT idAtor FROM AtorParticipa WHERE idDVD IN (1,4);

SELECT DISTINCT(idAtor) FROM AtorParticipa WHERE idDVD in (1,4);

SELECT * FROM DVD WHERE ano BETWEEN 1990 AND 2001;

SELECT * FROM DVD WHERE tituloPort LIKE '%Vingança%';

SELECT * FROM DVD WHERE (tituloPort BETWEEN 'A' AND 'T') AND (genero LIKE 'F%');

O ORDER BY é usado depois do predicado, e serve para ordenar nossa resposta segundo algum critério. Pode ser indicado um ou mais campos para ordenação. Pode ser informado o nome ou o número do campo. Para cada campo, a ordenação pode ser ascendente (ASC) ou descendente (DESC). Ex:

SELECT * FROM DVD ORDER BY genero DESC, tituloPort ASC;

O Count serve para contar o número de tuplas retornadas por uma query. Ex:

SELECT COUNT(*) FROM DVD WHERE ano = 2005;

O AVG() tira a média aritmética de um conjunto de valores. Ex:

SELECT AVG(population) MEDIA FROM WORLD.COUNTRY WHERE continent = 'Asia';

O MAX() retorna o maior dos valores do conjunto, e o MIN() o menor dos valores. Ex:

SELECT MAX(population) MAIOR, MIN(population) MENOR FROM WORLD.COUNTRY WHERE continent = 'Europe';

O NOW() retorna o dia e hora atuais.

A cláusula GROUP BY funciona como um agrupador, baseado em um ou mais campos. Ex:

SELECT continent CONTINENTE, SUM(population) SOMA FROM WORLD.COUNTRY GROUP BY continent;

Se a função de agregação precisar ser testada, devemos usar o HAVING. Ex:

SELECT continent CONTINENTE, SUM(population) SOMA FROM WORLD.COUNTRY GROUP BY continent HAVING SUM(population) > 0;

Uma subquery é um SELECT dentro de outro. O resultado do SELECT mais interno é usado pelo SELECT mais externo. Ex:

SELECT DISTINCT(idAtor) FROM AtorParticipa WHERE idDVD IN

(SELECT idDVD FROM DVD WHERE tituloPort = 'Matrix' OR tituloPort = 'V de Vingança');

O produto cartesiano entre duas tabelas, A e B, junta cada linha de A com cada linha de B. è caracterizado por ter mais de uma tabela depois do FROM, separadas por vírgula.Ex:

```
SELECT ATOR.nomeAtor FROM ATORPARTICIPA, ATOR
WHERE ATORPARTICIPA.idAtor = ATOR.idAtor AND
ATORPARTICIPA.idDVD =
(SELECT idDVD FROM DVD
WHERE tituloPort LIKE 'O Senhor dos An%');
```

Uma visão é simplesmente uma query SQL, que por algum motivo, é armazenada para ser reusada. Ex:

```
CREATE VIEW listaAtoreseFilmes AS
SELECT DVD.tituloPort TituloPortugues, ator.nomeAtor NomeDoAtor
FROM DVD, atorParticipa atp, ator
WHERE DVD.idDVD = atp.idDVD and ator.idAtor = atp.idAtor;
```

Depois de criada, uma visão pode ser usada como uma tabela em um SELECT, ou seja, pode ser usada para seleção simples, em produtos

cartesianos, em Joins, para criar outra visão etc. Mas as visões não geram dados repetidos no BD, são apenas um elemento lógico.

- Joins: Forma de ligar duas ou mais tabelas por campos em comum. Temos os seguintes tipos de joins:
- Inner Join: Retorna linhas quando existe pelo menos uma combinação em ambas as tabelas;
- Left Join: Retorna todas as linhas da tabela da esquerda, mesmo que as linhas não combinem com a tabela da direita.
- Right Join: Retorna todas as linhas da tabela da direita, mesmo que as linhas não combinem com a tabela da esquerda.
- Full join: retorna linhas quando tem uma combinação em uma das tabelas.

Exemplo de Joins:

```
SELECT city.id, city.name Cidade, country.name Pais,
city.population Populacao FROM city
INNER JOIN country
ON city.countrycode = country.code;
```

O UNION é usado para a operação de união da álgebra relacional. Ele une o resultado de dois ou mais SELECT, eliminando as tuplas repetidas. O UNION ALL mantém as tuplas repetidas. As queries devem retornar os mesmos campos, na mesma ordem. Ex:

```
select city.id, city.name Cidade, country.Name Pais,
CITY.COUNTRYCODE, city.population Populacao
from city
left join country
on city.countrycode = country.code
UNION
select city.id, city.name Cidade, country.Name Pais,
CITY.COUNTRYCODE, city.population Populacao
from city
```

```
right join country
on city.countrycode = country.code
ORDER BY 3;
```

O Data Control Language é a parte do SQL que trata do controle de acesso aos objetos do Banco de Dados. Tem basicamente dois comandos, o GRANT e o REVOKE.

O GRANT serve para dar permissão de uma operação DML a um usuário em determinado objeto. Ex:

```
GRANT DELETE, INSERT, SELECT ON DBEMPRESTIMO.AMIGO TO FULANO;
```

O REVOKE cancela (revoga) uma permissão:

```
REVOKE DELETE ON DBEMPRESTIMO.AMIGO FROM FULANO;
```

O TOP N retorna as N primeiras linhas da query. Ex:

```
SELECT TOP 3 * FROM COUNTRY ORDER BY POPULATION DESC;
```

O EXISTS testa se o resultado de uma subquery retornou pelo menos uma linha. Ex:

```
SELECT * FROM AMIGO WHERE EXISTS
(SELECT * FROM EMPRESTIMO
WHERE AMIGO.CPF = EMPRESTIMO.CPF);
```

O ANY teste se um campo está contido em uma lista (semelhante de certa forma ao IN). Ex:

```
SELECT * FROM AMIGO WHERE cpf <> ANY
(SELECT CPF FROM EMPRESTIMO);
```

Dependência funcional, em apertada síntese, é um relacionamento que existe entre atributos de uma Relação. Se um atributo pode ser determinado por outro, então depende funcionalmente deste outro.

A normalização de dados é uma série de passos que se segue no projeto de um banco de dados que permite um armazenamento consistente e um eficiente acesso aos dados em um banco de dados relacional. Esses passos reduzem a redundância de dados e as chances dos dados se tornarem inconsistentes.

Um projeto de Banco de Dados está na primeira formal normal,ou 1FN, se todos os atributos são atômicos, ou seja, não existem atributos monovalorados ou compostos.

Um projeto de Banco de Dados está na segunda formal normal, ou 2FN, se está na 1FN e os atributos que fazem parte de sua chave primária definem, em conjunto, todos os demais atributos. Ou seja, eu não posso ter um atributo na chave primária que, sozinho, define um outro atributo não chave.

Um projeto de Banco de Dados está na terceira formal normal, ou 3FN, se está na 2FN e não existe dependência transitiva. Ou seja, um campo não chave não pode ser determinado por outro(s) campo(s) não chave.

A Forma Normal de Boyce-Codd, ou BCNF é uma forma de 3FN um pouco mais refinada.

Dados são fatos que podem ser analisados e que possuem um significado implícito.

Já informação é o resultado do processamento, manipulação e organização de dados, de tal forma que represente uma modificação (quantitativa ou qualitativa) no conhecimento do sistema (pessoa, animal ou máquina) que a recebe.

A Mineração de Dados surgiu com a motivação de "garimpar" informações relevantes das Bases de Dados, de forma automática.

A Mineração de Dados é o processo de descoberta automática de informações úteis em grandes depósitos de dados. As técnicas de Mineração de Dados são usadas para agir sobre grandes Bancos de Dados com o intuito de descobrir padrões úteis e recentes que poderiam, de outra forma, permanecer ignorados. Elas também oferecem capacidade de previsão do resultado de uma observação futura.

O processo de KDD, no qual o Data Mining está inserido, é composto geralmente por 6 fases, que são: seleção de dados, limpeza,

enriquecimento, transformação/codificação, mineração, apresentação e interpretação dos resultados.

O resultado da Mineração de Dados pode descobrir os seguintes tipos de informação "nova":

- Regras de associação: Por exemplo, se um cliente compra uma máquina fotográfica, pode querer comprar também um cartão de memória.
- Padrões seqüenciais: Por exemplo, determinado cliente pega um empréstimo para comprar um carro. Depois da quarta parcela, começa a atrasar o pagamento. Depois de um ano, deixa de pagar. Isso pode se repetir de forma mais ou menos igual para diversos clientes, e pode definir um padrão. Assim, quando o cliente começa a atrasar muito, a empresa já pode se preparar para ele deixar de pagar a dívida.
- Regras de classificação: Por exemplo, clientes podem ser classificados por freqüência de visitas, por tipo de financiamento utilizado, por quantidade comprada, por afinidades com alguns itens e assim sucessivamente. Algumas estatísticas reveladoras podem ser geradas para cada classe de clientes.

Tarefas de Data Mining:

- Predição: Procura mostrar como certos atributos dos dados vão se comportar no futuro. Por exemplo, podemos analisar transações de compra para prever o que os clientes tendem a comprar se dermos descontos, o volume de vendas que pode ser alcançado em dado período, e se a exclusão de uma dada linha de produtos pode aumentar os lucros.
- Identificação: Padrões de dados podem ser usados para identificar a existência de um item, um evento ou uma atividade. Por exemplo, sistemas de segurança podem identificar a predisposição de alguém para invadir uma casa,

ou causar outro tipo de confusão, pela sequência de atitudes tomadas por uma pessoa.

- Otimização: Um objetivo de Mineração de Dados pode ser otimizar o uso de recursos limitados (ex tempo, dinheiro, materiais etc), maximizando as variáveis de saída (por exemplo vendas ou lucro) sob determinados conjuntos de restrições. Ou seja, um algoritmo de data mining pode analisar que dado que um empresário tem X reais para investir, tem Y equipamento, funcionários etc, e dado o histórico de vendas, qual seria o melhor caminho (escolha) que deveria ser tomado.
- Classificação: é o processo de identificar, entre classes previamente definidas, a qual classe cada elemento pertence. Imaginem que vocês possuem uma série de documentos que baixaram da web com artigos sobre diversas áreas (ex direito constitucional, direito administrativo, Banco de Dados, auditoria etc etc). Classificação em Data Mining é o processo de determinar de forma automática em qual classe cada elemento se encaixa.
- Associação: Determina com certo grau de certeza que determinado item costuma aparecer na presença de outro. É aquele exemplo da loja, que quando alguém compra uma máquina fotográfica, também compra um cartão de memória. Ou quando alguém compra pão, tende a comprar margarina ou manteiga. Essas regras de associação podem ser utilizadas para gerar uma aplicação bastante difundida na Mineração de Dados, que é a recomendação. Ela analisa seu padrão passado de consumo, o padrão de consumidores "semelhantes" a você, e com base nisso recomenda produtos que podem ser do seu interesse.

- Padrões seqüenciais: Semelhante à associação, mas analisa eventos em determinado período de tempo. Por exemplo, se um paciente fez ponte de safena para artérias bloqueadas e um aneurisma (sai para lá!), e depois desenvolveu ureia alta no sangue no período de um ano, ele está propenso a sofrer problemas renais nos próximos 18 meses.
- Agrupamento (clustering): É semelhante a classificação. Só que cada agrupamento não é predefinido. Assim, posso usar um algoritmo de agrupamento para dividir minha população em agrupamentos, de forma que cada elemento de um agrupamento é mais semelhante com os demais elementos do mesmo agrupamento, do que com elementos de outros agrupamentos. Por exemplo, imagine que um banco pega toda a sua base clientes e quer dividi-lo em agrupamentos. Depois do processo, são encontrado 4 agrupamentos. Em um estão clientes que fazem muitos empréstimos para financiar seus negócios, em outro temos clientes que fazem poupanças para garantir o futuro, em outro temos clientes que fazem aplicações arriscadas, e no último temos clientes que apenas usam a conta para receber salário, sem comprar produtos do Banco. Nos algoritmos de agrupamento, no máximo definimos o número de grupos. Para alguns desses algoritmos, nem isso é definido.

Sobre Dados e conjunto de dados, recomendo ler o capítulo.

O processo de classificação tem 3 etapas, o treino, onde o classificador é ensinado, o teste, onde se verifica se ele aprendeu de forma correta a classificar, e a aplicação no problema.

Sobre os classificadores, recomendo ler o capítulo.