**Dependencias funcionais:**

**Super-Chave** – É um conjunto de um ou mais atributos que, tomados coletivamente, nos permitem identificar de maneira unívoca uma entidade em um conjunto de entidades. Em outras palavras, não podem existir duas ou mais linhas da tabela com o(s) mesmo(s) valores de uma Super-Chave.

X → todos os atributos de R.

**Chave Candidata** – São super-chaves de tamanho mínimo, candidatas a serem chaves primárias da relação. Ou seja, atributo ou conjunto de atributos que permitem identificar de forma inequívoca qualquer tupla dessa relação. Este conjunto não pode ser reduzido sem perder qualidade.

Y é subconjunto próprio de X, se Y ⊆ X ∧ Y X.

**DF’s triviais:** X → Y é trivial se Y ⊆ X

**Transitividade:** Se X → Y e Y → Z, então X → Z. **“Augmentation”:** Se X → Y, então XZ → Y. **Pseudotransitividade:** Se X → Y e YW → Z, então XW → Z. **Decomposição:** Se X → Y, então X → A, para todo A pertencente a Y. **União:** Se X → Y e X → Z, então X → YZ. **Composição:** Se X → Y e Z → W, então XZ → YW. **Unificação:** Se X → Y e Z → W, então X(Z-Y) → YW.

**Fecho de um conjunto de atributos:** o fecho de X designa-se por X+ (X+ é o conjunto de todos os atributos que são funcionalmente determinados por X).

**Algoritmo para calcular X+:** X+ = X; se A1, A2, ... , Am → B, e todos os A’s estão contidos em X+, adicionar B a X+; termina quando não conseguirmos adicionar mais nada a X+.

**Normalização:**

**1ª forma normal** (os domínios de todos os atributos consistem apenas em valores atómicos e não existem subgrupos de atributos repetidos); **2ª forma normal** (estiver na 1FN, todos os atributos que não pertencem à chave dependem de toda a chave e não de um subconjunto da chave); **3ª forma normal** (estiver na 2FN e os atributos que não pertencem à chave não dependem de nenhum atributo que também não pertence à chave);

**Forma Normal de Boyce Codd:** uma relação está na FNBC quando todo o determinante da relação for uma chave candidata (é necessário quando: uma entidade tem várias chaves candidatas, as chaves candidatas são compostas e sobrepõe-se porque possuem pelo menos um atributo em comum);

**Álgebra Relacional:**

**Operações sobre conjuntos:** **∪, ∩, −**; remover linhas (**seleção**), remover colunas (**projeção**). operações que combinam informação contida em várias relações: **produtos cartesianos** e **joins**. mudar o nome a relações e atributos.

**Seleção(σ):** σc(R), onde c e uma condição.

**Projeção(π):** πL(R), onde L é uma lista de atributos de R.

**Produto Cartesiano(x):** R = R1 × R2; Faz a concatenação de cada tuplo t1 de R1 com cada tuplo t2 de R2 e coloca o tuplo t1t2 em R.

**Theta Join:** R = R1 ⋈c R2 e equivalente a: R = σc(R1 × R2)

**Natural Join:** R = R1 ⋈ R2 e equivalente a um theta join em que a condição c diz que os atributos com o mesmo nome são igualados. Depois, uma das colunas é projetada.

**Mudança de nome a relações e atributos:** ρ S(A1,A2,...,An)(R) produz uma relação de nome S idêntica a R e com os atributos chamados A1, A2, ... , An.

**Operações sobre sacos (bags):** Um conjunto não tem elementos repetidos. Um saco pode ter.Em ambos os casos a ordem dos elementos não interessa; t ocorre n + m vezes em **R ∪ S**; t ocorre min(n, m) vezes em **R ∩ S**; t ocorre max(0, n − m) vezes em **R – S**; Seleção, produto cartesiano e joins idêntico aos conjuntos; **Eliminação de repetidos (δ)** - Converte um saco de tuplos num conjunto de tuplos; Operadores de agregação: **SUM, AVG, MIN** e **MAX** (pode ser aplicado a valores numéricos ou a strings de caracteres; MIN=1o, MAX=ultimo, na ordem lexicográfica), **COUNT** (numero de valores de uma coluna, incluindo repetidos); **Agrupamento de tuplos (γ):** Cria grupos de acordo com o valor de um ou mais atributos; Pode-se aplicar agregações (SUM, AVG, MAX, ...) a cada grupo. Forma geral: γL(R); L é uma lista de elementos, em que cada qual pode ser: I um atributo de R (sobre o qual R será agrupado); I uma agregação aplicada a um atributo de R. **Projeção extendida (πL(R)):** Cada elemento de L pode ser - Um atributo de R; Uma expressão x → y (em que x e y são nomes de atributos, equivalente a um “rename” de x para y); Uma expressão E → z (em que E pode envolver atributos de R, constantes, e operadores aritméticos e de strings); **Ordenação (τL(R)):** Converte R para uma lista ordenada segundo os atributos especificados na lista L. Exemplo: τB,C (R) ordena os tuplos de R por B, e em caso de empate por C. **Outerjoin (⟗** **):** Adiciona ao output tuplos que falham a condição de join. Esses tuplos ficam com o símbolo null (⊥) nos atributos que não possuem. Existem variantes: **Outer Join, Left Outer Join (⟕ ), Right Outer Join (⟖ )**.

**PERGUNTAS DE TESTES:**

**Tomando como base um diagrama E-R à sua escolha, explique o que é um relacionamento recursivo:** Relacionamento recursivo é quando uma entidade se relaciona consigo própria. Podem ser (1:1), (1:N) ou (N:M) dependendo do modelo de negócio. Um exemplo é a gerência de funcionários, onde o gerente é um funcionário que possui um relacionamento com outros funcionários que lhe são subordinados.

**Quais são as principais responsabilidades de um administrador de uma base de dados:** Criação e testes de backup para garantir a recuperabilidade dos dados no caso de falha de hardware ou outros problemas severos; Instalar e atualizar as ferramentas do banco de dados; Alocar o espaço do sistema reservado ao banco e garantir um alocamento futuro no sistema; Saber modificar a estrutura do banco de dados; Saber os comandos básicos e exclusivos de cada SGBD; Verificar e zelar pela integridade do banco de dados; Ter um controle de acesso, ou privilégios, aos dados como quem pode acessar e o que pode acessar e talvez quando possa acessar; Garantir o acesso ao banco de dados no maior tempo possível; Garantir o máximo de desempenho para as consultas ao banco de dados; Auxiliar a equipe de desenvolvimento e a equipe de testes a maximizar o uso e desempenho do banco de dados; Contatar suporte técnico em caso de certos problemas com o banco de dados.