Modelação Conceptual: O processo de construir um modelo de dados de uma empresa, independente das considerações físicas.
Objetivo: Construir um modelo de dados conceptual dos requisitos de dados da empresa.

Metodologia:
- Identificação e caracterização de entidades: Obter as entidades a partir da análise de requisitos.

- Identificação e caracterização de relacionamentos: Obter os relacionamentos entre as entidades a partir da análise de requisitos.

(Depois destes dois passos já é possível observar o diagrama ER correspondente)

- Identificação e caracterização dos atributos das entidades e relacionamentos: Obter os atributos (informação necessária guardar para cada entidade e relacionamento) a partir da análise de requisitos.

- Determinar dominio dos atributos: Determinar os valores que os atributos podem tomar, assim como a natureza dos mesmos.

- Determinar chaves candidatas, primárias e alternativas: Primeiro identificar as chaves candidatas, e depois a partir dessas escolher qual a mais apropriada para ser chave primária (chave que identifica inequivocamente uma ocorrência). As chaves candidatas não escolhidas são consideradas como chaves alternativas e poderão tornar-se indices no futuro, que aumentarão a eficiência das pesquisas.

- Considerar o uso de modelação avançada: Consideração do uso de especialização, generalização, agregação ou composição. Normalmente não se usa nada porque é mais importante manter o modelo simples para que o utilizador o perceba.

- Verificação de redundância: Um relacionamento é redundante quando é possivel obter a mesma informação a partir de outros relacionamentos. É relativamente fácil verificar a existência de múltiplos caminhos entre duas entidades do modelo, no entanto isto não implica que existam relacionamentos redundantes entre elas, pois tais relacionamentos podem representar associações distintas. É também importante verificar todos os relacionamentos de 1 para 1 para ver se não identificam a mesma entidade, e considerar a dimensão do tempo (exemplo do homem, mulher e criança).

- Validação do modelo com perguntas do utilizador: É necessário verificar que o modelo suporta as transacções pedidas pelo utilizador, sendo que isto pode ser feito descrevendo a transacção do utilizador através do modelo ou atráves de mapas de transacções. Se houverem partes do modelo que não aparecem em nenhum mapa de transacções, pode ser que esta informação não seja necessária guardar no modelo.

- Revisão do modelo conceptual com o utilizador: Rever o modelo para assegurar que se trata de uma verdadeira representação dos requisitos da empresa.

Modelação Lógica: O processo de construir um modelo de dados usado por uma empresa baseado num modelo de dados conceptual, mas independente de um SGDB em particular e de outras considerações fisicas.

Metodologia:

- Derivação das relações para o modelo lógico:
    - Entidades fortes -> Criar uma relação com todos os atributos simples da entidade.
    - Entidades fracas -> Criar uma relação com todos os atributos simples, mas a chave primária é derivada da entidade "dono" e por isso esta só vai poder ser definida depois mapear o relacionamento com a entidade pai.
    - Relacionamentos 1:* -> A entidade filho (lado do *) ganha uma nova chave estrangeira que é o identificador da entidade pai. (A entidade filho é sempre a que vai ter a chave estrangeira).
    - Relacionamentos 1:1 binários
        - Participação obrigatória de ambos os lados: Juntar as duas relações numa só e escolher a chave primária de uma delas como nova chave primária.
        - Participação obrigatória de um dos lados: Identificar a entidade filho como sendo aquela que tem participação obrigatória e colocar nela uma chave estrangeira para a entidade pai (por exemplo, Cliente especifica Preferência, a entidade Preferência é a que vai ter participação obrigatória, ou seja, nem todos os clientes vão ter preferência, e então preferência tem de ter uma chave estrangeira para Cliente).
        - Participação opcional de ambos os lados: A entidade que aparece menos vezes vai ser designada a entidade filho e vai ter uma chave estrangeira para a entidade pai.
    - Relacionamentos 1:1 recursivos -> Mesma coisa que relacionamentos 1:1 binários, mas a chave estrangeira vai tar dentro da mesma relação (porque pai e filho vão ser a mesma entidade).
    - Relacionamentos entre subclasse/superclasse -> A subclasse é definida como sendo a entidade filho e vai ter uma chave estrangeira para a entidade pai.
    - Relacionamentos N:M binários -> Criar uma nova relação que tem todos os atributos do relacionamento e cuja chave primária é uma chave composta de chaves estrangeiras para cada entidade que participa no relacionamento.
    - Relacionamentos complexos -> (como vimos nas relações ternárias) Criar uma nova relação com os atributos do relacionamento e com uma chave primária que é composta por chaves estrangeiras para todos os atributos do relacionamento.
    - Atributos multivalor -> Criar uma relação com uma chave composta com o identificador da entidade pai e outro atibuto/s que identifiquem unicamente cada ocorrência.

- Validação das relações através de normalização: Pelo menos até à 3FN.

- Validação das relações através das transacções dos utilizadores: Mais uma vez validar o modelo com transacções do utilizador, mas agora a utilizar o modelo lógico.

- Análise das restrições de integridade (As restrições de integridade são as restrições que queremos impor de forma a proteger a base de dados de ficar imcompleta e inconsistente. Assim, as restrições de integridade não apenas ajudam na consistência da base de dados, mas também a representar verdadeiramente os requisitos de dados da empresa):

- Necessidade de valores -> Alguns atributos devem conter sempre algum valor (não podem ser null). Isto devia já ser explicito no dicionario de dados

- Restrições do domínio dos atributos -> Cada atributo tem um dominuo e um conjunto de valores válidos. (p.e. o sexo de uma pessoa é M ou F, e portanto deve ser um único caracter constituido por uma dessas letras)

- Multiplicidade -> Diz respeito ao relacionamento entre entidades. Na conversão para o modelo lógico, surgiram novos atributos e relações que não existiam no modelo lógico, e estas devem ser verificadas (p.e. uma relacionamento N:M dá origem a dois relacionamentos 1:N).

- Integridade da entidade -> A chave primária de cada entidade não pode ser null.

- Integridade referencial -> Se uma chave estrangeira contém um valor, esse valor deve referir um tuplo existente na relação pai.

- Restrições gerais -> Os updates na entidade podem estar controlados por restrições associadas ao mundo real.

- Revisão do modelo lógico com o utilizador: Rever novamente o modelo lógico com o utilizador para ter a certeza que o modelo continuar a representar adequadamente a empresa.

- Juntar modelos lógicos de dados num modelo global (opcional):

O objetivo é juntar todos os modelos de dados lógicos num único modelo global que representa todas as vistas de utilizadores da base de dados.

- Rever os nomes e conteudos das relações/entidades e as suas chaves candidatas
- Rever os nomes e conteudos dos relacionamentos/chaves estrangeiras
- Juntar relações dos vários modelos (as que são iguais)
- Incluir (sem juntar) as relações únicas a cada modelo
- Juntar os relacionamentos/chaves estrangeiras dos vários modelos (os que são iguais)
- Incluir (sem juntar) os relacionamentos/chaves estrangeiras únicos a cada modelo
- Confirmar se faltam alguma relação ou relacionamento
- Confirmar chaves estrangeiras
- Confirmar restrições de integridade

- Desenhar o novo modelo global
- Atualizar a documentação

- Análise de crescimento futuro: Ver futuras mudanças que poderão ocorrer no modelo e se esse modelo poderá acolher essas mudanças com facilidade.

Modelação Fisica: O processo de produzir uma descrição da implementação da base de dados em memória secundária. Descreve as relações base, organização de ficheiros e indices usados para alcançar acesso eficiente aos dados, assim como quaisquer restriçoes associadas e medidas de segurança.

Metodologia:

- Tradução do modelo lógico para o SGBD pretendido
    - Desenho das relações base -> Criar as tabelas referentes ao modelo lógico, com os respetivos dominios, valores por defeito e restriçoes do sistema.
    - Desenho da representação de dados derivados -> Decidir como representar os dados derivados, e como os atualizar (através de transacções ou triggers).
    - Desenho das restrições gerais -> Criar as restriçoes gerais referentes às regras de negócio do modelo (ADD CONSTRAINT nome_da_restricao CHECK restriçao).

- Desenho da organização de ficheiros e indices (Determinar a organização de ficheiros optima para armazenar as realções base e os indices requeridos para atingir performance aceitável, isto é, a forma como as relações e tuplos serão guardados em disco)
    - Análise de transacções
        - Mapear todas os caminhos de transacções para relações
        - Determinar quais as relações que são acedidas mais frequentemente por transacções
        - Analisar o uso de dados das transacções que envolvem estas relações
    - Escolha de organização de ficheiros ->
    - Escolha de indices ->
    - Estimar espaço em disco necessário ->

- Desenho das vistas de utilizador

- Desenho de mecanismos de segurança

- Consideração de introdução de redundância controlada

- Monotorizar e afinar o sistema operativo

Escolha de organização de ficheiros

Um dos principais objetivos no desenho do esquema física de uma base de dados é guardar e aceder dados de uma forma eficiente. Por exemplo, se pretendermos obter os donativos pela Data de doação, armazenar o ficheiro pela ordem da data seria uma organização eficaz do ficheiro. No entanto, se pretendermos os donativos de um determinado Tipo, procurar num ficheiro ordenado por Data não é muito eficiente.
A ordem com que os registos são armazenados e acedidos no ficheiro depende da sua organização.

Os principais tipos de organização de ficheiros são:

-Heap (ficheiros não ordenados): registos são armazenados sem ordem

-Sequencial (ficheiros ordenados): registos estão ordenados pelo valor de um determinadocampo

-Hash: registos estão armazenados de acordo com uma função hash

O MySQL não organiza e armazena informação, o motor de base de dados decide a forma como organiza a informação. Como vamos criar a base de dados em MySQL, vamos utilizar o formato de armazenamento mais comum, o InnoDB. O motor de armazenamento InnoDB mantém a sua área de buffer para armazenar dados e índices em memória principal. Além disso, armazena todos os dados e índices numa estrutura de dados do tipo B+ Tree. A organização de ficheiros em B+Tree é uma estrutura de armazenamento mais versátil que hashing. Suporta consultas baseadas em chaves exatas e intervalo de valores. O índice B+Tree é dinâmica, cresce com a relação. Além
disso, mantem a ordem da chave de acesso mesmo quando o ficheiro é atualizado, tornando a obtenção de informação mais eficiente. O motor escolhido também oferece recursos para transações ACID, assim como o suporte para chaves estrangeiras.