Perguntas bolsistas:

Parte 1: questões sobre banco de dados

Pergunta 1: O que é um Metadados e para que serve?

Metadados são **dados sobre dados** — ou seja, informações que descrevem, explicam, organizam ou facilitam o uso de outros dados.

📌 Explicando de forma simples

Se um dado é uma *foto*, os **metadados** são:

- a data em que foi tirada
- o local (GPS)
- o modelo da câmera
- a resolução
- quem tirou a foto

Se um dado é um *arquivo*, os metadados são:

- tamanho
- nome
- tipo (PDF, PNG, DOCX)
- data de criação
- autor

Se um dado é uma *tabela de banco de dados*, os metadados são:

- nome da tabela
- nome da coluna
- tipo de dado
- tamanho
- se aceita nulo
- descrição do campo
- chave primária / estrangeira

🗂️ Tipos de metadados

1. Metadados Descritivos

Ajudam a identificar e localizar o dado.

Ex.: título, autor, palavras-chave de um documento.

2. Metadados Estruturais

Explicam como os dados estão organizados.

Ex.: páginas de um livro, formatação de um XML, estrutura de tabelas.

3. Metadados Administrativos

Informações para gestão técnica.

Ex.: permissões, data de criação, versão, tamanho de arquivo.

🎯 Para que servem os metadados?

✔ Facilitar busca
✔ Organizar dados
✔ Permitir integração entre sistemas
✔ Garantir qualidade e consistência
✔ Ajudar na auditoria e segurança
✔ Automatizar processos (ex.: SEO, catalogação, indexação)

Pergunta 2: O que é um dicionário de dados e para que serve?

Resposta simples:

É um documento que descreve **todo o conteúdo** do banco, normalmente em forma de tabela, contendo:

- Nome da tabela
- Nome do campo
- Tipo de dado
- Tamanho
- Chave primária?
- Chave estrangeira?
- Nulo permitido?
- Valor padrão
- Descrição funcional

Serve para:

- ✔ manter padronização
- ✔ orientar desenvolvedores e analistas
- ✔ facilitar integrações e auditorias

Resposta completa:

Um **dicionário de dados** é um documento (ou sistema) que descreve detalhadamente os dados utilizados em um sistema de informação ou banco de dados.
Ele funciona como um **"manual explicativo" dos dados**, reunindo informações técnicas e funcionais sobre cada elemento armazenado.

O que contém um dicionário de dados?

Normalmente, inclui:

- **Nome do campo** (ex: data_nascimento)
- **Descrição** (Data de nascimento do cliente)
- **Tipo de dado** (texto, inteiro, data, booleano etc.)
- **Tamanho máximo**
- **Obrigatoriedade** (obrigatório ou opcional)
- **Relacionamentos** (chaves primárias e estrangeiras)
- **Regras de validação**
- **Origem do dado**
- **Responsável pelo dado**

Exemplo simples:

Campo	Tipo	Descrição
id_cliente	Inteiro	Identificador único do cliente
nome	Texto	Nome completo do cliente
data_nascimento	Data	Data de nascimento do cliente

Para que serve?

1 Padronização

Evita interpretações diferentes sobre o mesmo dado.

2 Comunicação entre áreas

Facilita o entendimento entre:

- TI
- Analistas

- Gestores
- Desenvolvedores
- Auditores

3️⃣ Governança de dados

Ajuda a organizar responsabilidades e regras.

4️⃣ Desenvolvimento de sistemas

Permite que programadores saibam exatamente:

- Como armazenar
- Que tipo usar
- Quais regras aplicar

5️⃣ Integração de sistemas

Fundamental quando diferentes sistemas precisam “conversar”.

🏢 Onde é usado?

- Bancos de dados corporativos
- Sistemas de ERP
- Sistemas governamentais
- Projetos de Business Intelligence
- APIs
- Projetos de ciência de dados

💡 Em resumo

Um dicionário de dados é como um **glossário técnico estruturado dos dados de um sistema**, garantindo organização, clareza, padronização e segurança na gestão da informação.

Pergunta 3: Fale sobre boas práticas em bancos de dados:

Resposta genérica:

1. Organize bem as informações
- Use nomes claros nas tabelas e campos.
- Evite guardar a mesma informação em vários lugares.

2. Crie índices
- Eles deixam as buscas mais rápidas.
- Mas não exagere: poucos índices bem escolhidos bastam.

3. Escreva consultas simples
- Peça só as informações necessárias (evite SELECT *).
- Sempre filtre os dados (use WHERE).

4. Cuide da segurança
- Use senhas fortes.
- Não deixe todo mundo acessar o banco.
- Use conexões seguras.

5. Faça backups
- Tenha cópias automáticas.
- Teste se consegue restaurar.

6. Monitore
- Verifique se alguma consulta está lenta.
- Limpe dados antigos, se possível.

7. Documente
- Anote o que cada tabela e campo significa.
- Isso ajuda outras pessoas a entenderem o banco.

Resposta de boas práticas pensando na linguagem do Power Query e Python (possíveis ferramentas para o nosso banco de dados):

1. Modelagem orientada à análise

☑ *Estruture dados pensando em ETL/ELT*
- Tabelas organizadas e bem relacionadas.
- Preferir modelos dimensionais (fato e dimensão) para BI.
- Evitar colunas com múltiplos valores (ex: listas separadas por vírgula).

☑ *Dados atômicos*
Cada campo deve conter **apenas uma informação**.
Isso facilita transformações no:
- Power Query (M)
- Pandas (Python)

☑ *Datas bem estruturadas*
- Usar tipo DATE ou TIMESTAMP
- Evitar texto para datas
- Manter padrão ISO (YYYY-MM-DD)

2. Performance para consumo analítico

☑ *Criar views para análise*
Em vez de permitir que cada analista construa queries complexas:
- Criar *views otimizadas*
- Padronizar joins e regras de negócio

Isso melhora:
- Performance
- Padronização
- Governança

☑ *Índices estratégicos*
Especialmente em:

- Chaves estrangeiras
- Campos usados em filtros frequentes
- Datas para séries temporais

☑ *Evitar SELECT **

Power Query e Python carregam tudo na memória.
Trazer apenas colunas necessárias reduz:

- Consumo de RAM
- Tempo de processamento

3. Integração com Power Query

☑ *Transformar o máximo possível no banco*

Power Query é poderoso, mas:

- O banco é mais eficiente para grandes volumes.
- Preferir fazer agregações e filtros no SQL.

Regra prática:
Dados pesados → tratar no banco
Ajustes analíticos → tratar no Power Query

☑ *Nomeação amigável*

Power BI e Excel exibem nomes de campos diretamente:

- Evitar nomes técnicos excessivos.
- Usar descrições claras.

Exemplo:
dt_nasc ✗
data_nascimento ☑

☑ *Evitar mudanças frequentes de schema*

Alterações constantes quebram relatórios.

4. Integração com Python (Pandas / Análise)

☑ *Tipagem consistente*

Python depende de tipos corretos:

- Inteiros não devem vir como texto.
- Campos numéricos sem caracteres especiais.
- Booleanos como TRUE/FALSE, não "Sim/Não".

☑ *Cuidado com nulos*

- Padronizar NULL no banco.
- Evitar valores como:
  - "N/A"
  - "-"
  - "Sem informação"

☑ *Volume de dados*

Se os datasets forem grandes:

- Implementar paginação
- Trabalhar com chunking (chunksize no pandas)
- Avaliar particionamento de tabelas

5. Governança e rastreabilidade

☑ *Dicionário de dados atualizado*

Fundamental para:

- Analistas
- Cientistas de dados
- Usuários de BI

☑ *Controle de versões*

- Versionar scripts SQL
- Versionar notebooks Python
- Documentar alterações estruturais

☑ *Logs e auditoria*

Registrar:

- Alterações críticas
- Cargas de dados
- Atualizações massivas

6. Qualidade de dados

☑ *Constraints no banco*

Não confiar apenas no Power Query ou Python:

- NOT NULL
- CHECK
- UNIQUE
- FOREIGN KEY

☑ *Validação na origem*

Garbage in → Garbage out.

7. Arquitetura recomendada

Uma arquitetura madura costuma seguir:

Banco de dados (camada bruta)

↓

Views analíticas

↓

Power Query / Python

↓

Dashboards / Modelos / Relatórios

Separar:

- Camada transacional
- Camada analítica

8. Escalabilidade e manutenção

☑ *Separar ambientes*

- Desenvolvimento
- Teste
- Produção

☑ *Backup e plano de recuperação*

Dados usados em BI costumam ser estratégicos.

☑ *Monitoramento*

Acompanhar:

- Queries lentas
- Crescimento de tabelas
- Uso de CPU e memória

🎯 Resumo estratégico

Se você usa Power Query e Python, o banco deve ser:

✔ Bem modelado
✔ Tipado corretamente
✔ Otimizado para leitura
✔ Documentado
✔ Estável (schema consistente)
✔ Com boa governança

Parte 2: Questões específicas de sistemática de sistematização de banco de dados relacionada ao processo de registro e averbação de contratos no INPI

Pergunta 1: Como estruturar um banco de dados pensando no fluxo do INPI?

☑ 1. Estruture o banco pensando no fluxo do INPI

O processo do INPI segue etapas: **protocolo → análise → exigências → deferimento/indeferimento → publicação**.

Seu banco deve refletir isso.

Tabelas recomendadas

Contratos
Partes (empresas/pessoas)
Pedidos no INPI
Anexos/Documentos
Exigências/Manifestações
Publicações
Histórico de status

☑ 2. Campos essenciais para o processo de registro/averbação

Tabela: Contratos
Identificador do contrato
Tipo (franquia, licença, cessão, know-how etc.)
Data de assinatura
Data de vigência
País das partes
Número de referência interno
Observações jurídicas
Tabela: Pedido INPI
Número do processo no INPI
Data do protocolo
Tipo de ato (registro / averbação)
Situação atual (em análise, exigência, deferido etc.)
Data da última atualização
Código da taxa GRU e comprovante
Tabela: Exigências
Número do processo
Tipo de exigência
Data da publicação
Prazo para resposta
Status (pendente / cumprida)

☑ 3. Relacionamentos importantes

Contrato 1:N Pedidos
(um contrato pode ter vários pedidos no INPI — retificações, exigências, etc.)
Pedido 1:N Exigências
Pedido 1:N Documentos enviados
Pedido 1:N Publicações na RPI

☑ 4. Controle de versões e documentos

INPI exige documentos atualizados e muitas vezes revisados.

Seu banco deve:

Permitir **múltias versões** de cada anexo.
Registrar quem enviou, quando, e para qual exigência.
Guardar o hash ou nome do arquivo para auditoria.

☑ 5. Campos essenciais para acompanhar prazos

O INPI estabelece prazos fixos (geralmente 60 dias p/ exigências).
Inclua:
Data da publicação
Data limite do prazo
Campo “alerta gerado” (sim/não)
Campo “resposta enviada”
Permite automação de lembretes.

☑ 6. Registro do histórico
O INPI muda status com frequência; é essencial guardar **cada mudança**, não só o status atual.
Tabelas recomendadas:
histórico_status (processo, status, data, responsável)

☑ 7. Padronize cadastros para evitar falhas
Os dados enviados ao INPI devem estar claros e consistentes.
Inclua validações:
CNPJ/CPF das partes
País de origem
Tipo de contrato válido
Datas coerentes
Campos obrigatórios específicos do INPI

☑ 8. Auditoria
Para garantir compliance:
Log de quem alterou dados
Registro de alterações críticas
Controle de anexos enviados

☑ 9. Indicadores úteis para gestão
Com a base organizada, é possível extrair:
Tempo médio por etapa
Quantidade de contratos por tipo
Contratos aguardando exigência
Contratos atrasados
Taxas pendentes de pagamento

📌 Resumo extremamente objetivo
Você deve criar um banco que registre:
Contrato
Processo no INPI
Documentos enviados
Exigências e prazos
Publicações da RPI
Histórico de status
E que permita rastreamento, auditoria e acompanhamento de prazo.

Pergunta 2: Como estruturar as tabelas e relacionamentos para garantir o rastreamento de versões de documentos?

Resposta:

- Uma tabela principal para **Contratos**
- Tabela para **Pedidos/Processos no INPI** ligada ao contrato (1:N)
- Tabela para **Exigências**, ligada ao processo (1:N)
- Tabela para **Documentos**, permitindo **várias versões** por processo (1:N)
- Tabela de **Histórico de Status**, registrando cada mudança com data (1:N)
- Campos de prazo (data de publicação, data limite, status de atendimento)

Pergunta 3: Quais campos e validações evitam inconsistências nos dados enviados ao INPI?

Resposta:

- Validação de CNPJ/CPF das partes
- Verificação do tipo de contrato permitido pelo INPI
- Datas coerentes (vigência, assinatura, protocolo)
- Campos obrigatórios preenchidos (contratante, contratado, objeto do contrato)
- Checagem de anexos exigidos (contrato, traduções, procurações)
- Padronização de nomes e formatos (PDF, tamanho máximo)