



UNIVERSIDADE ESTADUAL  
VALE DO ACARAÚ  
Centro de Ciências Exatas e Tecnologia  
Curso de Ciências da Computação

# Revisão de Sistemas de Banco de Dados





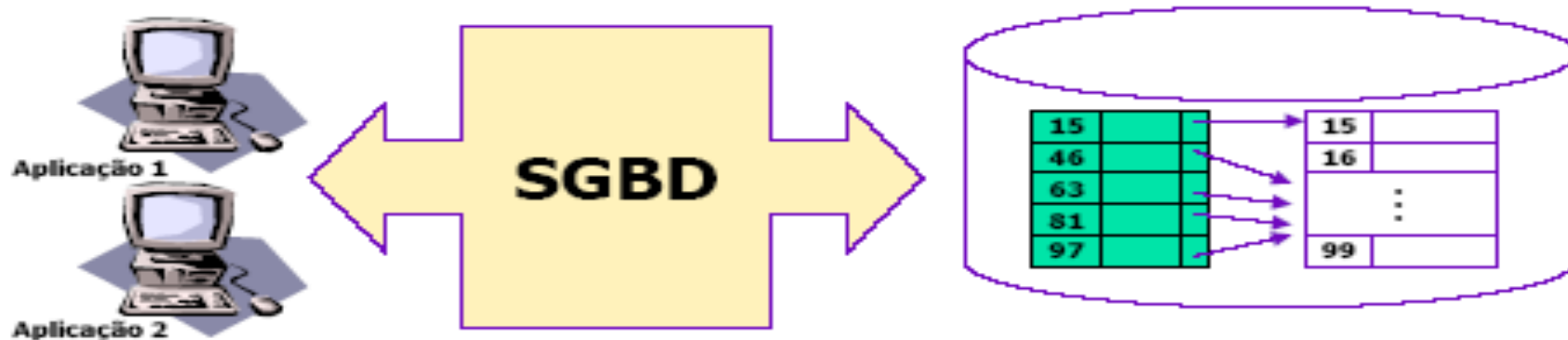
# Roteiro

- Conceitos e características de banco de dados
- Ciclo de vida de um sistema de banco de dados
- Modelo Entidade-Relacionamento
- Modelo Relacional
  - Normalização
  - Álgebra Relacional
  - SQL





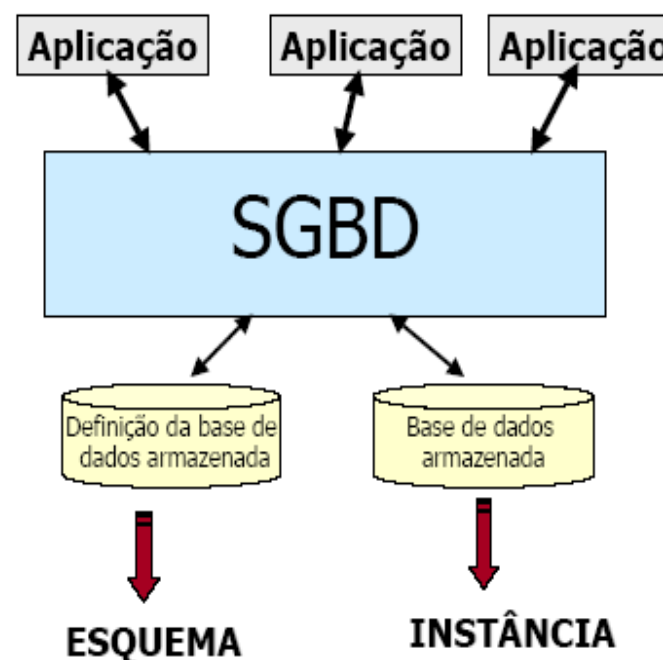
# Sistemas de Banco de Dados





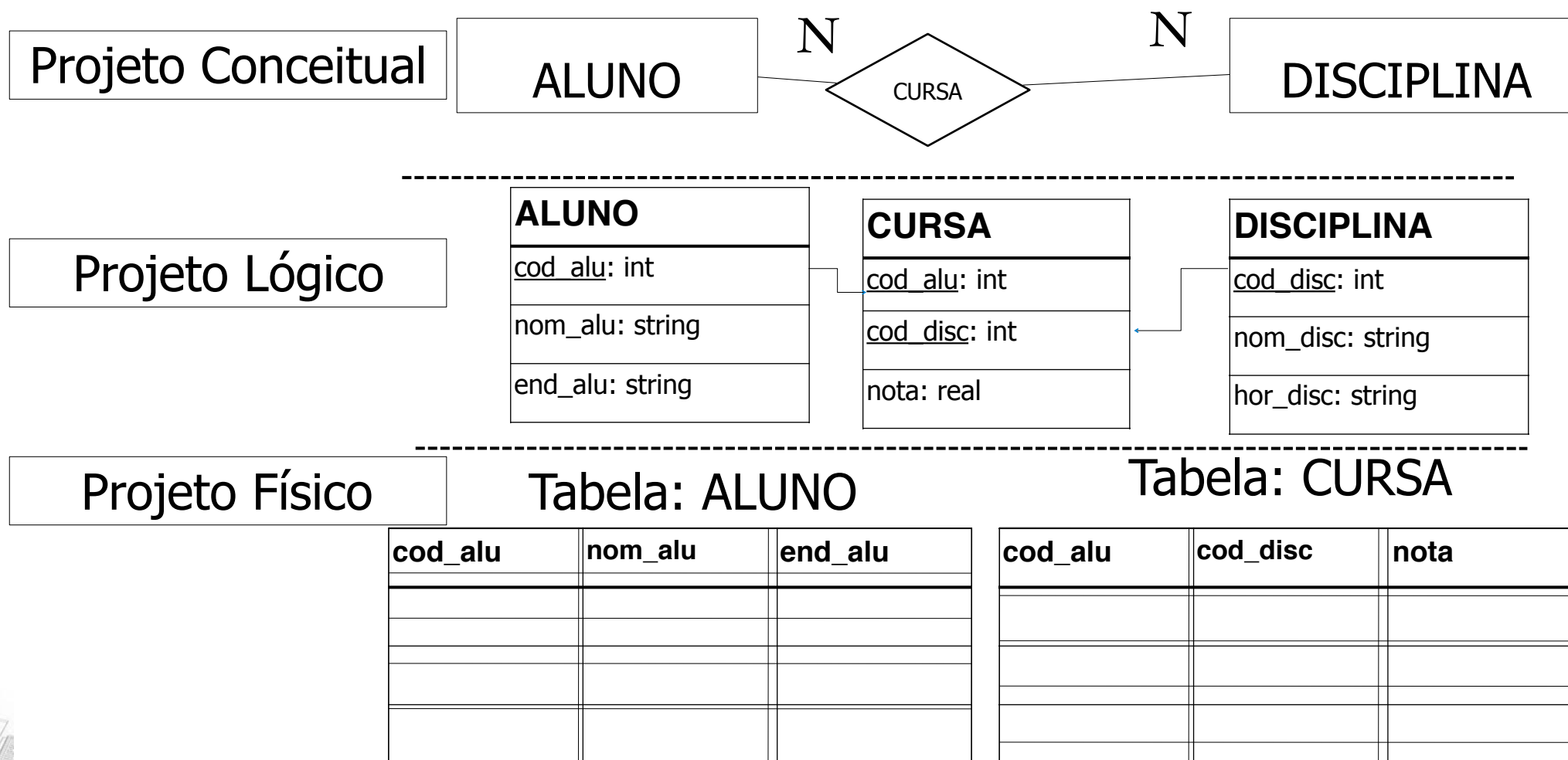
# Sistemas de Banco de Dados

- Abstração de dados
- Independência de dados
- Esquema e Instância





# Arquitetura em Três Camadas de Abstração





# Sistema Gerenciador de Banco de Dados

- Conjunto integrado de programas que permite descrever, armazenar, manipular, interrogar e tratar o conjunto de dados que compõem o banco de dados.
- Provê independência física e lógica dos dados
- Controle de redundância de dados
- Facilidade de consulta aos dados
- Garantia de integridade dos dados
- Compartilhamento dos dados
- Segurança dos dados





# Linguagens de Banco de Dados

- Linguagem de Definição de Dados (DDL)

- Define o esquema do banco de dados

```
CREATE TABLE ALUNO(  
    cod_alu      INT,  
    nom_alu      VARCHAR(35),  
    end_alu      VARCHAR(30)  
);
```

- Linguagem de Manipulação de Dados (DML)

- Manipula (consulta, insere, deleta, atualiza) as instâncias do banco de dados

```
SELECT nom_alu  
FROM ALUNO
```





# Ciclo de Vida de um Sistema de Banco de Dados

- Fase 1: Coleta e análise dos requisitos
- Fase 2: Projeto Conceitual usando um modelo de dados conceitual. Ex: MER
  - Escolha da tecnologia de banco de dados (tecnologia relacional, por exemplo) e do SGBD
- Fase 3: Projeto Lógico mapeamento do modelo conceitual (MER) para o modelo do SGBD escolhido (relacional)
- Fase 4: Projeto Físico implementação do banco de dados usando SGBD e vendo estruturas físicas de armazenamento através dos critérios:
  - tempo de resposta
  - espaço utilizado
  - número de transações







# Projeto Conceitual

- Desenho do esquema conceitual utilizando um modelo de dados
  - Alto nível de abstração
  - Independente do SGBD
    - MER – Modelo Entidade-Relacionamento





# Componentes Básicos do MER

- Entidades
  - Fortes e Fracas
- Relacionamentos
  - Grau
    - Binários, N-Ários ou Auto-relacionamentos
  - Cardinalidade
    - 1:1
    - 1:N
    - M:N





# Escolha do Paradigma de Banco de dados e de um SGBD

- Fatores técnicos, econômicos e políticos
- Custos:
  - do SGBD
  - do hardware
  - da manutenção
  - criação ou conversão
  - Pessoal
  - Treinamento
  - Operação
- Popularidade, familiaridade, desempenho, suporte...





# Projeto Lógico

- Transformar o Modelo de Dados Conceitual, em um modelo de dados lógico com base em alguma tecnologia de banco de dados.
- No modelo relacional, transformar as entidades (conceitual) em tabelas (lógico) e definir as restrições de integridade dos dados





# Modelo Lógico

- Modelo Relacional
  - Relações
    - Esquema da relação
    - Domínios
  - Atributos
  - Chaves
    - Candidata, Primária e Estrangeira
  - Restrições de Integridade
  - Álgebra Relacional





# Mapeamento de Modelos

- Mapear: mudar de representação
- Método: mapear as representações do Modelo ER para representações equivalentes no Modelo Relacional.
- Muitas ferramentas CASE (Computer-Aided Software Engineering) baseadas no modelo entidade-relacionamento (MER) convertem automaticamente o modelo lógico para um esquema de banco de dados relacional e geram a DDL para um SGBD específico.





# Mapeamento de Entidades

- Todas as entidades são mapeadas para uma tabela contendo os mesmos atributos do MER.
- Ex: EMPREGADO (RG, nome, idade)





# Mapeamentos de Entidades Fracas

- Identificador da entidade forte torna-se
  - parte da chave primária na tabela correspondente à entidade fraca (tabelaFrac)
  - chave estrangeira na tabelaFrac
    - Ex: *DEPENDENTE* (RG\_emp, RG\_dep, nome\_dep)







# Mapeamento de Atributos

- Mantenha nomes de atributos curtos e padronizados
- Indexe atributos muito consultados
- Atributos simples se transformam em colunas
- Compostos
  - Viram atributos simples
  - Ex: CLIENTE(cod\_cli, nom\_cli, rua\_cli, no\_cli, cid\_cli)
- Multivalorados
  - Viram outra tabela onde a parte da chave primária é a chave estrangeira da tabela “originária”





# Mapeamento de Especializações

- Três alternativas são geralmente adotadas
  1. tabela única para entidade genérica e suas especializações
  2. tabelas para a entidade genérica e as entidades especializadas
  3. tabelas apenas para as entidades especializadas





# Exemplos

1. Conta (no\_conta, sal\_conta, tx\_juros, lim\_esp, tipo\_conta)
2. Conta (no\_conta, sal\_conta)  
Poupança (no\_conta, tx\_juros)  
Corrente (no\_conta, lim\_esp)
3. Poupança (no\_conta, sal\_conta, tx\_juros)  
Corrente (no\_conta, sal\_conta, lim\_esp)





# Mapeamentos de Relacionamentos

- Recomendações de mapeamento baseiam-se na análise da cardinalidade dos relacionamentos
  - com base nesta análise, algumas alternativas de mapeamento podem ser adotadas
    1. entidades relacionadas podem ser fundidas em uma única tabela
    2. tabelas podem ser criadas para o relacionamento
    3. chaves estrangeiras podem ser criadas em tabelas a fim de representar adequadamente o relacionamento





# Relacionamento 1:1

- Escolha uma das relações envolvidas no relacionamento
  - E inclua como chave estrangeira nesta relação a chave primária da outra relação





# Relacionamento 1:N

- À tabela que representa a entidade no lado N, acrescenta-se como chave estrangeira a chave primária da entidade do lado 1
  - Ex: Pedido(cod\_ped, data\_ped, cod\_cli)





# Relacionamento de N:N

- É criada uma nova tabela contendo como chaves estrangeiras as chaves primárias das entidades participantes, mais os atributos do relacionamento.
- Ex: Pedido\_Contem\_Produto (cod\_ped, cod\_pro, qtde\_pro)





# Auto-Relacionamento

- Valem as mesmas recomendações anteriores
  - Ex: Empregados (RG, Nome, Idade)  
Gerência(RGe, RGg)

Empregados (RG, Nome, Idade, RGg)







# Relacionamento Ternário

- O mapeamento ocorre de forma semelhante ao descrito para relacionamentos N:N





# Normalização

- É uma ferramenta para ser utilizada no projeto lógico de BD
  - Objetivo 1: remover redundâncias entre atributos não-chave.
  - Objetivo 2: manter integridade dos dados após remoção, inserção, atualização.
- O processo de normalização consiste em decompor um conjunto de tabelas em um outro conjunto de tal forma que o novo conjunto não tenha anomalias tornando o esquema do bd mais simples e regular.





# Exemplo de Tabela não Normalizada

- Manutenção do histórico de entrada de estrangeiros no país
- Deseja controlar a entrada de cada estrangeiro
  - é necessário saber se o estrangeiro já tem ficha na Polícia Federal
  - para todos os controles relativos à entrada, é necessário saber as informações relativas à nacionalidade e respectiva embaixada
  - em caso de expatriamento por morte, a embaixada comunica à PF, e o estrangeiro é removido do sistema





# Exemplo de Tabela não Normalizada

<u>Data</u>	<u>Passap</u>	Nome	Vôo	Nacionalidade	Nasc	Já-ficha	Embaixada
10/09/12	Pas1	João	RG 121	Argentina	1/1/65	Não	Endereço 1
10/09/12	Pas2	José	AR876	Argentina	22/4/50	Sim	Endereço 1
10/09/12	Pas3	Hans	RG 121	Alemanha	12/09/50	Não	Endereço 2
11/09/12	Pas4	Maria	RG 121	EUA	12/07/70	Não	Endereço 3
11/09/12	Pas5	Kanda	JL 234	Japão	12/08/78	Não	Endereço 4
20/09/12	Pas1	João	VS 987	Argentina	1/1/65	Não	Endereço 1





# Redundância e Anomalias

- A tabela do exemplo exhibe dados redundantes.
- Estas redundâncias produzem várias anomalias:
  - Anomalia de atualização
  - Anomalia de remoção
  - Anomalia de inserção





# Anomalias de Inserção

- É necessário inserir valores nulos para os atributos cujo valor ainda não foi determinado ou valores inválidos, quando esses atributos pertencem à chave primária, o que conduz a inconsistência
- No exemplo, quando se insere um novo estrangeiro é necessário introduzir valores nulos e valores inválidos para os atributos que ainda não são conhecidos. A "alternativa" consiste em não inserir dados de um novo estrangeiro enquanto a alocação não for efetuada
- A duplicação de alguns dados poderá dar origem a erros de inserção, o que resulta em inconsistência.





# Anomalias de Alteração

- A existência de redundância conduz ao perigo de após uma atualização, apenas parte dos dados terem sido atualizados.
- No exemplo, se alterar o endereço da embaixada da Argentina, poderá ocorrer uma anomalia deste tipo se não se atualizaram todas as ocorrências da mesma embaixada





# Anomalias de Remoção

- A remoção de determinados dados podem levar à eliminação de outra informação que não se pretendia apagar.
- A remoção de uma estrangeiro pode conduzir à perda de informação relativa a uma embaixada







# Dependência Funcional

- Considere dois conjuntos de atributos **X e Y de** uma relação **R**.
- $X \rightarrow Y$  (o atributo X determina funcionalmente o atributo Y)  $\Leftrightarrow$  sempre que duas tuplas quaisquer de R tiverem o mesmo valor para X, elas possuem também o mesmo valor para Y. Ou seja, para qualquer par de tuplas *t1 e t2 em R*, se

$$t1[X] = t2[X], \text{ então } t1[Y] = t2[Y]$$

- Se  $v(X)$  ocorrer em duas ou mais linhas em R,  $v(Y)$  deve ser o mesmo em todas estas linhas.

Determinante: X

Determinado: Y





# Dependência Funcional

## Exemplos

- Empregado(matrícula, nome, CPF, salário, matricula\_supervisor)
  - matrícula  $\rightarrow$  nome
  - matrícula  $\rightarrow$  CPF
  - CPF  $\rightarrow$  matrícula
  - {CPF, matrícula}  $\rightarrow$  {nome, CPF }
  - matrícula  $\rightarrow$  {matricula\_supervisor, salário}
- Projeto(número, nome, verba, localização)
  - número  $\rightarrow$  {nome, localização}
- Alocação (matrícula\_empr, número\_proj, qtd\_horas)
  - {matrícula\_empr, número\_proj}  $\rightarrow$  qtd\_horas





# Propriedades de uma DF

- Dependência funcional total (completa):
  - Se  $X$  determina totalmente  $Y$ , então  $Y$  não é funcionalmente dependente de nenhum subconjunto dos atributos que compõem  $X$ 
    - cidade  $\rightarrow$  ddd (total)
    - cidade, nome  $\rightarrow$  ddd (parcial)
- Sejam  $a, b, g$  atributos (simples ou compostos)
  - Transitividade:
    - se  $a \rightarrow b$  e  $b \rightarrow g$ , então  $a \rightarrow g$ 
      - cpf  $\rightarrow$  cidade e cidade  $\rightarrow$  DDD, então cpf  $\rightarrow$  DDD





# Normalização x DF

- O conhecimento de dependência funcional é útil na detecção de redundâncias.
- Considere uma relação  $R$  com três atributos,  $XYZ$ .
  - Se nenhuma dependência funcional existe, então não há redundância.
  - Porém se  $X \rightarrow Y$ , temos várias tuplas em  $R$  podendo ter o mesmo valor para  $X$  e, conseqüentemente, terão o mesmo valor para  $Y$ .
- Sintetizando: o processo de normalização consiste basicamente em remover dependências funcionais.





# Formas Normais

- Para eliminar redundâncias e anomalias de um conjunto de relações, precisamos levar este conjunto através de diversas formas normais:
  - 1a Forma Normal (1FN)
  - 2a Forma Normal (2FN)
  - 3a Forma Normal (3FN)
- Dizemos que uma relação está em uma certa FN, se esta relação obedece ao conjunto de regras estabelecidas por esta FN.
- A normalização é um processo em cadeia:
  - Para uma relação estar na 3FN, ela deve estar na 2FN que por sua vez deve estar na 1FN





# 1ª Forma Normal – 1FN

- **Uma tabela está na 1FN se:**
  - **Não tem grupos repetidos de atributos**
  - **Cada um de seus atributos é atômico**
    - Um atributo é atômico se não há necessidade de decompor este valor





# 2ª Forma Normal – 2FN

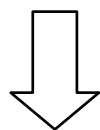
- **Uma tabela está na 2FN se está na 1FN, e cada um dos atributos não pertencentes à chave primária for dependente total dessa chave.**
- Ou seja, se uma coluna de uma tabela não pertence à chave e pode ter seu valor determinado por parte da chave é dita como “dependente parcialmente da chave”.
- Portanto, para levar um conjunto de tabelas à 2FN, o objetivo é remover dependências parciais.





# 2ª Forma Normal – 2FN

- Carro = ( placa, licença\_dono, nome\_dono, modelo, qtd\_km\_rodados, qtd\_km\_rodados\_por\_dono)



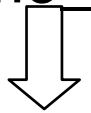
- Relação1 = ( licença\_dono, nome\_dono )
- Relação2 = ( placa\_carro, modelo, qtd\_km\_rodados)
- Relação3 = ( placa\_carro, licença\_dono, qtd\_km\_rodados\_por\_dono)







# 3ª Forma Normal – 3FN

- Uma tabela está na 3FN se está na 2FN, e cada um dos atributos não pertencentes à chave primária **NÃO** possui dependência funcional transitiva com essa chave.
- Carro ( placa, modelo, quantidade\_km\_rodados, código\_fabricante, nome\_fabricante )  

  - Carro ( placa, modelo, quantidade\_km\_rodados, código\_fabricante )
  - Fabricante ( código, nome )





# Voltando ao Exemplo Inicial

- Quanto objetos a tabela ESTRANGEIROS descreve ao mesmo tempo?
  - países (nacionalidade)
  - estrangeiros (passaporte)
  - entradas no país (data, passaporte)
- Quantos relacionamentos a tabela ESTRANGEIROS descreve ao mesmo tempo?
  - estrangeiros e entrada no país
  - estrangeiros e sua nacionalidade





# Dependências Funcionais

- Estrangeiros
  - passaporte → nome, nacionalidade, nasc, já-ficha
- Nacionalidade
  - nacionalidade → embaixada, ex -visto
- Entrada
  - data, passaporte → voo





# Depois da Normalização

## ESTRANGEIROS

### ENTRADAS

Data	Passap	Voo
10/09	Pas1	RG121
10/09	Pas2	AR876
10/09	Pas3	RG121
11/09	Pas4	RG121
11/09	Pas5	JL234
20/09	Pas1	VS987

Passap	Nome	Nacionalidade	Nasc	Já-ficha
Pas1	João	Argentina	1/1/65	Não
Pas2	José	Argentina	22/4/50	Sim
Pas3	Hans	Argentina	12/09/50	Não
Pas4	Maria	EUA	12/07/70	Não
Pas5	Kanda	Japão	12/08/78	Não

### NACIONALIDADE

Nacionalidade	Embaixada
Argentina	Endereço 1
Alemanha	Endereço 2
EUA	Endereço 3
Japão	Endereço 4





# Projeto Físico

- Especificação em SQL do esquema relacional para o SGBD escolhido.
- Critérios a atender :
  - Tempo de resposta
  - Espaço de armazenamento
  - Volume de transações suportado
- Estruturas de armazenamento e de recuperação de informações
- Mecanismos de acesso devem ser escolhidos, visando sempre o aprimoramento da performance dos aplicativos de BD.





# Projeto Físico

- Devem ser especificados não apenas as tabelas criadas, mas também
  - os índices necessários,
  - as restrições de integridade ,
  - algumas operações de inclusão, exclusão e atualização de dados para cada tabela,
  - bem como as consultas que a aplicação deve realizar.





# Implementação do Sistema de Banco de Dados

- Com instruções da DDL e da DML (SQL)
  - Faz-se a carga do Banco de Dados





UNIVERSIDADE ESTADUAL  
VALE DO ACARAÚ  
Centro de Ciências Exatas e Tecnologia  
Curso de Ciências da Computação

Dúvidas???

Perguntas???

Questionamentos???

