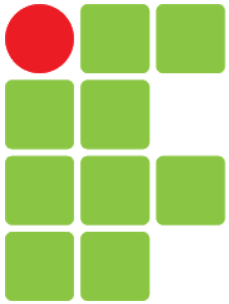


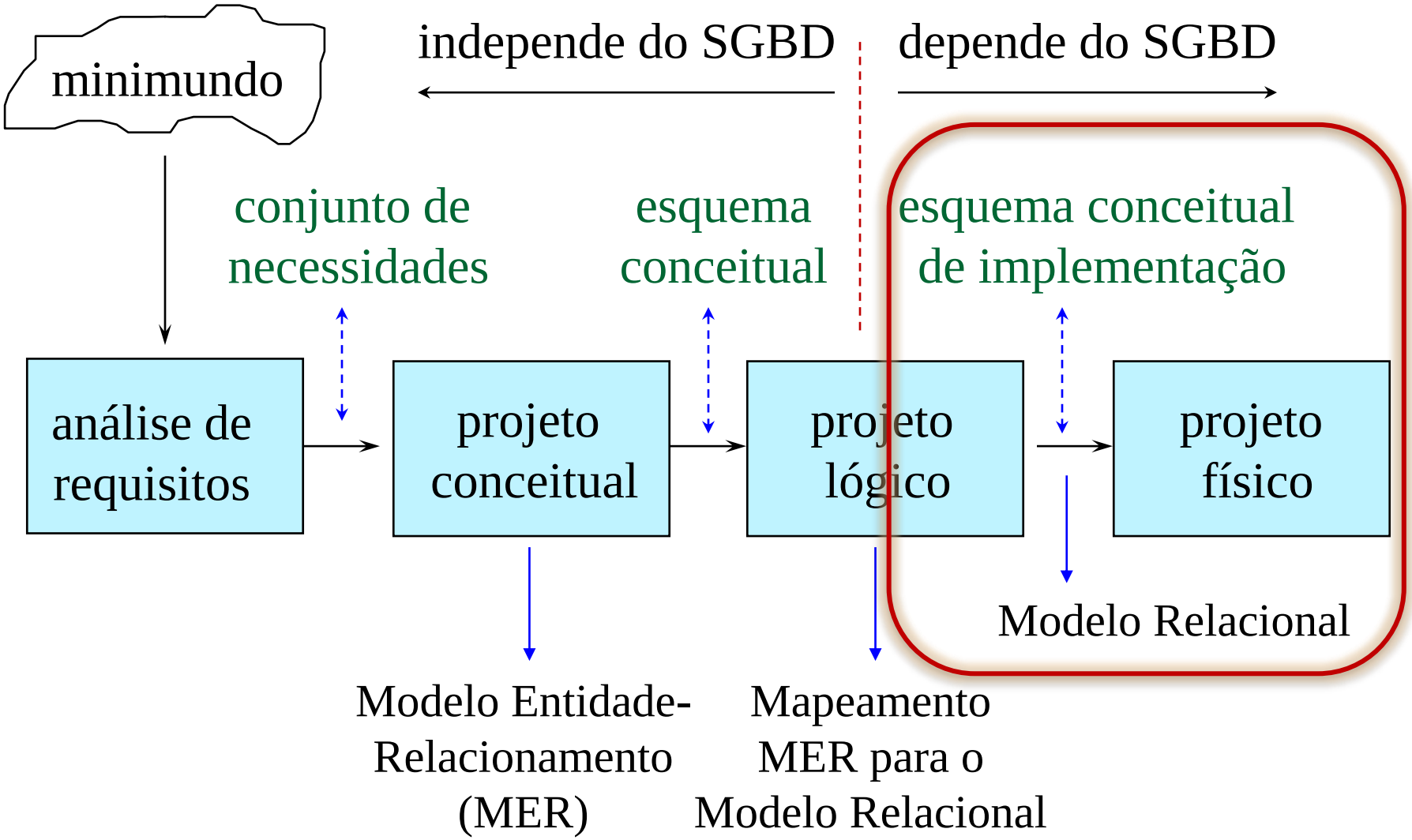
INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
BAIANO
Campus Guanambi

Linguagem SQL: DML

Eber Chagas Santos
professor.eberchagas@gmail.com

Slides baseados no material elaborado pelos professores:
Cristina D. A. Ciferri e **Ricardo R. Ciferri**

Projeto de BD



SQL DML

- `INSERT INTO ...`
  - insere dados em uma tabela
- `SELECT ... FROM ... WHERE ...`
  - lista atributos de uma ou mais tabelas de acordo com alguma condição
- `DELETE FROM ... WHERE ...`
  - remove dados de tabelas já existentes
- `UPDATE ... SET ... WHERE ...`
  - altera dados específicos de uma tabela

Inserção

- Realizada através da especificação
  - de uma tupla particular
  - de uma consulta que resulta em um conjunto de tuplas a serem inseridas
- Valores dos atributos das tuplas inseridas
  - devem pertencer ao domínio do atributo
- Atributos sem valores
  - especificados por NULL ou valor DEFAULT

INSERT

INSERT INTO nome_tabela
VALUES (V_1, V_2, ..., V_N) ;

- Ordem dos atributos deve ser mantida

Inserindo um registro na tabela Aluno:

```
INSERT INTO Aluno VALUES ('2023001',
'Fulano');
```

INSERT

INSERT INTO nome_tabela (A_1, A_2, ..., A_n)
VALUES (V_1, V_2, ..., V_N) ;

- Ordem dos atributos não precisa ser mantida

Inserindo outro registro na tabela Aluno:

```
INSERT INTO Aluno (nome,
matricula)VALUES('Beltrano','2023002');
```

INSERT

```
INSERT INTO nome_tabela
   SELECT ...
      FROM ...
   WHERE ...   ;
```

- Tuplas resultantes da cláusula SELECT serão inseridas na tabela nome_tabela

SELECT

```
SELECT <lista de atributos e funções>
FROM <lista de tabelas>
[ WHERE predicado ]
[ GROUP BY <atributos de agrupamento> ]
[ HAVING <condição para agrupamento> ]
[ ORDER BY <lista de atributos> ] ;
```

SELECT

- Cláusula SELECT
  - lista os atributos e/ou as funções a serem exibidos no resultado da consulta
- Cláusula FROM
  - especifica as relações a serem examinadas na avaliação da consulta

Listando os alunos cadastrados:

```
SELECT matricula FROM Aluno;
SELECT nickname FROM Aluno;
SELECT * FROM Aluno;
```

SELECT

- Cláusula WHERE
  - especifica as condições para a seleção das tuplas no resultado da consulta
    - as condições devem ser definidas sobre os atributos das relações que aparecem na cláusula FROM
  - pode ser omitida

Listando um aluno específico:

```
SELECT * FROM Aluno
WHERE matricula = '2023001';
```

SELECT

- Resultado de uma consulta
  - ordem de apresentação dos atributos
    - ordem dos atributos na cláusula SELECT
  - ordem de apresentação dos dados
    - ordem ascendente ou descendente de acordo com a cláusula ORDER BY
    - sem ordenação
  - duas ou mais tuplas podem possuir valores idênticos de atributos
    - eliminação de tuplas duplicadas
      - SELECT DISTINCT

Cláusula WHERE

SELECT

FROM

WHERE <atributo> <operador>
<valor | atributo | lista de valores>

- Operador
  - conjunção de condições: AND
  - disjunção de condições: OR
  - negação de condições: NOT

Cláusula WHERE

- Operadores de comparação

igual a	=	diferente de	< >
maior que	>	maior ou igual a	>=
menor que	<	menor ou igual a	<=
entre *dois* valores	BETWEEN ... AND	de cadeias de caracteres	LIKE *ou* NOT LIKE

- Precedência
  - NOT; operadores de comparação; AND; OR

Cláusula WHERE

- Operadores de comparação de cadeias de caracteres
  - % (porcentagem): substitui qualquer *string*
  - _ (underscore): substitui qualquer *caractere*
- Característica
  - operadores sensíveis ao caso***
    - letras maiúsculas são consideradas diferentes de letras minúsculas
    - *** = Varia de SGBD para SGBD

Cláusula WHERE

- Exemplos
  - WHERE nome LIKE ‘Ful%’
    - qualquer string que se inicie com ‘Ful’
  - WHERE nome LIKE ‘Ful_’
    - qualquer string de 4 caracteres que se inicie com ‘Ful’

Listando os alunos cujo nome sejam iniciados com “Ful”:

```
SELECT * FROM Aluno
WHERE nome LIKE 'Ful%';
```

Cláusula AS

- Renomeia
  - atributos
    - deve aparecer na cláusula SELECT
    - útil para a visualização das respostas na tela
  - relações
    - deve aparecer na cláusula FROM
    - útil quando a mesma relação é utilizada mais do que uma vez na mesma consulta
- Sintaxe
  - nome_antigo AS nome_novo

Listando os alunos por nome, mas exibindo o referido campo como "nickname":

```
SELECT nome as nickname
FROM Aluno;
```

Cláusula ORDER BY

- Ordena as tuplas que aparecem no resultado de uma consulta
  - asc (padrão): ordem ascendente
  - desc: ordem descendente
- Ordenação pode ser especificada em vários atributos
  - a ordenação referente ao primeiro atributo é prioritária. Se houver valores repetidos, então é utilizada a ordenação referente ao segundo atributo, e assim por diante

Cláusula ORDER BY

Listando os alunos em ordem crescente/ascendente (por nome):

```
SELECT * FROM Aluno
ORDER BY nome ASC;
```

Listando os alunos em ordem decrescente/descendente (por nome):

```
SELECT * FROM Aluno
ORDER BY nome DESC;
```

Funções de Agregação

- Funções
  - Média → AVG( )
  - Mínimo → MIN( )
  - Máximo → MAX( )
  - Total → SUM( )
  - Contagem → COUNT( )

- Observação
  - DISTINCT: não considera valores duplicados
  - ALL: inclui valores duplicados

Funções de Agregação

- Características
  - recebem uma coleção de valores como entrada
  - retornam um único valor
- Entrada
  - sum( ) e avg( ): conjunto de números
  - demais funções: tipos de dados numéricos e não-numéricos

Cláusula GROUP BY

*** **Exemplos serão vistos nos exercícios**

- Funcionalidade
  - permite aplicar uma função de agregação não somente a um conjunto de tuplas, mas também a um grupo de conjunto de tuplas
- Grupo de conjunto de tuplas
  - conjunto de tuplas que possuem o mesmo valor para os atributos de agrupamento
- Semântica da respostas
  - atributos de agrupamento no GROUP BY também devem aparecer no SELECT

Cláusula HAVING

- Funcionalidade
  - permite especificar uma condição de seleção para grupos, ao invés de uma condição para tuplas individuais
- Resposta
  - recupera os valores para as funções somente para aqueles grupos que satisfazem à condição imposta na cláusula HAVING

***** Exemplos serão vistos nos exercícios**

DELETE

```
DELETE
FROM nome_tabela
WHERE predicado ;
```

- Cláusula WHERE
  - é opcional:
    - todas as tuplas da tabela são eliminadas
    - a tabela continua a existir
- Predicado
  - pode ser complexo

DELETE ...

- Remove tuplas inteiras
- Opera apenas em uma relação
- Tuplas de mais de uma relação a serem removidas:
  - um comando DELETE para cada relação

- A remoção de uma tupla de uma relação é propagada para tuplas em outras relações?
  - restrições de integridade referencial

DELETE ...

Excluindo o aluno cuja matricula é 2019002:

```
DELETE FROM Aluno
WHERE matricula = '2023002';
```

Excluindo todos os registros de uma tabela:

```
DELETE FROM Aluno;
```

UPDATE

UPDATE nome_tabela
SET coluna = <*valor*>
WHERE predicado ;

- Cláusula WHERE
  - é opcional
- Exemplos de <*valor*>
  - NULL
  - ´string´
  - UPPER ´string´

Atualizando o nome de um aluno:

```
UPDATE Aluno
SET nome = 'Fulano
Casado'
WHERE matricula =
'2023001';
```

UPDATE ...

- Opera apenas em uma relação

- A atualização da chave primária é propagada para tuplas em outras relações?
  - restrições de integridade referencial