Laboratório de Banco de Dados

Profa. Marilde Terezinha Prado Santos

Sumário

- Plano de ensino da disciplina
- Calendário
- Critérios de avaliação
- Projetos
- Divisão da turma em grupos para execução dos projetos
- Orientação para execução da primeira etapa do projeto – documento de requisitos, projeto ER e projeto de BD relacional.

Plano de Ensino

- Prática de implementação de sistemas de banco de dados:
  - BD Relacional
  - BD Relacional/XML
- SGBDs
  - Oracle
  - DB2
  - SQL Server
  - PostgreSQL
  - MySQL

MARÇO

D	S	T	Q	Q	S	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

ABRIL

D	S	T	Q	Q	S	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

MAIO

D	S	T	Q	Q	S	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23/30	24/31	25	26	27	28	29

JUNHO

D	S	T	Q	Q	S	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

Turma A (segundas)	Turma C (quartas)
08/3 – Plano ensino/revisão	10/3
15/3 – Doc. Requisitos	17/3
22/3 – Proj. Conc. E Lógico	24/3 – P1 etapa 1
29/3 – Inic implementação	31/3
05/4 – Query/Update/View	07/4
12/4 – Procedure/Function	14/4 – P1 etapa 2
19/4	**21/4 - feriado**
26/4 – Trigger/Cursor/Seq	28/4
03/5 – Assertion/Index	05/5
10/5 – P1 etapa 3 (prova 1)	12/5 – P1 etapa 3 (prova 1)
17/5 – Projeto Rel/XML	19/5
24/5 – Implementação XML	26/5 – P2 etapa 1
31/5 – Inserção dado XML	02/6
07/6 – Consulta XML	09/6
14/6 – Consulta XML	16/6
21/6 – P2 etapa 2 (prova 2)	23/6 – P2 etapa 2 (prova 2)
28/6 – revisões	30/6 – revisões

Critérios de Avaliação

- Serão avaliados 2 grandes projetos quanto a grau de dificuldade, sintonia com a matéria vista em sala de aula, correção na documentação e implementação geradas (são previstas etapas de entrega para cada projeto – atrasos na entrega das etapas podem acarretar perda de pontos)
- Concomitante com a entrega da última etapa de cada projeto será aplicada uma avaliação individual referente ao projeto

  - P1: Projeto relacional – 10 ou 12 de Maio
    - 3 etapas P1= (etapas * 0,5 + avaliação * 0,5) * 0,4
  - P2: Projeto relacional/XML – 21 ou 23 de Junho
    - 2 etapas P2= (etapas * 0,5 + avaliação * 0,5) * 0,3

- Serão avaliados o cumprimento de tarefas dirigidas em sala de aula (não serão admitidos trabalhos de quem não esteve presente em sala)
  - Nota Participação: Média aritmética das tarefas propostas * 0,3

Repositório de Projetos

Problema: Nos últimos anos, tem-se assistido ao desenvolvimento de um grande número de projetos por parte de alunos. Muitos dos resultados desses projetos ficam esquecidos por não terem uma divulgação conveniente.

Solução: Neste projeto, pretende-se criar um repositório digital de projetos que terá uma interface Web para permitir aos utilizadores consultar os dados dos projetos e ter acesso aos códigos de programação dos mesmos.

O que deve ser feito para atingir os objetivos :

1. Desenvolva um estudo sobre a metainformação necessária para catalogar um projeto, toda documentação que o descreve e os códigos de implementação gerados.
2. Pretende-se que o grupo de estudantes parta deste estudo para gerar: (a) um documento de requisitos o mais completo possível, (b) um esquema conceitual do repositório, (c) um esquema relacional para guardar este tipo de informação.
3. Posteriormente deverão ser desenvolvidas duas interfaces Web para o repositório: uma de administração que permitirá acrescentar projetos, atualizá-los com controle de versão, eliminá-los, limitar acessos, e outra de consulta e downloads de arquivos.

Repositório de Publicações

Problema: Nos últimos anos, tem-se produzido um grande número de publicações por parte de pesquisadores do PPG-CC/UFSCar, algumas gerando premiações a seus autores. Todo início de ano, existe a árdua tarefa de construir um relatório com todas as publicações produzidas no último ano para apresentar para a CAPES para efeito de avaliação do programa de pós-graduação. Muitos dos resultados obtidos por pesquisadores são omitidos do relatório por não estarem organizados em meio de acesso conveniente. Existe dificuldade também na divulgação e reuso desse material.

Solução: Neste projeto, pretende-se criar um repositório digital de publicações que terá uma interface Web para permitir aos utilizadores consultar os dados dos publicações e ter acesso ao documento digital referente aos mesmos.

O que deve ser feito para atingir os objetivos :

1. Desenvolva um estudo sobre a metainformação necessária para catalogar uma publicação, toda informação que descreve onde e como foi divulgada.
2. Pretende-se que o grupo de estudantes parta deste estudo para gerar: (a) um documento de requisitos o mais completo possível, (b) um esquema conceitual do repositório, (c) um esquema relacional para guardar este tipo de informação.
3. Posteriormente deverão ser desenvolvidas duas interfaces Web para o repositório: uma de administração que permitirá acrescentar publicações, atualizá-las com controle de versão, eliminá-las, e outra de consulta e downloads de arquivos.

Repositório de Exames

Problema: No que diz respeito a provas de avaliação, a tarefa de um professor é muito repetitiva e, normalmente, há uma grande sobreposição em exames da mesma disciplina.

Solução: Criar um repositório de questões e as correspondentes ferramentas de gestão com as quais será possível construir exames, e manter viva toda uma memória digital.

O que deve ser feito para atingir os objetivos:

1. Desenvolva um estudo sobre a metainformação necessária para catalogar um exame, constando temas analisados, questões utilizadas e respectivo gabarito. O grupo de estudantes deverá fazer um levantamento do tipo de questões e exames que poderão existir.
2. Pretende-se que o grupo de estudantes parta deste estudo para gerar: (a) um documento de requisitos o mais completo possível, (b) um esquema conceitual do repositório, (c) um esquema relacional para guardar este tipo de informação.
3. Numa fase posterior, deverá ser implementada uma ferramenta de gestão com a qual seja possível construir exames (com facilidade de consulta às questões da base de dados), acrescentar questões, criar novas versões de questões já existentes (mantendo controle de versão), e outras operações que surjirão.

Orientação Primeira Etapa P1

- Formato
  - Cada grupo deverá gerenciar uma wiki com a documentação de seu projeto
  - A folha de rosto da wiki deve conter as seguintes informações:
    - Nome da Universidade/ Departamento
    - Nome (ou sigla) do projeto do grupo
    - Nome do SGBD que abrigará o sistema de BD
    - Identificação da Turma (A ou C)
    - Identificação (RA, Nome e email) dos membros do grupo

Orientação Primeira Etapa P1

- Documento de requisitos
  - Deve seguir formato apresentado na disciplina de Engenharia de Software
- Projeto Entidade-Relacionamento
- Projeto Banco de Dados Relacional
- SQL básica no Oracle
- Exercício para entrega

SQL Básica: Tabelas

```
CREATE TABLE [esquema.]tabela (
coluna1 tipo_dado [DEFAULT expr]
          [constraint_coluna],
...
colunaN tipo_dado [DEFAULT expr]
          [constraint_coluna],
[constraint_tabela]
);
```

SQL Básica: Tipos de Dados

- Integer, Float, Real...
- Char (n)
- Varchar2 (n)
- Clob
- Long
- Blob
- Raw e Long Raw
- Number (p,e)
- Date
- Timestamp
- Interval Year (p) to month
- Interval Day (dp) to second (sp)

SQL Básica: Constraints

- Grupo 1
  - NOT NULL
  - Unique
- Grupo 2
  - Check
  - Primary key
  - Foreign key

```
CONSTRAINT nome tipo expr
  constraint cod Primary Key
  constraint fcod Foreign Key references
  tabela(coluna)
  constraint chk Check (uf in ('SP',
  'MG'))
```

Exemplo

EQUIPE
(#Codigo_Equipe,Nome_Equipe,Cidade,Estado)

```
CREATE TABLE equipe (
  codigo_equipe INTEGER constraint
     equipe_pk Primary Key,
  nome_equipe Varchar2(20) NOT NULL,
  cidade Varchar2(10),
  estado Varchar2(10)
);
```

Exemplo

JOGADOR

(#Codigo_Jogador,Nome_Jogador,Posição_Jog,#Codigo_Equipe)

Codigo_jogador → INTEGER

Nome_jogador → Varchar2(20)

Posicao_jog → Varchar2(15)

Codigo_Time → INTEGER

Exemplo

JOGADOR
(#Codigo_Jogador,Nome_Jogador,Posição_Jog,Codigo_equipe)

```
CREATE TABLE jogador (
  codigo_jogador INTEGER constraint jogador_pk Primary
  Key,
  nome_jogador Varchar2(20) NOT NULL,
  posicao_jog Varchar2(15),
  codigo_equipe INTEGER,
   constraint jogador_fk Foreign Key (codigo_equipe)
  references equipe(codigo_equipe)
);
```

Exemplo

PARTIDA

(#Codigo_Partida,Cidade,Estado,Nome_Juiz,Data
)

Codigo_partida → INTEGER
cidade → Varchar2(10)
estado → Varchar2(10)
nome_juiz → Varchar2(20)
data → date

Exemplo

PARTIDA
(#Codigo_Partida,Cidade,Estado,Nome_Juiz,Data)

```
CREATE TABLE partida (
  codigo_partida INTEGER constraint
     partida_pk Primary Key,
  cidade Varchar2(10) NOT NULL,
  estado Varchar2(10),
  nome_juiz Varchar2(20) NOT NULL,
  data Date
);
```

SQL Básica: Mais Constraints!

- ON DELETE
  - SET NULL
  - CASCADE
  - SET DEFAULT
- ON UPDATE
  - SET NULL
  - CASCADE
  - SET DEFAULT

SQL Básica: Exemplo

JOGA

(#Codigo_Jogador,#Codigo_Partida,Numero_Gols)

codigo_jogador → Integer
    Se apagar jogador, apaga.
codigo_partida → Integer
    Se apagar partida, apaga.
numero_gols → Integer

SQL Básica: Exemplo

JOGA

(#Codigo_Jogador,#Codigo_Partida,Numero_Gols)

```
CREATE TABLE joga (
  codigo_jogador integer,
  constraint joga1_fk foreign key (codigo_jogador)
      references jogador(codigo_jogador)
      on delete cascade,
  codigo_partida integer,
  constraint joga2_fk foreign key (codigo_partida)
      references partida(codigo_partida)
      on delete cascade,
  numero_gols integer
);
```

SQL Básica: Alteração de Tabelas

```
ALTER TABLE [esquema.]tabela
[add coluna tipo_dado [DEFAULT expr]
  [constraint_coluna] ]
[modify coluna tipo_dado [DEFAULT expr]
  [constraint_coluna] ]
[add constraint_coluna/constraint_tabela]
[drop constraint_coluna/constraint_tabela
  [cascade]]
[enable constraint_coluna/constraint_tabela]
[disable constraint_coluna/constraint_tabela]
  ;
```

Exemplo

- Alterar a tabela joga para que o valor default de Numero_gols seja 0;

```
ALTER TABLE joga MODIFY numero_gols
  INTEGER DEFAULT '0';
```

- Adicionar uma Primary Key em joga

```
ALTER TABLE joga ADD constraint
  joga_pk Primary Key
  (codigo_jogador, codigo_partida);
```

SQL Básica: Apagando Tabelas

```
DROP TABLE [esquema.]tabela [CASDADE
  CONSTRAINTS];
```

- CASCADE CONSTRAINTS elimina todas as restrições presentes em outras tabelas que façam referência à tabela que está sendo eliminada.

SQL Básica: Índices

- Criar

```
CREATE [UNIQUE] INDEX índice ON
  tabela (coluna [ASC | DESC]);
```

- UNIQUE → Índice não aceita valores repetidos.
- É criado um índice UNIQUE sempre que uma Primary Key é criada.
- Apagando o índice

```
DROP INDEX índice;
```

Exemplo

```
select index_name from user_indexes;

CREATE UNIQUE INDEX my_index ON partida
  (nome_juiz);

select index_name from user_indexes;
```

SQL Básica: Linguagem de manipulação de dados

- Inserindo dados

```
INSERT INTO [esquema.]tabela
(coluna1, coluna2, ... colunaN)
VALUES
(valor1, valor2 ... valorN);
```

SQL Básica: Inserindo dados

- Se for inserir na mesma ordem da definição da tabela:

```
INSERT INTO [esquema.]tabela VALUES
  (valor1, ... valorN);
```

- Inserção em determinados campos

```
INSERT INTO [esquema.]tabela (colunaX,
  colunaY) VALUES (valorX, valorY);
```

Exemplo

- Criar o time Saravá Saci Soccer

```
INSERT INTO equipe
  (codigo_equipe,nome_equipe,cidade
  ,estado) VALUES (1,'SSS','São
  Carlos', 'São Paulo');
```

Opção

```
INSERT INTO equipe VALUES (1,
  'SSS', 'São Carlos', 'São
  Paulo');
```

Exemplo

- Inserir time “Tiradentes”, de Brasília, DF

```
INSERT INTO equipe VALUES
  (2,'Tiradentes','Brasilia','DF');
```

- Inserir time “Enc97FC”

```
INSERT INTO equipe (codigo_equipe,
  nome_equipe) VALUES
  (3,'Enc97FC');
```

SQL Básica: Inserindo dados

- Cuidados com as restrições (constraints)!

```
INSERT INTO JOGADOR VALUES
  (1,'Juliano','goleiro',4);
INSERT INTO equipe (codigo_equipe,
  nome_equipe) VALUES (4, 'Selecao');
INSERT INTO JOGADOR VALUES
  (1,'Juliano','goleiro',4);
```

SQL Básica: Atualizando dados

```
UPDATE tabela SET coluna = valor [, coluna
  = valor...] [WHERE condição];
```

- Exemplo: Juliano mudou para atacante!

```
UPDATE jogador SET posicao_jog = 'atacante'
  where codigo_jogador = 1;
```

SQL Básica: Apagando dados

```
DELETE [FROM] tabela
[WHERE condição];
```

- Exemplo: Apagar Enc97FC

```
DELETE FROM equipe WHERE codigo_equipe = 3;
```

SQL Básica: Apagando dados

- Cuidados com as restrições (constraints)!

```
DELETE FROM equipe WHERE codigo_equipe = 4;
```

FALHA!

```
DELETE jogador;
```

```
DELETE FROM equipe WHERE codigo_equipe = 4;
```

OK!

SQL Básica: Selecionando dados

- Forma básica:

SELECT <lista de atributos>

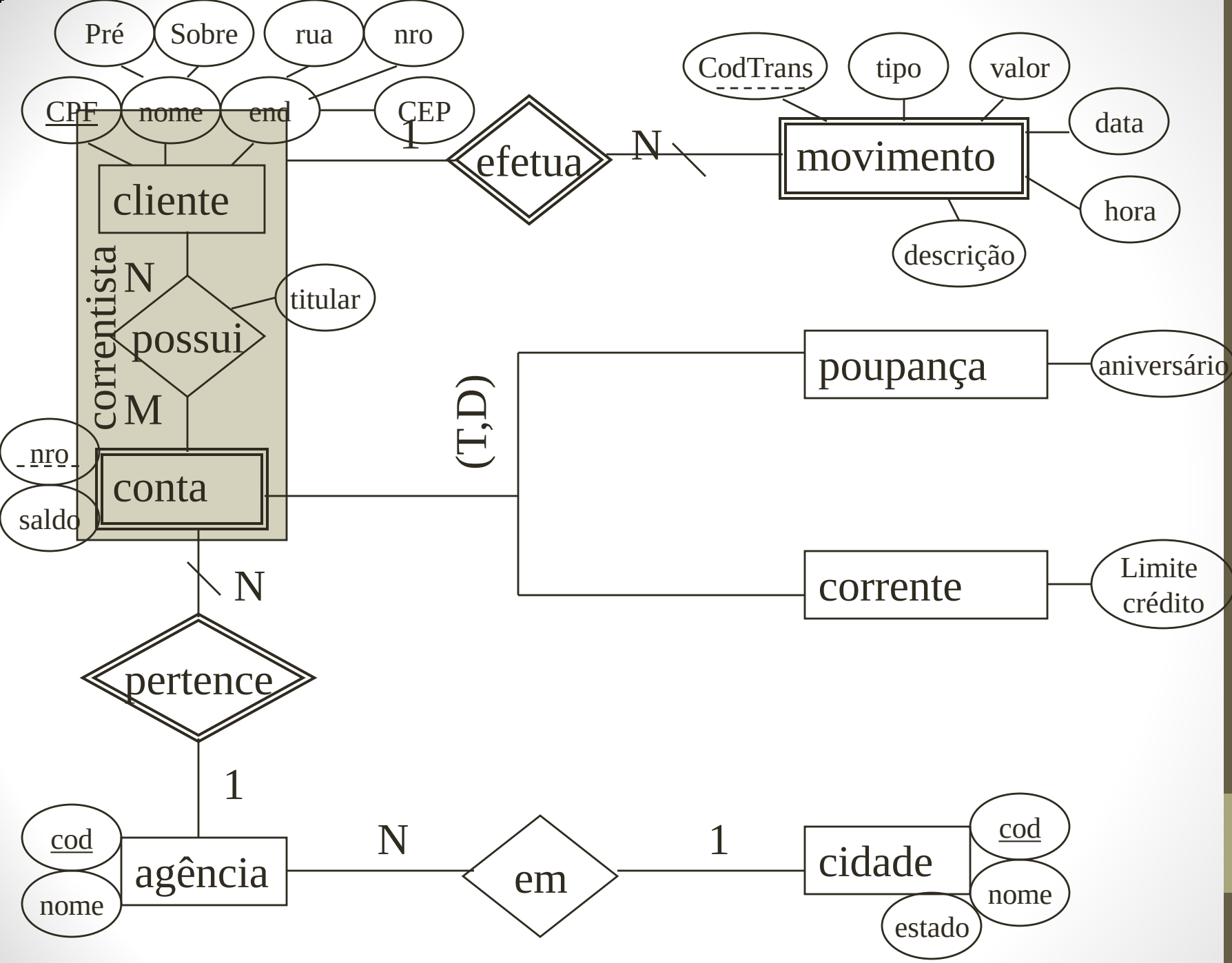
FROM <lista de tabelas>

WHERE <condição>

SQL Básica: Selecionando dados

```
SELECT [DISTINCT] {*, colunas
[AS alias], expressões, funções..}
FROM {tabelas [AS alias]}
[WHERE condição]
[GROUP BY colunas]
[HAVING condição]
[ORDER BY colunas [ASC | DESC]];
```

Controle Bancário


Pré
Sobre
rua
nro
CPF
nome
end
CEP
cliente
correntista
1
efetua
N
CodTrans
tipo
valor
movimento
data
hora
descrição
N
possui
M
titular
nro
conta
saldo
(T,D)
poupança
aniversário
corrente
Limite crédito
N
pertence
1
cod
agência
nome
N
em
1
cidade
cod
nome
estado

Tarefas

- Efetuar o mapeamento para um esquema relacional
- Fazer scripts para criação de tabelas no Oracle
- Fazer scripts para alimentar o banco de dados

Tarefas

- Efetuar as seguintes consultas nas relações:
  - Listar as contas corrente conjuntas e seus respectivos correntistas onde o titular da conta tem o sobrenome “Silva”
  - Listar a movimentação (descrição, tipo, valor, data) da(s) conta(s) corrente cujo titular é “André Silva” no período entre 01/03/2004 e 26/03/2004.