Consulta a Bancos de Dados Relacionais: SQL – Structured Query Language

Disciplina Bancos de Dados 1 (INE 5613 – 2006-1)

Curso de Sistemas de Informação

Prof. Renato Fileto

INE/CTC/UFSC

Tópicos

- Histórico
- Descrição e uso da SQL
- Resumo dos comandos de manipulação de dados
- Consultas em SQL

Histórico

1970	Codd publica "A Relational Model of Data Large Shared Banks" com os princípios para o modelo relacional.
1974	Chamberlin e outros definem a *SEQUEL* (*Structured English Query Language*) no "*IBM San Jose Research Center*".
1975	Primeiro Protótipo IBM (*SEQUEL-XRM*).
1976	Revisão de *SEQUEL* ⇒ *SEQUEL/2* (posteriormente, SQL).
1977	Sistema R, implementando *SEQUEL/2*, torna-se operacional.
1977	*Oracle* é lançado no mercado.

Histórico (continuação)

1983	IBM lança DB2. Outros produtos relacionais lançados no mercado (Sybase, Ingres, etc). SQL é um padrão "di facto".
1986	SQL torna-se um padrão ANSI para linguagem relacional.
1987	Padrão ANSI para SQL é aceito pela ISO (SQL/86).
1989	SQL incorpora características de reforço de integridade (SQL/89).
1992	Comitês ISO e ANSI apresentam SQL2 (SQL/92).
1999	SQL 3

Descrição da *SQL*

Linguagem para acesso a Sistemas Gerenciadores de Bases de Dados Relacionais (padrão para diversos produtos comerciais).

Uma base de dados é entendida como uma coleção de tabelas (modelo relacional).

Recursos:

- Manipulação e controle de dados (*DML*);
- Definição de bases de dados (*DDL*).

Utilização da *SQL*

Formas de uso:

- Interativamente;
- Embutida em linguagem de programação (PL/I, Cobol, C, etc).

Tipos de tabelas:

- Tabelas básicas: Têm existência própria.
- Tabelas virtuais (visões): São definidas a partir de outras tabelas (básicas e/ou virtuais).

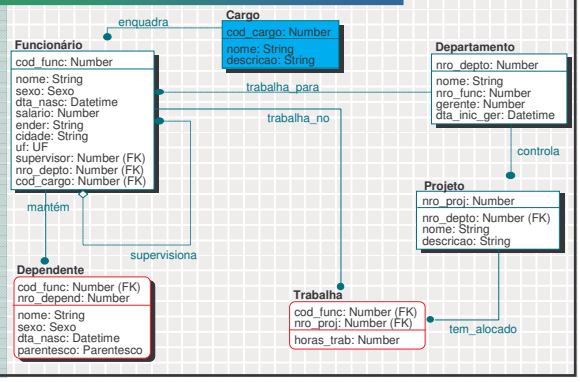
Comandos *SQL*

- **Definição do Esquema do Banco de Dados**
  - CREATE TABLE — definição de tabelas
  - DROP TABLE — remoção de tabelas
  - ALTER TABLE — alteração de tabelas
  - CREATE... INDEX — ... índices
  - CREATE... VIEW — ... visões
- **Especificação de restrições de consistência, integridade e segurança**

Comandos *SQL* (continuação)

- **Manipulação de Dados**
  - SELECT — consulta
  - INSERT — inserção de registros
  - DELETE — remoção de registros
  - UPDATE — atualização de registros

Banco de Dados para Consultas



Consultas em SQL

Instrução SELECT:

SELECT <lista_colunas>
FROM <lista_tabelas>
WHERE <condição>

Equivalente em álgebra relacional:

$$\pi_{\text{<lista\_colunas>}} (\sigma_{\text{<condição>}} (X_{\text{<lista\_tabelas>}}))$$

Exemplos:

```
SELECT  nome, ender
FROM  funcionario
WHERE  cidade = "Campinas" AND sexo = 'M';
```

Consultas em SQL (continuação)

```
SELECT  departamento.nome, funcionario.nome
FROM  funcionario, departamento
WHERE  funcionário.cod_depto = departamento.cod_depto;

SELECT  D.nome AS nome_depto, F.nome AS nome_func
FROM  funcionario F, departamento D
WHERE  F.nro_depto = D.nro_depto;

SELECT  *
FROM  funcionario
WHERE  salario >= 1800 ;

SELECT  D.*,  F.nome
FROM funcionario F, departamento D
WHERE F.nro_depto = D.nro_depto;
```

Junção

```
SELECT F.nome AS Funcionario, D.nome AS Dependente
FROM  funcionario F INNER JOIN dependente D
                    ON F.cod_func = D.nro_func;

SELECT F.nome AS Funcionario, D.nome AS Dependente
FROM  funcionario F LEFT JOIN dependente D
                    ON F.cod_func = D.nro_func;
```

Tipos de junção:

- INNER JOIN
- LEFT JOIN
- RIGHT JOIN

Conjuntos em SQL

```
SELECT  ALL  salario            (default)
FROM  funcionario;

SELECT  DISTINCT  salario
FROM  funcionario;
```

Operação UNION:

```
SELECT  nome FROM  funcionario UNION
SELECT  nome FROM  dependente;
```

Operação IN:

```
SELECT  nome FROM  funcionario
WHERE uf  IN ("PR", "SC", "RS");
```

União

```
SELECT  F.nome, F.endereco
FROM  funcionario F, departamento D
WHERE F.nro_depto = D.nro_depto
UNION
SELECT  F.nome, F.endereco
FROM  funcionario F, trabalha T
WHERE F.cod_func = T.cod_func;
```

Obs.: A operação de união requer que os argumentos tenham o mesmo esquema

Valores Nulos

```
SELECT  nome
FROM  funcionario
WHERE ender IS NULL ;
```

Consultas aninhadas

```
SELECT  *
FROM  funcionario
WHERE cod_func IN ( SELECT cod_func
                    FROM dependente ) ;

SELECT  *
FROM  funcionario F
WHERE  EXISTS  (    SELECT *
                    FROM dependente
                    WHERE cod_func = F.cod_func ) ;

SELECT  *
FROM  funcionario F
WHERE  UNIQUE  (    SELECT *
                    FROM dependente
                    WHERE cod_func = F.cod_func ) ;
```

Consultas aninhadas (continuação)

```
SELECT  nome
FROM  funcionario F
WHERE (cod_func, nro_depto) NOT IN
                        (    SELECT cod_func, nro_depto
                             FROM trabalha T, projeto P
                             WHERE  T.nro_proj = P.nro_proj  ) ;

SELECT  nome
FROM  funcionario
WHERE cod_func NOT IN (    SELECT cod_func
                           FROM trabalha )
      AND
      nro_depto NOT IN (   SELECT nro_depto
                           FROM projeto );
```

Operadores de Comparação e Ordenação

```
SELECT  nome
FROM  funcionario F
WHERE   salario  >=  ALL  ( SELECT salario
                            FROM funcionario ) ;

SELECT  nome
FROM  funcionario S
WHERE  salario  <  ANY  (   SELECT salario
                            FROM funcionario
                            WHERE supervisor = S.cod_func) ;
```

Ordenação

```
SELECT  nome, salario
FROM  funcionario F
ORDER BY salario DESC, nome ASC ;          (ASC é default)
```

Funções de Agregação e Agrupamentos

```
SELECT  COUNT(*)
FROM funcionario;

SELECT SUM(salario), MIN(salario), MAX(salario), AVG(salario)
FROM funcionario;

SELECT  COUNT(*) , AVG(salario)
FROM funcionario F, departamento D
WHERE F.nro_depto = D.nro_depto AND D.nome="Pesquisa";

SELECT  D.nome, SUM(salario), AVG(salario), COUNT(*)
FROM funcionario F, departamento D
WHERE F.nro_depto = D.nro_depto
GROUP BY D.nome;
```

Funções de Agregação e Agrupamentos (continuação)

```
SELECT  F.nome, count(*)
FROM funcionario F, trabalha T
WHERE  F.cod_func = T.cod_func
GROUP BY F.nome HAVING COUNT(*) > 3;

SELECT  D.nome, count(*)
FROM funcionario F, departamento D
WHERE F.nro_depto = D.nro_depto AND
        F.salario > (SELECT avg(salario) FROM funcionario)
GROUP BY D.nome HAVING COUNT(*) > 3;

SELECT  D.nome, AVG(salario) / (  SELECT AVG(salario)
                                  FROM funcionario )
FROM funcionario F, departamento D
WHERE F.nro_depto = D.nro_depto
GROUP BY D.nome
ORDER BY AVG(salario) DESC ;
```

Síntese da Instrução SELECT

Utilizada para realizar consultas em um BD

Formato:

```
SELECT  [ ALL | DISTINCT ] <coluna1>, <coluna2>, ...
FROM <tabela1> , <tabela2>, ...
[ WHERE <condição> ]
[ GROUP BY <colunaX>, <colunaY>, ...
    [HAVING <restrição-grupo>] ]
[ ORDER BY <colunaA>, <colunaB>, ... [ ASC | DESC ] ]
```

Síntese da Instrução SELECT (cont. 1)

Operadores aritméticos

+, -, *, /, ...

Funções de agregação

COUNT:	Conta o número de valores na coluna.
SUM:	Soma dos valores na coluna.
AVG:	Calcula a média dos valores na coluna.
MAX:	Encontra o maior valor na coluna.
MIN:	Encontra o menor valor na coluna.

Síntese da Instrução SELECT (cont. 2)

Operadores de comparação

>, <, =, <=, >=, <>

Operadores sobre relações

IN, EXISTS, UNIQUE

Comparação de valores com elementos de conjuntos

ANY (SOME), ALL