

SQL - Introdução

- A linguagem SQL (*Structured Query Language* – Linguagem de Consulta Estruturada) representa um conjunto de comandos responsáveis pela definição das tabelas, campos e atualização dos dados em um SGBD.
- É uma linguagem comercial de definição e manipulação de banco de dados relacional.
- A linguagem SQL (inicialmente chamada SEQUEL) surgiu no centro de pesquisa de San Jose da IBM, dentro do projeto System R (anos 70).
- SQL é *padrão de fato* (praticamente todos SGBDs oferecem uma interface SQL).
- SQL é *padrão de direito* (ISO):
 - SQL1 aprovado em 1986, com modificações em 1989
 - SQL2 aprovado em 1992 (mais amplamente aceita)
 - SQL3 aprovado em 1999

SQL - Padronização

- SQL é uma linguagem completa de manipulação de banco de dados que oferece funcionalidades de criação, alteração e consulta dos dados.
- Os comandos existentes na linguagem são subdivididos em dois grupos:
 - DDL (Data Definition Language): Usada para definição do esquema da base de dados. É o conjunto de comandos responsáveis pela criação, alteração e deleção da estrutura das tabelas e índices de um sistema.
 - DML (Data Manipulation Language): Usada para programação de consultas e transações que inserem, removem e alteram linhas de tabelas. É o conjunto de comandos responsáveis pela consulta e atualização dos dados armazenados em um banco de dados.

SQL - Padronização

■ Tipos de dados mais comuns:

- INTEGER ou INT (-32768 a 32767)
- SMALLINT (0 a 65535)
- FLOAT (-2.147.483.648 a 2.147.483.647),
- REAL (1.17E-38 a 3.4E+38)
- DOUBLE PRECISION (2.2E-308 a 1.7E+308)
- DECIMAL (i,j) onde i é o total de casas decimais e j é o total de casas decimais depois do ponto (até 18)
- CHAR (n) onde n é a quantidade *fixa* de caracteres (até 60.000)
- VARCHAR (n) onde n é a quantidade *máxima* de caracteres (até 60000)
- BIT(n) onde n é a quantidade *fixa* de bits
- BIT VARYING(n) onde n é a quantidade *máxima* de bits
- DATE no formato aaaa-mm-dd
- TIME no formato hh:mm:ss

SQL - Padronização

■ Operadores:

| Lógicos | Aritméticos | | Relacionais | |
|--------------|-------------|---------------|-------------|--------|
| AND | + | < <= | | |
| OR | - | > >= | | |
| NOT | * | <> = | | |
| | / | LIKE, BETWEEN | | |
| Conjunturais | | | | |
| =ANY | <ANY | <ALL | EXISTS | IN |
| >ANY | >=ANY | >ALL | NOT EXISTS | NOT IN |
| <=ANY | <>ANY | <>ALL | | |

SQL - Padronização

■ Funções mais comuns:

- AVG - obtém o valor médio de uma coluna
- COUNT - obtém a soma da quantidade de linhas analisadas
- MAX - obtém o maior valor de uma coluna
- MIN - obtém o menor valor de uma coluna
- SUM - obtém a soma de valores de uma coluna

SQL - Exemplo

- Para ilustrar o assunto será adotado o seguinte exemplo (UFSCar):
A empresa construtora de veículos especiais "Star Trek" necessita armazenar, em seu Banco de Dados, informações sobre as peças que utiliza em cada projeto de veículo e os fornecedores dessas peças. O Banco de Dados deve ser capaz de oferecer respostas sobre peças, fornecedores e projetos realizados, bem como associações entre esses elementos, ou seja, a quantidade de peças fornecida por uma determinada empresa e utilizadas em um projeto.
As PEÇAS são identificadas por um número, sendo utilizada a cor das gavetas onde estão colocadas as peças para uma localização visual mais rápida por parte dos funcionários. Para a especificação das compras são necessários o nome e o preço atual de cada peça.
Para a emissão correta das notas de compra e dos pagamentos é necessário o nome dos FORNECEDORES bem como a cidade e um número de identificação. A categoria de cada fornecedor é utilizada para indicar a qualidade de seus produtos e serviços.
OS PROJETOS de veículos construídos pela "Star Trek" possuem um nome, número de identificação, a duração para a montagem do veículo e o custo de cada veículo (incluindo peças e serviços).

SQL - Exemplo

- Considere que o seguinte esquema relacional foi desenvolvido:

Peça (PeNro, PeNome, PePreço, PeCor)

Fornecedor (FNro, FNome, FCidade, FCateg)

Projeto (PNro, PNome, PDuração, PCusto)

Fornece_Para (PeNro, FNro, PNro, Quant)

PeNro → Peça

Fnro → Fornecedor

Pnro → Projeto

SQL - Exemplo

- Considere também as tabelas a seguir, que mostram instâncias do banco de dados em um determinado momento.

- **Peça:**

| <u>PeNro</u> | PeNome | PePreço | PeCor |
|--------------|----------|---------|----------|
| PE1 | Cinto | 22 | Azul |
| PE2 | Volante | 18 | Vermelho |
| PE3 | Lanterna | 14 | Preto |
| PE4 | Limpador | 09 | Amarelo |
| PE5 | Painel | 43 | Vermelho |

SQL - Exemplo

■ Fornecedor:

| <u>FNro</u> | FNome | FCidade | FCateg |
|-------------|------------|------------|--------|
| F1 | Plastec | Campinas | B |
| F2 | C&M | São Paulo | D |
| F3 | Kirurgic | Campinas | A |
| F4 | Piloto's | Piracicaba | A |
| F5 | Equipament | São Carlos | C |

SQL - Exemplo

■ Projeto:

| <u>PNro</u> | PNome | PDuração | PCusto |
|-------------|---------|----------|--------|
| P1 | Detroit | 5 | 43.000 |
| P2 | Pegasus | 3 | 37.000 |
| P3 | Alfa | 2 | 26.700 |
| P4 | Sea | 3 | 21.200 |
| P5 | Paraíso | 1 | 17.000 |

SQL - Exemplo

■ Fornece_Para:

| <u>PeNro</u> | <u>FNro</u> | <u>PNro</u> | <u>Quant</u> |
|--------------|-------------|-------------|--------------|
| PE1 | F5 | P4 | 5 |
| PE2 | F2 | P2 | 1 |
| PE3 | F3 | P4 | 2 |
| PE4 | F4 | P5 | 3 |
| PE5 | F1 | P1 | 1 |
| PE2 | F2 | P3 | 1 |
| PE4 | F3 | P5 | 2 |

SQL

- SQL/2 não oferece instruções para criação de Bancos de Dados
Alguns produtos (SQL/Server) têm instruções de DDL:

- Create Database - cria uma base de dados vazia
- Drop Database - elimina uma base de dados

Outros têm abordagens variadas:

- Oracle cria o BD como parte da instalação do software
- INGRES tem um utilitário

- SQL oferece três instruções para definição do esquema:

- Create Table - define a estrutura de uma tabela, suas restrições de integridade e cria a tabela vazia
- Drop Table - elimina a tabela da base de dados
- Alter Table - permite modificar a definição de uma tabela

SQL – Comandos DDL

Create Table

- Objetivo: Criar a estrutura de uma tabela definindo as colunas (campos) e as chaves primárias e estrangeiras existentes.

Sintaxe:

```
CREATE TABLE <nome-tabela>
(<nome-coluna> <tipo-do-dado> [NOT NULL]
                                [NOT NULL WITH DEFAULT] )
PRIMARY KEY (nome-coluna-chave)
FOREIGN KEY (nome-coluna-chave-estrangeira) REFERENCES
(nome-tabela-pai) ON DELETE [RESTRICT]
                                [CASCADE]
                                [SET NULL]
```

SQL – Comandos DDL

onde:

- nome-tabela - Representa o nome da tabela que será criada.
- nome-coluna - Representa o nome da coluna que será criada. A definição das colunas de uma tabela é feita relacionando-as em uma lista.
- tipo-do-dado - Cláusula que define o tipo e tamanho dos campos definidos para a tabela.
- NOT NULL - Exige o preenchimento do campo, ou seja, no momento da inclusão é obrigatório que possua um conteúdo.
- NOT NULL WITH DEFAULT - Preenche o campo com valores pré-definidos, de acordo com o tipo do campo, caso não seja especificado o seu conteúdo no momento da inclusão do registro. Os valores pré-definidos são:
 - Campos numéricos - Valor zero.
 - Campos alfanuméricos - Caracter branco.
 - Campo formato Date - Data corrente.
 - Campo formato Time - Horário no momento da operação.

SQL – Comandos DDL

- PRIMARY KEY (nome-coluna-chave) - Define a coluna que será a chave primária da tabela. Caso haja mais de um coluna como chave, elas deverão ser relacionadas entre os parênteses.
- FOREIGN KEY (nome-coluna-chave-estrangeira) REFERENCES (nome-tabela-pai) - Define as colunas que são chaves estrangeiras. No campo REFERENCES deve ser especificada a tabela na qual a coluna é chave primária.
- ON DELETE - Esta opção especifica os procedimentos do SGBD quando houver uma exclusão de um registro na tabela pai quando existe um registro correspondente nas tabelas filhas. As opções disponíveis são:
 - RESTRICT - Esta opção não permite a exclusão na tabela pai de um registro cuja chave primária exista em alguma tabela filha.
 - CASCADE - Esta opção realiza a exclusão em todas as tabelas filhas que possua o valor da chave que será excluída na tabela pai.
 - SET NULL - Esta opção atribui o valor NULO nas colunas das tabelas filhas que contenha o valor da chave que será excluída na tabela pai.

SQL – Comandos DDL

■ Exemplos:

Peca (PeNro, PeNome, PePreço, PeCor)

```
CREATE TABLE Peca (  
    PeNro          CHAR(5)          NOT NULL,  
    PeNome         CHAR(30)         NOT NULL,  
    PePreco        FLOAT            NOT NULL,  
    PeCor          CHAR(20),  
    PRIMARY KEY (PeNro) );
```

Fornecedor (FNro, FNome, FCidade, FCateg)

```
CREATE TABLE Fornecedor (  
    FNro          CHAR(5)          NOT NULL PRIMARY KEY,  
    FNome         CHAR(30)         NOT NULL,  
    FCidade       CHAR(20)         NOT NULL,  
    FCateg        CHAR(1) );
```


SQL – Comandos DDL

Alter Table - Altera a estrutura de uma tabela acrescentando, alterando, retirando e/ou alterando nomes, formatos das colunas e integridade.

Sintaxe:

ALTER TABLE <nome-tabela>

DROP <nome-coluna>

ADD <nome-coluna> <tipo-do-dado> [NOT NULL/NOT NULL WITH DEFAULT]

RENAME <nome-coluna> <novo-nome-coluna>

RENAME TABLE <novo-nome-tabela>

MODIFY <nome-coluna> <tipo-do-dado> [NULL/NOT NULL/NOT NULL WITH DEFAULT]

ADD PRIMARY KEY <nome-coluna>

DROP PRIMARY KEY <nome-coluna>

ADD FOREIGN KEY (nome-coluna-chave-estrangeira) **REFERENCES**
(nome-tabela-pai) **ON DELETE** [RESTRICT/CASCADE/SET NULL]

DROP FOREIGN KEY (nome-coluna-chave-estrangeira) **REFERENCES**
(nome-tabela-pai)



SQL – Comandos DDL

onde:

- nome-tabela - Representa o nome da tabela que será atualizada.
- nome-coluna - Representa o nome da coluna que será criada.
- tipo-do-dado - Cláusula que define o tipo e tamanho dos campos definidos para a tabela.
- DROP <nome-coluna> - Realiza a retirada da coluna especificada na estrutura da tabela.
- ADD <nome-coluna> <tipo-do-dado> - Realiza a inclusão da coluna especificada. Na coluna correspondente a este campo nos registros já existentes será preenchido o valor NULL (Nulo). As definições NOT NULL e NOT NULL WITH DEFAULT são semelhantes à do comando CREATE TABLE.
- RENAME <nome-coluna> <novo-nome-coluna> - Realiza a troca do nome da coluna especificada.
- RENAME TABLE <novo-nome-tabela> - Realiza a troca do nome da tabela especificada.



SQL – Comandos DDL

- MODIFY <nome-coluna> <tipo-do-dado> - Permite a alteração na característica da coluna especificada.

Opções:

- Além das existentes na opção ADD (NOT NULL e NOT NULL WITH DEFAULT), a opção NULL altera a característica do campo passando a permitir o preenchimento com o valor Nulo.
 - ADD PRIMARY KEY <nome-coluna> - Esta opção é utilizada quando é acrescentado um novo campo como chave primária da tabela.
 - DROP PRIMARY KEY <nome-coluna> - Esta opção é utilizada quando é retirado um campo como chave primária da tabela.
 - ADD FOREIGN KEY <nome-coluna> - Esta opção é utilizada quando é acrescentada uma nova chave estrangeira.
 - DROP FOREIGN KEY <nome-coluna> - Esta opção é utilizada quando é retirada uma chave estrangeira da estrutura da tabela.

SQL – Comandos DDL

■ Exemplo:

Peca (PeNro, PeNome, PePreço, PeCor)

```
CREATE TABLE Peca (  
    PeNro          CHAR(5)          NOT NULL,  
    PeNome         CHAR(30)         NOT NULL,  
    PePreco        FLOAT            NOT NULL,  
    PeCor          CHAR(20),  
    PRIMARY KEY (PeNro) );
```

```
ALTER TABLE Peca  
    ADD COLUMN material CHAR(20);
```

SQL – Comandos DDL

Drop Table

- Objetivo: Apagar a estrutura e os dados existentes em uma tabela. Após a execução deste comando estarão deletados todos dados, estrutura e índices de acessos que estejam a ela associados.

Sintaxe:

DROP TABLE <nome-tabela>

onde:

- nome-tabela - Representa o nome da tabela que será apagada.

SQL – Comandos DDL

Exemplo:

Peca (PeNro, PeNome, PePreço, PeCor)

```
CREATE TABLE Peca (  
    PeNro          CHAR(5)          NOT NULL,  
    PeNome         CHAR(30)         NOT NULL,  
    PePreco        FLOAT            NOT NULL,  
    PeCor          CHAR(20),  
    PRIMARY KEY (PeNro) );
```

DROP TABLE Peca

SQL – Exercício

1. Faça um script SQL de criação de tabelas para o banco de dados abaixo (defina os tipos de dados conforme sua interpretação do contexto). Após a criação, insira o campo Codigo como chave para a tabela Docente, retire o campo HsSem da tabela Materia, renomeie o campo Categoria da tabela Docente para "Cargo" e apague a tabela Uso, que não será mais utilizada.

Docente (Professor, Categoria)

Materia (Disciplina, QtdeAlunos, HsSem)

CompraLivro (Professor, Disciplina, ISBN, Uso)

Professor → Docente

Disciplina → Materia

ISBN → Livro

Uso → Utilidade

Livro (ISBN, Titulo, Autor, Preco)

Utilidade (Uso, Qtde)

SQL – Comandos DML

Insert

- Objetivo: Incluir um novo registro em uma tabela do Banco de Dados.

Sintaxe:

INSERT INTO <nome-tabela> [(<nome-coluna>, [<nome-coluna>])]

VALUES (<relação dos valores a serem incluídos>)

onde:

- nome-tabela - Representa o nome da tabela onde será incluído o registro.
- nome-coluna - Representa o nome da(s) coluna(s) que receberão conteúdo no momento da operação de inclusão.

SQL – Comandos DML

Este comando pode ser executado de duas maneiras:

- 1) Quando todos os campos da tabela terão conteúdo - Neste caso não é necessário especificar as colunas, entretanto a relação dos valores a serem incluídos deverão obedecer a mesma sequência da definição da tabela.
- 2) Quando apenas parte dos campos da tabela terão conteúdo - Neste caso devem ser especificadas todas as colunas que terão conteúdo e os valores relacionados deverão obedecer esta sequência. Para os campos que não têm conteúdo especificado será preenchido o valor NULL.

SQL – Comandos DML

Update

- Objetivo: Atualiza os dados de um registro ou de um grupo de registros em uma tabela do Banco de Dados .

Sintaxe:

UPDATE <nome-tabela>

SET <nome-coluna> = <novo conteúdo para o campo>,
[<nome-coluna> = <novo conteúdo para o campo>]

WHERE <condição>

onde:

- nome-tabela- Representa o nome da tabela cujo conteúdo será alterado.
- nome-coluna- Representa o nome da(s) coluna(s) terão seus conteúdos alterados com o novo valor especificado.
- condição- Representa a condição para a seleção dos registros que serão atualizados. Esta seleção poderá resultar em um ou vários registros. Neste caso a alteração irá ocorrer em todos os registros selecionados.

SQL – Comandos DML

Delete

- Objetivo: Apaga um registro ou um grupo de registros em uma tabela do Banco de Dados.

Sintaxe:

DELETE FROM <nome-tabela>
WHERE <condição>

onde:

- nome-tabela - Representa o nome da tabela cujos registros serão apagados.
- condição - Representa a condição para a deleção dos registros. Esta seleção poderá resultar em um ou vários registros. Neste caso a operação irá ocorrer em todos os registros selecionados.

SQL – Comandos DML

Select

- Objetivo: Seleciona um conjunto de registros em uma ou mais tabelas que atenda a uma determinada condição definida pelo comando. O comando SELECT ... FROM ... WHERE ... possibilita consultar uma ou mais tabelas de acordo com critérios estabelecidos.

Sintaxe:

SELECT <nome-coluna> [, <nome-coluna>]
FROM <nome-tabela> [, <nome-tabela>]
WHERE <condição>
GROUP BY <nome-coluna>
HAVING <condição>
ORDER BY <nome-campo> ASC/DESC

SQL – Comandos DML

onde:

- nome-coluna – Representa o nome da(s) coluna(s) envolvida(s) na consulta.
- nome-tabela - Representa o nome da(s) tabela(s) que contem as colunas que serão utilizadas para execução da consulta.
- condição - Representa a condição para a seleção dos registros. Este seleção poderá resultar em um ou vários registros.
- nome-coluna - Representa a(s) coluna(s) cujos resultados são agrupados para atender à consulta.
- WHERE - Especifica o critério de seleção dos registros nas tabelas.
- GROUP BY - Especifica o(s) campo(s) que serão agrupados na consulta.
- HAVING - Especifica uma condição para seleção de um grupo de dados. Esta opção só é utilizada combinada com a opção GROUP BY.
- ORDER BY - Esta opção quando utilizada apresenta o resultado da consulta ordenado de forma crescente ou decrescente pelos campos definidos.

SQL – Comandos DML

Resumindo as Cláusulas:

- | | |
|----------|---|
| SELECT | - <u>o que</u> se deseja na tabela de resultado |
| FROM | - <u>de onde</u> retirar os dados necessários |
| WHERE | - <u>condições para busca</u> dos resultados |
| GROUP BY | - agrupamento de resultados |
| HAVING | - condições para a definição de grupos no resultados |
| ORDER BY | - estabelece a ordenação lógica da tabela de resultados |

SQL – Comandos DML – Exemplos

Exemplo 1: Obtenha os nomes de todas as peças

```
SELECT  PeNome
FROM    Peca
```

| PeNome |
|----------|
| Cinto |
| Volante |
| Lanterna |
| Limpador |
| Painel |

Exemplo 2 (Where): Obtenha o nome e código dos fornecedores de Campinas

```
SELECT  FNome, FNro
FROM    Fornecedor
WHERE   FCidade = 'Campinas'
```

| FNome | FNro |
|----------|------|
| Plastec | F1 |
| Kirurgic | F3 |

SQL – Comandos DML – Exemplos

Exemplo 3 (Operador Aritmético): Obtenha o nome e a duração em dias de cada projeto

```
SELECT  PNome, (PDuracao * 30) as
Dur_Dias
FROM    Projeto
```

| PNome | Dur_Dias |
|---------|----------|
| Detroit | 150 |
| Pegasus | 90 |
| Alfa | 60 |
| Sea | 90 |
| Paraíso | 30 |

Exemplo 4 (Operador Relacional): Obtenha o nome dos projetos de custo menor que \$28000

```
SELECT  PNome
FROM    Projeto
WHERE   PCusto < 28000
```

| PNome |
|---------|
| Alfa |
| Sea |
| Paraíso |

SQL – Comandos DML – Exemplos

Exemplo 5 (Operador Lógico): Obtenha os nomes das peças de cor vermelha e com preço maior que \$ 25

```
SELECT  PeNome
FROM    Peca
WHERE   (PeCor = 'Vermelho'
        AND  PePreco > 25)
```

| PeNome |
|--------|
| Painel |

SQL – Comandos DML – Exemplos

Exemplo 6 (Uso explícito da relações): Obtenha o código dos fornecedores para o projeto P5

```
SELECT  Fornece_para.FNro
FROM    Fornece_para
WHERE   Fornece_para.PNro = 'P5'
```

| FNro |
|------|
| F4 |
| F3 |

Exemplo 7 (Variáveis): Obtenha o código dos fornecedores concorrentes

```
SELECT  X.FNro, Y.FNro
FROM    Fornece_para AS X, Fornece_para AS Y
WHERE   (X.PeNro = Y.PeNro
        AND  X.FNro <> Y.FNro )
```

| X.FNro | Y.FNro |
|--------|--------|
| F4 | F3 |
| F3 | F4 |

SQL – Comandos DML – Exemplos

Exemplo 8 (Variáveis): Obtenha o código das peças de mesma cor

```
SELECT  X.PeNro, Y.PeNro
FROM    Peca AS X, Peca AS Y
WHERE   (X.PeCor = Y.PeCor
        AND X.PeNro <> Y.PeNro )
```

| X.PeNro | Y.PeNro |
|---------|---------|
| PE2 | PE5 |
| PE5 | PE2 |

SQL – Comandos DML – Exemplos

Operador UPPER

Exemplo 9 (Operador Relacional): Obtenha a duração e o custo do projeto de nome Pegasus.

```
SELECT  PDuracao, PCusto
FROM    Projeto
WHERE   PNome = "Pegasus"
```

| PDuracao | PCusto |
|----------|--------|
| ----- | ----- |

Utilizando UPPER:

```
SELECT  PDuracao, PCusto
FROM    Projeto
WHERE   Pnome = UPPER("Pegasus")
```

| PDuracao | PCusto |
|----------|--------|
| 3 | 37000 |

SQL – Comandos DML – Exemplos

Operador BETWEEN ...AND ...

Exemplo 10 (Between): Obtenha o nome dos projetos com preço entre \$20000 e \$30000

```
SELECT PNome
FROM Projeto
WHERE PCusto BETWEEN 20000 AND 30000
```

| PNome |
|-------|
| Alfa |
| Sea |

SQL – Comandos DML – Exemplos

Operador IN

Exemplo 11 (In): Obtenha, em ordem crescente de preço, o nome das peças de cor vermelha ou amarela e com preço de \$09, \$ 18, \$ 22 , \$40 ou \$ 90

```
SELECT PeNome
FROM Peca
WHERE ( ( PeCor = 'Vermelho'
OR PeCor = 'Amarelo' )
AND PePreco IN (09,18,22,40,90))
ORDER BY PePreco ASC
```

| PeNome |
|----------|
| Limpador |
| Volante |

SQL – Comandos DML – Exemplos

Operador LIKE

Exemplo 12 (Like): Obtenha o nome dos fornecedores residentes em cidades iniciadas com a letra S

```
SELECT FNome  
FROM Fornecedor  
WHERE FCidade LIKE 'S%'
```

| FNome |
|------------|
| C&M |
| Equipament |

SQL – Comandos DML – Exemplos

Operador IS NULL

Exemplo 13 (IS NULL): Obtenha o nome das peças que não possuem cor

```
SELECT PeNome  
FROM Peca  
WHERE PeCor IS NULL;
```

| PeNome |
|-----------|
| Parabrisa |

SQL – Comandos DML – Exemplos

Utilizando a negação (NOT)

| Operador | Significado |
|-----------------|------------------------|
| NOT BETWEEN | NÃO ENTRE DOIS VALORES |
| NOT IN | NÃO ENTRE UMA LISTA |
| NOT LIKE | NÃO ENTRE UM PADRÃO |
| IS NOT NULL | NÃO IGUAL A NULL |

Exemplo 13 (Not): Obtenha o nome das peças cujo preço não é \$ 09, \$14, \$60

```
SELECT PeNome
FROM Peca
WHERE PePreco NOT IN (09,14,60)
```

| PeNome |
|---------|
| Cinto |
| Volante |
| Painel |

SQL – Comandos DML – Exemplos

CLÁUSULA DISTINCT

Exemplo 14 (Distinct): Obtenha os código de todas as peças fornecidas

```
SELECT DISTINCT PeNro
FROM Fornece_para
```

| PeNro |
|-------|
| PE1 |
| PE2 |
| PE3 |
| PE4 |
| PE5 |

SQL – Comandos DML – Exemplos

CLÁUSULA ORDER BY

Exemplo 15 (Order by): Obtenha, em ordem decrescente de preço, os nomes das peças de cor vermelha e com preço maior que \$ 15

```
SELECT  PeNome
FROM    Peca
WHERE   (PeCor = 'Vermelho'
AND     PePreco > 15)
ORDER BY PePreco DESC
```

| PeNome |
|---------|
| Painel |
| Volante |

SQL – Comandos DML – Exemplos

CLÁUSULA GROUP BY

Exemplo 16 (Group By): Obtenha a quantidade de cada peça utilizada em todos o projetos

```
SELECT  PeNro, SUM (Quant)
FROM    Fornece_para
GROUP BY PeNro
```

| PeNro | SUM(Quant) |
|-------|------------|
| PE1 | 5 |
| PE2 | 2 |
| PE3 | 2 |
| PE4 | 5 |
| PE5 | 1 |

SQL – Comandos DML – Exemplos

Cláusula HAVING

Exemplo 17 (Having): Obtenha os códigos das peças que são utilizadas em uma quantidade inferior a 5 unidades na somatória de todos os projetos

```
SELECT  PeNro, SUM (Quant)
FROM    Fornece_para
GROUP BY PeNro
HAVING  SUM (Quant) < 5
```

| PeNro | SUM(Quant) |
|-------|------------|
| PE2 | 2 |
| PE3 | 2 |
| PE5 | 1 |

SQL – Comandos DML – Exemplos

JOINS

Exemplo 18 (Join): Obtenha o nome das peças utilizadas no projeto P4

```
SELECT  Peca.PeNome
FROM    Peca, Fornece_para
WHERE   (Fornece_para.PNro = 'P4'
AND     Fornece_para.PeNro=Peca.PeNro)
```

| PeNome |
|----------|
| Cinto |
| Lanterna |

SQL – Comandos DML – Exemplos

Exemplo 19 (Natural Join): Obtenha o nome das peças que tenham o preço maior que \$20

```
SELECT DISTINCT PeNome
FROM(Peca NATURAL JOIN Fornece_para)
WHERE PePreco > 20
```

| PeNome |
|--------|
| Cinto |
| Painel |

| PeNro | PeNome | PePreço | PeCor | FNro | PNro | Quant |
|-------|----------|---------|----------|------|------|-------|
| PE1 | Cinto | 22 | Azul | F5 | P4 | 5 |
| PE2 | Volante | 18 | Vermelho | F2 | P2 | 1 |
| PE2 | Volante | 18 | Vermelho | F3 | P5 | 2 |
| PE3 | Lanterna | 14 | Preto | F3 | P4 | 2 |
| PE4 | Limpador | 09 | Amarelo | F4 | P5 | 3 |
| PE4 | Limpador | 09 | Amarelo | F3 | P5 | 2 |
| PE5 | Painel | 43 | Vermelho | F1 | P1 | 1 |

SQL – Comandos DML – Exemplos

Exemplo 20 (Equi Join): Obtenha o nome das peças fornecidas que não são vermelhas e as suas cores

```
SELECT DISTINCT PeNome, PeCor
FROM (Peca JOIN Fornece_para ON PeNro = PeNro)
WHERE PeCor <> 'Vermelho'
```

| PeNome | PeCor |
|----------|---------|
| Cinto | Azul |
| Lanterna | Preto |
| Limpador | Amarelo |

SQL – Comandos DML – Exemplos

Exemplo 21 (Outer Join): Obtenha informações sobre todas as peças

```
SELECT *  
FROM(Peca LEFT OUTER JOIN Fornece_para  
ON PeNro = PeNro)
```

| PeNro | PeNome | PePreco | PeCor | FNro | PNro | Quant |
|-------|----------|---------|----------|------|------|-------|
| PE1 | Cinto | 22 | Azul | F5 | P4 | 5 |
| PE2 | Volante | 18 | Vermelho | F2 | P2 | 1 |
| PE2 | Volante | 18 | Vermelho | F3 | P5 | 2 |
| PE3 | Lanterna | 14 | Preto | F3 | P4 | 2 |
| PE4 | Limpador | 09 | Amarelo | F4 | P5 | 3 |
| PE4 | Limpador | 09 | Amarelo | F3 | P5 | 2 |
| PE5 | Painel | 43 | Vermelho | F1 | P1 | 1 |
| PE6 | Calota | 70 | Azul | NULL | NULL | NULL |

SQL – Comandos DML – Exemplos

SubQueries (Subconsultas)

Exemplo 22 (SubSelect): Obtenha o nome das peças utilizadas no projeto P5

```
SELECT Peca.PeNome  
FROM Peca  
WHERE Peca.PeNro IN (  
SELECT DISTINCT  
Fornece_para.PeNro  
FROM Fornece_para  
WHERE Fornece_para.PNro = 'P5' )
```

| PeNome |
|----------|
| Limpador |

| PeNro |
|-------|
| PE4 |

SQL – Comandos DML – Exemplos

Exemplo 23 (SubSelect): Obtenha o nome das peças fornecidas pelos fornecedores F4 e F3

```
SELECT Peca.PeNome
FROM Peca
WHERE Peca.PeNro IN (
    SELECT Fornece_para.PeNro
    FROM Fornece_para
    WHERE Fornece_para.FNro = 'F4' )
AND Peca.PeNro IN (
    SELECT Fornece_para.PeNro
    FROM Fornece_para
    WHERE Fornece_para.FNro = 'F3' )
```

| PeNome |
|----------|
| Limpador |

| PeNro |
|-------|
| PE4 |

| PeNro |
|-------|
| PE3 |
| PE4 |

SQL – Comandos DML – Exemplos

Exemplo 24 (SubSelect): Obtenha o nome das peças utilizadas nos projetos com duração maior que 3 meses

```
SELECT Peca.PeNome
FROM Peca
WHERE Peca.PeNro IN (
    SELECT Fornece_para.PeNro
    FROM Fornece_para
    WHERE Fornece_para.PNro IN(
        SELECT Projeto.PNro
        FROM Projeto
        WHERE Projeto.PDuração>3))
```

| PeNome |
|--------|
| Painel |

| PeNro |
|-------|
| PE5 |

| PNro |
|------|
| P1 |

SQL – Comandos DML – Exemplos

Exemplo 25 (Exists): Obtenha os nomes dos fornecedores que fornecem a peça PE2

```
SELECT Fornecedor.FNome
FROM Fornecedor
WHERE EXISTS (
  SELECT *
  FROM Fornece_para
  WHERE (Fornece_para.FNro = Fornecedor.FNro
  AND Fornece_para.PeNro = 'PE2') )
```

| FNome |
|-------|
| C&M |

SQL – Comandos DML – Exemplos

OPERADOR ANY

Exemplo 26 (Any): Obtenha o nome das peças fornecidas por algum fornecedor de Piracicaba

```
SELECT Peca.PeNome
FROM Peca
WHERE Peca.PeNro IN (
  SELECT PeNro
  FROM Fornece_para
  WHERE FNro =ANY (
    SELECT FNro
    FROM Fornecedor
    WHERE FCidade='Piracicaba'))
```

| PeNome |
|----------|
| Limpador |

| PeNro |
|-------|
| PE4 |

| FNro |
|------|
| F4 |

SQL – Comandos DML – Exemplos

OPERADOR ALL

Exemplo 27 (All): Obtenha o nome das peças não fornecidas por fornecedores de categoria A

```
SELECT  PeNome
FROM    Peca
WHERE   PeNro IN (
        SELECT  DISTINCT PeNro
        FROM    Fornece_para
        WHERE   FNro <> ALL (
            SELECT  FNro
            FROM    Fornecedor
            WHERE   FCateg = 'A' ) )
```

| PeNome |
|---------|
| Cinto |
| Volante |
| Painel |

| PeNro |
|-------|
| PE1 |
| PE2 |
| PE5 |

| FNro |
|------|
| F3 |
| F4 |

SQL – Comandos DML – Exemplos

UNION

Exemplo 28 (Union): Obtenha os códigos das peças com preço menor que \$ 20 ou que possuem a cor vermelha

```
SELECT  PeNro
FROM    Peca
WHERE   PePreco < 20
UNION
SELECT  PeNro
FROM    Peca
WHERE   PeCor = 'Vermelho'
```

| PeNro |
|-------|
| PE2 |
| PE3 |
| PE4 |

| PeNro |
|-------|
| PE2 |
| PE5 |

| PeNro |
|-------|
| PE2 |
| PE3 |
| PE4 |
| PE5 |

SQL – Comandos DML – Exemplos

INTERSECT

Exemplo 29 (Intersect): Obtenha os códigos das peças com preço menor que \$ 20 e que possuem a cor vermelha

```
SELECT PeNro
FROM Peca
WHERE PePreco < 20
INTERSECT
SELECT PeNro
FROM Peca
WHERE PeCor = 'Vermelho'
```

| PeNro |
|-------|
| PE2 |
| PE3 |
| PE4 |

| PeNro |
|-------|
| PE2 |

| PeNro |
|-------|
| PE2 |
| PE5 |

SQL – Comandos DML – Exemplos

MINUS

Exemplo 30 (Minus): Obtenha os códigos das peças com preço menor que \$ 20 e que não possuem a cor vermelha

```
SELECT PeNro
FROM Peca
WHERE PePreco < 20
MINUS
SELECT PeNro
FROM Peca
WHERE PeCor = 'Vermelho'
```

| PeNro |
|-------|
| PE2 |
| PE3 |
| PE4 |

| PeNro |
|-------|
| PE2 |
| PE5 |

| PeNro |
|-------|
| PE3 |
| PE4 |