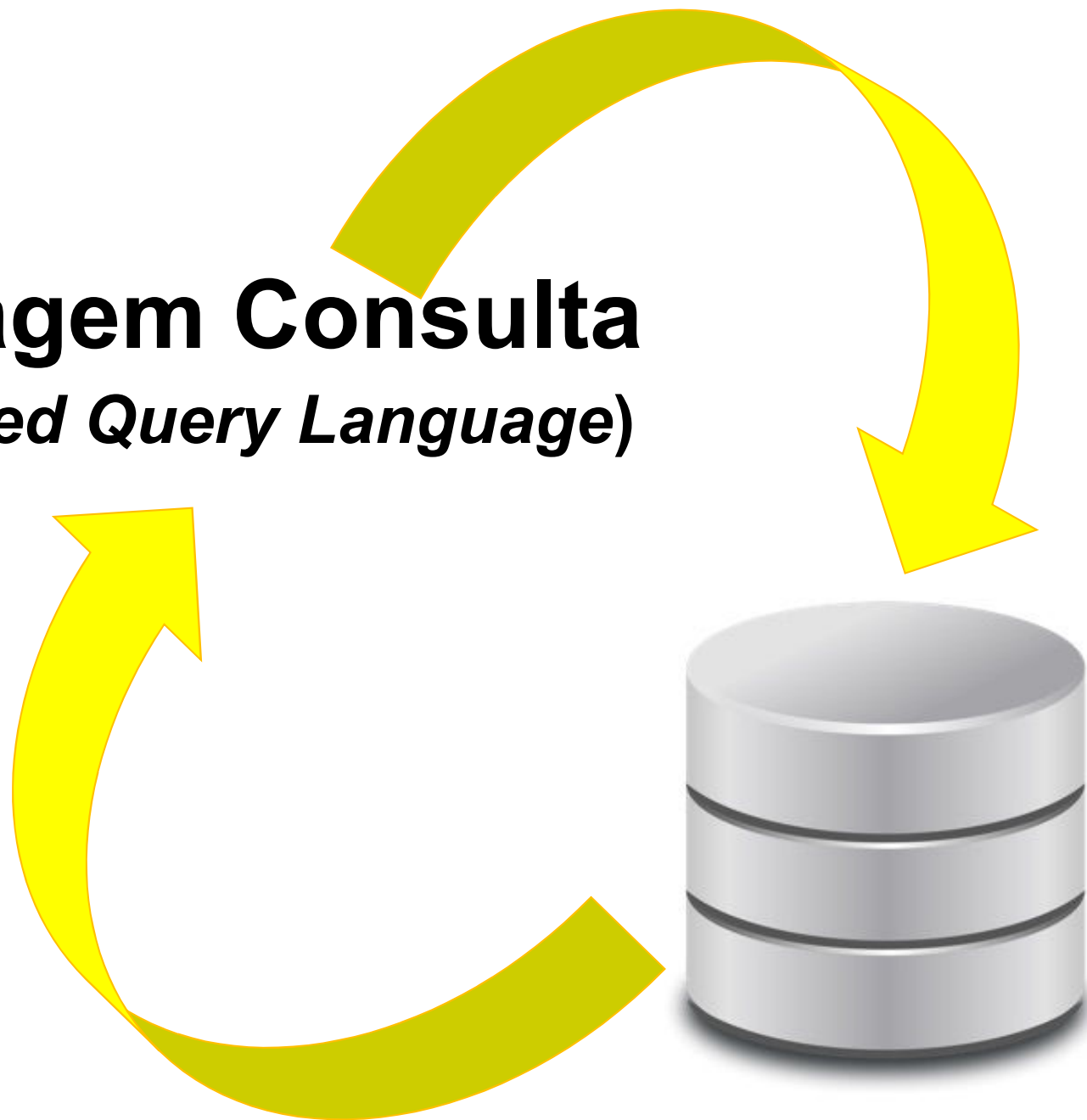




Universidade Federal de Mato Grosso-UFMT
Sistemas de Informação
Laboratório de Banco de Dados
Prof. Dr. Clóvis Júnior

Linguagem Consulta

(Structured Query Language)



Structure Query Language - História

- **1970: Codd define o Modelo Relacional**
- **1974: IBM desenvolve o projecto SYSTEM/R com a linguagem SEQUEL**
- **1979: É lançado o primeiro SGBD comercial (ORACLE)**
- **1981: É lançado o SGBD INGRES**
- **1983: IBM anuncia o DB2**
- **1986, 1987: É ratificada a norma SQL que fica conhecida como SQL-86 (ANSI X3.135-1986 e ISO 9075:1987)**
- **1989: É ratificada a norma SQL-89 quer pela ANSI quer pela ISO**
- **1992: É ratificada a norma: SQL2**
- **1999: É ratificada a norma SQL1999, anteriormente conhecida como SQL3**
- **2006: SQL:2006, define a forma como o SQL pode ser usado em conjunção com o XML (ANSI/ISO/IEC 9075-14:2006)**

Structure Query Language - Definição

- **SQL é uma linguagem normalizada para definição, acesso, manipulação e controle de Bancos de Dados Relacionais**
- **Na maioria dos SGBDR, esta linguagem pode ser utilizada:**
 - **Interativamente**
 - **Embutida em linguagens de programação**

Structure Query Language – Esquema Relacional

Empregado (cod-emp, nome_emp, data_admissão,
cod_cat, cod_dept, cod_emp_chefe)

Departamento (cod-dept, nome_dept, localização)

Categoria (cod-cat, designação, salario_base)

Structure Query Language – Esquema Relacional

Categoria

cod_cat	designação	salario_base
1	CategoriaA	300
2	CategoriaB	250
3	CategoriaC	160
...

Departamento

cod_dept	nome_dept	localização
1	Contabilidade	Lisboa
2	Vendas	Porto
3	Investigação	Coimbra
...

Empregado

cod_emp	nome_emp	data_admissão	cod_cat	cod_dept	cod_emp_chefe
1	António Abreu	13-Jan-75	1	1	1
2	Bernardo Bento	1-Dec-81	1	2	1
3	Carlos Castro	4-Jun-84	3	3	1
...
20	Manuel Matos	7-Feb-90	3	2	2
...

Structure Query Language – Comandos

Qual o salário do empregado 'Machado de Assis' e o nome do departamento a que pertence?

```
SELECT nome_emp, salario_base, nome_dept  
FROM Empregado, Departamento, Categoria  
WHERE nome_emp = 'Machado de Assis'  
        AND Empregado.cod_cat = Categoria.cod_cat  
        AND Departamento.cod_dept = Empregado. cod_dept
```

Structure Query Language – Componentes

DDL (Data Definition Language)

DML (Data Manipulation Language)


TML (Transaction Manipulation Language)

DCL (Data Control Language)

Structure Query Language – Manipulação de Dados

SELECT Acesso aos dados da B.D.

INSERT
UPDATE
DELETE



Manipulação dos dados da B.D.

Structure Query Language – Select e From

```
SELECT  [ DISTINCT ] coluna, ... | *  
FROM    tabela
```

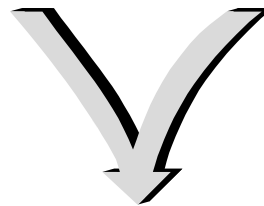
O símbolo * é utilizado quando se pretende selecionar todos os atributos da tabela especificada na clausula FROM

DISTINCT é aplicado a todas as colunas especificadas na clausula SELECT e elimina as repetições existentes

Structure Query Language – Select e From - Projeção

Empregado

cod_emp	nome_emp	data_admissão	cod_cat	cod_dept	cod_emp_chefe
1	António Abreu	13-Jan-75	1	1	1
2	Bernardo Bento	1-Dec-81	1	2	1
3	Carlos Castro	4-Jun-84	3	3	1
...
20	Manuel Matos	7-Feb-90	3	2	2
...



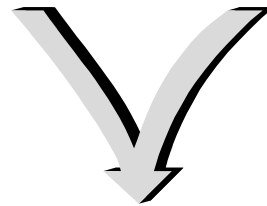
Clausulas
Select
From

```
SELECT  cod_emp, nome_emp  
FROM    empregado
```

Structure Query Language – Select e From - Restrição

Categoria

cod_cat	designação	salario_base
1	CategoriaA	300
2	CategoriaB	250
3	CategoriaC	160
...



Clausula Where

```
SELECT  *  
FROM    categoria  
WHERE   salario_base > 200
```

Structure Query Language – Select e From - Junção

Empregado

cod_emp	nome_emp	data_admissão	cod_cat	cod_dept	cod_emp_chefe
1	Antônio Abreu	13-Jan-75	1	1	1
2	Bernardo Bento	1-Dec-81	1	2	1
3	Carlos Castro	4-Jun-84	3	3	1
...
20	Manuel Matos	7-Feb-90	3	2	2
...

A partir do produto cartesiano
seleciona-se somente as linhas que
satisfazem a condição

EMPREGADO.COD_DEPT= DEPTARTAMENTO.COD_DEPT

Departamento

cod_dept	nome_dept	localização
1	Contabilidade	Cuiabá
2	Vendas	Rondonópolis
3	Investigação	Campo Grande
...

SQL – Junções Múltiplas

Categoria

cod_cat	designação	salario_base
1	CategoriaA	300
2	CategoriaB	250
3	CategoriaC	160
...

Departamento

cod_dept	nome_dept	localização
1	Contabilidade	Lisboa
2	Vendas	Porto
3	Investigação	Coimbra
...

Empregado

cod_emp	nome_emp	data_admissão	cod_cat	cod_dept	cod_emp_chefe
1	António Abreu	13-Jan-75	1	1	1
2	Bernardo Bento	1-Dec-81	1	2	1
3	Carlos Castro	4-Jun-84	3	3	1
...
20	Manuel Matos	7-Feb-90	3	2	2
...

```
SELECT categoria.cod_cat, nome_emp, nome_dept, salario_base
FROM empregado, departamento, categoria
WHERE empregado.cod_dept = departamento.cod_dept
AND empregado.cod_cat = categoria.cod_cat
```

Structure Query Language – Select e From - Junção

```
SELECT  nome_emp, empregado.cod_dept, nome_dept  
FROM    empregado, departamento  
WHERE   empregado.cod_dept = departamento.cod_dept
```

Caso o nome de uma coluna seja igual em várias tabelas então a REGRA é

Nome_Tabela.Nome_Coluna

em qualquer local da cláusula SELECT

SQL – Select e From – Junção, Restrição e Junção

Qual o nome dos empregados pertencentes ao departamento de Vendas

```
SELECT empregado.cod_dept, nome_emp  
FROM empregado, departamento  
WHERE empregado.cod_dept = departamento.cod_dept  
AND  
nome_dept = 'Vendas'
```

Projeção

Restrição

Junção

SQL – Alias (*Correlation Name*)

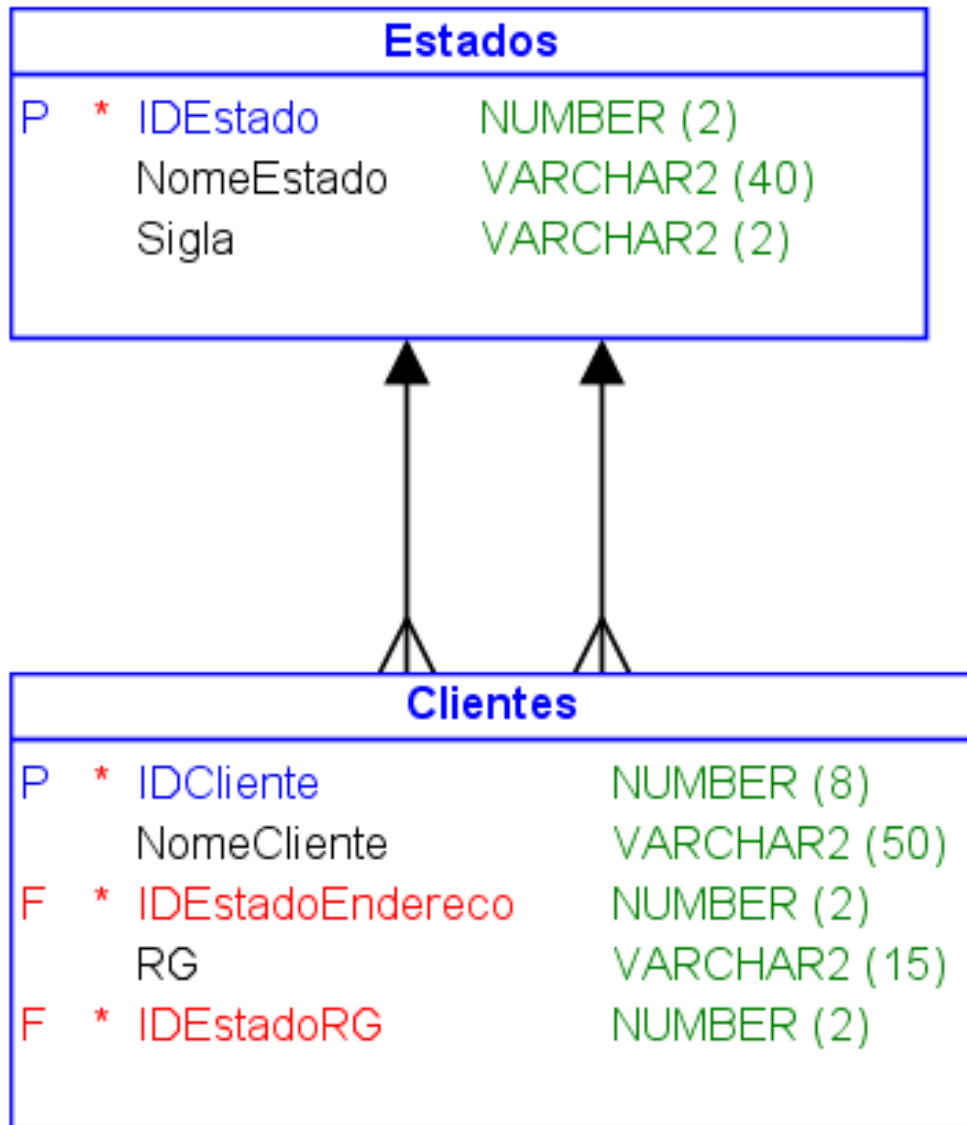
```
SELECT  cod_emp, D.cod_dept, nome_dept  
FROM    empregado E, departamento D  
WHERE   E.cod_dept = D.cod_dept
```

Útil quando se pretende usar a mesma tabela com significados diferentes

Pretende-se o nome de cada empregado e o nome do respectivo chefe

```
SELECT  E.nome, CH.nome  
FROM    empregado E, empregado CH  
WHERE   E.cod_emp_chefe = CH.cod_emp
```


SQL – Alias (*Correlation Name*) - Exemplo



Usuário: SQL_Alias

```
CREATE TABLE Clientes
( IDCliente NUMBER (8) ,
  NomeCliente VARCHAR2 (50) NOT NULL ,
  RG VARCHAR2 (15) NOT NULL ,
  IDEstadoRG NUMBER (2) NOT NULL ,
  IDEstadoEndereco NUMBER (2) NOT NULL
);
```

```
CREATE TABLE Estados
( IDEstado NUMBER (2) NOT NULL ,
  NomeEstado VARCHAR2 (40) ,
  Sigla VARCHAR2 (2)
);
```

```
ALTER TABLE Estados ADD CONSTRAINT Estados_PK
PRIMARY KEY ( IDEstado ) ;
```

```
ALTER TABLE Clientes ADD CONSTRAINT Clientes_PK
PRIMARY KEY ( IDCliente ) ;
```

```
ALTER TABLE Clientes ADD CONSTRAINT Estados_RG_FK
FOREIGN KEY ( IDEstadoRG)
REFERENCES Estados (IDEstado);
```

```
ALTER TABLE Clientes ADD CONSTRAINT Estados_Endereco_FK
FOREIGN KEY
(IDEstadoEndereco) REFERENCES Estados (IDEstado);
```

SQL – União

Suponha que tem as seguintes tabelas:

CLIENTE (nome, morada)

FORNECEDOR (nome, morada)

Resulta em uma lista com endereço tanto de clientes quanto fornecedores

```
SELECT nome, endereco  
FROM cliente  
UNION  
SELECT nome, endereco  
FROM fornecedor
```

SQL – Intersecção

Resulta em endereços de clientes que também são fornecedores

```
SELECT  nome, endereco  
FROM    cliente  
INTERSECT  
SELECT  nome, endereco  
FROM    fornecedor
```

SQL – Diferença

Resulta em uma lista de endereços de clientes que não são fornecedores

```
SELECT nome, endereco  
FROM cliente  
EXCEPT  
SELECT nome, endereco  
FROM fornecedor
```

SQL – Clausula Where

```
SELECT [ DISTINCT ] coluna, ...|  
*  
FROM   tabela, [tabela,....]  
WHERE  condição-de-pesquisa
```

Uma condição de pesquisa é basicamente uma coleção de predicados, combinados através dos operadores booleanos AND, OR, NOT e parêntesis.

SQL – Predicados

- Um predicado de comparação (WHERE NOME_EMP = 'João Silva')
- Um predicado de BETWEEN (WHERE COD_EMP **BETWEEN** 1 AND 5)
- Um predicado de LIKE (WHERE NOME_EMP **LIKE** ' M%')
- Um teste de valor nulo (WHERE COMISSÃO **IS** NULL)
- Um predicado de IN (WHERE COD_CAT **IN** (1,2))

SQL – Predicados - SubQuery

- Os predicados podem ser utilizados num contexto estático, sendo avaliados com base em valores constantes.

Ex: WHERE COD_CAT IN (1,2)

- Podem também ser utilizados com base em valores dinâmicos

Ex: WHERE COD_CAT IN
(SELECT COD_CAT FROM CATEGORIA)



SQL – Predicados – SubQuery – Integração de Consultas

Nomes dos empregados que trabalham nos departamentos
de Rondonópolis

```
SELECT cod_emp, nome_emp  
FROM empregado  
WHERE cod_dept IN ( SELECT cod_dept  
                        FROM departamento  
                        WHERE localização = 'Rondonópolis'  
                      )
```

SQL – Predicados – SubQuery – Integração de Consultas

Empregados cujo salário é superior a **todos** os salários dos empregados do departamento 1

```
SELECT      nome_emp
FROM empregado, categoria
WHERE      empregado.cod_cat = categoria.cod_cat
AND
salario_base > ALL
( SELECT salario_base
  FROM empregado, categoria
  WHERE empregado.cod_cat = categoria.cod_cat
    AND cod_dept = 1
)
```

SQL – Predicados – SubQuery – Integração de Consultas

Empregados cujo salário é superior a **algum** dos salários dos empregados do departamento 1

```
SELECT      nome_emp
FROM empregado, categoria
WHERE      empregado.cod_cat = categoria.cod_cat
AND
salário_base > ANY
( SELECT salario_base
  FROM empregado, categoria
  WHERE empregado.cod_cat = categoria.cod_cat
    AND cod_dept = 1
)
```

SQL – Predicados – SubQuery – Integração de Consultas

Operador EXISTS

Nome dos departamentos que têm empregados (pelo menos um)

```
SELECT nome_dept  
FROM departamento  
WHERE EXISTS  
    ( SELECT *  
      FROM empregado  
      WHERE departamento.cod_dept = empregado.cod_dept    )
```

A condição é VERDADEIRA se o resultado da subquery não for vazio

SQL – Predicados – SubQuery – Integração de Consultas

Nome dos departamentos que não têm empregados

```
SELECT      nome_dept
FROM departamento
WHERE NOT EXISTS
      ( SELECT *
        FROM empregado
        WHERE departamento.cod_dept =
empregado.cod_dept )
```

A condição é VERDADEIRA se o resultado da subquery for vazio

SQL – Clausula Order By

Clausula **ORDER BY** usada para ordenar os dados referentes a uma ou mais colunas
É a última clausula a ser especificada

```
SELECT    [ DISTINCT ] coluna, ... | *  
FROM      tabela  
WHERE     condição  
ORDER BY  coluna [ASC | DESC ], ...
```

SQL – Clausula Order By

```
SELECT      *  
FROM        empregado  
ORDER BY    nome_emp
```

Por **defeito**, os dados são ordenados **ascendentemente**

↑	Z	9	Recentes
	⋮	⋮	⋮
	A	0	Menos Recentes
	Caracter (Char)	Numérico (Number)	Data (Date)

SQL – Funções Agregadoras

Salário_base	
1	100
2	200
3	12,5
4	450
5	700
6	100
7	120
8	350
9	890
10	400

MIN = 12,5

MAX = 890

COUNT(*) = 10

SUM =

AVG = SUM / COUNT

SQL – Funções Agregadoras

```
SELECT MAX(salario_base)
FROM categoria
```

```
SELECT MIN(salario_base)
FROM categoria
```

```
SELECT COUNT(*)
FROM categoria
```

```
SELECT SUM(salario_base)
FROM categoria, empregado
WHERE empregado.cod_cat =
categoria.cod_cat
```

```
SELECT AVG(salario_base)
FROM categoria, empregado
WHERE empregado.cod_cat =
categoria.cod_cat
```

SQL – Funções Agregadoras com Restrições

```
SELECT    AVG(salario_base)  
FROM      empregado, categoria  
WHERE      cod_dept = 1  
            and  
            empregado.cod_cat = categoria.cod_cat
```

Média dos salários dos empregados do
departamento cujo código é 1

SQL – Agrupamentos

Cod_dept Salário_base

1	120	120
1	250	
1	150	
1	300	
1	250	
2	100	100
2	150	
2	230	
3	300	160
3	400	
3	200	
3	160	

Para cada departamento qual o
salário mínimo?

```
SELECT  cod_dept, min(salario_base)
FROM    empregado, categoria
WHERE    empregado.cod_cat = categoria.cod_cat
GROUP BY cod_dept
```

SQL – Agrupamentos Múltiplos

Cod_dept Tipo_cat Salário_base

1	A	120	120
1	A	250	
1	B	150	150
1	B	300	
1	B	250	
2	A	100	100
2	B	150	150
2	B	230	
3	B	300	300
3	B	400	
3	C	200	160
3	C	160	

Para cada categoria de cada departamento qual o salário mínimo?

```
SELECT  cod_dept, tipo_cat, min(salario_base)
FROM    empregado, categoria
WHERE   empregado.cod_cat = categoria.cod_cat
GROUP BY cod_dept, tipo_cat
```

SQL – Agrupamentos Múltiplos

SELECT	[DISTINCT] coluna, ... *
FROM	tabela, ...
WHERE	condição
GROUP BY	coluna, ...

Qualquer coluna que não seja uma função agregadora só pode estar na cláusula SELECT se estiver na cláusula GROUP BY

SELECT	COD_DEPT, min(salario_base)
FROM	empregado, categoria
WHERE	empregado.cod_cat = categoria.cod_cat
GROUP BY	COD_DEPT

SQL – Restrições sobre Grupos

Cod_dept Salário_base

1	120
1	250
1	150
1	300
1	250
2	100
2	150
2	230
3	300
3	400
3	200
3	160

AVG = 214

120

AVG = 160

100

AVG = 265

160

Para cada departamento qual o salário mínimo.

Seleccionar apenas os departamentos cujo salário médio seja superior a 200

```
SELECT    cod_dept, min(salario_base)
FROM      empregado, categoria
WHERE     empregado.cod_cat = categoria.cod_cat
GROUP BY  cod_dept
HAVING    avg (salario_base) > 200
```

SQL – Clausula Having

```
SELECT      [ DISTINCT ] coluna, ... | *  
FROM        tabela, ...  
WHERE       condição  
GROUP BY    coluna, ...  
HAVING      condição
```

WHERE OU HAVING ?

A cláusula WHERE nunca contém funções agregadoras

A cláusula HAVING deve sempre conter funções agregadoras

SQL – SubQueries com Funções Agregadoras

Nome do empregado que tem o maior salário

```
SELECT empregado.cod_emp, nome_emp
FROM   empregado, categoria
WHERE  empregado.cod_cat = categoria.cod_cat and
       salario_base = ( SELECT max(salario_base)
                        FROM categoria, empregado
                        WHERE empregado.cod_cat = categoria.cod_cat
                        )
```


SQL – SubQueries com Agrupamentos

Para cada departamento qual o empregado com maior salário

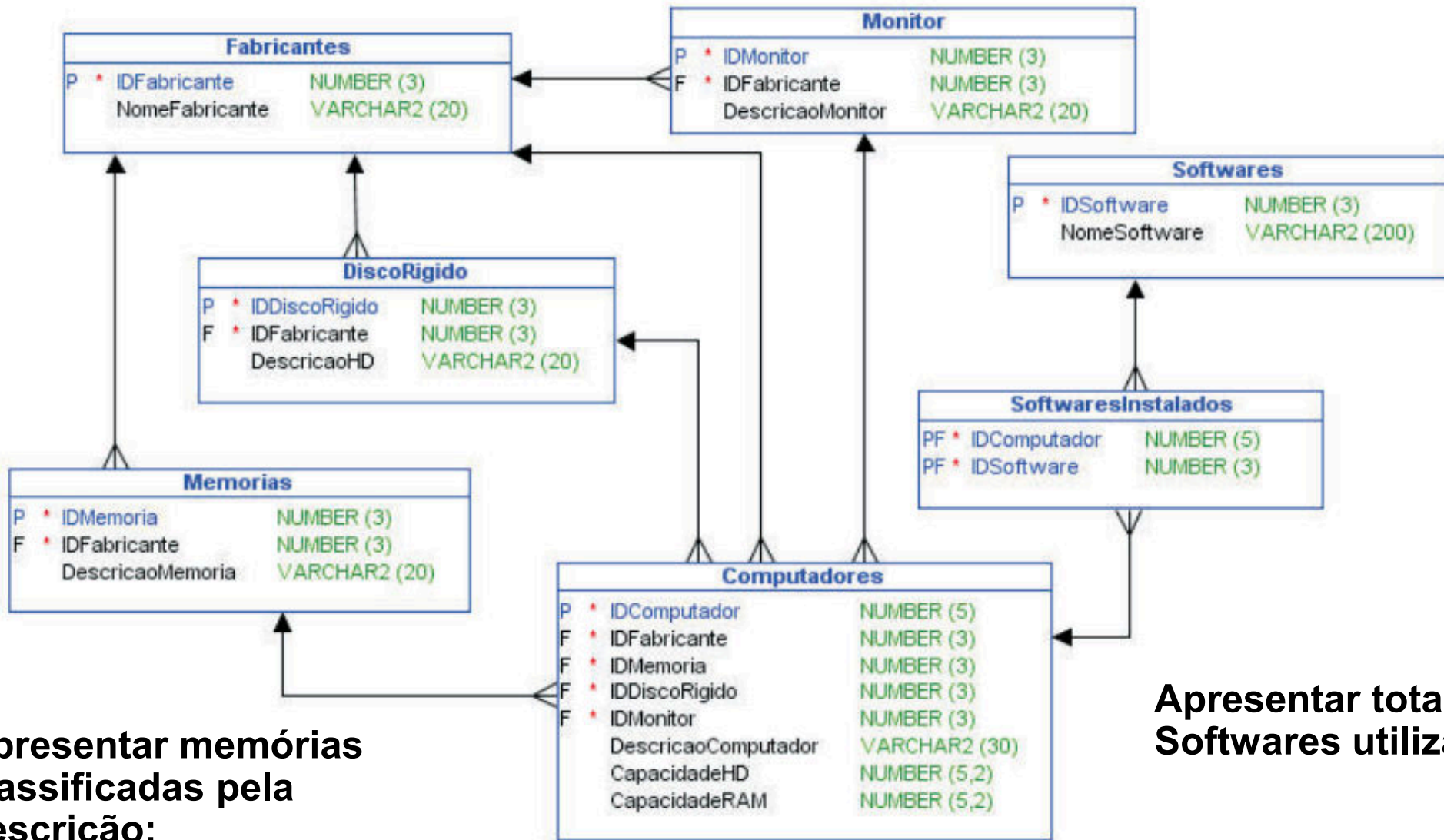
```
SELECT cod_dept, cod_emp, nome_emp
FROM empregado, categoria
WHERE empregado.cod_cat = categoria.cod_cat and
      (cod_dept, salario_base) IN
      ( SELECT cod_dept, max(salario_base)
        FROM categoria, empregado
        WHERE empregado.cod_cat = categoria.cod_cat
        GROUP BY cod_dept
      )
```

SQL – Exercícios

Apresentar Monitores e fabricantes;

Apresentar Memórias e fabricantes;

Apresentar total de softwares iniciados com a letra W;



Apresentar memórias
classificadas pela
descrição;

Apresentar total de
Softwares utilizados;

SQL – Exercícios

- 1) Descrição do computador, descrição do tipo de memória e capacidade para computadores com 32 Gb ou mais de RAM;**
- 2) Listar os computadores com as respectivas capacidades de HD cujos monitores sejam de LED;**
- 3) Apresentar a descrição dos computadores e softwares instalados em computadores com menos de 16 Mb de RAM;**
- 4) Soma da quantidade de memória RAM instaladas nos computadores agrupados pelo fabricante (memória);**