DCC011:
Introdução a Banco de Dados

Rodrygo Santos
rodrygo@dcc.ufmg.br

Departamento de Ciência da Computação
Universidade Federal de Minas Gerais

Programa

- Introdução
  - Conceitos básicos, características da abordagem de banco de dados, modelos de dados, esquemas e instâncias, arquitetura de um sistema de banco de dados, componentes de um sistema de gerência de banco de dados.
- Modelos de dados e linguagens
  - Modelo entidade-relacionamento (ER), modelo relacional, álgebra relacional, **SQL**.
- Projeto de bancos de dados
  - Fases do projeto de bancos de dados, projeto lógico de bancos de dados relacionais, normalização.
- Novas Tecnologias e Aplicações de Banco de Dados

1. Introdução

- Originalmente proposta para o System R desenvolvido nos laboratórios da IBM na década de 70 → SEQUEL (Structured English QUEry Language)
- Objeto de um esforço de padronização coordenado pelo ANSI/ISO:
  - SQL1 (SQL-86)
  - SQL2 (SQL-92)
  - SQL3 (SQL:1999)

SQL: linha do tempo

Year	Official standard	Informal name	Comments
1986 1987	ANSI X3.135:1986 ISO/IEC 9075:1987 FIPS PUB 127	SQL-86 SQL-87	First formalized by ANSI, adopted as FIPS PUB 127
1989	ANSI X3.135-1989 ISO/IEC 9075:1989 FIPS PUB 127-1	SQL-89	Minor revision that added integrity constraints, adopted as FIPS PUB 127-1
1992	ANSI X3.135-1992 ISO/IEC 9075:1992 FIPS PUB 127-2	SQL-92 SQL2	Major revision (ISO 9075), *Entry Level* SQL-92, adopted as FIPS PUB 127-2
1999	ISO/IEC 9075:1999	SQL:1999 SQL3	Added regular expression matching, recursive queries (e.g., transitive closure), triggers, support for procedural and control-of-flow statements, nonscalar types (arrays), and some object-oriented features (e.g., structured types), support for embedding SQL in Java (SQL/OLB) and vice versa (SQL/JRT)
2003	ISO/IEC 9075:2003	SQL:2003	Introduced XML-related features (SQL/XML), window functions, standardized sequences, and columns with autogenerated values (including identity columns)
2006	ISO/IEC 9075-14:2006	SQL:2006	Adds Part 14, defines ways that SQL can be used with XML. It defines ways of importing and storing XML data in an SQL database, manipulating it within the database, and publishing both XML and conventional SQL data in XML form. In addition, it lets applications integrate queries into their SQL code with XQuery, the XML Query Language published by the World Wide Web Consortium (W3C), to concurrently access ordinary SQL-data and XML documents.[31]
2008	ISO/IEC 9075:2008	SQL:2008	Legalizes ORDER BY outside cursor definitions. Adds INSTEAD OF triggers, TRUNCATE statement,[32] FETCH clause
2011	ISO/IEC 9075:2011	SQL:2011	Adds temporal data (PERIOD FOR)[33] (more information at Temporal database#History). Enhancements for window functions and FETCH clause.[34]
2016	ISO/IEC 9075:2016	SQL:2016	Adds row pattern matching, polymorphic table functions, operations on JSON data stored in character string fields
2019	ISO/IEC 9075-15:2019	SQL:2019	Adds Part 15, multidimensional arrays (MDarray type and operators)
2023	ISO/IEC 9075:2023	SQL:2023	Adds data type JSON (SQL/Foundation); Adds Part 16, Property Graph Queries (SQL/PGQ)

Introdução

- SQL é considerada a razão principal para o sucesso de bancos de dados relacionais comerciais
- Tornou-se a linguagem padrão para bases relacionais
- Funciona entre diferentes produtos
- *Embedded SQL*: Java, C/C++, Cobol...

Introdução

- SQL = LDD + LMD + LCD + LC
  - LDD – Ling. Definição de Dados
    - CREATE SCHEMA / TABLE / VIEW
    - DROP SCHEMA / TABLE / VIEW
    - ALTER TABLE
  - LMD – Ling. Manipulação de Dados
    - INSERT, UPDATE, DELETE
  - LCD – Ling. Controle de Dados
    - GRANT, REVOKE
  - LC – Ling. de Consulta
    - SELECT

Introdução

- Conceitos:
  - Tabela/Table = Relação
  - Linha/Row = Tupla
  - Coluna/Column = Atributo

1. Consultas Básicas em SQL

- Formato básico do comando SELECT:

  SELECT <lista de atributos>
  FROM <lista de tabelas>
  [**WHERE** <condição>;]

- Exemplo:

  SELECT BDATE, ADDRESS
  FROM EMPLOYEE
  WHERE FNAME= 'John' **AND**
  MINIT= 'B' **AND**
  LNAME= 'Smith' ;

$\pi_{bdate,Address}\ \sigma_{Fname= 'John' \text{ AND } Minit= 'B' \text{ AND } Lname= 'Smith'}(EMPLOYEE)$

Consultas básicas e Álgebra

- Operações Básicas
  - *Seleção* (σ) Seleciona um sub-conjunto de linhas da relação ➔FROM, WHERE
  - *Projeção* (π) Mantém apenas colunas específicas ➔ SELECT
  - Junção (×, ⋈): WHERE

Consultas Básicas em SQL

A. SELECT FROM WHERE

(Q1)

```
SELECT LNOME, ENDERECO
FROM EMPREGADO, DEPARTAMENTO
WHERE DNOME='Research' AND dno=dnumero;
```

condição de seleção

condição de junção

$\pi_{lnome,\ endereco}\ \sigma_{dnome='Research'}\ (EMPREGADO \bowtie_{dno=dnumero} DEPARTAMENTO)$

EMPLOYEE(ssn, fname, lname, address,bdate, superssn, dno)
  superssn REFERENCIA EMPLOYEE
  dno REFERENCIA DEPARTMENT
DEPARTMENT (dnum, dname, mgrssn, mgrinitialdate)
 mgrssn REFERENCIA EMPLOYEE.ssn

Consultas Básicas em SQL

B. Atributos Ambíguos e Pseudônimos (alias)

```
SELECT dname, dlocation
FROM
   DEPARTMENT AS D, DEPT_LOCATIONS AS DL
WHERE D.dnum = DL.dnumber;
```

(Q8)

```
SELECT e.fname, e.lname, s.fname, s.lname
FROM EMPLOYEE AS E, EMPLOYEE AS S
WHERE e.superssn=s.ssn;
```

EMPLOYEE(ssn, fname, lname, address,bdate, superssn, dno)
  superssn REFERENCIA EMPLOYEE
  dno REFERENCIA DEPARTMENT
DEPARTAMENT (dnum, dname, mgrssn, mgrinitialdate)
  mgrssn REFERENCIA EMPLOYEE.ssn
PROJECT (pnumber, pname, plocation, dnum)
  dnum REFERENCIA DEPARTAMENT
DEPT_LOCATIONS (dnumber,dlocation)
  dnumber REFERENCIA DEPARTAMENT

Consultas Básicas em SQL

B. Atributos Ambíguos e Pseudônimos (alias)

(Q2)

- **SELECT** pnumber, dnum, lname, address, bdate
  FROM PROJECT P, DEPARTMENT D, EMPLOYEE E
  WHERE plocation= 'Stafford' **AND**

condição de seleção

D.dnum=P.dnum **AND** D.mgrssn=E.ssn;

condição de junção

$\Pi_{pnumber,dnum,lname,address,bdate}\ \sigma_{plocation= 'Stafford'}$

$(EMPLOYEE \bowtie_{ssn=mgrssn} (DEPARTMENT \bowtie PROJECT))$

EMPLOYEE(ssn, fname, lname, address,bdate, superssn, dno)
  superssn REFERENCIA EMPLOYEE
  dno REFERENCIA DEPARTMENT
DEPARTAMENT (dnum, dname, mgrssn, mgrinitialdate)
 mgrssn REFERENCIA EMPLOYEE.ssn
PROJECT (pnumber, pname, plocation, dnum)
  dnum REFERENCIA DEPARTAMENT

Consultas Básicas em SQL

C. SELECT FROM sem o WHERE

SELECT ssn, lname, salary
FROM EMPLOYEE;

(Q10) **SELECT** lname, dname
FROM EMPLOYEE, DEPARTMENT
WHERE dno=dnumber; ← **com a junção**

Atenção! A consulta em vermelho corresponde a um produto cartesiano das tabelas EMPLOYEE e DEPARTMENT :

$\pi_{lname,dname}$ (EMPLOYEE x DEPARTMENT)

EMPLOYEE(ssn, fname, lname, address,bdate, superssn, dno)
superssn REFERENCIA EMPLOYEE
dno REFERENCIA DEPARTMENT
DEPARTAMENT (dnum, dname, mgrssn, mgrinitialdate)
mgrssn REFERENCIA EMPLOYEE.ssn

Consultas Básicas em SQL

D. TODOS OS ATRIBUTOS

- Consultas a todos os atributos

(Q1C)

```
SELECT *
FROM EMPLOYEE
WHERE Dno=5;
```

(Q1D)

```
SELECT *
FROM EMPLOYEE, DEPARTMENT
WHERE Dname= 'Research' AND Dno=Dnumber;
```

EMPLOYEE(ssn, fname, lname, address,bdate, superssn, dno)
  superssn REFERENCIA EMPLOYEE
  dno REFERENCIA DEPARTMENT
DEPARTAMENT (dnum, dname, mgrssn, mgrinitialdate)
  mgrssn REFERENCIA EMPLOYEE.ssn

Consultas Básicas em SQL

E. TABELAS COMO CONJUNTOS

- SQL trata uma tabela como um multi-conjunto
- Tuplas duplicadas **PODEM** aparecer em uma tabela
  - E no **resultado** de uma consulta

- **SQL não elimina automaticamente as duplicatas porque...**
  - Eliminação de duplicatas é uma operação cara (ordenar)
  - O usuário pode estar interessado nelas
  - Funções de agregação utilizam duplicatas *(funções de agregação serão explicadas a seguir)*

- Operações
  - SELECT DISTINCT, SELECT ALL
  - ∪: UNION, \ : EXCEPT, ∩: INTERSECT

Tabelas como Conjuntos

EMPLOYEE(ssn, fname, lname, address,bdate, superssn, dno)
  superssn REFERENCIA EMPLOYEE
  dno REFERENCIA DEPARTMENT
DEPARTAMENT (dnum, dname, mgrssn, mgrinitialdate)
  mgrssn REFERENCIA EMPLOYEE.ssn
PROJECT (pnumber, pname, plocation, dnum)
  dnum REFERENCIA DEPARTAMENT

```
SELECT salary
FROM EMPLOYEE;
```

Não elimina linhas (tuplas) duplicadas

Para eliminar precisa usar DISTINCT, por exemplo:

(Q11)

```
SELECT DISTINCT salary
FROM EMPLOYEE;
```

(Q4)

```
(SELECT pnumber
 FROM PROJECT, DEPARTMENT, EMPLOYEE
 WHERE dnum=dnumber AND mgrssn=ssn AND
lname= 'Smith' )
 UNION
(SELECT pnumber
 FROM PROJECT, WORKS_ON, EMPLOYEE
 WHERE pnumber=pno AND essn=ssn AND
            lname= 'Smith' );
```

2. Facilidades Adicionais

A. JOINS

- Uso do operador JOIN, na cláusula FROM
- **SELECT** FNAME, LNAME, ADDRESS
  FROM (EMPLOYEE **JOIN** DEPARTMENT
  ON DNO=DNUMEBR**)**
  WHERE DNAME= 'Research' ;
- A cláusula FROM contém então uma única tabela resultante da junção de Empregado e Departamento

EMPREGADO (ssn, pnome, minicial, unome,
datanasc, endereco, sexo, salario, superssn, dno)
superssn REFERENCIA EMPREGADO
dno REFERENCIA DEPARTAMENTO
DEPARTAMENTO (dnumero, dnome, gerssn,
gerdatainicio)
gerssn REFERENCIA EMPREGADO.ssn

Facilidades Adicionais
A. JOINS

- Pode-se especificar outros tipos de junção na cláusula FROM
- Junção natural: equijoin em cada par de atributos com o mesmo nome
- **SELECT** DNAME, DLOCATION

  FROM (DEPARTMENT **NATURAL JOIN**

  DEPT_LOCATIONS**)**;

DEPARTAMENTO (**dnumero**, dnome, gerssn, gerdatainicio)
  gerssn REFERENCIA EMPREGADO.ssn
DEPTO_LOCALIZACOES (**dnumero**, dlocalizacao)
  dnumero REFERENCIA DEPARTAMENTO

Facilidades Adicionais
A. JOINS

Renomeando atributos para o natural join:

- **SELECT** pnome, unome, dnome
  FROM (EMPREGADO **NATURAL JOIN**
  (DEPARTMENTO AS DEPT (**dno**, dnome, gerssn, gerdatainicio))**)**
  WHERE dnome = 'Pesquisa';

EMPREGADO (ssn, pnome, minicial, unome, …superssn, dno)
  superssn REFERENCIA EMPREGADO
  dno REFERENCIA DEPARTAMENTO
DEPARTAMENTO (dnumero, dnome, gerssn, gerdatainicio)
  gerssn REFERENCIA EMPREGADO.ssn

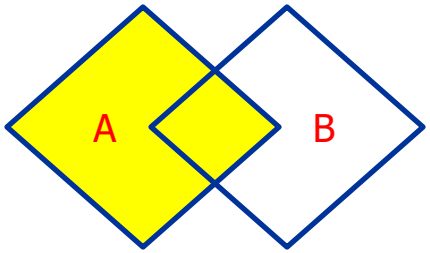
Facilidades Adicionais

A. JOINS

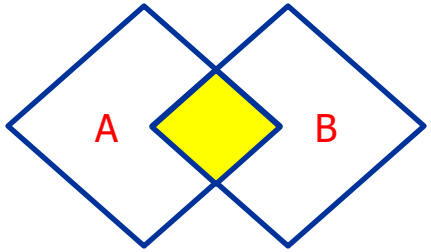
- INNER JOIN: somente pares empregado/depto
- OUTER JOIN: inclui empregados que não têm departamento
- LEFT/RIGHT/FULL OUTER JOIN
- **SELECT** fname, lname, dependent_name
  FROM (EMPLOYEE **LEFT OUTER JOIN**
  DEPENDENT **ON** ssn=essn);

EMPREGADO (ssn, pnome, minicial, unome,
datanasc, endereco, sexo, salario, superssn, dno)
superssn REFERENCIA EMPREGADO
dno REFERENCIA DEPARTAMENTO
DEPARTAMENTO (dnumero, dnome, gerssn,
gerdatainicio)
gerssn REFERENCIA EMPREGADO.ssn

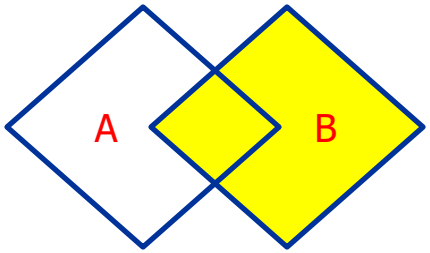
SQL JOINS



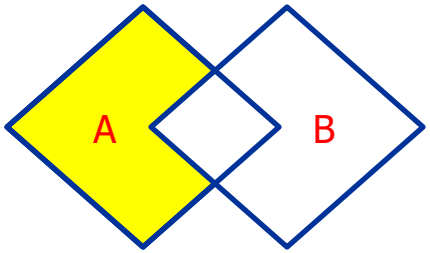

```
SELECT …
FROM A  LEFT JOIN  B
ON A.key = B.key
```



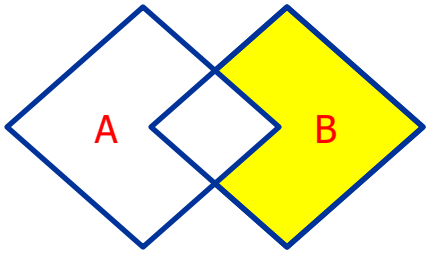

```
SELECT …
FROM A  INNER JOIN  B
ON A.key = B.key
```



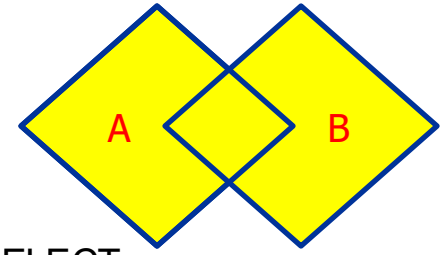

```
SELECT …
FROM A  RIGHT JOIN  B
ON A.key = B.key
```



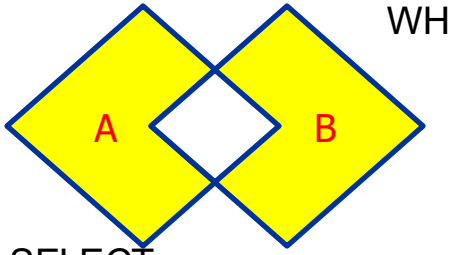

```
SELECT …
FROM A  LEFT JOIN  B
ON A.key = B.key
WHERE B.key IS NULL
```




```
SELECT …
FROM A  RIGHT JOIN  B
ON A.key = B.key
WHERE A.key IS NULL
```




```
SELECT …
FROM A  FULL OUTER JOIN  B
ON A.key = B.key
```




```
SELECT …
FROM A  FULL OUTER JOIN  B
ON A.key = B.key
WHERE A.key IS NULL
      OR B.key IS NULL
```

Facilidades Adicionais

B. FUNÇÕES DE AGREGAÇÃO

- Funções de agregação: **COUNT, SUM, MAX, MIN, AVG**
- **SELECT SUM**(SALARY), **MAX**(SALARY), **MIN**(SALARY), **AVG**(SALARY)
  FROM EMPLOYEE;

- **SELECT SUM**(SALARY), **MAX**(SALARY), **MIN**(SALARY), **AVG**(SALARY)
  FROM EMPLOYEE, DEPARTMENT
  WHERE DNO=DNUMBER **AND** DNAME= 'Research' ;

- **SELECT COUNT**(*)
  FROM EMPLOYEE, DEPARTMENT
  WHERE DNO=DNUMBER **AND** DNAME= 'Research' ;

Consultas Complexas em SQL

- Consultas aninhadas (*fetch* valores existentes)

```
SELECT FNAME, LNAME, ADDRESS
FROM EMPLOYEE
WHERE DNO IN (SELECT DNUMBER
                  FROM DEPARTMENT
                  WHERE DNAME= 'Research' );
```

é equivalente à consulta

```
SELECT FNAME, LNAME, ADDRESS
FROM EMPLOYEE, DEPARTMENT
WHERE DNO=DNUMBER AND DNAME= 'Research' ;
```

EMPREGADO (ssn, pnome, minicial, unome, …superssn, dno)
  superssn REFERENCIA EMPREGADO
  dno REFERENCIA DEPARTAMENTO
DEPARTAMENTO (dnumero, dnome, gerssn, gerdatainicio)
  gerssn REFERENCIA EMPREGADO.ssn

Consultas Complexas em SQL

- Comparação de conjuntos

(Q4A)

```
SELECT DISTINCT PNUMBER
FROM PROJECT
WHERE PNUMBER IN (SELECT PNUMBER
                  FROM PROJECT, DEPARTMENT,
                       EMPLOYEE
                  WHERE DNUM=DNUMBER AND
                        MGRSSN=SSN AND
                        LNAME= 'Smith' )
      OR
      PNUMBER IN (SELECT PNO
                  FROM WORKS_ON, EMPLOYEE
                  WHERE ESSN=SSN AND
                        LNAME= 'Smith' );
```

EMPREGADO (ssn, … superssn, dno)
  superssn REFERENCIA EMPREGADO
  dno REFERENCIA DEPARTAMENTO
DEPARTAMENTO (dnumero, dnome, gerssn, gerdatainicio)
  gerssn REFERENCIA EMPREGADO.ssn
PROJETO (pnumero, pjnome, plocalizacao, dnum)
  dnum REFERENCIA DEPARTAMENTO
TRABALHA_EM (essn, pno, horas)
  essn REFERENCIA EMPREGADO
  pno REFERENCIA PROJETO

A primeira consulta seleciona números de projetos que tem Smith como gerente; a segunda consulta seleciona número dos projetos que tem Smith como empregado.

Consultas Complexas em SQL

- Comparação de conjuntos

```
SELECT DISTINCT ESSN
FROM WORKS_ON
WHERE (PNO, HOURS) IN (SELECT PNO, HOURS
                       FROM WORKS_ON
                       WHERE ESSN='123456789');

SELECT LNAME, FNAME
FROM EMPLOYEE
WHERE SALARY > ALL (SELECT SALARY
                    FROM EMPLOYEE
                    WHERE DNO=5);
```

EMPREGADO (ssn, … superssn, dno)
  superssn REFERENCIA EMPREGADO
  dno REFERENCIA DEPARTAMENTO
TRABALHA_EM (essn, pno, horas)
  essn REFERENCIA EMPREGADO
  pno REFERENCIA PROJETO

Consultas Complexas em SQL

- Uso da função EXISTS

(Q16B)

```
SELECT E.FNAME, E.LNAME
FROM EMPLOYEE AS E
WHERE EXISTS (SELECT *
                FROM DEPENDENT
                WHERE E.SSN=ESSN AND
                      E.SEX=SEX AND
                      E.FNAME=DEPENDENT_NAME);
```

(Q6)

```
SELECT FNAME, LNAME
FROM EMPLOYEE
WHERE NOT EXISTS (SELECT *
                    FROM DEPENDENT
                    WHERE SSN=ESSN);
```

Consultas Complexas: Agrupamento

- Aplicar funções de agregação a subgrupos de tuplas em uma relação
- Exemplo: média de salário em cada departamento
- **SELECT DNO, COUNT(*), AVG(SALARY)**
  FROM EMPLOYEE
  GROUP BY DNO;

(a)

PNOME	MINICIAL	LNOME	SSN	• • •	SALARIO	SUPERSSN	DNO
John	B	Smith	123456789		30000	333445555	5
Franklin	T	Wong	333445555		40000	888665555	5
Ramesh	K	Narayan	666884444		38000	333445555	5
Joyce	A	English	453453453	• • •	25000	333445555	5
Alicia	J	Zelaya	999887777		25000	987654321	4
Jennifer	S	Wallace	987654321		43000	888665555	4
Ahmad	V	Jabbar	987987987		25000	987654321	4
James	E	Bong	888665555		55000	null	1

DNO	COUNT (*)	AVG (SALARIO)
5	4	33250
4	3	31000
1	1	55000

Resultado da Q24

Agrupamento das tuplas EMPREGADO por meio do valor de DNO

Agrupamento

- **SELECT DNO, COUNT(*), AVG(SALARY)**

  FROM EMPLOYEE

  GROUP BY DNO;
- Se um dno = NULL?!
  - Cria-se um grupo separado para todas as tuplas nas quais o atributo é NULL

Agrupamento

- **SELECT** Pnumber, Pname, COUNT(*)
  FROM PROJECT, WORKS_ON
  WHERE Pnumber=Pno
  GROUP BY Pnumber, Pname;
- Neste caso, GROUP BY é aplicado após a junção de PROJECT e WORKS_ON

Agrupamento condicional

- Agrupa as tuplas; dos grupos resultantes, queremos as que satisfazem uma condição
- Agrupamento com a cláusula HAVING
- Somente os grupos que satisfazem a condição especificada em HAVING são retornados
- **SELECT** PNUMBER, PNAME, COUNT(*)
  FROM PROJECT, WORKS_ON
  WHERE PNUMBER=PNO
  GROUP BY PNUMBER, PNAME
  HAVING COUNT(*) > 2;
- WHERE → tuplas; HAVING → grupos de tuplas

PNOME	PNUMERO		ESSN	PNO	HORAS
ProdutoY	2		123456789	2	7,5
ProdutoY	2		453453453	2	20,0
ProdutoY	2		333445555	2	10,0
Automacao	10	. . .	333445555	10	10,0
Automacao	10		999887777	10	10,0
Automacao	10		987987987	10	35,0
Reoganizacao	20		333445555	20	10,0
Reoganizacao	20		987654321	20	15,0
Reoganizacao	20		888665555	20	null
NovosBeneficios	30		987987987	30	5,0
NovosBeneficios	30		987654321	30	20,0
NovosBeneficios	30		999887777	30	30,0

Depois da aplicação da condição da cláusula HAVING

PNOME	COUNT (*)
ProdutoY	3
Automacao	3
Reorganizacao	3
NovosBenefícios	3

Resultado da Q26
(PNUMERO não apresentado)

```
SELECT PNUMBER, PNAME, COUNT(*)
FROM PROJECT, WORKS_ON
WHERE PNUMBER=PNO
GROUP BY PNUMBER, PNAME
HAVING COUNT(*) > 2;
```

Exemplo

- Para cada departamento que tem mais de 5 empregados, retorne o número do departamento e o número de seus empregados que ganham mais de 40.000.

EMPREGADO (ssn, pnome, minicial, unome, …,
salario, superssn, dno)
superssn REFERENCIA EMPREGADO
dno REFERENCIA DEPARTAMENTO
DEPARTAMENTO (dnumero, dnome, gerssn, gerdatainicio)
gerssn REFERENCIA EMPREGADO.ssn

Exemplo

Para cada departamento que tem mais de 5 empregados, retorne o número do departamento e o número de seus empregados que ganham mais de 40.000.

- **SELECT** dnumero, COUNT(*)
  FROM DEPARTAMENTO, EMPREGADO
  WHERE dnumero=dno AND salario>40000
  GROUP BY dnumero
  HAVING COUNT (*) > 5;

ERRADO: Queremos listar o número de empregados (com salario > 40000) dos departamentos com mais de 5 empregados, não dos departamentos com mais de 5 empregados com salario > 40000
EXEMPLO: Um departamento com 7 empregados, dos quais somente 2 têm salario > 40000, é válido!

Exemplo

Para cada departamento que tem mais de 5 empregados, retorne o número do departamento e o número de seus empregados que ganham mais de 40.000.

```
SELECT dnumero, COUNT(*)
        FROM DEPARTAMENTO, EMPREGADO
        WHERE dnumero=dno AND salario>40000
                AND dno IN (SELECT dno
                                FROM EMPREGADO
                                GROUP BY dno
                                HAVING COUNT (*) > 5)
        GROUP BY dnumero;
```

Considerações finais

- As consultas são avaliadas *conceitualmente* na seguinte ordem:
  1. FROM, identifica as tabelas/junções
  2. WHERE
  3. GROUP BY
  4. HAVING
  5. ORDER BY
- Se a consulta não tem group by, having e order by
  1. Para cada combinação de linhas (uma de cada relação especificada em FROM)
  2. Avalia a cláusula WHERE
  3. SE é true, adiciona os atributos de SELECT da combinação de linhas no resultado