JUNÇÃO DE TABELAS (JOIN)

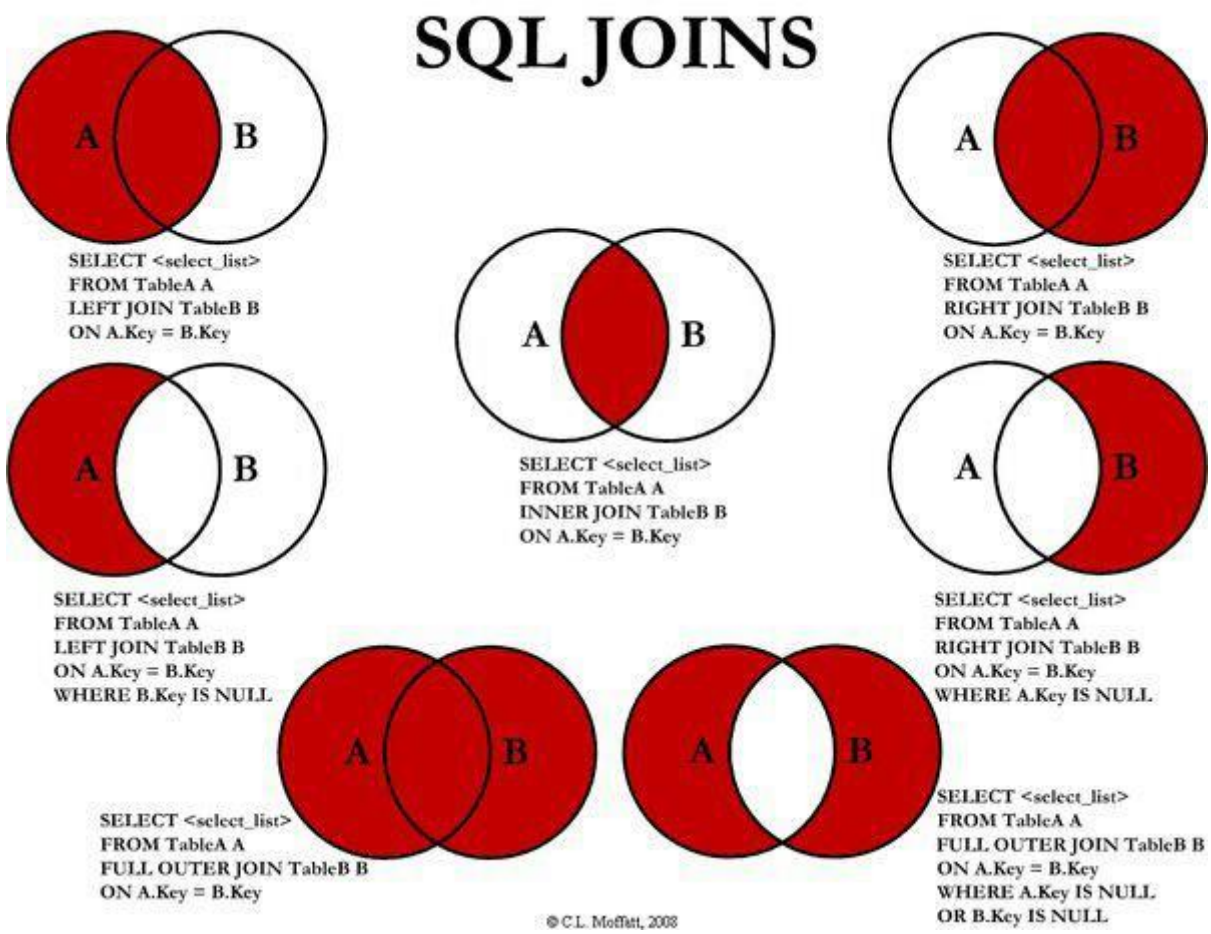

Figura 1 – Tipos de Join – padrão ANSI (por Elâine Okada via Facebook)

Os dados, na maioria das vezes, precisam atender uma determinada solicitação, porém encontram-se em diversas tabelas. Para isto, se faz necessário criar um critério de relacionamento entre eles. Podemos utilizar quantas tabelas forem necessárias, mesmo que uma das tabelas não possua campos a serem mostrados. Podemos usar uma junção para conectar qualquer quantidade de tabelas. E para isto utilizamos a seguinte fórmula:

QUANTIDADE DE JOINS=QUANTIDADE DE TABELAS DA CONSULTA – 1

Como estamos tratando de um banco de dados relacional, frequentemente devemos consultar informações em tabelas distintas. Por exemplo, caso fosse necessário listar o nome dos professores por disciplinas: primeiro dado está em PROFESSOR e o segundo em DISCIPLINA.

As condições de junção são definidas na cláusula WHERE, como qualquer outra condição da consulta, e possuem influência direta sobre a escolha do método de junção que será realizado pelo otimizador. O que as torna diferentes das demais condições, é que elas são utilizadas para realizar a união entre os dados contidos nas diferentes fontes. Para executar a junção, o Oracle combina os registros das fontes, formando um único registro contendo os campos definidos na cláusula SELECT. Conforme as condições de junção definidas pelos desenvolvedores. É possível obter os tipos de junção: EQUIJOIN, OUTERJOIN ou PRODUTO CARTESIANO.

Quando um *join* é omitido ou invalidado, ocorre **PRODUTO CARTESIANO (CROSS JOIN)** onde todas as linhas da primeira tabela foram unidas a todas as linhas da segunda tabela.

Para resolver um comando com mais de uma fonte de dados, o Oracle, primeiramente, determina a melhor forma de acesso aos dados de cada fonte isoladamente. Em seguida, faz a junção das fontes em pares, mesmo que a consulta possua mais fontes. O Oracle determina o par mais vantajoso para iniciar a junção e, após concluir a primeira operação, busca uma nova fonte para realizar a junção com o resultado anterior. Este processo se repete até que todas as fontes tenham sido utilizadas. Exemplo:

```
SELECT e.first_name, e.last_name, d.department_name, j.job_title
  FROM   employees e, departments d, jobs j
 WHERE  e.department_id = d.department_id
   AND   e.job_id              =  j.job_id;
```

No exemplo, para resolver a consulta, o Oracle analisa a melhor forma de acesso aos dados nas tabelas “employees”, “departments” e “jobs”.,Em seguida, procura a melhor ordem de unir estas informações. A melhor ordem encontrada é unir primeiramente as tabelas “jobs” e “employees”. Somente após encontrar o resultado desta junção é que a tabela “departments” será acessada.

As junções são classificadas como:

- **PATENTEADAS PELA ORACLE**: *EquiJoin, NoEquiJoin, OuterJoin e SelfJoin.*
- **COMPATÍVEIS COM O SQL 1999:** Híbridas, naturais, Using, Externas (dos dois lados ou completas) e junções externas.

RELACIONAMENTO EQUIJOIN

É o relacionamento mais usado, reúne campos iguais de tabelas diferentes. Através do relacionamento entre chave primária e chave estrangeiras correspondente. Nesta junção é utilizado o operador de igualdade. Por exemplo, unindo as tabelas PROFESSOR e DISCIPLINA, que têm em comum o campo CODPROF, poderiam ser mostrados os nomes dos professores e respectivas disciplinas que cada professor ministra. Exemplo:

```
SELECT nomeprof, disciplina
  FROM  professor, disciplina
 WHERE professor.codprof = disciplina.codprof;
```

QUALIFICAR COLUNAS

Sempre que existir colunas com o mesmo nome em duas ou mais tabelas unidas, o nome da coluna deverá ser qualificado pelo nome da tabela para especificar qual coluna está sendo mencionada. Nessa instrução SELECT, o nome da coluna ITEM_NUMBER é definido em ambas as tabelas, portanto, ele precisa ser qualificado pelo nome da tabela. Se as colunas tiverem nomes diferentes, nenhuma qualificação será necessária. Ou seja, podemos também usar *alias* ao invés do nome da tabela para identificá-las. Exemplo:

```
SELECT nomeprof, disciplina
  FROM professor a, disciplina b
 WHERE a.codprof = b.codprof;
```

RELACIONAMENTO NOTEQUIJOIN

Usado quando for necessário reunir campos de tabelas que não tenham nada em comum. Ou seja, é utilizada para obter dados de tabelas que não possuem relacionamentos preestabelecidos. Utiliza-se neste tipo de junção os operadores de maior (>), menor (<), maior ou igual (>=), menor ou igual (<=), e BETWEEN...AND. Suponha que existe uma tabela, FAIXA, cujos registros fossem listrados da seguinte maneira:

DESCRIÇÃO	MÍNIMO	MÁXIMO
ALTO	7.00	10.00
MEDIO	4.00	6.99
BAIXO	0	3.99

Exemplo: O comando abaixo classifica os preços por hora de professor na disciplina, segundo sua faixa de preço:

```
SELECT nomeprof, coddisciplina
  FROM professor a, disciplina b, faixa c
 WHERE a.codprof = b.codprof
   AND a.preco_hora BETWEEN mínimo AND maximo;
```

RELACIONAMENTO SELFJOIN OU AUTOJOIN

Útil quando houver dois campos em uma mesma tabela que sejam do mesmo tipo. Retorna linhas unidades na MESMA tabela. E para isto deve-se utilizar um apelido de tabela diferente para identificar cada referência à tabela na consulta. A relação abaixo mostra o pré-requisito de cada curso que possua um. Exemplo:

```
SELECT a.nm_curso, b. nm_curso "PRE_REQUISITO"
  FROM CURSO a, CURSO b
WHERE b.pre_requisito = a.cod_curso;
```

JUNÇÃO HÍBRIDA – CROSS JOIN

Retorna um produto cartesiano (híbrido) e para isso utiliza a cláusula CROSS JOIN. Exemplo: SELECT nomeprof, disciplina

```
        FROM professor
    CROSS JOIN disciplina;
```

JUNÇÃO NATURAL – NATURAL JOIN

Cria uma junção das colunas com o mesmo nome nas tabelas relacionadas, assim, verifica quais são os valores iguais destas colunas. Exemplo:

```
SELECT nomeprof, disciplina
   FROM  professor
NATURAL JOIN disciplina;
```

JUNÇÃO COM USING

Quando as colunas possuem o mesmo nome mas os tipos de dados não forem correspondentes. Usamos a cláusula USING especifica a coluna a ser usada na EQÜIJOIN, assim estabelecemos uma junção com uma coluna somente quando há várias correspondentes. Não podemos usar apelidos de coluna que são usadas na coluna de join.

Exemplo: SELECT p.nomeprof, d.disciplina

FROM professor p **JOIN** disciplina d **USING** (codprof);

JUNÇÃO COM ON

Um dos tipos da NATURAL JOIN. Ajuda a entender o script pois deixa a coluna de junção explicitamente declarada. A condição é separada de outras condições (cláusula WHERE). É um EqüiJoin de todas colunas com o mesmo nome. Exemplo:

```
SELECT p.nomeprof, d.disciplina
  FROM professor p JOIN disciplina d
    ON (p.codprof = d.codprof);
```

JUNÇÕES EXTERNAS

Recupera linhas de uma equijunção, sendo que em alguma das tabelas envolvidas na junção pode não haver registro correspondente. Ou seja, retorna uma linha MESMO QUANDO uma das colunas na condição da junção tenha um valor nulo.

RELACIONAMENTO OUTERJOIN

Além de mostrar registros cujos campos em comum estejam presentes nas duas tabelas, ainda mostra os que faltam. O operador de junção patenteado da Oracle é o (+). Este operador pode aparecer qualquer um dos lados da condição de junção, mas sempre do lado oposto da coluna que contém o valor nulo.

Por exemplo, no EquiJoin mostramos apenas os que possuem disciplinas. No OuterJoin mostramos também quem não possui disciplinas. Exemplo:

```
SELECT nomeprof, disciplina
  FROM  professor a, disciplina b
 WHERE a.codprof = b.codprof(+)
```

Observação: o operador de adição está junto ao campo em cuja tabela "FALTAM" registros, isto é, a tabela relacionada.

JUNÇÃO LEFT OUTER JOIN

Recupera todas as linhas da tabela à esquerda que não há correspondência do lado direito. Exemplo:

```
SELECT nomeprof, disciplina FROM professor a
LEFT OUTER JOIN disciplina b
  ON (a.codprof = b.codprof);
```

JUNÇÃO RIGHT OUTER JOIN

Recupera todas as linhas da tabela à direita que não há correspondência do lado esquerdo. Exemplo:

```
SELECT nomeprof, disciplina
  FROM  professor a RIGHT OUTER JOIN disciplina b
ON (a.codprof = b.codprof);
```

JUNÇÃO FULL OUTER JOIN

Recupera todas as linhas da tabela à direita que não há correspondência do lado esquerdo e, todas as linhas da tabela à esquerda que não tenha correspondência à direita. Exemplo:

```
SELECT nomeprof, disciplina
  FROM  professor a FULL OUTER JOIN disciplina b
ON (a.codprof = b.codprof);
```

EXERCÍCIOS

1. Crie uma consulta para exibir o sobrenome do funcionário, sua matrícula e o nome do departamento que ele está alocado.

2. Crie uma lista única de todos os cargos existentes no departamento 80. Inclua a localização deste departamento.

3. Crie uma consulta para exibir o sobrenome do funcionário, o nome do departamento, a localização e a cidade de todos os funcionários que recebem comissão.

4. Exiba o sobrenome do funcionário e o nome do departamento para todos os funcionários que possuem um *a* em seus sobrenomes.

5. Crie uma consulta para exibir o sobrenome, o cargo, o número e o nome do departamento para todos os funcionários que trabalham em Toronto.

6. Exiba o sobrenome e o número do funcionário junto com o sobrenome e o número do gerente. Coloque um label nas colunas.