Există situații când trebuie să accesăm date din mai multe tabele. În exemplul de mai sus, rezultatul afişat conține date din două tabele separate.
- atributul EMPNO există în tabelul EMP;
- atributul DEPTNO există în ambele tabele EMP şi DEPT;
- atributul LOC există în tabelul DEPT.
Pentru a obține rezultatul dorit trebuie realizată o legătură între tabelele EMP şi DEPT.

Definirea joncțiunilor:

Vom folosi o condiție de joncțiune ori de câte ori trebuie să accesăm date din mai multe tablele din baza d e date. Se poate crea o corespondență între liniile unui tabel şi liniile altui tabel pe baza valorilor comune existente în coloanele corespondente, care sunt de obicei chei primare şi străine. Pentru afişarea datelor din două sau mai multe tabele aflate în relație se va scrie o simplă condiție
de joncțiune în clauza WHERE:

```
SELECT tabel1.coloana1, tabel1.coloana2, tabel2.coloana3
FROM tabel1, tabel2
WHERE tabel1.coloana1 = tabel2.coloana3;
```
unde:
tabel.coloana indică tabelul şi coloana de unde este extrasă data
tabel1.coloana1 = tabel2.coloana2 este condiția care leagă cele două tabele

Observații:
- în momentul scrierii unei expresii SELECT care conține o condiție de joncțiune, este indicat ca numele c oloanelor să fie precedate de numele tabelului de care aparțin; în acest fel este mărită claritatea codului S QL şi se îmbunătățeşte accesul la baza de date.
- pentru a realiza o legătură între n tabele este nevoie de minim n-1 condiții de joncțiune (e.g. pentru a leg a 4 tabele sunt necesare 3 joncțiuni). Această regula s-ar putea să nu se aplice dacă tabelul are o cheie p rimară formată din mai multe atribute şi astfel este necesară mai mult de o coloană pentru a identifica în mod unic fiecare linie.

Produsul Cartezian:

Atunci când o condiție de joncțiune este invalidă sau complet omisă, rezultatul este un produs cartezian în care vor fi afişate toate combinațiile de linii din tabelele implicate. Un produs cartezian tinde să genereze un număr mare de linii, iar rezultatul său este în general nefolositor. De aceea trebuie inclusă întotdeauna o condiție de joncțiune validă în clauza WHERE, cu excepția cazului când se doreşte în mod explicit com binarea tuturor liniilor din tabele implicate în relație.
Exemplul următor afişează numele fiecărui angajat şi numele departamentului în care lucrează din tabelel e EMP şi DEPT. Deoarece nu a fost specificată clauza WHERE, toate liniile (14) din tabelul EMP sunt co mbinate cu toate liniile (4) din tabelul DEPT, generând astfel 56 de linii în tabelul rezultat.

```
SELECT emp.ename, dept.dname FROM emp, dept;
```

Tipuri de condiții de joncțiune:

Principalele tipuri de condiții de joncțiune sunt:
1. echi-joncțiune;
2. non-echi-joncțiune.
Pe lângă acestea mai există şi alte tipuri de condiții de joncțiune:
3. joncțiune externă;
4. joncțiune între un tabel şi el însuşi.

1. Echi-joncțiuni:

Pentru a determina numele departamentului unui angajat trebuie comparată valoarea din coloana DEPTN O din tabelul EMP cu valorile DEPTNO din tabelul DEPT. Legătura astfel creată între tabelele EMP şi DE PT este o echi-joncțiune - valorile din coloana DEPTNO din ambele tabele
trebuie să coincidă.

```
SELECT emp.empno, emp.ename, emp.deptno,dept.deptno, dept.loc
FROM emp, dept
WHERE emp.deptno=dept.deptno;
```

În exemplul de mai sus:
- Clauza SELECT specifică numele coloanelor ce vor fi afişate
- numele şi numărul angajatului şi numărul departamentului, care sunt coloane în tabelul
EMP;
- numărul, numele şi locația departamentului, sunt coloane în tabelul DEPT.
- Clauza FROM specifică cele 2 tabele ce conțin informațiile utile:
- tabelul EMP
- tabelul DEPT
- Clauza WHERE specifică modul în care se va realiza legătura între tabele.

Suplimentar condiției de joncțiune, clauza WHERE poate conține şi alte criterii necesare operației de sele cție a datelor. De exemplu, pentru a afişa codul angajatului KING, numele, numărul şi locația departament ului este nevoie de o condiție suplimentară în clauza WHERE.

```
SELECT emp.empno, emp.ename, dept.deptno, dept.loc
FROM emp, dept
WHERE emp.deptno= dept.deptno AND INITCAP(emp.ename) = 'King';
```

Alias-uri de tabele:

Calificarea coloanelor cu ajutorul numelui tabelului poate consuma mult timp, mai ales în cazul în care nu mele tabelului este un şir lung. Pentru simplificarea codului SQL se pot folosi alias-uri de tabele în locul n umelor acestora.

```
SELECT e.empno, e.ename, e.deptno, d.deptno, d.loc
FROM emp e, dept d
WHERE e.deptno=d.deptno;
```

Observatii:
- Alias-urile de tabel pot avea lungimea maximă de 30 caractere;
- dacă în clauza FROM se introduce un alias pentru tabel, atunci acel alias trebuie să înlocuiască numele tabelului în toată instrucțiunea SELECT;
- alias-urile de tabel ar trebui sa fie semnificative;
- alias-ul unui tabel este valid numai pentru SELECT-ul curent.

2. Non-echi-joncțiuni:

Relația dintre tabelul EMP si SALGRADE este o non-echi-joncțiune, în sensul că nici o coloana din tabelul EMP nu corespunde direct unei coloane din tabelul SALGRADE. Legătura dintre cele două tabele este ur mătoarea: coloana SAL din EMP este cuprinsă între coloanele LOSAL şi HISAL din tabelul SALGRADE. Legătura se obține folosind un operator, altul decât operatorul egal (‘=’).

```
CREATE TABLE salgrade ( grade NUMBER, losal NUMBER, hisal NUMBER );

INSERT INTO salgrade VALUES (1, 700, 1200);
INSERT INTO salgrade VALUES (2, 1201, 1400);
```

```
INSERT INTO salgrade VALUES (3, 1401, 2000);
INSERT INTO salgrade VALUES (4, 2001, 3000);
INSERT INTO salgrade VALUES (5, 3001, 9999);

SELECT emp.ename, emp.sal, salgrade.grade
FROM emp, salgrade
WHERE emp.sal BETWEEN salgrade.losal AND salgrade.hisal;
```

Exemplul de mai sus crează o non-echi-joncțiune pentru a determina gradul de salarizare al unui angajat. Salariul trebuie să fie între (between) limita inferioară (LOSAL) şi cea superioară (HISAL) a unui nivel de s alarizare.

3. Joncțiune externă:

Dacă o linie (înregistrare) nu satisface condiția de joncțiune, acea linie nu va apare în rezultatul interogării . De exemplu, în condiția de echi-joncțiune a tabelelor EMP şi DEPT, departamentul OPERATIONS nu va apare pentru că nu există nici o persoană care lucrează în acel departament.
Liniile lipsă pot fi returnate dacă în condiția de joncțiune se utilizează operatorul de joncțiune externă. Ope ratorul este semnul "+" între paranteze - (+) şi este plasat în acea parte a condiției de joncțiune corespunz ătoare tabelului deficient în informații. Acest operator are ca efect crearea uneia sau a mai multor linii nule la care se poate adaugă una sau mai multe linii din tabelul ce conține
informațiile neselectate.

```
SELECT tabel.coloana1, tabel.coloana2, ...
FROM tabel1, tabel2
WHERE tabel1.coloana1 = tabel2.coloana2(+) ;
```
unde:
tabel1.coloana1 = este condiția care realizează legătura între tabele
tabel2.coloana2(+) este simbolul pentru joncțiune externă; poate fi plasat in oricare parte a clauzei WHER E, dar nu în ambele părți simultan. Se va plasa operatorul de joncțiune externă după numele coloanei din tabelul deficitar în informații.

Următorul exemplu afişează toate numerele şi numele departamentelor. Departamentul OPERATIONS, c are nu are nici un angajat este de asemenea afişat.
```
SELECT e.ename, d.deptno, d.dname
FROM emp e, dept d
WHERE e.deptno(+) = d.deptno
ORDER BY e.deptno ;
```

Restricții în cazul utilizării unei condiții de joncțiune externă:
- operatorul de joncțiune externă poate apare numai într-o singură parte a unei expresii - partea care nu d eține informația ce nu este returnată de interogare. El returnează acele linii din tabel care nu au corespon dent direct în celălalt tabel.
- o condiție ce implică operatorul de joncțiune externă nu poate utiliza operatorul IN şi nici nu poate fi lega tă de o altă condiție prin operatorul OR.

4. Condiții de joncțiune ale unui tabel cu el însuşi:

"MGR din tabelul WORKER este egal cu EMPNO din tabelul MANAGER"

Unele aplicații impun realizarea unei joncțiuni între un tabel şi el însuşi. De exemplu, pentru a găsi numele managerului lui Blake va trebui să:
- găsim înregistrarea corespunzătoare angajatului Blake în tabelul EMP, inspectând coloana ENAME;
- găsim codul managerului lui Blake din coloana MGR. Codul managerului lui Blake este 7839.
- găsim numele managerului având codul EMPNO egal cu 7893 în coloana ENAME. Codul angajatului Ki

ng este 7839. Deci King este managerul lui Blake.

Pe parcursul acestui proces am căutat în tabel de două ori. Prima dată pentru a-l găsi pe Blake în coloan a ENAME şi pentru a citi valoarea MGR - 7839. A doua oară am căutat în coloana EMPNO valoarea 7839 şi am extras din coloana ENAME valoarea King.

```
SELECT worker.ename || 'works for' || manager.ename
FROM emp worker, emp manager
WHERE worker.mgr = manager.empno;
```

Exemplul de mai sus crează o legătură între tabelul EMP şi el însuşi. Pentru a simula două tabele în clauz a FROM, se folosesc două aliasuri, numite WORKER şi MANAGER pentru acelaşi tabel EMP. În acest ex emplu, clauza WHERE conține condiția de joncțiune care înseamnă "pentru care codul managerului unui angajat coincide cu codul de angajat al managerului".