Hier ein einfaches (INNER) JOIN Beispiel (INNER ist beim JOIN der Default):

```
SELECT DISTINCT
       c.department
FROM Project p JOIN p.customer c
WHERE (p.name = 'Project1'
  AND  e.address.state = 'NRW')
```

Hier anders formuliert als mehrfaches JOIN Beispiel:

```
SELECT DISTINCT d
FROM Project p JOIN p.customers c
               JOIN c.department d
               JOIN c.address a
WHERE (p.name = 'Project1' AND a.state = 'NRW')
```

Hier ein anderes Beispiel zur Verwendung mehrfacher JOINs im FROM Teil:

```
SELECT DISTINCT p
FROM Department d JOIN d.customers c
                  JOIN c.projects p
WHERE (p.name = 'Project2' AND p.startdate IS SYSDATE)
```

Hier noch ein Beispiel zur Verwendung des “IS NOT EMPTY“ Operators:

```
SELECT d, c
FROM Department d,
     Employee c
WHERE (d = c.department AND c.phonenumbers IS NOT EMPTY)
```

Während einer Performance Tuning Phase ist diese Schreibweise eines JPQL JOIN Statements eher sprechender:

```
SELECT DISTINCT
       d.id,
       d.name
FROM project p,
     customerprojects cps,
     customer c,
     department d,
     address a
WHERE (p.id = cps.project_id
  AND  cps.cust_id = c.id
  AND  c.dept_id = d.id
  AND  c.address_id = a.id
  AND  p.name = 'Project1'
  AND  a.state = 'NRW')
```

Hier die Beispiele für Map JOINs unter Verwendung der Schlüsselwörter KEY, VALUE:

```
SELECT c.name,
       VALUE(p)
FROM Customer c JOIN c.phones p

SELECT c.name,
       KEY(p),
       VALUE(p)
FROM Customer (AS) c JOIN c.phones p
WHERE KEY(p) IN ('Festnetz', 'HandyBeruf', 'HandyPrivat')
```

Hier ein Beispiel zum LEFT/RIGHT OUTER JOIN:

```
SELECT cust, ord
FROM Customer (AS) cust LEFT/RIGHT OUTER JOIN cust.orders (AS) ord
WHERE (ord.delivery = SYSDATE)
```

Hier ein Beispiel zum LEFT/RIGHT OUTER JOIN mit FETCH Option (JOIN-Default ist immer LEFT INNER):

```
SELECT cust, ord
FROM Customer (AS) cust
LEFT/RIGHT OUTER JOIN FETCH cust.orders (AS) ord
JOIN FETCH cust.bills (AS) bill
WHERE (ord.delivery = SYSDATE)
```