

Join Example

Owner Table

owner_id	name
ABCD1234	Mr John Smith
EFGH5678	Lord Realestate
IJKL9012	Mega Corporation

Property Table

owner_id	property_i	town	property_type
EFGH5678	1	Banbury	1
EFGH5678	2	Banbury	1
EFGH5678	3	Banbury	1
EFGH5678	4	Banbury	1
IJKL9012	FLT001	London	2
IJKL9012	FLT002	London	2
IJKL9012	FLT003	London	2
IJKL9012	OFF101	London	4

```

SELECT  own.name,
        pro.town
FROM    owner      own,
        property   pro
WHERE   own.owner_id = pro.owner_id /* join */
AND     own.owner_id = 'IJKL9012' /* rest */
AND     pro.property_type = 4 /* rest */
    
```

name	town
Mega Corporation	London

Join Example – Cart Prod

```

SELECT  pro.owner_id, own.name,
        pro.property_id, pro.town
FROM    owner      own,
        property   pro
WHERE   own.owner_id = 'IJKL9012' /* rest */
    
```

Owner Table

owner_id	name
ABCD1234	Mr John Smith
EFGH5678	Lord Realestate
IJKL9012	Mega Corporation

Property Table

owner_id	property_i	town	property_type
EFGH5678	1	Banbury	1
EFGH5678	2	Banbury	1
EFGH5678	3	Banbury	1
EFGH5678	4	Banbury	1
IJKL9012	FLT001	London	2
IJKL9012	FLT002	London	2
IJKL9012	FLT003	London	2
IJKL9012	OFF101	London	4

Result

owner_id	name	property_i	town
EFGH5678	Mega Corporation	1	Banbury
EFGH5678	Mega Corporation	2	Banbury
EFGH5678	Mega Corporation	3	Banbury
EFGH5678	Mega Corporation	4	Banbury
IJKL9012	Mega Corporation	FLT001	London
IJKL9012	Mega Corporation	FLT002	London
IJKL9012	Mega Corporation	FLT003	London
IJKL9012	Mega Corporation	OFF101	London

- Previous joins are *equijoins* (=)
- Other operators can be used
e.g. List all my employees older than their manager

```
SELECT emp.name
FROM employee emp,
      manager man
WHERE emp.manager_name = man.name
AND emp.date_of_birth > man.date_of_birth
```

- != to find rows outside the natural join
 - Unlikely to be used on its own as it would generate a semi-cartesian product.
 - Join of all rows to all other rows except for the ones where the columns match

```
SELECT emp.name
FROM employee emp,
      manager man
WHERE emp.manager_name = man.name
AND emp.date_of_birth != man.date_of_birth
```

- Correlation names differentiate between each instance of the table
- They are two separate data sources
- E.g. Find all properties with the same postcode as the selected property

```
SELECT prp2.property_id
FROM property
      property
WHERE prp1.property_id
AND prp1.postcode
```

```
prp1,
prp2
= 'ABCD1234'
= prp2.postcode
```

Join the two instances

Restriction on first instance of property

Sub-Select Joins

- Right hand side can be a sub-select

- *expr operator (sub-select)*

```
-- Select Owners with properties in London
SELECT name
FROM owner
WHERE owner_id IN (SELECT owner_id
                   FROM property
                   WHERE town = 'London')

--
-- Select properties with a price > the average
SELECT property_id
FROM property
WHERE price > (SELECT avg(price)
               FROM property)
```

- Sub-select evaluated first
- Sub-select must return one value for comparisons (=, !=, > etc)

Correlated Sub-Queries

- Inner or sub-query references outer query

- *Test for failed joins - Outer Joins are better (later)*

```
SELECT customer_id, name
FROM customer cus
WHERE NOT EXISTS
      (SELECT * FROM requirement req
       WHERE req.customer_id = cus.customer_id)
```

- *Selection based on an aggregate restriction*

```
SELECT order_number
FROM order ord
WHERE customer_id = 1234
AND order_date = (SELECT max(ord2.order_date)
                 FROM order ord2
                 WHERE ord.customer_id =
                 ord2.customer_id)
```

Join – Alternative Syntax

- Introduced to support outer joins
- Improves query construction
- New syntax in the FROM clause

```
source join-type join source
      on join-condition
```
- Produces intermediate result tables
- Allows nesting of joins
- Default *join-type* is inner
- Others LEFT | RIGHT | FULL [OUTER]

Join – Alternative Syntax

- Traditional Syntax

```
SELECT      own.name ,
            pro.town
FROM        owner own,
            property pro
WHERE       own.owner_id = pro.owner_id /* join */
AND         own.owner_id = 'IJKL9012' /* rest */
AND         pro.property_type = 4 /* rest */
```

- New Syntax

```
SELECT      own.name ,
            pro.town
FROM        owner own
            JOIN property pro
            ON own.owner_id = pro.owner_id /* join */
WHERE       own.owner_id = 'IJKL9012' /* rest */
AND         pro.property_type = 4 /* rest */
```

- Don't mix restrictions and Joins in the ON clause

Joins – Alternative Syntax

- New syntax allows nesting of joins
- Example
 - `(s1 join s2 on s1.col = s2.col) join (s3 join s4 on s3.col = s4.col) on s1.col = s3.col`
 - S1 joined to s2
 - S3 joined to s4
 - Two intermediate results are joined using `s1.col = s3.col`
- Restrictions in the ON clause do not apply to the final result set
 - Restriction is used to evaluate the join
 - Example coming up
- CROSS Join
 - Cartesian Product of the 2 tables
 - A CROSS JOIN B

Joins – Alternative Syntax

- USING clause can be used instead of ON clause
 - Tables must have identical column names to join on

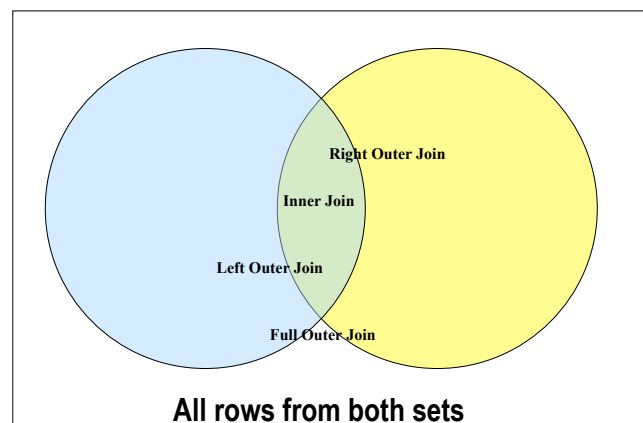
```
SELECT      own.name,  
            pro.town  
FROM        owner own  
            JOIN property pro  
            USING owner_id /* join */  
WHERE       own.owner_id = 'IJKL9012' /* rest */  
AND         pro.property_type = 4 /* rest */
```

- Multiple column names allowed
USING col1, col2, col3...

- **Outer joins return 2 sets of data**
 - Rows where the join succeeded with values from both tables
 - Rows where the join failed with null values from one of the tables.
- **LEFT [OUTER] JOIN**
 - Joins from Left source to right source
 - Returns nulls on the right if join fails
- **RIGHT [OUTER] JOIN**
 - Joins from Right source to Left source
 - Return nulls on the left if join fails
- **Join failure can be tested in the WHERE clause**
 - `WHERE column IS NULL`
 - Useful for testing for non-existence
 - Single SQL to join to multiple 'specialization' tables

Left/Right Outer Join

All rows from One set plus the rows that join in the other set



Outer Joins

- A left join B
same as
B right join A
- FULL [OUTER] JOIN
 - Union of a left join and a right join
 - Nulls from both tables as required
 - Can't have a completely null row
- Join conditions can include ordinary restrictions
 - Example: price < 500.00
 - Affects whether null values are returned, not the numbers of rows returned by the join

Outer Join Examples

- Restriction in the ON clause

```
SELECT own.name,  
       pro.town  
FROM   owner          own  
LEFT JOIN  
       property       pro  
ON     own.owner_id  = pro.owner_id  
AND    pro.price     < 500.00
```

name	town
Mr John Smith	
Lord Realestate	Banbury
Lord Realestate	Banbury
Lord Realestate	Banbury
Lord Realestate	Banbury
Mega Corporation	

- Restriction in the WHERE clause

```
SELECT own.name,  
       pro.town  
FROM   owner          own  
LEFT JOIN  
       property       pro  
ON     own.owner_id  = pro.owner_id  
WHERE  pro.price     < 500.00
```

name	town
Lord Realestate	Banbury
Lord Realestate	Banbury
Lord Realestate	Banbury
Lord Realestate	Banbury

Outer Join Examples

- Check for failed join

```
SELECT own.owner_id,  
       own.name  
FROM   owner      own  
LEFT JOIN  
       property   pro  
ON     own.owner_id = pro.owner_id  
WHERE  pro.owner_id IS NULL
```

owner_id	name
ABCD1234	Mr John Smith

Outer Join Examples

- Combined Syntax

```
SELECT own.name,  
       pro.town,  
       hou.number_of_beds  
FROM   owner      own  
LEFT JOIN  
       property   pro  
ON     own.owner_id = pro.owner_id, /* comma ends outer join */  
       house      hou  
WHERE  pro.owner_id = hou.owner_id /* house join in where clause */  
AND    pro.property_id = hou.property_id
```

owner_id	name	owner_id
ABCD1234	Mr John Smith	
EFGH5678	Lord Realestate	EFGH5678
EFGH5678	Lord Realestate	EFGH5678
EFGH5678	Lord Realestate	EFGH5678
EFGH5678	Lord Realestate	EFGH5678
IJKL9012	Mega Corporation	IJKL9012
IJKL9012	Mega Corporation	IJKL9012
IJKL9012	Mega Corporation	IJKL9012
IJKL9012	Mega Corporation	IJKL9012
XYZZ9999	Mr W.O.N. Lottery	XYZZ9999