5 Joining Tables

5.1 Cartesian Product

5.1.1 Multiple tables in the FROM clause

There may be instances when a query requires information from two or more tables to be collated in some way, for example, to find out who owns each property. This would require the information in the **`property`** table to be combined with information from the **`owner`** table.

To two or more tables together, SQL implements the equivalent of the mathematical operation `Cartesian Product.` This is written in SQL by listing the tables to be connected in the `FROM` clause, separated by commas.

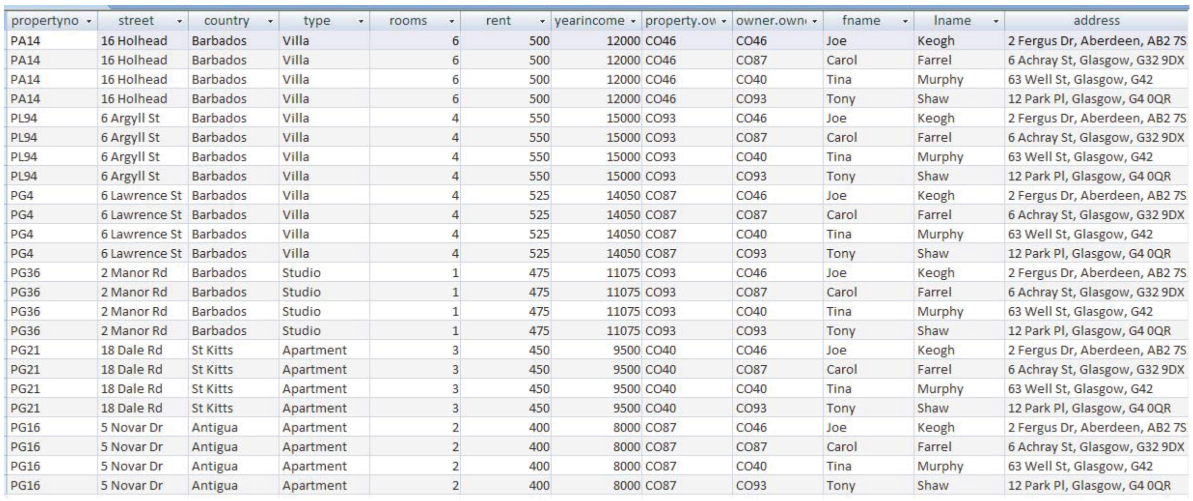
A multi-table `FROM` statement is of the form:

```
FROM tablename [,tablename [,tablename …
```

A query to combine the **`property`** and **`owner`** tables would look like this:

```
SELECT *
FROM property, owner;
```

This would produce the following results:



propertyno	street	country	type	rooms	rent	yearincome	property.ow	owner.own	fname	lname	address
PA14	16 Holhead	Barbados	Villa	6	500	12000	CO46	CO46	Joe	Keogh	2 Fergus Dr, Aberdeen, AB2 7S
PA14	16 Holhead	Barbados	Villa	6	500	12000	CO46	CO87	Carol	Farrel	6 Achray St, Glasgow, G32 9DX
PA14	16 Holhead	Barbados	Villa	6	500	12000	CO46	CO40	Tina	Murphy	63 Well St, Glasgow, G42
PA14	16 Holhead	Barbados	Villa	6	500	12000	CO46	CO93	Tony	Shaw	12 Park Pl, Glasgow, G4 0QR
PL94	6 Argyll St	Barbados	Villa	4	550	15000	CO93	CO46	Joe	Keogh	2 Fergus Dr, Aberdeen, AB2 7S
PL94	6 Argyll St	Barbados	Villa	4	550	15000	CO93	CO87	Carol	Farrel	6 Achray St, Glasgow, G32 9DX
PL94	6 Argyll St	Barbados	Villa	4	550	15000	CO93	CO40	Tina	Murphy	63 Well St, Glasgow, G42
PL94	6 Argyll St	Barbados	Villa	4	550	15000	CO93	CO93	Tony	Shaw	12 Park Pl, Glasgow, G4 0QR
PG4	6 Lawrence St	Barbados	Villa	4	525	14050	CO87	CO46	Joe	Keogh	2 Fergus Dr, Aberdeen, AB2 7S
PG4	6 Lawrence St	Barbados	Villa	4	525	14050	CO87	CO87	Carol	Farrel	6 Achray St, Glasgow, G32 9DX
PG4	6 Lawrence St	Barbados	Villa	4	525	14050	CO87	CO40	Tina	Murphy	63 Well St, Glasgow, G42
PG4	6 Lawrence St	Barbados	Villa	4	525	14050	CO87	CO93	Tony	Shaw	12 Park Pl, Glasgow, G4 0QR
PG36	2 Manor Rd	Barbados	Studio	1	475	11075	CO93	CO46	Joe	Keogh	2 Fergus Dr, Aberdeen, AB2 7S
PG36	2 Manor Rd	Barbados	Studio	1	475	11075	CO93	CO87	Carol	Farrel	6 Achray St, Glasgow, G32 9DX
PG36	2 Manor Rd	Barbados	Studio	1	475	11075	CO93	CO40	Tina	Murphy	63 Well St, Glasgow, G42
PG36	2 Manor Rd	Barbados	Studio	1	475	11075	CO93	CO93	Tony	Shaw	12 Park Pl, Glasgow, G4 0QR
PG21	18 Dale Rd	St Kitts	Apartment	3	450	9500	CO40	CO46	Joe	Keogh	2 Fergus Dr, Aberdeen, AB2 7S
PG21	18 Dale Rd	St Kitts	Apartment	3	450	9500	CO40	CO87	Carol	Farrel	6 Achray St, Glasgow, G32 9DX
PG21	18 Dale Rd	St Kitts	Apartment	3	450	9500	CO40	CO40	Tina	Murphy	63 Well St, Glasgow, G42
PG21	18 Dale Rd	St Kitts	Apartment	3	450	9500	CO40	CO93	Tony	Shaw	12 Park Pl, Glasgow, G4 0QR
PG16	5 Novar Dr	Antigua	Apartment	2	400	8000	CO87	CO46	Joe	Keogh	2 Fergus Dr, Aberdeen, AB2 7S
PG16	5 Novar Dr	Antigua	Apartment	2	400	8000	CO87	CO87	Carol	Farrel	6 Achray St, Glasgow, G32 9DX
PG16	5 Novar Dr	Antigua	Apartment	2	400	8000	CO87	CO40	Tina	Murphy	63 Well St, Glasgow, G42
PG16	5 Novar Dr	Antigua	Apartment	2	400	8000	CO87	CO93	Tony	Shaw	12 Park Pl, Glasgow, G4 0QR

With many more fields not shown.

Activity: Try this query out and verify that it performs as suggested

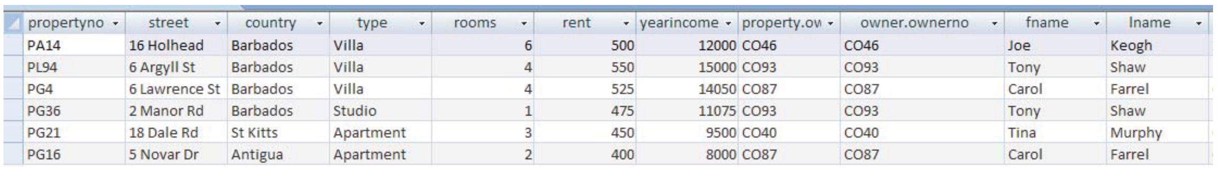
In fact, the Cartesian Product merges every row from every table together – the **`property`** table contains 6 records, the `OWNER` table contains 4 records, hence the final query results in 6*4 = 24 records. The database cannot assume (or deduce) how the table should be connected. To do this an extra term must be added to the `WHERE` clause, specifying which two (or more) columns should contain values that are to be compared in some way (typically using equality '=').

With this example, it is clear that there are two columns that should be matched to produce the correct results – the **`property`** table contains an **`ownerno`** column, the **`owner`** table contains an **`ownerno`**. Only those rows which contain the same value should be retained.

A correct version of the `SELECT` clause for this example would look like this:

```
SELECT *
FROM property, owner
WHERE property.ownerno=owner.ownerno;
```

Notice how we have to refer to the different **`ownerno`** attributes using the **`tablename.columnname`** syntax. This is required as the two tables share a same named column heading `OWNERNO`.



propertyno	street	country	type	rooms	rent	yearincome	property.ow	owner.ownerno	fname	lname
PA14	16 Holhead	Barbados	Villa	6	500	12000	CO46	CO46	Joe	Keogh
PL94	6 Argyll St	Barbados	Villa	4	550	15000	CO93	CO93	Tony	Shaw
PG4	6 Lawrence St	Barbados	Villa	4	525	14050	CO87	CO87	Carol	Farrel
PG36	2 Manor Rd	Barbados	Studio	1	475	11075	CO93	CO93	Tony	Shaw
PG21	18 Dale Rd	St Kitts	Apartment	3	450	9500	CO40	CO40	Tina	Murphy
PG16	5 Novar Dr	Antigua	Apartment	2	400	8000	CO87	CO87	Carol	Farrel

Here it is clear from the two **`ownerno`** columns that the records have been matched in the correct sequence.

Activity: Try out the refined query and verify that it produces this result.

```
SELECT *
FROM property, owner
WHERE property.ownerno=owner.ownerno;
```

To join *three* tables it is normally necessary to specify *two* join conditions, *four* tables usually require *three* join conditions etc.

Temporary labels may be specified in the `FROM` clause to save repeating the full table names repeatedly in the query:

```
SELECT *
FROM property p, owner o
WHERE p.ownerno=o.ownerno;
```

Other Boolean operators to specify a further subset of the selection can be appended as normal:

```
SELECT *
FROM property p, owner o
WHERE p.ownerno=o.ownerno AND p.type="Villa";
```

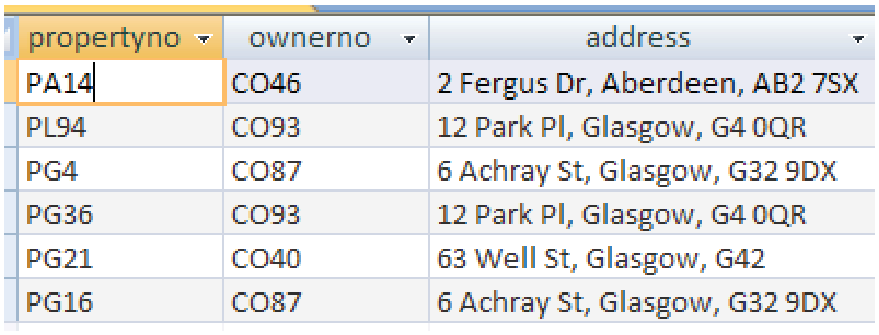
5.1.2 Join and Cartesian Product Examples

Activity: Type these example queries in and verify that they produce similar results.

Examples

Display property number, owner number and the address of the owner.

```
SELECT propertyno, owner.ownerno, address
FROM property, owner
WHERE property.ownerno=owner.ownerno;
```

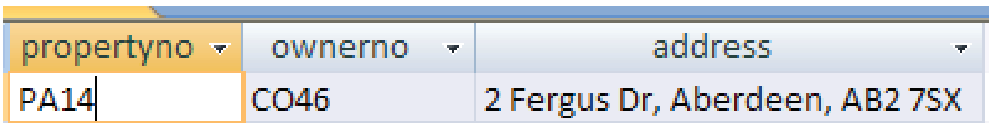


propertyno	ownerno	address
PA14	CO46	2 Fergus Dr, Aberdeen, AB2 7SX
PL94	CO93	12 Park Pl, Glasgow, G4 0QR
PG4	CO87	6 Achray St, Glasgow, G32 9DX
PG36	CO93	12 Park Pl, Glasgow, G4 0QR
PG21	CO40	63 Well St, Glasgow, G42
PG16	CO87	6 Achray St, Glasgow, G32 9DX

Note that the **ownerno** must be prefixed with the tablename **owner** – this is to clarify which department number is included in the output.

Find out where the Mr Keogh is based.

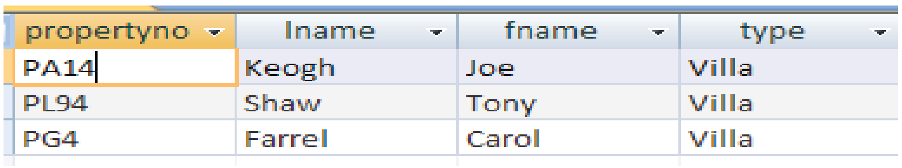
```
SELECT propertyno, owner.ownerno, address
FROM property, owner
WHERE property.ownerno=owner.ownerno
AND lname="Keogh";
```



propertyno	ownerno	address
PA14	CO46	2 Fergus Dr, Aberdeen, AB2 7SX

Find all the property numbers and their owner names that are Villas.

```
SELECT propertyno, lname, fname, type
FROM property, owner
WHERE property.ownerno=owner.ownerno
AND type="Villa";
```



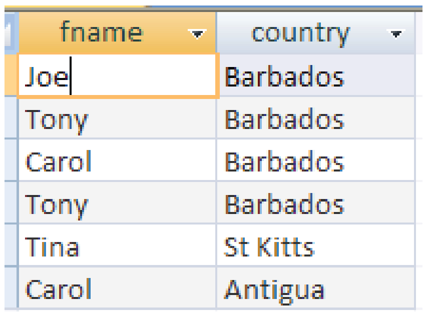
propertyno	lname	fname	type
PA14	Keogh	Joe	Villa
PL94	Shaw	Tony	Villa
PG4	Farrel	Carol	Villa

5.2 Exercises – Join, Selection and Projection

In the following exercises, the query must be specified to produce the suggested result. There are spaces for you to write the SQL query. Use the AS command to get correct column headings in SQL.

1. Display the owner first name and country of their properties.

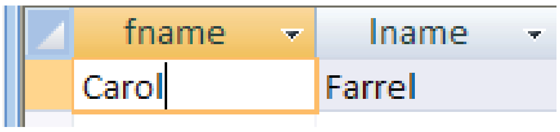
SQL:



fname	country
Joe	Barbados
Tony	Barbados
Carol	Barbados
Tony	Barbados
Tina	St Kitts
Carol	Antigua

2. Display the owner first and last name for Apartments in Antigua.

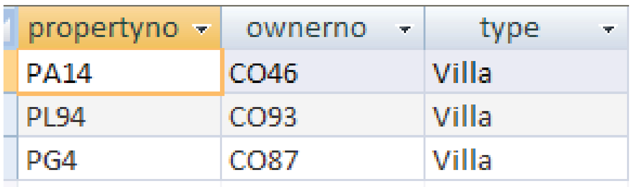
SQL:



fname	lname
Carol	Farrel

3. Display the property number and owner number for all Villas.

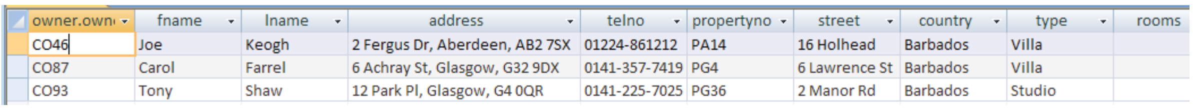
SQL:



propertyno	ownerno	type
PA14	CO46	Villa
PL94	CO93	Villa
PG4	CO87	Villa

4. Display the properties and their owner details with a rental income between £460 and £530.

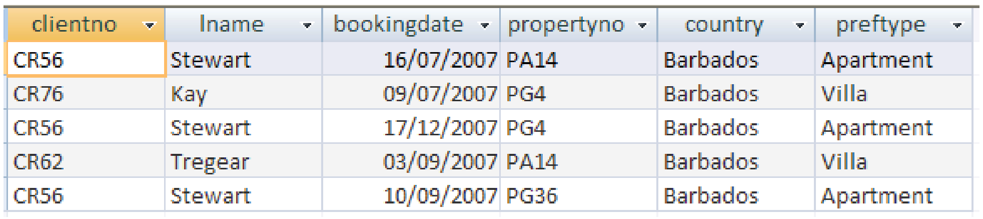
SQL:



owner.own	fname	lname	address	telno	propertyno	street	country	type	rooms
CO46	Joe	Keogh	2 Fergus Dr, Aberdeen, AB2 7SX	01224-861212	PA14	16 Holhead	Barbados	Villa	
CO87	Carol	Farrel	6 Achray St, Glasgow, G32 9DX	0141-357-7419	PG4	6 Lawrence St	Barbados	Villa	
CO93	Tony	Shaw	12 Park Pl, Glasgow, G4 0QR	0141-225-7025	PG36	2 Manor Rd	Barbados	Studio	

5. Display the employee client number, client last name, their booking date, the property number of the booked property, the country that the booked property is in and the clients preference type (Hint – you need to use 3 tables)

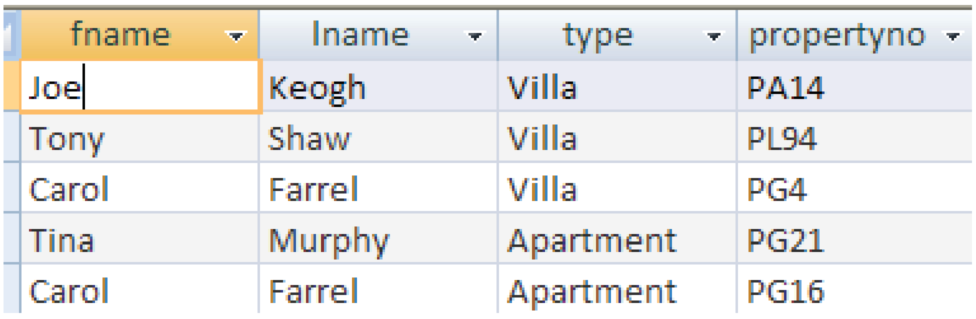
SQL:



clientno	lname	bookingdate	propertyno	country	preftype
CR56	Stewart	16/07/2007	PA14	Barbados	Apartment
CR76	Kay	09/07/2007	PG4	Barbados	Villa
CR56	Stewart	17/12/2007	PG4	Barbados	Apartment
CR62	Tregear	03/09/2007	PA14	Barbados	Villa
CR56	Stewart	10/09/2007	PG36	Barbados	Apartment

6. Display all the owners first, last names, type and property number whose property type are Villa and Apartments (Hint – you need to use brackets in the WHERE clause after the AND)

SQL:



fname	lname	type	propertyno
Joe	Keogh	Villa	PA14
Tony	Shaw	Villa	PL94
Carol	Farrel	Villa	PG4
Tina	Murphy	Apartment	PG21
Carol	Farrel	Apartment	PG16

5.3 Summary

More often than not SQL queries need to combine information from two or more tables. To join records two or more tables together, SQL implements the equivalent of the mathematical operation `Cartesian Product.` This is written in SQL by listing the tables to be connected in the `FROM` clause, separated by commas.

A multi-table `FROM` statement is of the form:

```
FROM tablename [,tablename [,tablename …]]
```

Recall that the general form for the complete SELECT statement is:

```
SELECT [DISTINCT | ALL] {*| column [AS new_name]] [, …]}
FROM Tablename [alias] [, tablename]…
[WHERE conditional statement]
[GROUP BY column_list] [HAVING condition]
[ORDER BY column_list]
```

When records are retrieved from more than one table the `WHERE` clause MUST contain a conditional statement indicating the keys which the tables are joined on. For example;

```
SELECT *
FROM property, owner
WHERE property.ownerno=owner.ownerno;
```