Lab 6

Objectives

In this lab, we will use the Select statement statement to look at multi table and outer Joins.

- Retrieve data from more than two tables.
- Perform other types of Join.

Join

Sometimes we need to retrieve information from more than two tables. The rule goes that the number of *JOINs* a Select statement has is 1 less than the number of tables involved. So if we have three tables in the query then we have two JOINs, if we have four tables in the query then we would have 3 JOINs and so on.

Example One

For each loan, if we wanted the student's name who borrowed the book, the ISBN of the book borrowed, and the dates loaned and returned.

There are three things to do here:

1. Identify which tables have the data you are looking for, student name is in the Student table, ISBN is in the Book and BookCopy tables, loan details are in the Loan table. Since tables Loan and BookCopy can be joined, we will use BookCopy instead of table Book. The tables required are: Student, Loan and BookCopy. Since there are 3 tables, we need 2 joins.

2. Identify which columns are the primary and foreign keys in these tables. To join the Student and Loan tables, we use the studentId in the Student and studentId in the Loan table. To join the BookCopy and Loan tables we use the copyId in the BookCopy table and copyId in the Loan table. We use these in the ON clause.

3. Identify which columns we want the query to output; in this case it is fName, lName from the Student table, ISBN from the BookCopy table, dateOut and dateBack from the Loan table. We will use these in our SELECT.

```
SELECT CONCAT(fname,' ',lname) AS Name, isbn, dateOut, dateBack
FROM bookcopy JOIN loan
ON  bookcopy.copyId = loan.copyId
JOIN student
ON loan.studentId = student.studentId;
```

Example Two

The previous example gives us some information about the book loans per student but we still do not know the book title!

We can expand the above example to return the book title instead of the ISBN. For each loan, we want the student's name who borrowed the book, the title of the book borrowed, and the dates loaned and returned.

There are three things to do here:

1. Identify which tables have the data you are looking for, student name is in the *Student* table, Book Title is in the *Book* table, loan details are in the *Loan* table. Since table Book is not related to any of these tables, we also require BookCopy as it has ISBN as a foreign key and BookCopy can be joined with Loan. The tables required are: *Student*, *Book*, *Loan* and *BookCopy*. Since there are 4 tables, we need 3 joins.

2. Identify which columns are the primary and foreign keys in these tables. To join the Student and Loan tables, we use the *studentId* in the *Student* and *studentId* in the *Loan* table. To join the BookCopy and Loan tables we use the *copyId* in the *BookCopy* table and *copyId* in the *Loan* table. To join the BookCopy and Book tables we use the *ISBN* in the *BookCopy* table and *ISBN* in the *Book* table. We use these in the *ON* clause.

3. Identify which columns we want the query to output; in this case it is *fName*, *lName* from the Student table, *title* from the Book table, *dateOut* and *dateBack* from the Loan table. We will use these in our *SELECT*.

```
SELECT CONCAT(fname,' ',lname)  Name, title, dateOut, dateBack
FROM bookcopy JOIN book
ON  bookcopy.isbn = book.isbn
JOIN loan
ON bookcopy.copyId = loan.copyId
JOIN student
ON student.studentId = loan.studentId;
```

Exercises (Multi Table Join)

1. List the books by title that are on loan at present (dateBack is null). Label the title *Books currently on loan*. Sort in alphabetical order by title.

Books currently on loan
jQuery for Novices
Macro Economics
Maths for Business

2. List the books by title that are on loan at present, the student by name who borrowed the book, the date the book was borrowed and the date the book was due back. Label the title *Books currently on loan* and label the student name *Student Name*. Sort in alphabetical order by title.

Books currently on loan	Student Name	dateOut	dateDue
jQuery for Novices	Philip Walsh	2019-08-01	2019-08-21
Macro Economics	Martin Roche	2019-08-03	2019-08-29
Maths for Business	Steven Ryan	2019-08-08	2019-08-27

3. Return the number of loans per book title. Output the count with the label Number of Loans.

title	Number of Loans
Database Design	1
Finanacial Accounting	3
JavaScript	1
jQuery for Novices	3
Learning JavaScript	1
Macro Economics	2
Maths for Business	2

Movies Exercise 1

For these exercises, load the *movies* database and remember to select the database.

To do:

1. For each rating retrieve the reviewer name, film title, number of stars, and date of rating.

Reviewer Name	Film title	Number of Stars	Date of Rating
Sarah Martinez	Gone with the Wind	2	2018-01-22
Sarah Martinez	Star Wars	4	2018-01-27
Daniel Lewis	Snow White	4	NULL
Brittany Harris	The Sound of Music	2	2018-01-20
Brittany Harris	E.T.	4	2018-01-12
Brittany Harris	Raiders of the Lost Ark	2	2018-01-30
Mike Anderson	Gone with the Wind	3	2018-01-09
Chris Jackson	The Sound of Music	3	2018-01-27
Chris Jackson	E.T.	2	2018-01-22
Chris Jackson	Raiders of the Lost Ark	4	NULL
Elizabeth Tho...	Avatar	3	2018-01-15
Elizabeth Tho...	Snow White	5	2018-01-19
James Cameron	Avatar	5	2018-01-20
Ashley White	E.T.	3	2018-01-02

2. In a previous query you returned the number of films reviewd by Chris Jackson. Now, return the film titles of the films reviewed.

Films Reviewed by Chris Jackson
The Sound of Music
E.T.
Raiders of the Lost Ark

Outer Join

Please enter the following statement which returns all Student records:

```
SELECT * FROM student;
```

Now, enter the following statement which returns all Loan records:

```
SELECT * FROM loan;
```

If you look at the loan data, not all of the students have taken a book loan (as yet). Now, say we want to produce a query which lists all students and their loan details (if they have any). The query would be as follows:

```
SELECT concat(fname,' ', lname) as 'Name', copyId
FROM student JOIN loan
ON student.studentid = loan.studentid;
```

This will return the same number of records as the previous statement (SELECT * FROM loan;)

As you can see a JOIN SELECT will only show the student record if it has a loan. But, how can we return all student records and their loans even if they have NO loan record associated? We must use a **LEFT** or **RIGHT** ***OUTER JOIN***.

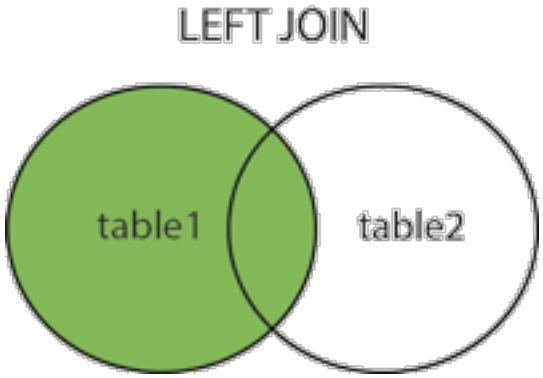
An outer join extends the result of a simple join. An outer join returns all rows that satisfy the join condition and also returns some or all of those rows from one table for which no rows from the other satisfy the join condition.

Left Outer Join

The **LEFT JOIN** keyword returns all rows from the left (first) table (table1), with the matching rows in the right table (table2). The result is NULL in the right side when there is no match.

Syntax:

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name=table2.column_name;
```



Here is the previous example written with the *LEFT OUTER JOIN* syntax:

```
SELECT *
FROM student LEFT JOIN loan
ON student.studentid = loan.studentid;
```

Execute the statement. Do you see all the null values?
A more realistic SELECT would be:

```
SELECT concat(fname,' ', lname) as 'Name', copyId
FROM student LEFT JOIN loan
```

```
ON student.studentid = loan.studentid;
```

Notes:

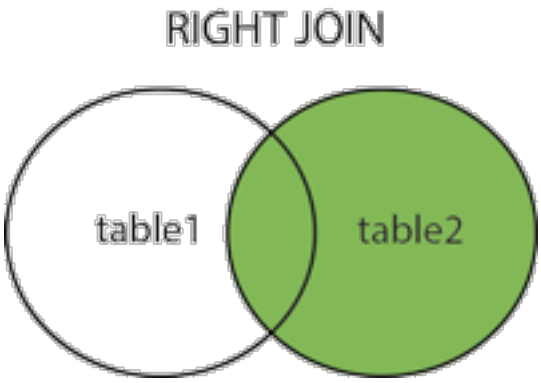
- The LEFT JOIN keyword returns all the rows from the left table (Student), even if there are no matches in the right table (Loan).
- A student name will appear more than once if he/she has borrowed more than one book.

Right Outer Join

The **RIGHT JOIN** keyword returns all rows from the right (secons) table (table2), with the matching rows in the left table (table1). The result is NULL in the left side when there is no match.

Syntax:

```
SELECT column_name(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name=table2.column_name;
```



Here is the previos example written with the *RIGHT OUTER JOIN* syntax:

```
SELECT concat(fname,' ', lname) as 'Name', copyId
FROM loan RIGHT JOIN student
ON student.studentid = loan.studentid;
```

Note:

- The RIGHT JOIN keyword returns all the rows from the right table (Student), even if there are no matches in the left table (Loan).

Example Three

The following example produces for each bookcopy - the copyId, its' book title, the loan details (i.e. the date it was borrowed (if it was borrowed), and the date it was returned). This will require joining tables - bookcopy, loan and book.

1. Identify which tables have the data you are looking for, copyId is in the *BookCopy* and *Loan* tables; title is in the *Book* table; and loan details are in the *Loan* table. The tables required are: *Book*, *Loan* and *BookCopy*. Since there are 3 tables, we need 2 joins. We need an Outer Join as we want to return all BookCopy records and their loan details whether they have a loan or not.

2. Identify which columns are the primary and foreign keys in these tables. To join the Book and BookCopy tables, we use the *ISBN* in the *Book* and *ISBN* in the *BookCopy* table. To join the BookCopy and Loan tables we use the *copyId* in the *BookCopy* table and *copyId* in the *Loan* table. We use these in the *ON* clause. We will join the *Book* and *BookCopy* tables together with an Equijoin, and then finally left join the *Loan* table to include all BookCopy details even if there is no match with the Loan table.

3. Identify which columns we want the query to output; in this case it is *copyId* from the *BookCopy* table, *title* from the *Book* table, *dateOut* and *dateBack* from the *Loan* table. We will use these in our *SELECT*.

```
SELECT bookcopy.copyId, title, dateOut, dateBack
FROM book JOIN bookcopy
ON book.isbn = bookcopy.isbn
LEFT JOIN loan
ON bookcopy.copyId = loan.copyId;
```

A bookcopy will appear more than once if it is loaned out more than once.
You can also write the query as follows using the RIGHT join:

```
SELECT bookcopy.copyId, title, dateOut, dateBack
FROM loan RIGHT JOIN bookcopy
ON bookcopy.copyId = loan.copyId
JOIN book
ON book.isbn = bookcopy.isbn;
```

Example Four

The following example produces for each Student **-** the full student name, the title of the books borrowed, the date it was borrowed, and the date it was returned even if the student has (or had) no loans.

1. Identify which tables have the data you are looking for, student name is in the *Student* table, Book Title is in the *Book* table, loan details are in the *Loan* table. Since table Book is not related to any of these tables, we also require BookCopy as it has ISBN as a foreign key and BookCopy can be joined with *Loan*. The tables required are: *Student*, *Book*, *Loan* and *BookCopy*. Since there are 4 tables, we need 3 joins. We need an Outer Join as we want to return all Student records and their loan details whether they have a loan or not.

2. Identify which columns are the primary and foreign keys in these tables. To join the Student and Loan tables, we use the *studentId* in the *Student* and *studentId* in the *Loan* table. To join the BookCopy and Loan tables we use the *copyId* in the *BookCopy* table and *copyId* in the *Loan* table. To join the BookCopy and Book tables we use the *ISBN* in the *BookCopy* table and *ISBN* in the *Book* table. We use these in the *ON* clause.

3. Identify which columns we want the query to output; in this case it is *fName*, *lName* from the *Student* table, *title* from the *Book* table, *dateOut* and *dateBack* from the *Loan* table. We will use these in our *SELECT*.

As we saw previously we need the Book table to obtain the title column, but because it is not linked with the Loan table, we need BookCopy also. We can join these three tables together with an Equijoin, and then finally right join the Student table to include all students even if there is no match with the Loan table.

```
SELECT concat(fname,' ', lname) as 'Name', title, dateOut, dateBack
FROM book JOIN bookcopy
ON book.isbn=bookcopy.isbn
JOIN loan
ON bookcopy.copyid = loan.copyid
RIGHT JOIN student
ON student.studentid = loan.studentid;
```

Another way of expressing the query (using LEFT join) is as follows:

```
SELECT concat(fname,' ', lname) as 'Name', title, dateOut, dateBack
FROM student LEFT JOIN loan
ON student.studentid = loan.studentid
LEFT JOIN bookcopy
ON bookcopy.copyid = loan.copyid
LEFT JOIN book
ON book.isbn=bookcopy.isbn;
```

In this example, because we selected from the student table first, we now need to left join it with all the other tables to ensure that all students are included as after the first join **-** loanId and copyId will have null values for students who have no loans AND in order to join these values with other tables we need an Outer Join.

Movies Exercise 2

For these exercises, load the *movies* database, remember to select the database, and execute the following statement:

```
insert into reviewer values(209, 'Liam Collins');
```

1. Return the film title, reviewer name and number of stars for all films (even if it is not reviewed yet).

Film Title	Reviewer Name	Number of Stars
Gone with the Wind	Sarah Martinez	2
Gone with the Wind	Mike Anderson	3
Star Wars	Sarah Martinez	4
The Sound of Music	Brittany Harris	2
The Sound of Music	Chris Jackson	3
E.T.	Brittany Harris	4
E.T.	Chris Jackson	2
E.T.	Ashley White	3
Titanic	NULL	NULL
Snow White	Daniel Lewis	4
Snow White	Elizabeth Tho...	5
Avatar	Elizabeth Tho...	3
Avatar	James Cameron	5
Raiders of the Lost...	Brittany Harris	2
Raiders of the Lost...	Chris Jackson	4

2. Return the reviewer name, film title, and number of stars for all reviewers (even if they have not left a review yet).

Reviewer Name	Film Title	Number of Stars
Sarah Martinez	Gone with the Wind	2
Sarah Martinez	Star Wars	4
Daniel Lewis	Snow White	4
Brittany Harris	The Sound of Music	2
Brittany Harris	E.T.	4
Brittany Harris	Raiders of the Lost Ark	2
Mike Anderson	Gone with the Wind	3
Chris Jackson	The Sound of Music	3
Chris Jackson	E.T.	2
Chris Jackson	Raiders of the Lost Ark	4
Elizabeth Tho...	Avatar	3
Elizabeth Tho...	Snow White	5
James Cameron	Avatar	5
Ashley White	E.T.	3
Liam Collins	NULL	NULL

Geography Exercise

For these exercises, load the *geography* database, and remember to select the database:

To Do:

1. Return the Continent, Country and City for all countries (even if it has not a city).

Continent	Country	City
Oceania	American Samoa	Fagatogo
Oceania	American Samoa	Tafuna
Oceania	Australia	Adelaide
Oceania	Australia	Brisbane
Oceania	Australia	Cairns
Oceania	Australia	Canberra
Oceania	Australia	Central Coast
Oceania	Australia	Geelong
Oceania	Australia	Gold Coast
Oceania	Australia	Hobart
Oceania	Australia	Melbourne
Oceania	Australia	Newcastle
Oceania	Australia	Perth
Oceania	Australia	Sydney
Oceania	Australia	Townsville
Oceania	Australia	Wollongong
Oceania	Christmas Island	Flying Fish Cove

...

Oceania	Northern Mariana Islands	Garapan
Oceania	Palau	Koror
Oceania	Papua New Guinea	Port Moresby
Oceania	Pitcairn	Adamstown
Oceania	Samoa	Apia
Oceania	Solomon Islands	Honiara
Oceania	Tokelau	Fakaofo
Oceania	Tonga	Nuku´alofa
Oceania	Tuvalu	Funafuti
Oceania	United States Minor Outlying Islands	NULL
Oceania	Vanuatu	Port-Vila
Oceania	Wallis and Futuna	Mata-Utu
Antarctica	Antarctica	NULL
Antarctica	Bouvet Island	NULL
Antarctica	French Southern territories	NULL
Antarctica	Heard Island and McDonald Islands	NULL
Antarctica	South Georgia and the South Sandwich Islands	NULL