SQL Week 6

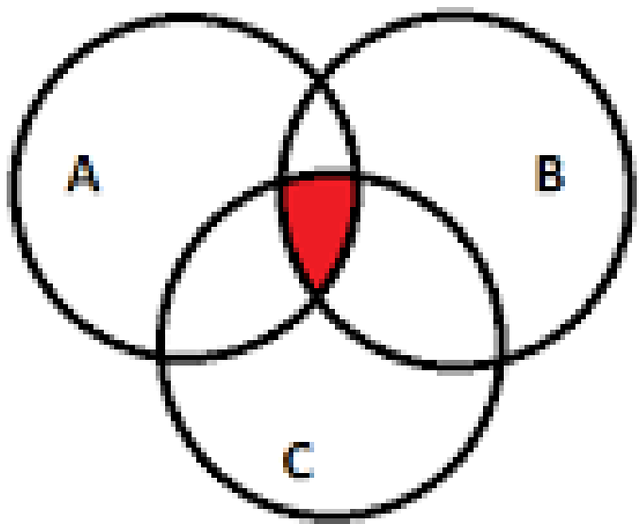
Topics List

- Multi-table JOIN
- OUTER JOIN

SQL Week 6

Multi-table Join

- A Multi-table SQL join is an instruction to combine data from three or more sets of data (i.e. three or more tables)



SQL Week 6

Multi-table Join

Book(ISBN, title, publisher, publishedDate, category, price)
Primary key ISBN

BookCopy(copyId, ISBN, dateAcquired, dateDestroyed)
Primary key copyId
Foreign key ISBN references Book(ISBN)

Student(studentId, fName, lName, street, town, county, course, year)
Primary key studentId

Loan(loanId, copyId, studentId, dateOut, dateDue, dateBack)
Primary key loanId
Foreign key copyId references BookCopy(copyId)
Foreign key studentId references Student(studentId)

SQL Week 6

Multi-table Join

Author(authorId, fName, lName)
Primary key authorId

Authorship(authorId, ISBN)
Primary key authorId, ISBN
Foreign key authorId references Author(authorId)
Foreign key ISBN references Book(ISBN)

SQL Week 6

Multi-table Join

- For each loan we want to return the student's name who borrowed the book, the Title of the book borrowed, and the dates loaned and returned.

- The tables required are Student, Book, BookCopy and Loan.

SQL Week 6

Multi-table Join

- Again, in each SELECT statement that requires a JOIN, there are four things to do:
  - Identify which tables have the data you are looking for. These table names are used in the *JOIN* clause.
  - Identify which columns are the primary and foreign keys in these tables. We use these in the *ON* clause.
- Identify which columns we want the query to output. We put these in the *SELECT* clause of the statement.
- Add any conditions necessary (if there are any).

SQL Week 6

Multi-table Join

```
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;
```

Topics List

- Multi-table JOIN
- OUTER JOIN

SQL Week 6

Outer Join

- **Outer joins**. When performing an inner **join**, rows from either table that are unmatched in the other table are not returned.

- In an **outer join**, unmatched rows in one or both tables can be returned.

SQL Week 6

Outer Join

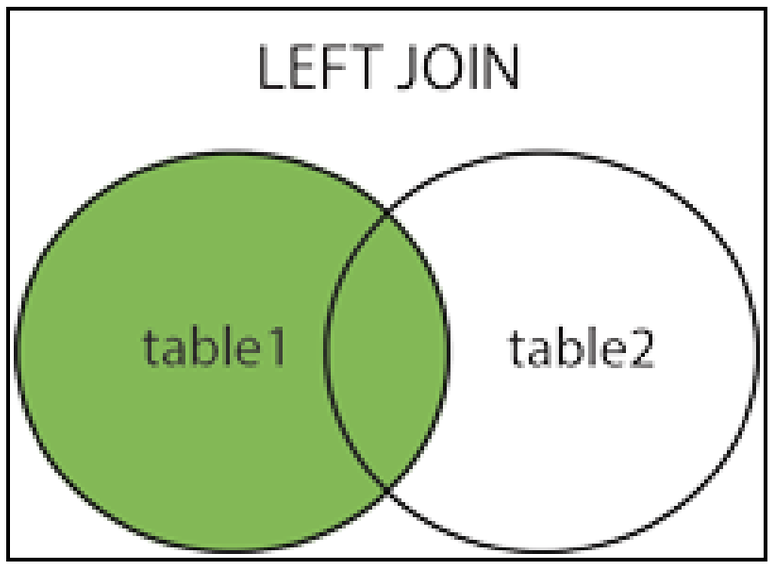
- To perform an outer join in MySQL, we use either the LEFT JOIN or the RIGHT JOIN.

- LEFT JOIN returns only unmatched rows from the left table.

- RIGHT JOIN returns only unmatched rows from the right table

SQL Week 6
Left 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.



SQL Week 6

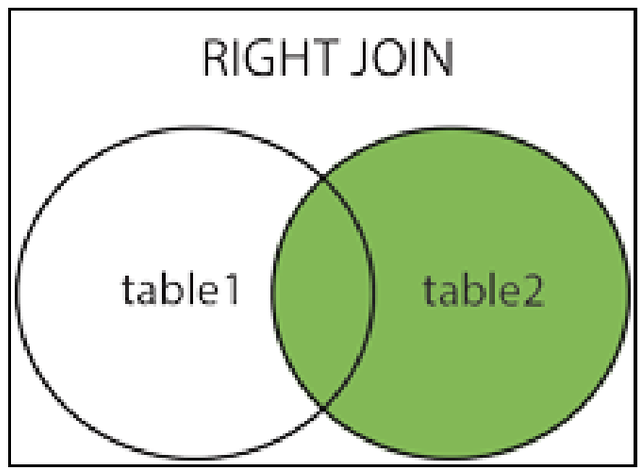
Left Join

- In the following example, we select the student name and the copyId of the book on loan for each student. If the student has no loans, their name is returned and the copyId value will be NULL.

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

SQL Week 6
Right Join

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



SQL Week 6

Right Join

- Here is the previous example written with the *RIGHT OUTER JOIN* syntax. This time all the records from the second (right) table are returned.

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