Tutorial 3 Summary

Following I have summed up the activities and commands discussed in the third tutorial session.

MySQL Joins

Joins in relational database, and in our case MySQL, can be used to fetch results for almost any query, once you know understand the uses and applications of the various types of joins. They are usually used to put together data from different tables into a single table with respect to certain matching columns. They are primarily classified as

■ CROSS JOIN

One of the simplest and easiest type of join. Given two tables A and B, a **cross join** of them returns a table that contains each row from table A combined with each row of table B. The query is also very simply generated as

```
SELECT * FROM course CROSS JOIN prereq;
```

OR

```
SELECT * FROM course, prereq;
```

(Note the use of a comma to generate the cross join.)

■ SELF JOIN

When a table is cross joined by itself its referred to as a **self join**. The syntax, again, is the same.

```
SELECT * FROM course AS c1, course AS c2;
```

(Note the use of two different aliases for the same table because while performing the join they are treated as different tables and need to be aliased as such.)

■ INNER JOIN

Let's say you want to match all the rows from the table 'course' to all those rows in table 'prereq' where there is a certain match of the 'course_id'. Basically you want to draw a table that shows the prerequisite courses (if any) for any given course. This means that we need to find an intersection of two tables with similarity based on the 'course_id' column. So an INNER JOIN returns the records that have matching values in particular columns of both the tables.

```
SELECT * FROM course INNER JOIN prereq ON course.course_id = prereq.course_id;
```

OR

```
SELECT * FROM course INNER JOIN prereq USING(course_id);
```

*(Note the use of the keyword **USING** to shorten the query when the two tables have the same column names for the columns that are tallied.)*

■ OUTER JOIN

What if you have decided the course you want to take this and are listed in the course table and you just want to check that if the course you are planning to credit have any prerequisite courses or not, and if they do then which are those. This calls for a kind of join that keeps all the entities from a particular table and only matches the matching rows from the other table. This is called an Outer Join. There are 3 types:

- Left Outer Join

  If you left outer join table A with table B, then all rows from table A are kept and only the matching rows from table B are returned. For the course, prerequisite case, we have

  SELECT * FROM course LEFT OUTER JOIN prereq ON course.course_id = prereq.course_id;

  OR

  SELECT * FROM course LEFT JOIN prereq USING(course_id);

  *(Note the omission of the keyword **OUTER** in the second query. MySQL by default understands that a LEFT JOIN is a LEFT OUTER JOIN.)*

- Right Outer Join

  Just like the left join, if you right outer join table A with table B, it keeps all the rows from the right table, or in our case table B and only returns the matching rows from table A.

  SELECT * FROM course RIGHT OUTER JOIN prereq ON course.course_id = prereq.course_id;

  OR

  SELECT * FROM course RIGHT JOIN prereq USING(course_id);

  *(Note that **A** LEFT JOIN **B** is exactly the same as **B** RIGHT JOIN **A**.)*

- Full Outer Join

  Although there is no direct keyword for a full outer join in MySQL, it is hardly needed in retrieving any query. The places where it is needed, a full join can be generated by combining the outputs of the left and the right outer joins. Basically a full join return all records from both the tables whether there is corresponding match for it in the other table or not. A full join query uses the UNION keyword.

  SELECT c.course_id, p.prereq_id, c.title, c.dept_name, c.credits FROM course c LEFT JOIN prereq p USING(course_id)

  UNION

  SELECT p.course_id, p.prereq_id, c.title, c.dept_name, c.credits FROM course c RIGHT JOIN prereq p USING(course_id);

  (Note a UNION returns all non repeating rows whereas UNION ALL returns all rows irrespective of duplicates.)

NOTE: If you intend to use the 'course', 'prereq', 'customer', 'salesman' an discussed in class, download the text file containing the command to build the tables in MySQL from this link.