CMI/Feb-Mar 2021/RDBMS, SQL & Visualization Feb 2021

Tutorial 2 Summary

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

MySQL Aggregate Functions

Aggregate functions are very commonly used and extremely helpful functions that perform some very basic querying without the use of complex syntax, like

- COUNT():

  The **COUNT()** function returns the number of rows for a particular column of a table or a selection. Let's say you want to get the number of rows you have in the employee table. You may query over all columns (i.e. using *****) or a particular column, let's say 'EmpCode'

  SELECT COUNT(*) FROM Employee;
  or
  SELECT COUNT(EmpCode) FROM Employee;

  Counting over a selection is usually done when we are querying with certain criteria, like, the number of employees that have received a commission or the number of **DISTINCT** jobs that have received commission.

  SELECT COUNT(*) FROM Employee where commission > 0;
  or
  SELECT COUNT(DISTINCT Job) FROM Employee where commission > 0;

- SUM():

  Just as the name suggests, **SUM()**, iterates over a particular numerical column of a table or a selection and **sums** all the numbers. Note that it ignores null values. Let's say you want to figure out how much commission the company needs to pay its employees in total,

  SELECT SUM(commission) FROM Employee;

  or how much commission do you need to pay, let's say, to the 'SALESMEN',

  SELECT SUM(commission) FROM Employee WHERE job = 'salesman';

  or maybe even 'job' wise sum of salary,

  SELECT job, SUM(salary) FROM Employee GROUP BY job;

  *(The **GROUP BY**) keyword helps you collate items with similar values in a particular column, It is very commonly used with all the aggregate functions to generate detailed grouped outputs*

- AVG():

  Very similar to **SUM()** and **COUNT()**, the **AVG()** function returns the simple arithmetic mean of the desired column or selection. The **'average'** function very basically does $\frac{\textbf{sum}}{\textbf{count}}$. It's uses are similar to that of SUM. For example if you want to query the average salary to be payed job title wise, then

  ```
  SELECT job, AVG(salary) FROM Employee GROUP BY job;
  ```

  Or let's say average payout (= salary+commission) per employee.

  ```
  SELECT AVG(salary + commission) FROM Employee;
  ```

  (Note the adding of respective values of two columns by column names.)

- MIN(), MAX():

  Both the functions are exactly similar in use except the fact they return the opposite extreme ties of a numerical column or selection. You may gather the maximum salary that an employee draws as such

  ```
  SELECT MAX(Salary) FROM Employee;
  ```

  Or maybe all the details of the employee that draws the minimum salary,

  ```
  SELECT * FROM Employee WHERE Salary = (SELECT MIN(Salary) FROM Employee);
  ```

  Or maybe all the details of the employee that draws the second highest salary,

  ```
  SELECT * FROM Employee WHERE Salary = (SELECT MAX(Salary) FROM Employee where
  Salary != (SELECT MAX(Salary) FROM Employee));
  ```

NOTE: If you intend to use the same table 'employee' as discussed in class, download the text file containing the command to build the table in MySQL from this link.