# 1. MySQL – MySQLD – MySQLAdmin

\_ MySQLD: is the server executable (one of them), đây là một server dùng để manage database

\_ MySQL: is the command line client.

\_ MySQLAdmin: is a maintainance or administrative utility.

# 2. Storage Engine

**Address:** http://ktmt.github.io/blog/2013/05/07/storageenginemysql/

Storage Engine thực chất là cách MySQL lưu trữ dữ liệu trên đĩa cứng.

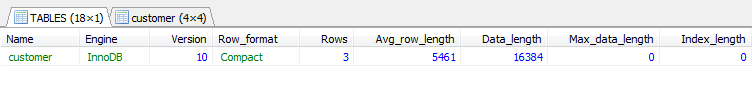
**InnoDB engine:** Là engine phức tạp nhất trong các engine của MySQL

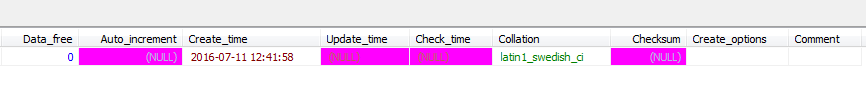
InnoDB engine có hỗ trợ transaction;

Use this query to show information table:

**show** **table** **status** **like** 'customer';

**result:**



****

Transaction define:

http://vietjack.com/sql/transaction\_trong\_sql.jsp

# 3. Select shortest and longest

Query the two cities in **STATION** with the shortest and longest CITY names, as well as their respective lengths (i.e.: number of characters in the name). If there is more than one smallest or largest city, choose the one that comes first when ordered alphabetically:

**select city, char\_length(city) city\_length from station order by city\_length, city limit 1;**

**select city, char\_length(city) city\_length from station order by city\_length desc, city limit 1;**

# 4. Select multiple conditions

Query the list of CITY names starting with vowels (i.e., a, e, i, o, or u) from **STATION**. Your result cannot contain duplicates.

**select distinct city from station where city like 'a%'**

**union**

**select distinct city from station where city like 'e%'**

**union**

**select distinct city from station where city like 'i%'**

**union**

**select distinct city from station where city like 'o%'**

**union**

**select distinct city from station where city like 'u%'**

# 5. Rlike

Query the list of CITY names from **STATION** which have vowels (i.e., a, e, i, o, and u) as both their first and last characters. Your result cannot contain duplicates.

**select distinct city from station where city rlike '^[aeiouAEIOU].\*[aeiouAEIOU]$';**

# 6. RIGHT Substring

**select name from students where marks>75 order by RIGHT(name, 3),id**

# 7. Select case when

**select case**

**when A+B<=C or A+C<=B or B+C<=A then "Not A Triangle"**

**when A=B and A=C then "Equilateral"**

**when A=B or A=C or B=C then "Isosceles"**

**else "Scalene"**

**end as triangle\_type**

**from TRIANGLES;**

# 8. Group by, concat

Generate the following two result sets:

1. Query an *alphabetically ordered* list of all names in **OCCUPATIONS**, immediately followed by the first letter of each profession as a parenthetical (i.e.: enclosed in parentheses). For example: AnActorName(A), ADoctorName(D), AProfessorName(P), and ASingerName(S).
2. Query the number of ocurrences of each occupation in **OCCUPATIONS**. Sort the occurrences in *ascending order*, and output them in the following format:
3. There are a total of [occupation\_count] [occupation]s.

where [occupation\_count] is the number of occurrences of an occupation in **OCCUPATIONS** and [occupation] is the *lowercase* occupation name. If more than one *Occupation* has the same [occupation\_count], they should be ordered alphabetically.

**Note:** There will be at least two entries in the table for each type of occupation.

**select concat(name,"(",substr(occupation,1,1),")") from occupations**

**order by name asc;**

**select concat('There are a total of ',count(occupation),' ',lower(occupation),'s.') as n from occupations**

**group by occupation**

**order by n asc;**

# 9. Đếm giá trị

**set @r1=0, @r2=0, @r3=0, @r4=0;**

**select min(Doctor), min(Professor), min(Singer), min(Actor)**

**from(**

**select case when Occupation='Doctor' then (@r1:=@r1+1)**

**when Occupation='Professor' then (@r2:=@r2+1)**

**when Occupation='Singer' then (@r3:=@r3+1)**

**when Occupation='Actor' then (@r4:=@r4+1) end as RowNumber,**

**case when Occupation='Doctor' then Name end as Doctor,**

**case when Occupation='Professor' then Name end as Professor,**

**case when Occupation='Singer' then Name end as Singer,**

**case when Occupation='Actor' then Name end as Actor**

**from OCCUPATIONS**

**order by Name**

**) Temp**

**group by RowNumber;**