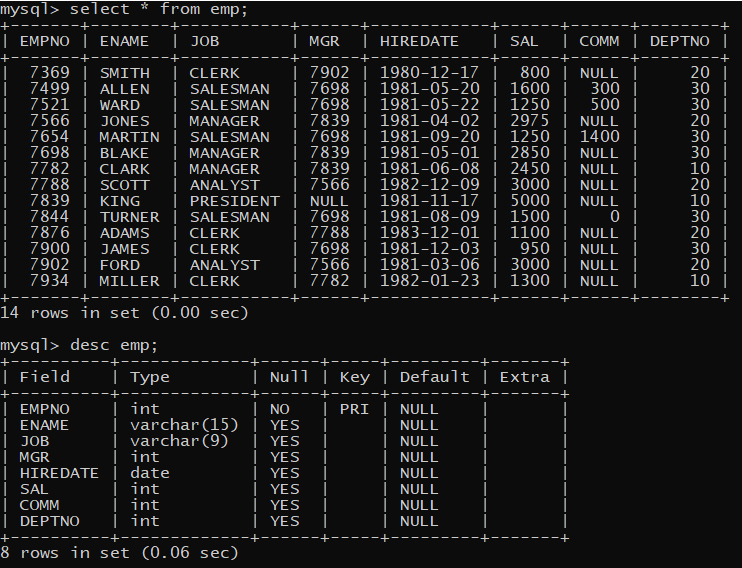


\* selects all the columns not the rows by default it takes rows because of absence of where clause



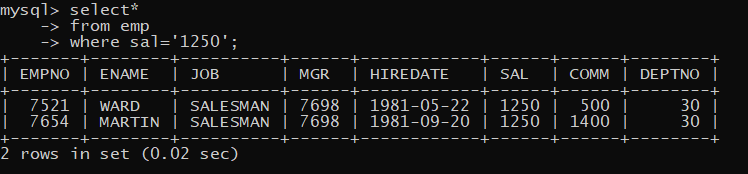
. Where only needs a boolean value if its false it will not show anything and if its true then it will give records

. There are some compile time errors as if we give a wrong table name or a column name then will get an error

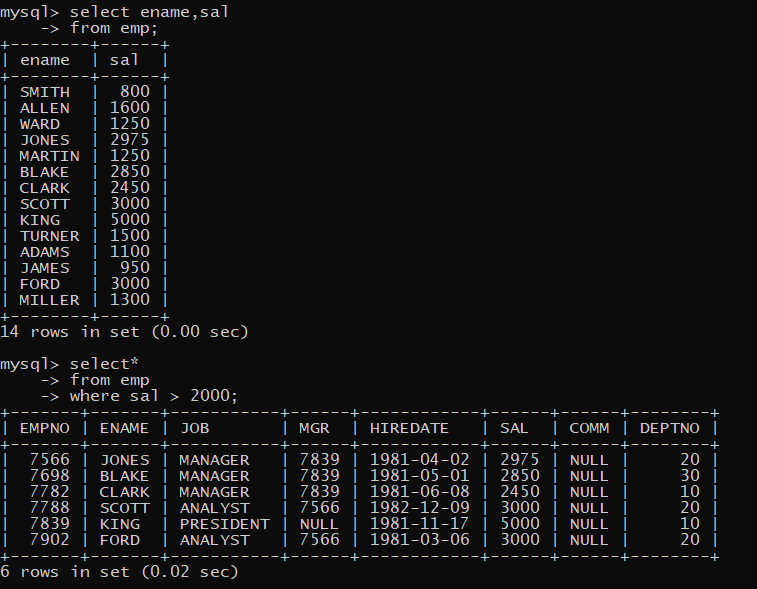
. **compile time errors**

1. Syntax error key words spelled wrong
2. Wrong column name
3. Wrong table name
4. No permission to access the table (not in control of developer work for DBA)
5. Connection error
6. Mismatched data types of values
7. Wrong type casting or no type casting

. **ANSI SQL supports implicit type casting**



. Here we used salary as a string in single quotes 1250 -> ‘1250’ still we are getting output because SQL type caste from string to int



( -- provides comments in SQL ) eg. [-- this is a comment ]

. We should always confirm that out put is correct or not

. There are 2 types

. Logically correct and logically wrong

. Example logically wrong

**Display records of emps who earn salary >= 3000**

*Select\**

*From emp*

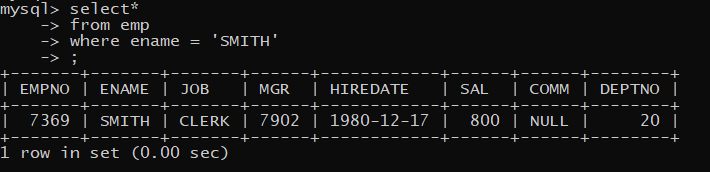
*Where sal > 3000;*

We should include >=  **point is always make sure logic is clear and correct**

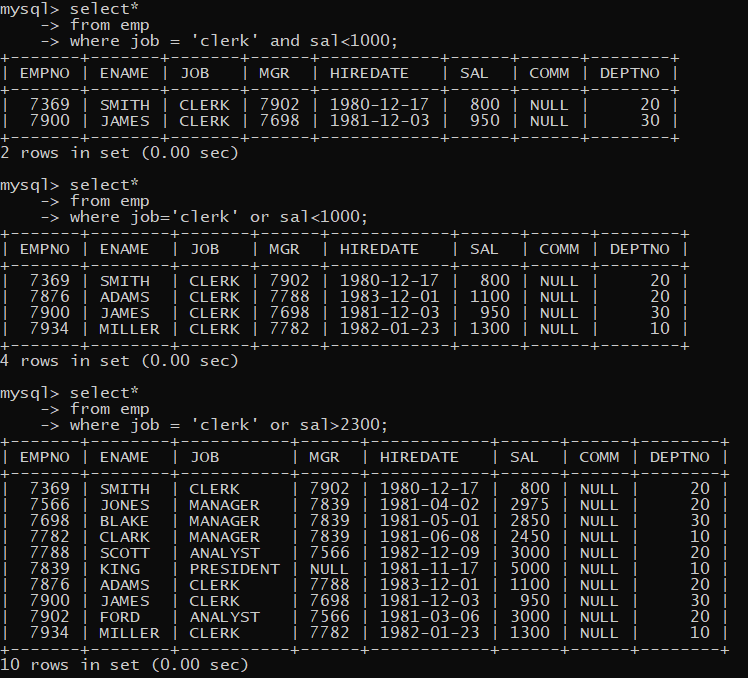
. **MYSQL** supports both ‘\_’ and “\_” for strings but **oracle** does not support “\_”

. Always use ‘\_’ single quotes

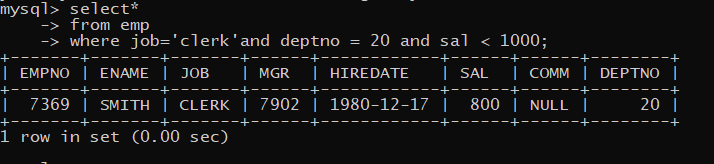
. **MYSQL** and **SQL Server** supports data values as case insensitive **oracle** is case sensitive



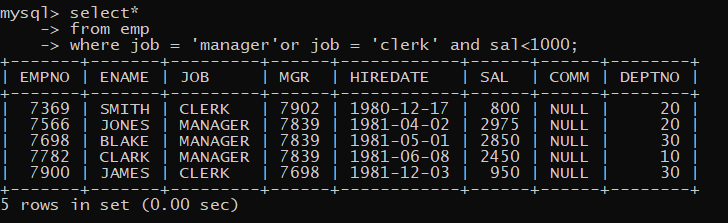
. Logical **and or** operators



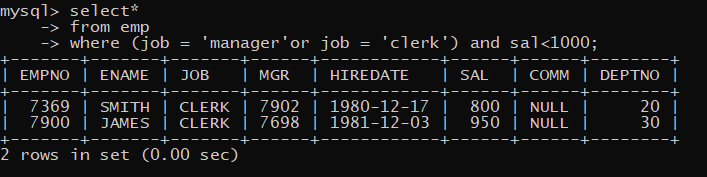
. Display records from job type clerk for dept no 20 who earn salary less than 1000



. Combination of **And Or** and always has **1st priority** afte**r and or** gets executed

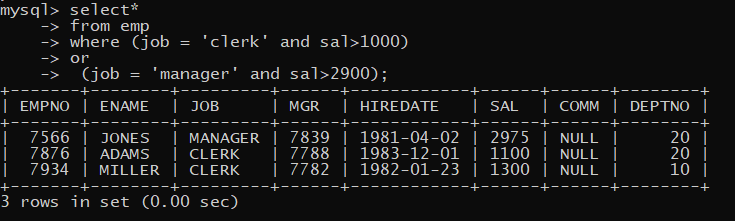


. **Parenthesis ()** will **override** the **default priority of AND OR**



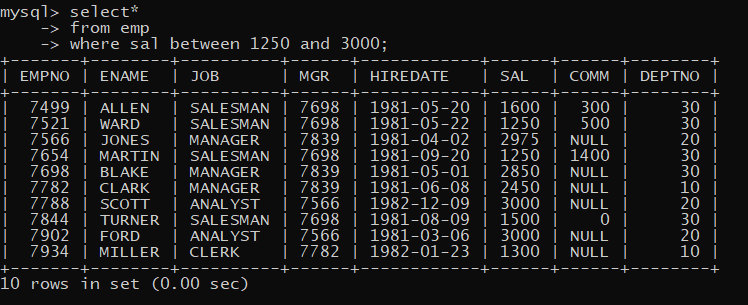
. **Multiple sets of conditions**

. Display records of clerk who earn salary above 1000 as well as display records of managers who earn salary above 2900

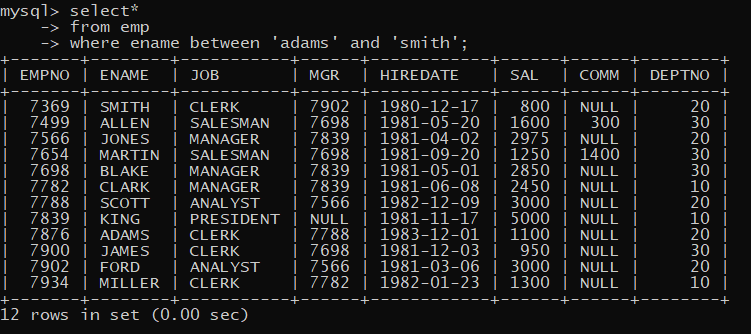


. **SQL Special Operators :**

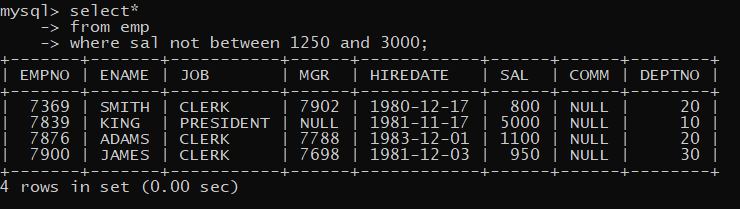
-1 **Between** it works with a range of values -> both **lower and higher** values are **inclusive**



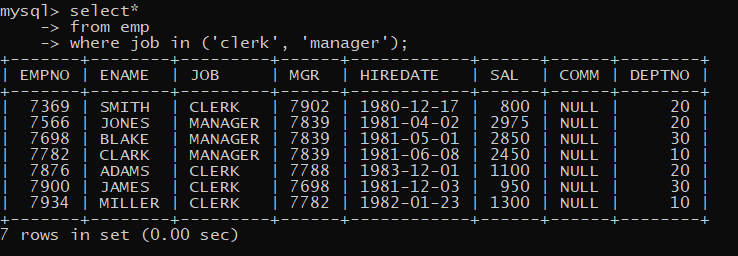
. Between is an overloaded operator it **also works with string** and arranges as per **alphabetical order**



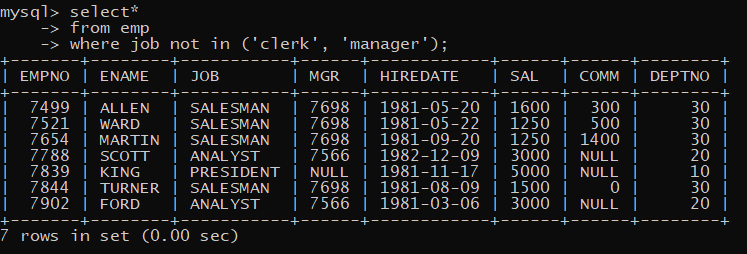
-2 **Not between** -> **higher and lower** values are not **included**

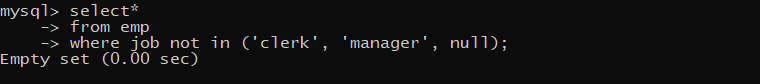


-3 **In** -> use full for equating multiple values of single column **[in acts as logical Or operator]**

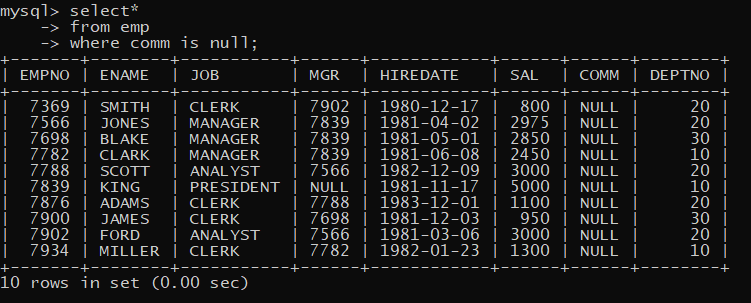


-4  **not in** -> acts as logical **and** operator | due to this if we had a **null value** comes it **will not work**

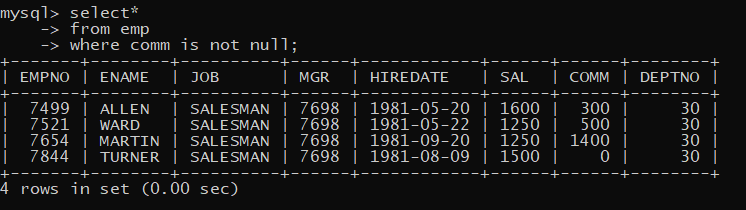




-5  **Is Null ->** to see the **null/missing/blank** value records



-6  **Is Not Null ->**



. SQL only considers **Null** value as null [**white space / ‘\_’ zero length string** are not considered as null]

-7 **Like** -> it is a pattern matching operator **|** it will **not** search for an E**xact value**

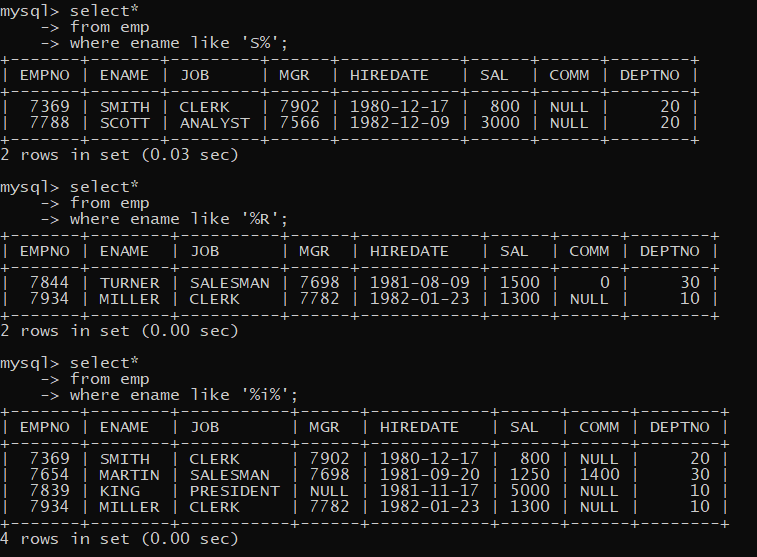
There are 2 wildcards associated with Like operator **%** and **\_**

% it will replace any character and any no of characters

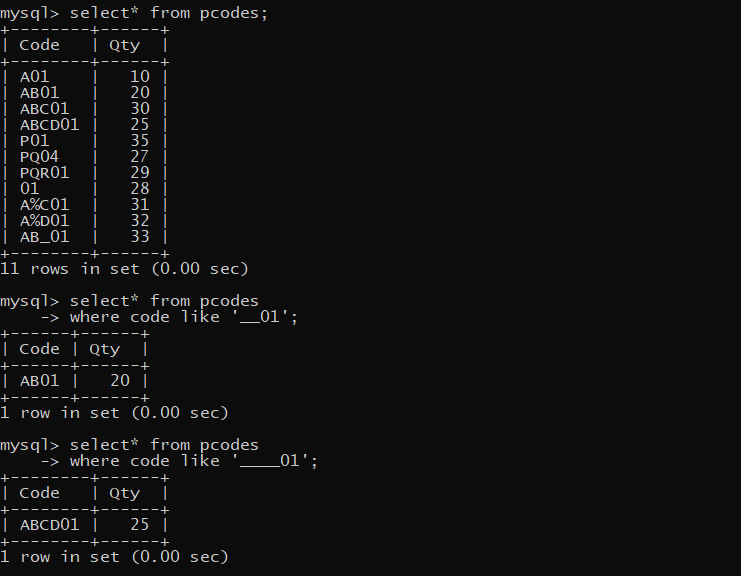
Display records where name starts with S /

Display records where name end with R /

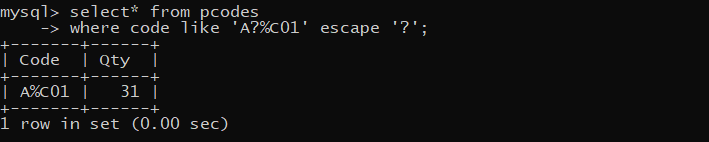
Display records where name has i in between



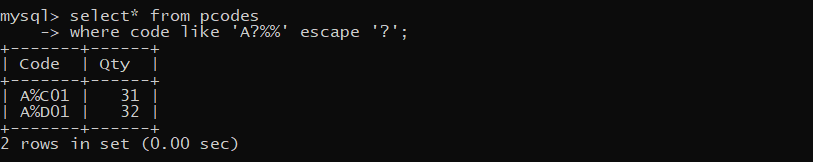
\_ single underscore replace **single character**



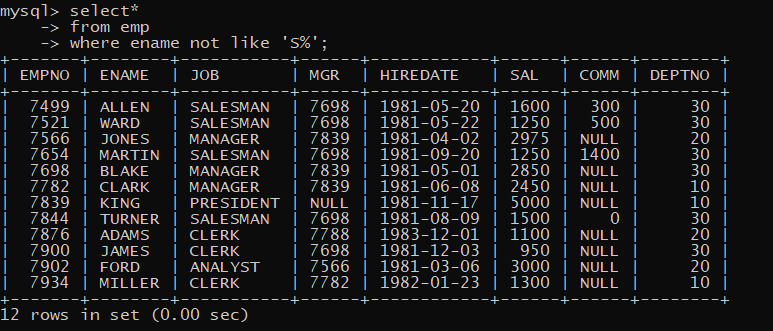
. **Escape character** -> to tell compiler that % is not a operator its **data value** form row



. **Escape char** only works for **immediate next** char not after that

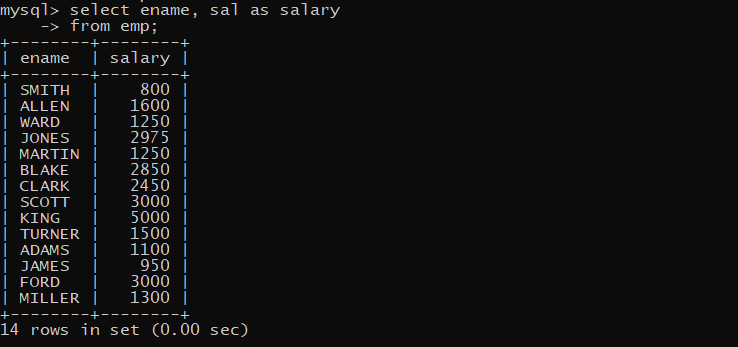


-8 **Not Like** ->

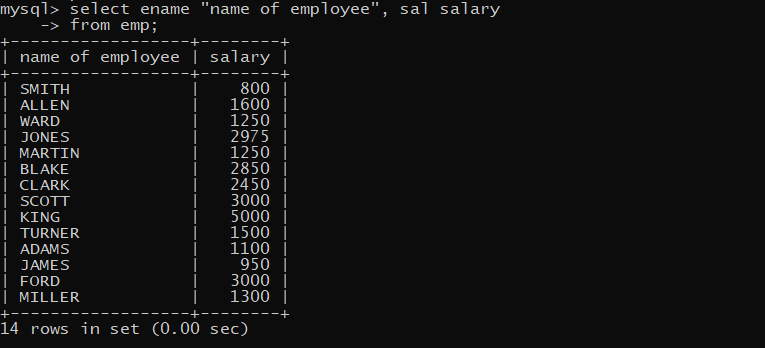


**. Column Aliases ->** only in the output of select statement the column will have a customized

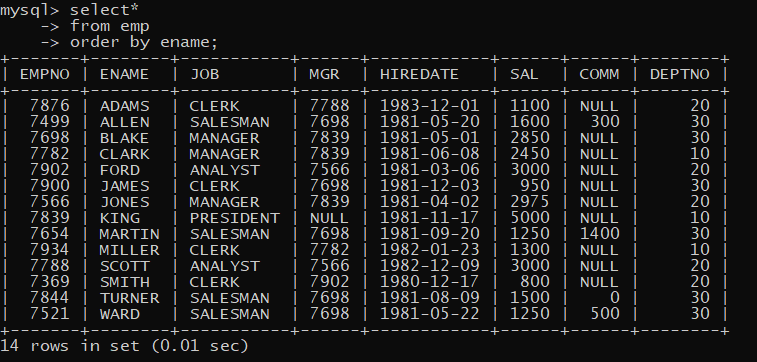
**(user friendly)** heading



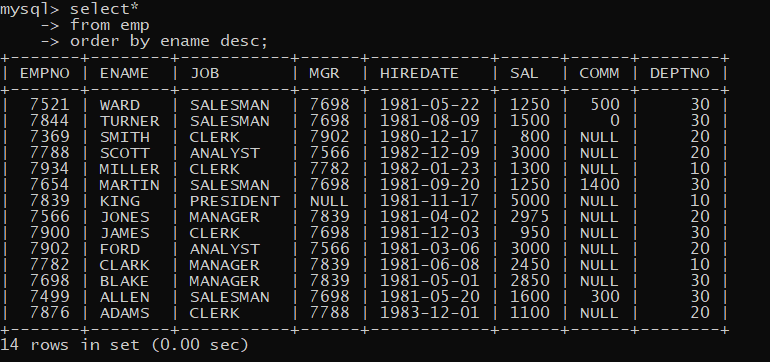
. Use **“-”** or **‘-’** for multi word string

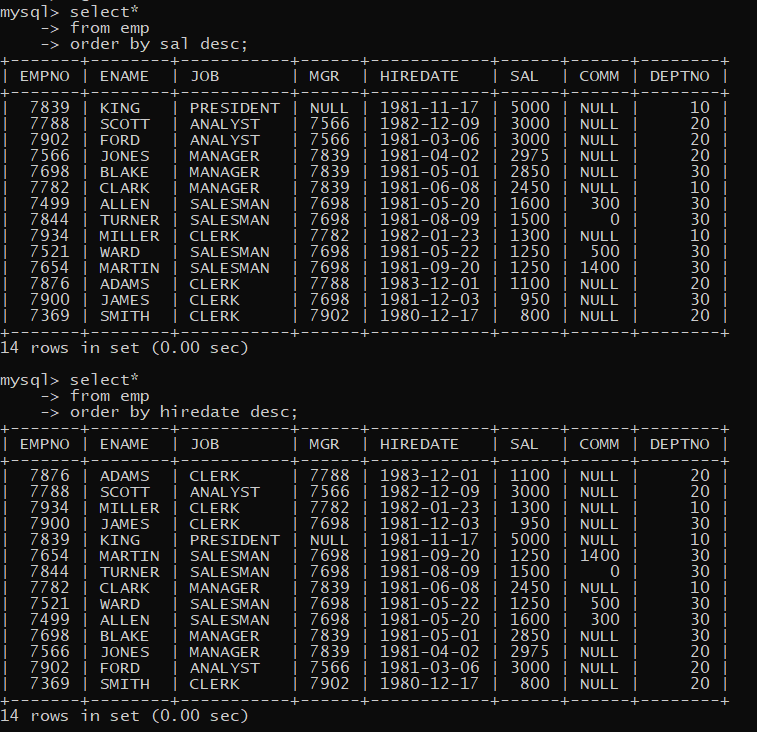


. **Order by clause** -> use full for sorting I.e. rearranging the sequence of rows ascending or descending as per some column(s)

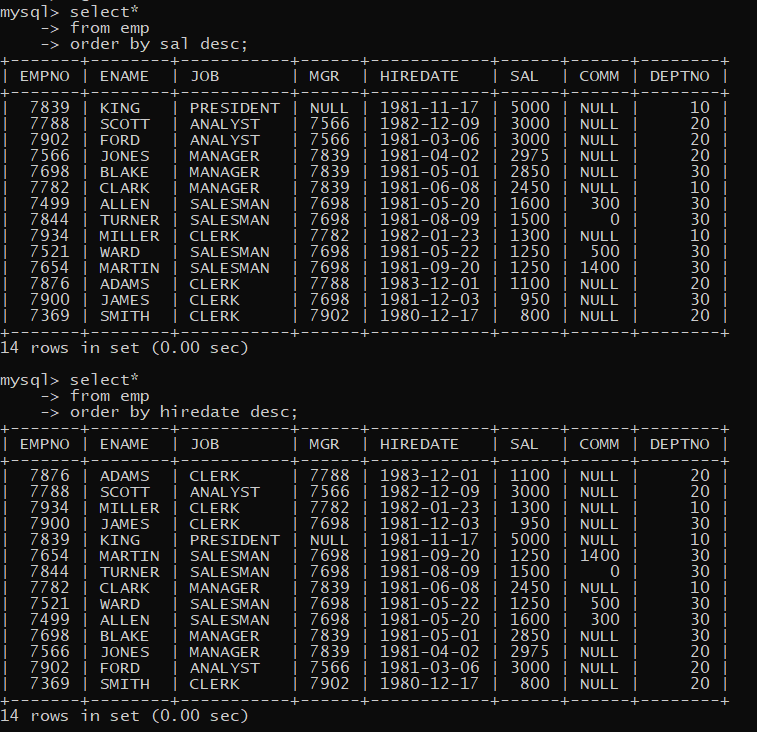


. **Desc** and **asc** for sorting descending and ascending | also works with numerical data and Dates

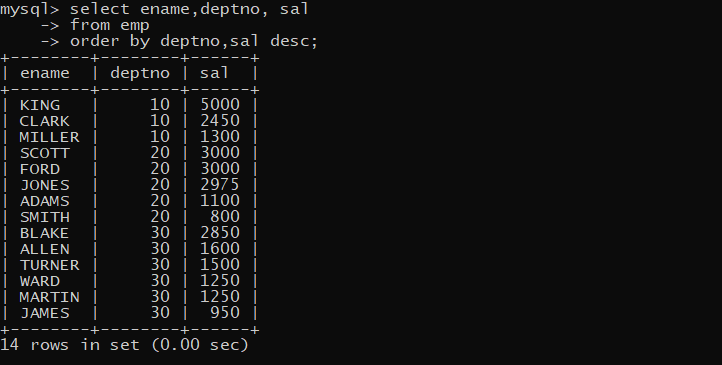




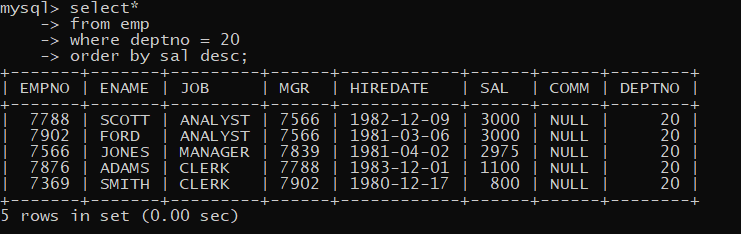
.



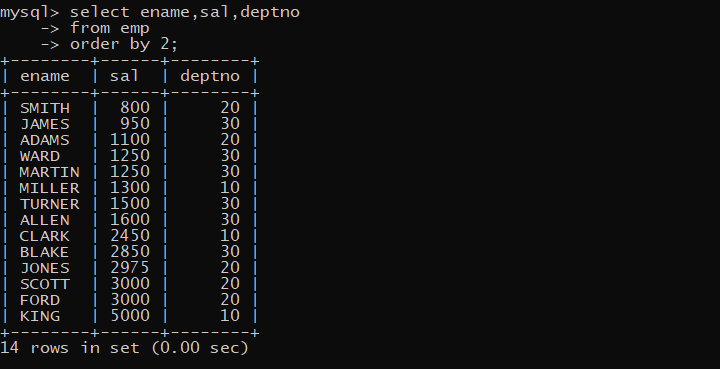
. **Multiple column sorting ->**



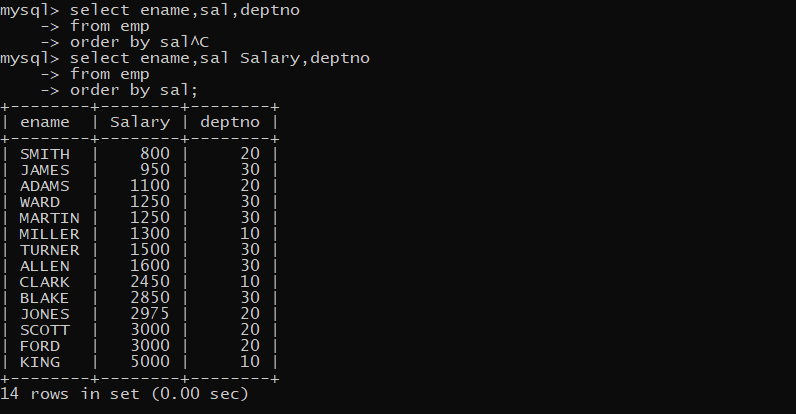
. We can use **order** and **where** with each other



. Column **position no** can be given in the order by

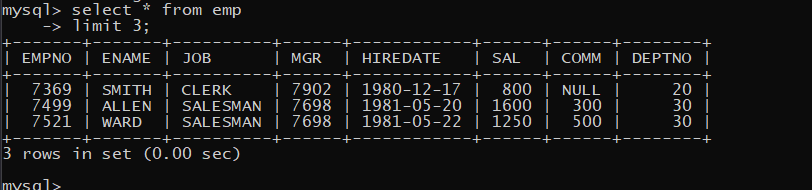


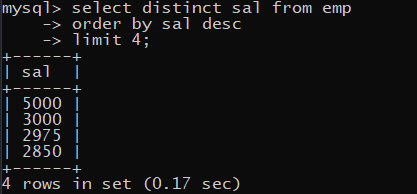
. Column **Alias name** can be given in the order by

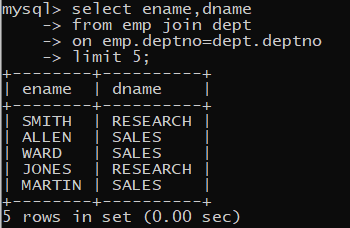


**Limit clause - ( it is my SQL proprietary )**

It will fetch top certain number of rows from table







**Limit with offset**

Offset is handy to skip number of rows from top.

Display 4th to 8th records

