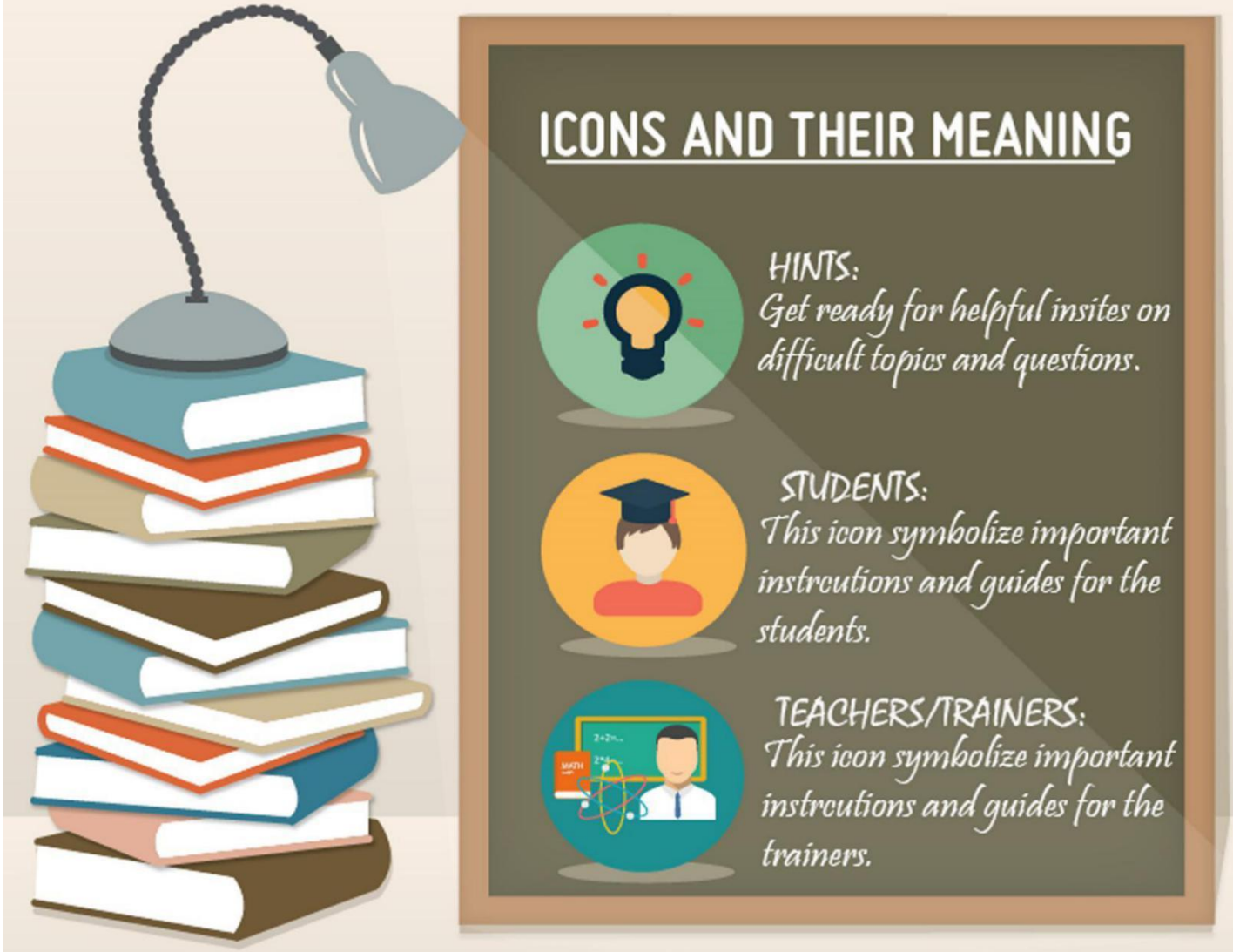
MySQL

MANUAL V8.3

MODULE CODE:

ANUDIP FOUNDATION


ICONS AND THEIR MEANING
HINTS:
Get ready for helpful insites on difficult topics and questions.
STUDENTS:
This icon symbolize important instrcutions and guides for the students.
TEACHERS/TRAINERS:
This icon symbolize important instrcutions and guides for the trainers.

Grouping and Sorting Data

Objective: After completing this lesson you will be able to : * Gain an understanding of how to use ORDER BY for sorting a result set * Use ORDER BY to perform natural sorting	**Materials Required:** 1. Computer 2. Internet access
Theory Duration: 80 minutes	**Practical Duration:** 0 minute
Total Duration: 80 minutes	

Chapter 21

ORDER BY: Sort a Result Set

Users selecting rows can expect the MySQL server to return them in any random order. It is possible to instruct MySQL to sort the result in any specific order. An ORDER BY clause can be added for sorting a set of results. This clause names the columns which are to be sorted.

Syntax

The code given below is the SQL SELECT command syntax and the ORDER BY clause for sorting a MySQL table's data.

SELECT field1, field2,...fieldN table_name1, table_name2...

ORDER BY field1, [field2...] [ASC [DESC]]

- The result can be sorted on any field, given that the field is listed out.
- Users can sort results on one or multiple fields.
- The ASC and DESC keywords help to sort results in ascending or descending order. Keywords are sorted in the ascending order by default.
- You can use the WHERE...LIKE clause in the usual way to put a condition.

ORDER BY clause utilized at the Command Prompt

Users can utilize the SQL SELECT command with the ORDER BY clause for retrieving data from the MySQL table named 'tutorials_tbl'.

Example

The example below is for returning result in an ascending order.

root@host# mysql -u root -p password;

Enter password:*******

mysql> use TUTORIALS;

Database changed

mysql> SELECT * from tutorials_tbl ORDER BY tutorial_author ASC

```
+-------------+----------------+-----------------+-----------------+
| tutorial_id | tutorial_title | tutorial_author | submission_date |
+-------------+----------------+-----------------+-----------------+
|       2     |  Learn MySQL   |    Arun Kumar   |    2009-06-22   |
|       1     |   Learn PHP    |   Rick James    |    2009-06-22   |
|       3     | JAVA Tutorial  |  Johnny Wilson  |    2009-06-07   |
+-------------+----------------+-----------------+-----------------+
```

3 rows in set (0.42 sec)

mysql>

Verifying all author names which are in the ascending order.

ORDER BY clause utilized in a PHP Script

Users can utilize similar ORDER BY clause syntax for a PHP function - mysql_query(). The function can execute an SQL command and a PHP function.

The selected data can be retrieved by using the mysql_fetch_array()

Example

Try out the following example, which returns the result in a descending order of the tutorial authors.

```
<?php
   $dbhost = 'localhost:3036';
   $dbuser = 'root';
   $dbpass = 'rootpassword';
   $conn = mysql_connect($dbhost, $dbuser, $dbpass);
   if(! $conn ) {
      die('Could not connect: ' . mysql_error());
   }
   $sql = 'SELECT tutorial_id, tutorial_title,
      tutorial_author, submission_date
      FROM tutorials_tbl
```

```
    ORDER BY   tutorial_author DESC';
mysql_select_db('TUTORIALS');

$retval = mysql_query( $sql, $conn );
if(! $retval ) {
    die('Could not get data: ' . mysql_error());
}
while($row = mysql_fetch_array($retval, MYSQL_ASSOC)) {
    echo "Tutorial ID :{$row['tutorial_id']}  <br> ".
        "Title: {$row['tutorial_title']} <br> ".
        "Author: {$row['tutorial_author']} <br> ".
        "Submission Date : {$row['submission_date']} <br> ".
        "--------------------------------<br>";
}
echo "Fetched data successfully\n";
mysql_close($conn);
?>
```

Natural Sorting with ORDER BY Clause

The first step is setting up a sample database table.

In this example, the CREATE TABLE statement is used for creating a new table 'items' -

```
CREATE TABLE items (
    id INT AUTO_INCREMENT PRIMARY KEY,
    item_no VARCHAR(255) NOT NULL
);
```

Some rows are then inserted into the items table -

```
INSERT INTO items(item_no)
```

```
VALUES ('1'),
       ('1C'),
       ('10Z'),
       ('2A'),
       ('2'),
       ('3C'),
       ('20D');
```

Then the data from the items table is sorted by the parameter item_no -

```
SELECT
    item_no
FROM
    items
ORDER BY item_no;
```

Most users do not expect to see the result shown above. The desired result is the one given in the image below -

Users desire the sorting results shown in the second image, which is known as natural sorting.

However, MySQL does not provide users a built-in natural sorting function or syntax. Users can rely on the ORDER BY clause to soft strings in a linear manner i.e. a character at a time, beginning from the first character.

MySQL natural sorting examples

To achieve natural sorting, the first step is splitting the item_no column into 2 columns i.e. prefix and suffix. The prefix column has the number section of the item_no, while the suffix column contains the alphabetical part. Data can be sorted based on columns. Example +

```
SELECT
    CONCAT(prefix, suffix)
FROM
    items
ORDER BY
```

prefix , suffix;

The query performs data sorting numerically at first. It then proceeds to perform alphabetical sorting of the data. Doing this gives back the expected naturally sorted result.

One downside of using the method mentioned above is that users have to break down the item_no into two parts prior to inserting it. Moreover, two columns have to be converted into one when the data is selected.

A user can use the query given below to carry out natural sorting without table structure modification. This can be done if the item_no data is in a standard format.

Example query -

```
SELECT
    item_no
FROM
    items
ORDER BY CAST(item_no AS UNSIGNED) , item_no;
```

In the above query, item_no data is first converted into an unsigned integer with type cast. Then the ORDER BY clause is utilized for numerically sorting rows first and then in an alphabetical order.

Let’s take a look at another common set of data that we often have to deal with.

Another example with a different data type -

```
TRUNCATE TABLE items;
INSERT INTO items(item_no)
VALUES('A-1'),
    ('A-2'),
    ('A-3'),
    ('A-4'),
    ('A-5'),
    ('A-10'),
    ('A-11'),
```

('A-20'),

('A-30');

Users expect this result -

The result given in the above image can be achieved by using the LENGTH function. The LENGTH function returns a string length. The user is aiming to sort the item_no data by length first, followed by sorting by column value -

```
SELECT
    item_no
FROM
    items
ORDER BY LENGTH(item_no) , item_no;
```

This results in the data being sorted naturally.

An alternative method is performing application layer natural sorting if the above methods are not useful.

Instructions: The progress of students will be assessed with the exercises mentioned below.

MCQ

1. By default MySQL returns selected rows in a _____ order.

a) descending

b) ascending

c) random

d) none of the mentioned

2. Which clause is used with the GROUP BY clause for sorting MySQL table data?

a) SELECT

b) CREATE

c) SORT

d) none of the mentioned

3. Results can be sorted on _____.

a) one field

b) two fields

c) both a and b

d) none of the mentioned

4. In a PHP script selected data can be fetched with _____.

a) mysql_fetch_string()

b) mysql_fetch_array()

c) mysql_fetch_data()

d) none of the mentioned

5. In which order is sorting done naturally?

a) numerically then alphabetically

b) alphabetically then numerically

c) no particular order

d) none of the mentioned