Ans-1

```
DELIMITER //

CREATE PROCEDURE Insval(var_id int,var_name varchar(20),var_phnno varchar(20))
BEGIN
insert into pro values(var_id,var_name,var_phnno);

END //

DELIMITER ;
```

```
Database changed
mysql> source Ans1.sql
Query OK, 0 rows affected (0.03 sec)

mysql> create table pro(id int,name varchar(20),phnno int);
Query OK, 0 rows affected (0.43 sec)

mysql> call Insval(1,'tushar',1);
Query OK, 1 row affected (0.08 sec)

mysql> call Insval(2,'raj',2);
Query OK, 1 row affected (0.04 sec)

mysql> call Insval(3,'verma',3);
Query OK, 1 row affected (0.04 sec)

mysql> call Insval(4,'abc',4);
Query OK, 1 row affected (0.06 sec)

mysql> call Insval(5,'cfg',5);
Query OK, 1 row affected (0.12 sec)

mysql>
```

Ans-2

```
DELIMITER //

CREATE PROCEDURE Insval(var_name varchar(20),var_phnno varchar(20))
BEGIN
DECLARE pk int;
select max(id) into pk from pro;
insert into pro values(pk+1,var_name,var_phnno);

END //

DELIMITER ;
```

```
mysql> drop procedure Insval;
Query OK, 0 rows affected (0.00 sec)

mysql> source Ans1.sql
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near '@pk i
nt
select max(id) into pk from pro;
insert into pro values(pk+1,var_name,va' at line 3
mysql> source Ans1.sql
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near 'selec
t max(id) into pk from pro;
insert into pro values(pk+1,var_name,var_phnno)' at line 4
mysql> source Ans1.sql
Query OK, 0 rows affected (0.00 sec)

mysql> call Insval('admin',2);
Query OK, 1 row affected (0.12 sec)

mysql> call Insval('employee',3);
Query OK, 1 row affected (0.05 sec)

mysql>
```

Ans3

Ans1.sql

```sql
DELIMITER $$

CREATE PROCEDURE condition_FUNC(
    IN  message varchar(100),
    OUT output  VARCHAR(100))
BEGIN
    case message

    when 'first' THEN
        SET output = 'first condition executed';

    when 'second' THEN
        SET output = 'second condition executed';
    when 'third' THEN
        SET output = 'third condition executed';
    ELSE
        SET output='Else condition executed';
    END CASE;
END$$

DELIMITER ;
```

Ans3.sql

```
mysql> source Ans3.sql
Query OK, 0 rows affected (0.00 sec)

mysql> call condition_FUNC('first',@output);
Query OK, 0 rows affected (0.00 sec)

mysql> select @output;
+-------------------------+
| @output                 |
+-------------------------+
| first condition executed |
+-------------------------+
1 row in set (0.00 sec)

mysql> call condition_FUNC('second',@output);
Query OK, 0 rows affected (0.00 sec)

mysql> select @output;
+--------------------------+
| @output                  |
+--------------------------+
| second condition executed |
+--------------------------+
```

Ans-4

Ans4.sql

```sql
DROP TABLE student_marks;
CREATE TABLE student_marks ( `student_id` INT NOT NULL, `marks` INT NOT NULL, `grade` VARCHAR(100) ,PRIMARY KEY (`student_id`));
CREATE PROCEDURE inser_ta()
BEGIN
insert into student_marks values (1, 70,null),(2,82,null),(3,94,null),(4,96,null),(5,68,null);
END;
CREATE PROCEDURE dem_ta()
BEGIN
DECLARE done INT DEFAULT FALSE;
DECLARE s_id int;
DECLARE mrks int;
DECLARE cur1 CURSOR FOR select student_id,marks from student_marks;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
OPEN cur1;
get_loop: LOOP
FETCH cur1 INTO s_id, mrks;
IF  done THEN LEAVE get_loop; END IF;
IF  mrks<50 THEN update student_marks set  grade ='D' where student_id = s_id;
ELSEIF  mrks<70 THEN update student_marks set  grade ='C' where student_id = s_id;
ELSEIF  mrks<90 THEN update student_marks set  grade ='B' where student_id = s_id;
ELSE update student_marks set  grade ='A' where student_id = s_id;
END IF;
END LOOP;
CLOSE cur1;
END;
CALL inser_ta;
CALL dem_ta;
select * from student_marks;
```

Q4.sql

```
mysql> source Ans4.sql
Query OK, 0 rows affected (0.16 sec)

Query OK, 0 rows affected (0.43 sec)

Query OK, 0 rows affected (0.43 sec)

Query OK, 0 rows affected (0.43 sec)

Query OK, 5 rows affected (0.47 sec)

Query OK, 0 rows affected (0.75 sec)

+------------+-------+-------+
| student_id | marks | grade |
+------------+-------+-------+
|          1 |    70 | B     |
|          2 |    82 | B     |
|          3 |    94 | A     |
|          4 |    96 | A     |
|          5 |    68 | C     |
+------------+-------+-------+
```

Ans-5

Ans5.sql

```sql
DELIMITER $$
CREATE TRIGGER before_employee_insert
BEFORE INSERT ON emp
FOR EACH ROW
BEGIN
INSERT INTO employe_audit
SET action = 'before insert',
employeeNumber = NEW.empno,
firstname = NEW.ename,
changedat = NOW();
END$$
DELIMITER ;

DELIMITER $$
CREATE TRIGGER after_employee_insert
AFTER INSERT ON emp
FOR EACH ROW
BEGIN
INSERT INTO employe_audit
SET action = 'after insert',
employeeNumber = NEW.empno,
firstname = NEW.ename,
changedat = NOW();
END$$
DELIMITER ;


DELIMITER $$
CREATE TRIGGER after_employee_update
AFTER UPDATE ON emp
FOR EACH ROW
BEGIN
INSERT INTO employe_audit
SET action = 'after update',
employeeNumber = OLD.empno,
firstname = OLD.ename,
changedat = NOW();
END$$
```

Terminal

```
mysql> select * from employe_audit;
+----+----------------+-----------+---------------------+---------------+
| id | employeeNumber | firstname | changedat           | action        |
+----+----------------+-----------+---------------------+---------------+
|  1 |              7 | raj       | 2020-03-01 14:22:28 | before insert |
|  2 |              7 | raj       | 2020-03-01 14:22:28 | after insert  |
|  3 |              4 | ttt       | 2020-03-01 14:26:12 | after update  |
|  4 |              7 | raj       | 2020-03-01 14:29:27 | before delete |
|  6 |              2 | tv        | 2020-03-01 14:31:10 | before delete |
|  8 |              2 | tv        | 2020-03-01 14:31:10 | after delete  |
+----+----------------+-----------+---------------------+---------------+
6 rows in set (0.00 sec)

mysql>
mysql>
```

Ans-6

i) **slow_query_log -** Boolean for turning the slow query log on and off.

- `slow_query_log`

Property	Value
Command-Line Format	`--slow-query-log[={OFF|ON}]`
System Variable	`slow_query_log`
Scope	Global
Dynamic	Yes
Type	Boolean
Default Value	OFF

Whether the slow query log is enabled. The value can be 0 (or **OFF**) to disable the log or 1 (or **ON**) to enable the log. The destination for log output is controlled by the `log_output` system variable; if that value is **NONE**, no log entries are written even if the log is enabled.

"Slow" is determined by the value of the `long_query_time` variable. See Section 5.4.5, "The Slow Query Log".

ii) **slow_query_log_file -** The absolute path for the query log file. The file's directory should be owned by the mysqld user and have the correct permissions to be read from and written.

- `slow_query_log_file`

Property	Value
Command-Line Format	`--slow-query-log-file=file_name`
System Variable	`slow_query_log_file`
Scope	Global
Dynamic	Yes
Type	File name
Default Value	`host_name-slow.log`

iv) log_queries_not_using_indexes - Boolean value whether to log queries that are not hitting indexes. When doing query analysis, it is important to log queries that are not hitting indexes.

- `log_queries_not_using_indexes`

Property	Value
Command-Line Format	`--log-queries-not-using-indexes[={OFF|ON}]`
System Variable	`log_queries_not_using_indexes`
Scope	Global
Dynamic	Yes
Type	Boolean
Default Value	`OFF`

If you enable this variable with the slow query log enabled, queries that are expected to retrieve all rows are logged. See Section 5.4.5, "The Slow Query Log". This option does not necessarily mean that no index is used. For example, a query that uses a full index scan uses an index but would be logged because the index would not limit the number of rows.

v) min_examined_row_limit - Sets a lower limit on how many rows should be examined. A value of 1000 would ignore any query that analyzes less than 1000 rows.

vii) profiling

- `profiling`

If set to 0 or `OFF` (the default), statement profiling is disabled. If set to 1 or `ON`, statement profiling is enabled and the `SHOW PROFILE` and `SHOW PROFILES` statements provide access to profiling information. See Section 13.7.5.32, "SHOW PROFILES Statement".

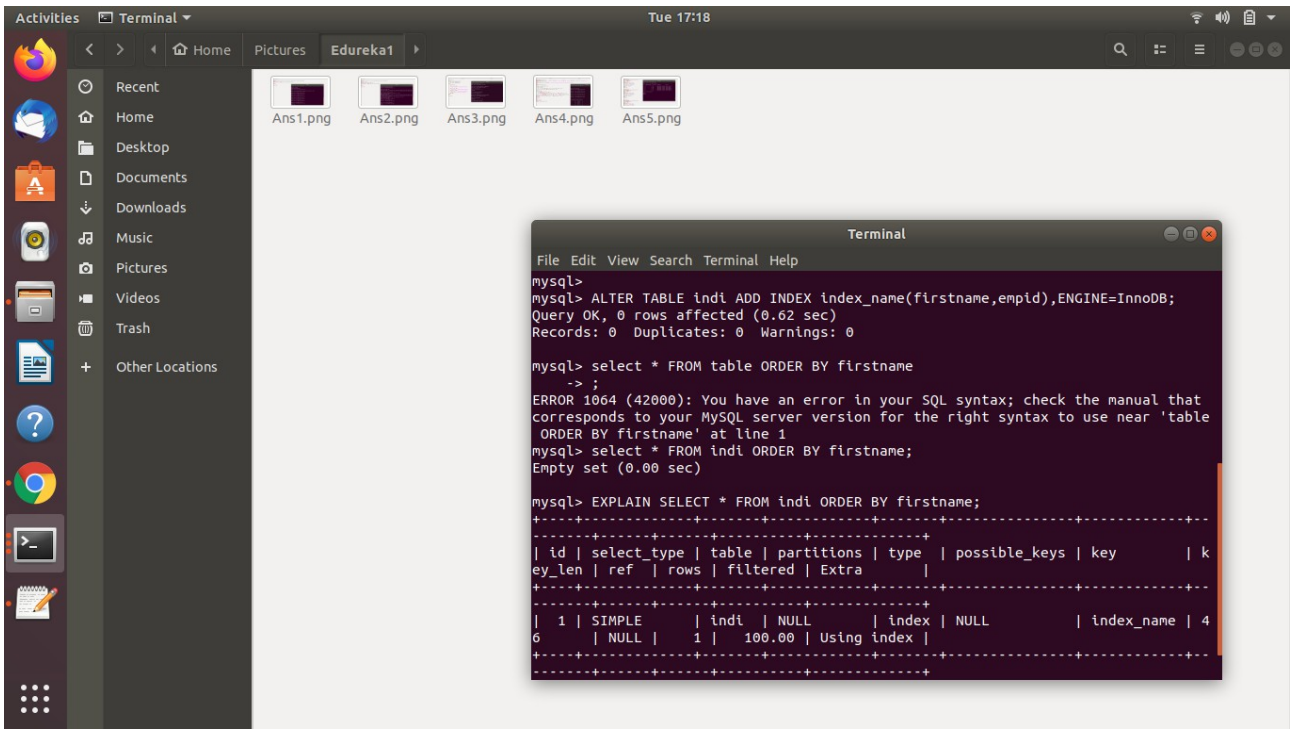
This variable is deprecated and will be removed in a future MySQL release.

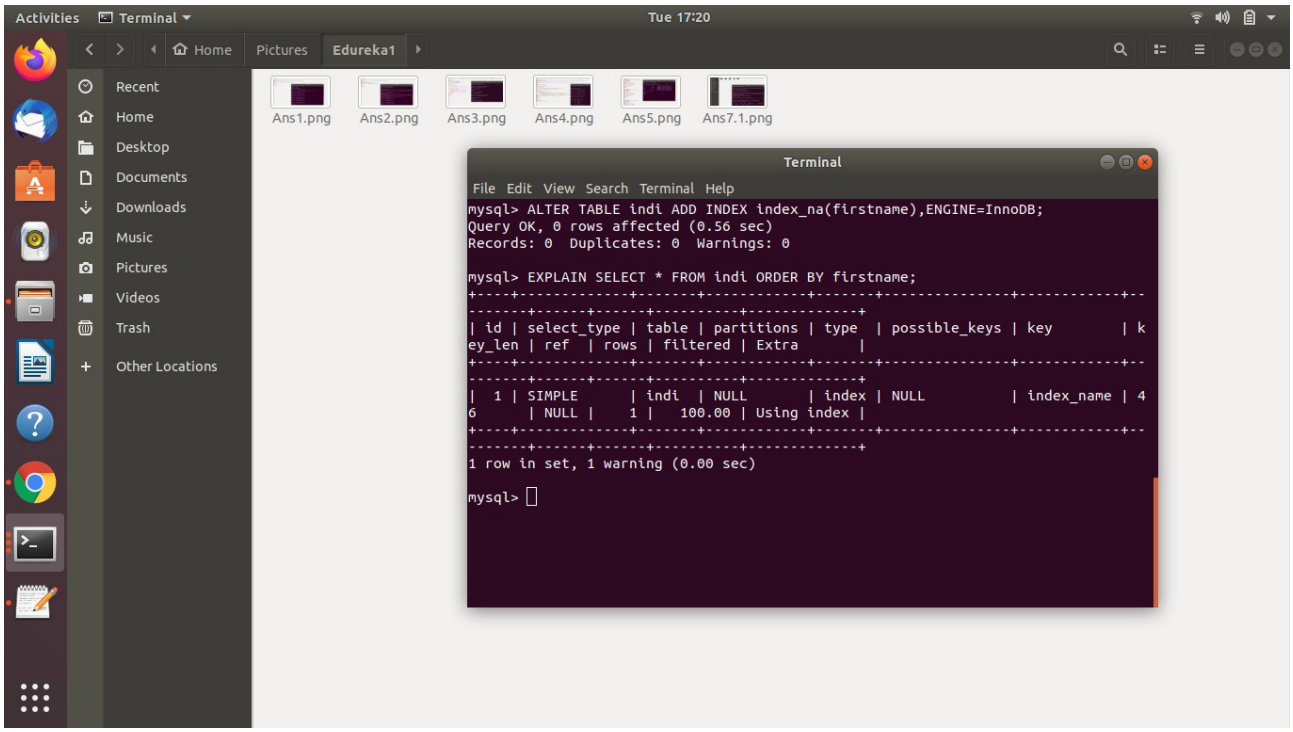
viii) profiling_history_size

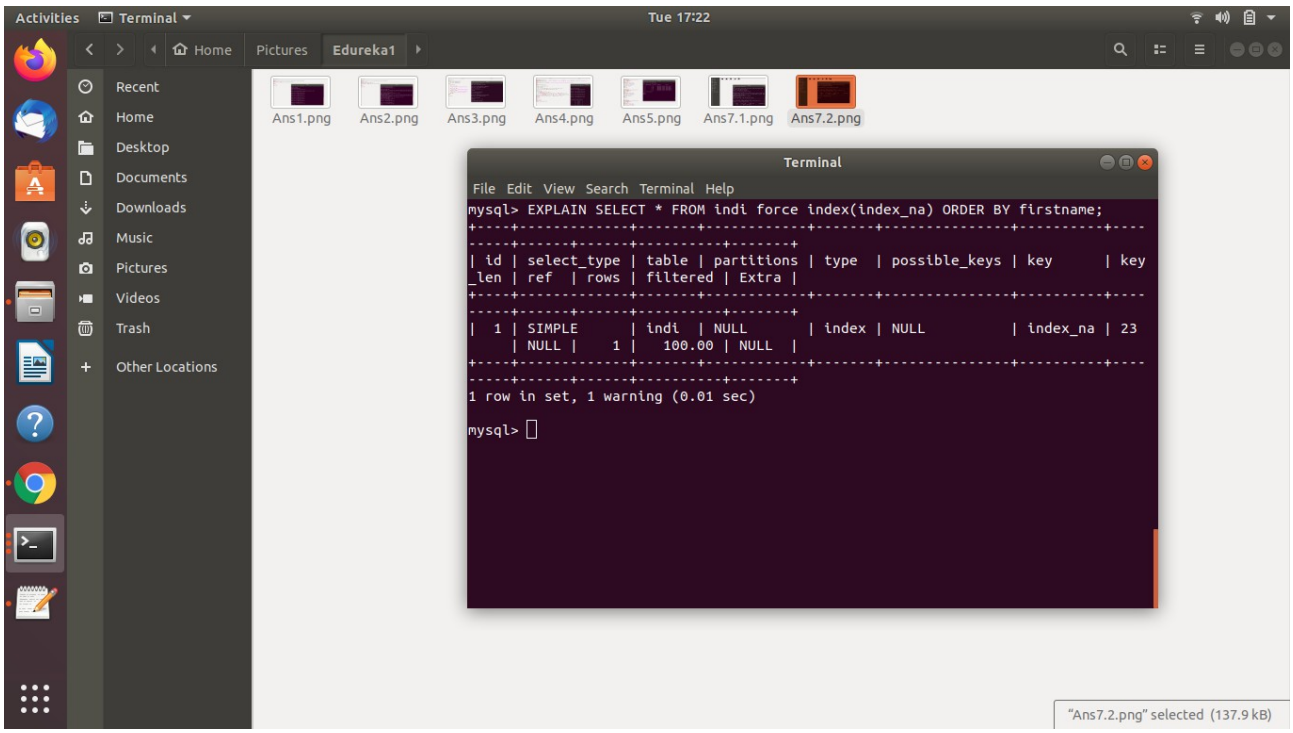
- `profiling_history_size`

The number of statements for which to maintain profiling information if `profiling` is enabled. The default value is 15. The maximum value is 100. Setting the value to 0 effectively disables profiling. See Section 13.7.5.32, "SHOW PROFILES Statement".

Ans-7


Activities
Terminal
Tue 17:18
Home
Pictures
Edureka1
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
Ans1.png
Ans2.png
Ans3.png
Ans4.png
Ans5.png
Terminal
File Edit View Search Terminal Help
mysql>
mysql> ALTER TABLE indi ADD INDEX index_name(firstname,empid),ENGINE=InnoDB;
Query OK, 0 rows affected (0.62 sec)
Records: 0 Duplicates: 0 Warnings: 0
mysql> select * FROM table ORDER BY firstname
-> ;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'table ORDER BY firstname' at line 1
mysql> select * FROM indi ORDER BY firstname;
Empty set (0.00 sec)
mysql> EXPLAIN SELECT * FROM indi ORDER BY firstname;
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
| 1 | SIMPLE | indi | NULL | index | NULL | index_name | 46 | NULL | 1 | 100.00 | Using index |


Activities
Terminal
Tue 17:20
Home
Pictures
Edureka1
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
Ans1.png
Ans2.png
Ans3.png
Ans4.png
Ans5.png
Ans7.1.png
Terminal
File Edit View Search Terminal Help
mysql> ALTER TABLE indi ADD INDEX index_na(firstname),ENGINE=InnoDB;
Query OK, 0 rows affected (0.56 sec)
Records: 0 Duplicates: 0 Warnings: 0
mysql> EXPLAIN SELECT * FROM indi ORDER BY firstname;
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
| 1 | SIMPLE | indi | NULL | index | NULL | index_name | 46 | NULL | 1 | 100.00 | Using index |
1 row in set, 1 warning (0.00 sec)
mysql>


Activities
Terminal
Tue 17:22
Home
Pictures
Edureka1
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
Ans1.png
Ans2.png
Ans3.png
Ans4.png
Ans5.png
Ans7.1.png
Ans7.2.png
Terminal
File Edit View Search Terminal Help
mysql> EXPLAIN SELECT * FROM indi force index(index_na) ORDER BY firstname;
+----+-------------+-------+------------+-------+---------------+----------+----
-----+------+------+----------+-------+
| id | select_type | table | partitions | type | possible_keys | key | key
_len | ref | rows | filtered | Extra |
+----+-------------+-------+------------+-------+---------------+----------+----
-----+------+------+----------+-------+
| 1 | SIMPLE | indi | NULL | index | NULL | index_na | 23
     | NULL | 1 | 100.00 | NULL |
+----+-------------+-------+------------+-------+---------------+----------+----
-----+------+------+----------+-------+
1 row in set, 1 warning (0.01 sec)
mysql>
"Ans7.2.png" selected (137.9 kB)