MySQL RDBMS

Trainer: Mr. Nilesh Ghule

Stored Procedure – PSM Syntax

VARIABLES

```
DECLARE varname DATATYPE;
DECLARE varname DATATYPE DEFAULT init_value;
SET varname = new_value;
SELECT new_value INTO varname;
SELECT expr_or_col INTO varname FROM table_name;
```

PARAMETERS

```
CREATE PROCEDURE sp_name(PARAMTYPE p1 DATATYPE)
BEGIN
    …
END;

-- IN param: Initialized by calling program.
-- OUT param: Initialized by called procedure.
-- INOUT param: Initialized by calling program and
modified by called procedure
-- OUT & INOUT param declared as session variables.

CREATE PROCEDURE sp_name(OUT p1 INT)
BEGIN
    SELECT 1 INTO p1;
END;

SET @res = 0;
CALL sp_name(@res);
SELECT @res;
```

mysql client

IF-ELSE

```
IF condition THEN
    body;
END IF;
```

```
IF condition THEN
    if-body;
ELSE
    else-body;
END IF;
```

```
IF condition THEN
    if1-body;
ELSE
    IF condition THEN
        if2-body;
    ELSE
        else2-body;
    END IF;
END IF;
```

```
IF condition THEN
    if1-body;
ELSEIF condition THEN
    if2-body;
ELSE
    else-body;
END IF;
```

LOOPS

loop
repeat

```
WHILE condition DO
    body;
END WHILE;
```

if cnd true.

like do-while executed at least once.

```
REPEAT
    body;
UNTIL condition
END REPEAT;
```

repeat if cnd is false.

```
label: LOOP
IF condition THEN
        ...
    LEAVE label;
    END IF;
    ...
END LOOP;
```

like infinite loop (no condn).

like "break" keyword in C. exit the loop.

ITERATE keyword is like "continue" in C.
skip further statements in the loop & continue executing next iteration of loop.

CASE-WHEN

```
CASE
WHEN condition THEN
    body;
WHEN condition THEN
    body;
ELSE
    body;
END CASE;
```

SHOW PROCEDURE

```
SHOW PROCEDURE STATUS
LIKE 'sp_name';
```

→ sp%

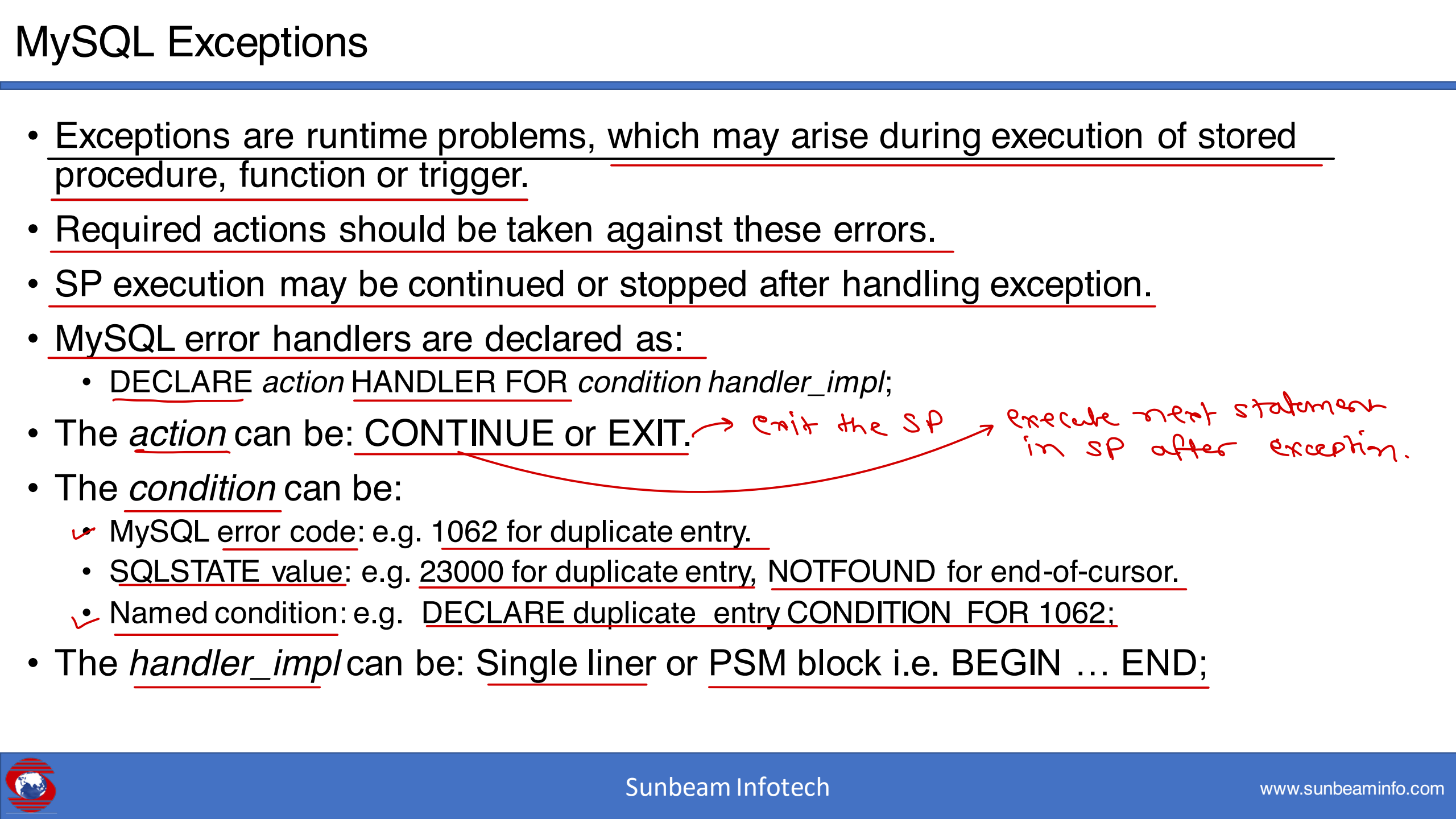
```
SHOW CREATE PROCEDURE sp_name;
```

DROP PROCEDURE

```
DROP PROCEDURE
IF EXISTS sp_name;
```

MySQL Exceptions

- Exceptions are runtime problems, which may arise during execution of stored procedure, function or trigger.
- Required actions should be taken against these errors.
- SP execution may be continued or stopped after handling exception.
- MySQL error handlers are declared as:
  - DECLARE *action* HANDLER FOR *condition handler_impl*;
- The *action* can be: CONTINUE or EXIT.
- The *condition* can be:
  - MySQL error code: e.g. 1062 for duplicate entry.
  - SQLSTATE value: e.g. 23000 for duplicate entry, NOTFOUND for end-of-cursor.
  - Named condition: e.g. DECLARE duplicate_entry CONDITION FOR 1062;
- The *handler_impl* can be: Single liner or PSM block i.e. BEGIN … END;



DECLARE v_cur CURSOR FOR SELECT * FROM dept;

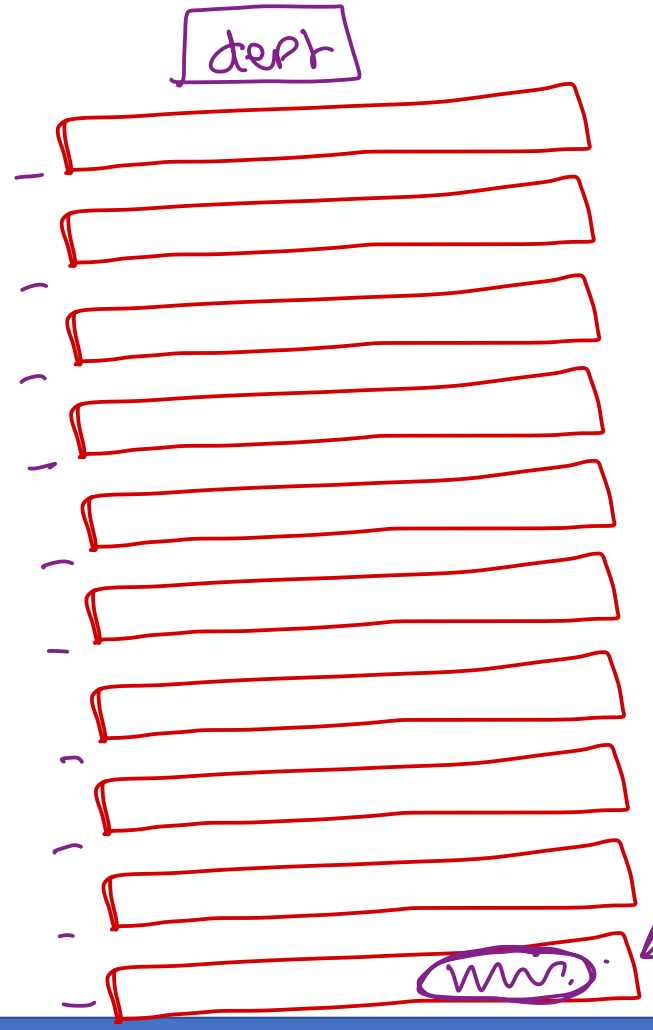
OPEN v_cur; ✓

LOOP

FETCH ... ✓

?

END LOOP;



Client 2

UPDATE dept

SET ...

WHERE ...

MySQL Stored Functions

- Stored Functions are MySQL programs like stored procedures.
- Functions can be having one or more parameters. MySQL allows only IN params.
- Functions must return some value using RETURN statement.
- Function entire code is stored in system table. (like procedure)
- Like procedures, functions allows statements like local variable declarations, if-else, case, loops, etc. One function can invoke another function/procedure and vice-versa. The functions can also be recursive.
- There are two types of functions: DETERMINISTIC and NOT DETERMINISTIC.

CREATE FUNCTION

```
CREATE FUNCTION fn_name(p1 TYPE)
RETURNS TYPE
[NOT] DETERMINISTIC
BEGIN
    body;
    RETURN value;
END;
```

SHOW FUNCTION

```
SHOW FUNCTION STATUS LIKE 'fn_name';

SHOW CREATE FUNCTION fn_name;
```

DROP FUNCTION

```
DROP FUNCTION IF EXISTS fn_name;
```

Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>