

Assignment 08

MySQL Workbench Screenshot:

Session 1 (Top):

```

3 -- BEGIN
4 --     DECLARE done INT DEFAULT 0;
5 --     DECLARE e_name VARCHAR(100);
6 --     DECLARE cur CURSOR FOR
7 --         SELECT emp_name FROM employees WHERE dept_id = did;
8
9 --     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
10
11 --     OPEN cur;
12
13 --     read_loop: LOOP
14 --         FETCH cur INTO e_name;
15 --         IF done THEN
16 --             LEAVE read_loop;
17 --         END IF;
18
19 --     END LOOP read_loop;
20
21 --     CLOSE cur;
22
23 --     INSERT INTO employees (emp_name, salary, dept_id)
24 --     VALUES (e_name, sal, did);
25
26 --     SELECT CONCAT('Employee ', e_name, ' added safely.') AS Message;
27
28 END
29
30 DELIMITER ;
31
32 CREATE PROCEDURE safe_add_employee(IN ename VARCHAR(100), IN sal DECIMAL(10,2), IN did INT)
33 BEGIN
34
35     DECLARE dept_exists INT;
36
37     SELECT COUNT(*) INTO dept_exists FROM departments WHERE dept_id = did;
38
39     IF dept_exists = 0 THEN
40         SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Invalid Department ID';
41     ELSE
42         INSERT INTO employees (emp_name, salary, dept_id)
43         VALUES (ename, sal, did);
44         SELECT CONCAT('Employee ', ename, ' added safely.') AS Message;
45     END IF;
46
47 END
48
49 CALL safe_add_employee('Grace', 60000, 9);

```

Session 2 (Bottom):

Table: **departments**

```

Columns:
dept_id int AI PK
dept_name varchar(50)

```

Output:

Action	Time	Action	Message
20	23:41:56	CREATE PROCEDURE delete_employee(IN eid INT) BEGIN	DECLARE rows_affected INT; DELETE F...
21	23:42:13	CALL delete_employee(2)	0 row(s) affected
22	23:43:17	CREATE PROCEDURE safe_add_employee(IN ename VARCHAR(100), IN sal DECIMAL(10,2), IN did INT)	B... 0 row(s) affected
23	23:43:28	CALL safe_add_employee('Grace', 60000, 99)	Err Code: 1644. Invalid Department ID
24	23:44:12	CALL safe_add_employee('Grace', 60000, 10)	Err Code: 1644. Invalid Department ID
25	23:44:25	CALL safe_add_employee('Grace', 60000, 9)	Err Code: 1644. Invalid Department ID

Snipping Tool message: Screenshot copied to clipboard. Automatically saved to screenshots folder.

Navigator

SCHEMAS

Filter objects

- ▶ bank
- ▶ employee
 - ▶ Tables
 - ▶ Views
 - ▶ Stored Procedures
 - ▶ Functions
- ▶ handler_lab
 - ▶ Tables
 - ▶ departments
 - ▶ employees
 - ▶ Views
 - ▶ Stored Procedures
 - ▶ Functions
- ▶ sakila
- ▶ student
- ▶ sys
- ▶ world

-- DELIMITER \$\$

-- CREATE PROCEDURE delete_employee(IN eid INT)

-- BEGIN

-- DECLARE rows_affected INT;

--

-- DELETE FROM employees WHERE emp_id = eid;

-- SET rows_affected = ROW_COUNT();

--

-- IF rows_affected = 0 THEN

-- SELECT 'No such employee' AS Message;

-- ELSE

-- SELECT CONCAT('Employee ID ', eid, ' deleted successfully') AS Message;

-- END IF;

-- END\$\$

-- DELIMITER ;

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

Message
Employee ID 2 deleted successfully

Result Grid

The screenshot shows the MySQL Workbench interface. The Navigator pane on the left lists databases: bank, employee, handler_lab, sakila, student, sys, and world. The 'employee' database is expanded, showing Tables, Views, Stored Procedures, and Functions. The 'handler_lab' database is also expanded, showing Tables (departments, employees), Views, Stored Procedures, and Functions. The query editor window on the right contains the following SQL code:

```
-- DELIMITER $$  
-- CREATE PROCEDURE count_employees()  
-- BEGIN  
--     DECLARE done INT DEFAULT 0;  
--     DECLARE emp_count INT DEFAULT 0;  
--     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;  
--  
--     SELECT COUNT(*) INTO emp_count FROM employees WHERE dept_id IS NOT NULL;  
--     SELECT CONCAT('Employees with department:', emp_count) AS Message;  
-- END$$  
-- DELIMITER ;  
CALL count_employees();
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:		
<table border="1"><thead><tr><th>Message</th></tr></thead><tbody><tr><td>Employees with department:11</td></tr></tbody></table>	Message	Employees with department:11			
Message					
Employees with department:11					

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree, which includes the 'employee' schema expanded to show 'Tables', 'Views', and 'Stored Procedures'. Below it is the 'handler_lab' schema, also expanded. The main area is a query editor containing the following SQL code:

```
-- DELIMITER $$  
-- CREATE FUNCTION annual_salary(monthly DECIMAL(10,2))  
-- RETURNS DECIMAL(10,2)  
-- DETERMINISTIC  
-- BEGIN  
--     RETURN monthly * 12;  
-- END$$  
-- DELIMITER ;  
  
SELECT annual_salary(5000);
```

Result Grid | Filter Rows: _____ | Export: _____ | Wrap Cell Content: _____

Result 5

Navigator

SCHEMAS

Filter objects

- bank
- employee
 - Tables
 - Views
 - Stored Procedures
 - Functions
- handler_lab
 - Tables
 - departments
 - employees
 - Views
 - Stored Procedures
 - Functions
- sakila
- student
- sys
- world

Code Editor

```

1 -- DELIMITER $$ 
2 -- CREATE PROCEDURE check_salary(IN sal DECIMAL(10,2))
3 -- BEGIN
4 --   DECLARE EXIT HANDLER FOR SQLEXCEPTION
5 --   BEGIN
6 --     SELECT 'Salary check failed: Below minimum allowed' AS Message;
7 --   END;
8
9 --   IF sal < 1000 THEN
10 --     SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Salary too low';
11 --   ELSE
12 --     SELECT CONCAT('Salary is valid: ', sal) AS Message;
13 --   END IF;
14 -- END$$
15 -- DELIMITER ;

```

Result Grid

Message
Salary check failed: Below minimum allowed

Administration **Schemas**

Information

Navigator

SCHEMAS

Filter objects

- bank
- employee
 - Tables
 - Views
 - Stored Procedures
 - Functions
- handler_lab
 - Tables
 - departments
 - employees
 - Views
 - Stored Procedures
 - Functions
- sakila
- student
- sys
- world

Code Editor

```

5 -- DECLARE CONTINUE HANDLER FOR 1062
6 -- BEGIN
7 --   SELECT CONCAT('Duplicate found for employee_', i, ', skipped.') AS Info;
8 -- END;
9
10 -- WHILE i <= 5 DO
11 --   INSERT INTO employees (emp_name, salary, dept_id)
12 --   VALUES (CONCAT('employee_', i), 40000 + (i * 1000), (i % 5) + 1);
13 --   SET i = i + 1;
14 -- END WHILE;
15
16 -- SELECT 'Bulk insert completed' AS Status;
17 -- END$$
18 -- DELIMITER ;
19

```

Result Grid

Status
Bulk insert completed

Result Grid

Schemas

```

3   -- BEGIN
4   -- DECLARE EXIT HANDLER FOR 1062
5   -- BEGIN
6   --     SELECT 'Employee already exists' AS Message;
7   -- END;
8
9   -- INSERT INTO employees (emp_name, salary, dept_id)
10  -- VALUES (ename, sal, did);
11
12  -- SELECT CONCAT('Employee ', ename, ' added successfully') AS Message;
13  -- END$$
14  -- DELIMITER ;
15
16 • CALL add_employee('John', 55000, 2);
17

```

Result Grid

Message
Employee John added successfully

Administration Schemas

Schemas

```

1   -- DELIMITER $$
2   -- CREATE PROCEDURE add_department(IN dname VARCHAR(50))
3   -- BEGIN
4   --     DECLARE EXIT HANDLER FOR 1062
5   --     BEGIN
6   --         SELECT 'Department already exists' AS Message;
7   --     END;
8
9   --     INSERT INTO departments (dept_name) VALUES (dname);
10  --     SELECT CONCAT('Department ', dname, ' added successfully') AS Message;
11  -- END$$
12  -- DELIMITER ;
13
14 • call add_department(12);

```

Result Grid

Message
Department 12 added successfully

Administration Schemas

Information

Result 1