

Exercise 3: Stored Procedures

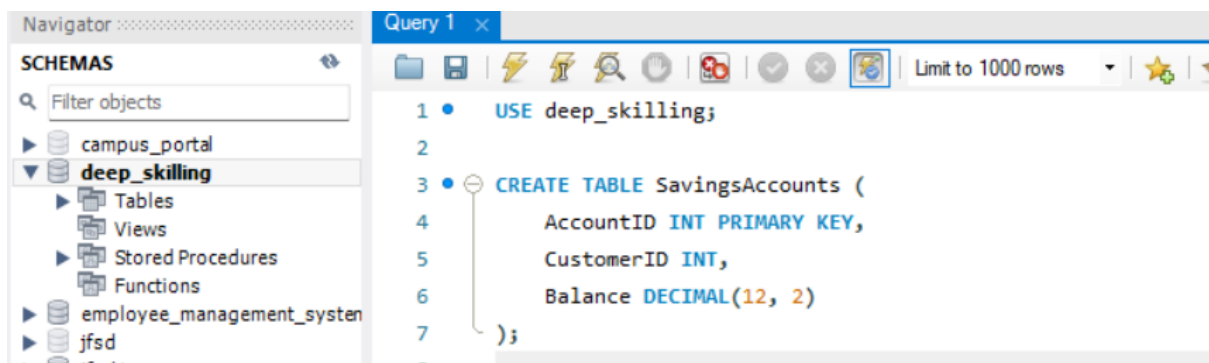
FROM FILE PLSQL Exercises

Create table Savingaccount:

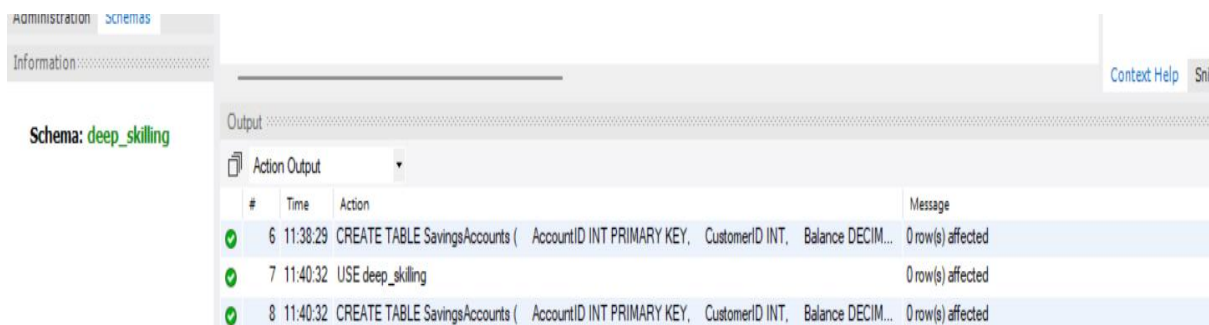
Code:

```
CREATE TABLE SavingsAccounts (  
    AccountID INT PRIMARY KEY,  
    CustomerID INT,  
    Balance DECIMAL(12, 2)  
);
```

Practice screen from Mysql workbench



Output screen:



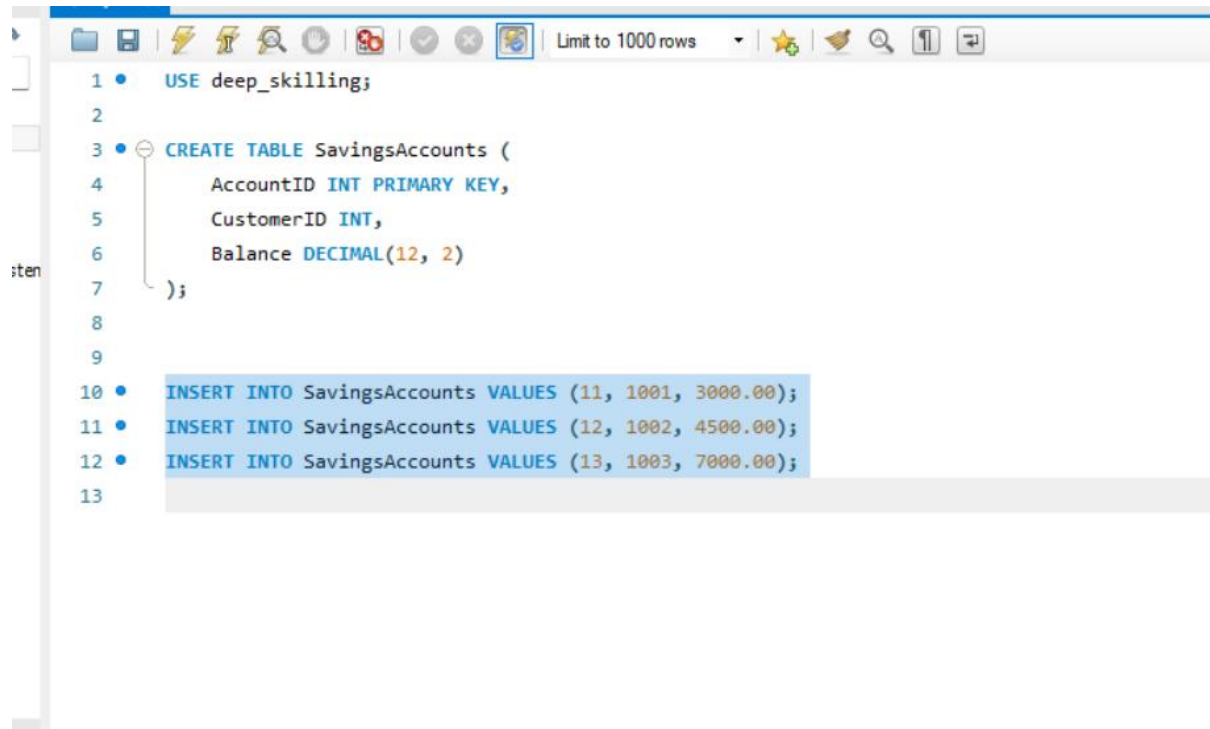
Insert the data into table code:

```
INSERT INTO SavingsAccounts VALUES (11, 1001, 3000.00);
```

```
INSERT INTO SavingsAccounts VALUES (12, 1002, 4500.00);
```

```
INSERT INTO SavingsAccounts VALUES (13, 1003, 7000.00);
```

Practice screen from Mysql workbench:



Output screen:

Information

Schema: deep_skilling

Output

Action Output

#	Time	Action	Message
✓ 6	11:38:29	CREATE TABLE SavingsAccounts (AccountID INT PRIMARY KEY, CustomerID INT, Balance DECIM...	0 row(s) affected
✓ 7	11:40:32	USE deep_skilling	0 row(s) affected
✓ 8	11:40:32	CREATE TABLE SavingsAccounts (AccountID INT PRIMARY KEY, CustomerID INT, Balance DECIM...	0 row(s) affected
✓ 9	11:40:53	INSERT INTO SavingsAccounts VALUES (11, 1001, 3000.00)	1 row(s) affected
✓ 10	11:40:53	INSERT INTO SavingsAccounts VALUES (12, 1002, 4500.00)	1 row(s) affected
✓ 11	11:40:53	INSERT INTO SavingsAccounts VALUES (13, 1003, 7000.00)	1 row(s) affected

Object Info

Session

In Stored procedures :

DELIMITER //

CREATE PROCEDURE ProcessMonthlyInterest()

BEGIN

 UPDATE SavingsAccounts

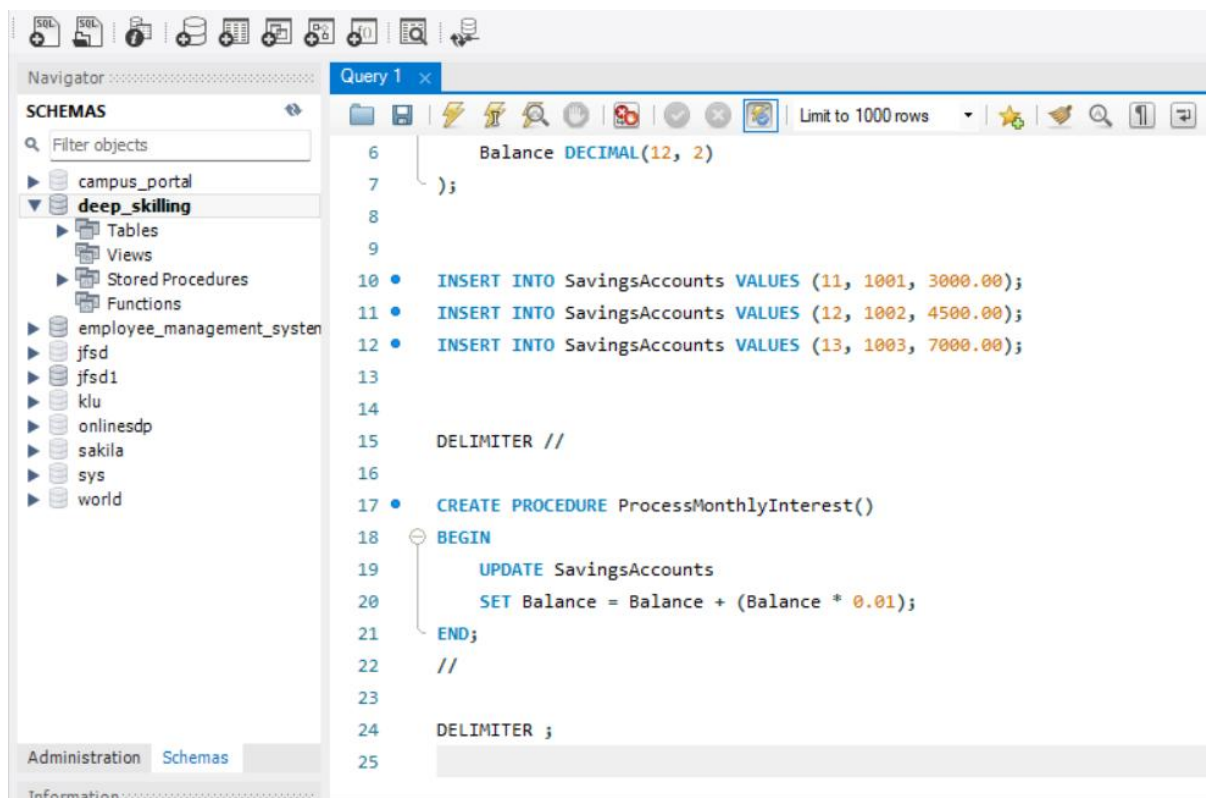
 SET Balance = Balance + (Balance * 0.01);

END;

//

DELIMITER ;

Practice screen from Mysql workbench:



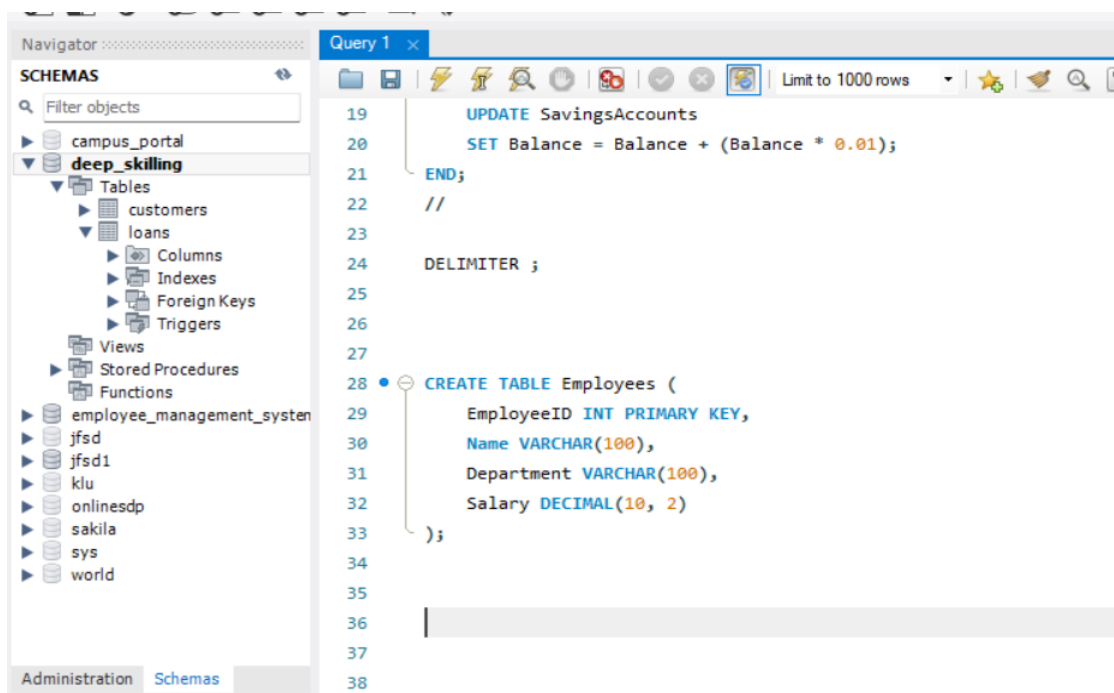
Output screen:

Information:.....			
Schema: deep_skilling			
Object Info Session			
Output			
Action Output			
#	Time	Action	Message
✓ 7	11:40:32	USE deep_skilling	0 row(s) affected
✓ 8	11:40:32	CREATE TABLE SavingsAccounts (AccountID INT PRIMARY KEY, CustomerID INT, Balance DECIM...	0 row(s) affected
✓ 9	11:40:53	INSERT INTO SavingsAccounts VALUES (11, 1001, 3000.00)	1 row(s) affected
✓ 10	11:40:53	INSERT INTO SavingsAccounts VALUES (12, 1002, 4500.00)	1 row(s) affected
✓ 11	11:40:53	INSERT INTO SavingsAccounts VALUES (13, 1003, 7000.00)	1 row(s) affected
✓ 12	11:51:33	CREATE PROCEDURE ProcessMonthlyInterest() BEGIN UPDATE SavingsAccounts SET Balance = Bala...	0 row(s) affected

Create Employees table

Code:

```
CREATE TABLE Employees (  
    EmployeeID INT PRIMARY KEY,  
    Name VARCHAR(100),  
    Department VARCHAR(100),  
    Salary DECIMAL(10, 2));
```



Output screen:

Schema: deep_skilling		Output			
		Action Output			
#	Time	Action	Message		
✓ 8	11:40:32	CREATE TABLE SavingsAccounts (AccountID INT PRIMARY KEY, CustomerID INT, Balance DECIM...	0 row(s) affected		
✓ 9	11:40:53	INSERT INTO SavingsAccounts VALUES (11, 1001, 3000.00)	1 row(s) affected		
✓ 10	11:40:53	INSERT INTO SavingsAccounts VALUES (12, 1002, 4500.00)	1 row(s) affected		
✓ 11	11:40:53	INSERT INTO SavingsAccounts VALUES (13, 1003, 7000.00)	1 row(s) affected		
✓ 12	11:51:33	CREATE PROCEDURE ProcessMonthlyInterest() BEGIN UPDATE SavingsAccounts SET Balance = Bala...	0 row(s) affected		
✓ 13	11:57:37	CREATE TABLE Employees (EmployeeID INT PRIMARY KEY, Name VARCHAR(100), Department VA...	0 row(s) affected		

Insert data:

INSERT INTO Employees VALUES (201, 'Nikhil', 'Finance', 25000);

INSERT INTO Employees VALUES (202, 'sruthi', 'IT', 30000);

INSERT INTO Employees VALUES (203, 'Manoj', 'Finance', 27000);

```
Query 1 x
Limit to 1000 rows

23
24 DELIMITER ;
25
26
27
28 CREATE TABLE Employees (
29     EmployeeID INT PRIMARY KEY,
30     Name VARCHAR(100),
31     Department VARCHAR(100),
32     Salary DECIMAL(10, 2)
33 );
34 INSERT INTO Employees VALUES (206, 'Nikhil', 'Finance', 25000);
35 INSERT INTO Employees VALUES (207, 'Sruthi', 'IT', 30000);
36 INSERT INTO Employees VALUES (208, 'Manoj', 'Finance', 27000);
37
38
39
40
```

Output screen:

✓	20	12:04:59	INSERT INTO Employees VALUES (206, 'Nikhil', 'Finance', 25000)	1 row(s) affected
✓	21	12:04:59	INSERT INTO Employees VALUES (207, 'Sruthi', 'IT', 30000)	1 row(s) affected
✓	22	12:04:59	INSERT INTO Employees VALUES (208, 'Manoj', 'Finance', 27000)	1 row(s) affected

Stored procedures

code:

DELIMITER //

```
CREATE PROCEDURE UpdateEmployeeBonus(  
    IN dept_name VARCHAR(100),  
    IN bonus_percent DECIMAL(5, 2)  
)  
BEGIN  
    UPDATE Employees  
    SET Salary = Salary + (Salary * bonus_percent / 100)  
    WHERE Department = dept_name;  
END;  
//  
  
DELIMITER ;
```

```

39 DELIMITER //
40
41 CREATE PROCEDURE UpdateEmployeeBonus(
42     IN dept_name VARCHAR(100),
43     IN bonus_percent DECIMAL(5, 2)
44 )
45 BEGIN
46     UPDATE Employees
47     SET Salary = Salary + (Salary * bonus_percent / 100)
48     WHERE Department = dept_name;
49 END;
50 //
51
52 DELIMITER ;
53
54

```

Output screen:

Schema: deep_skilling		Output		Context Help	Snippets
		Action Output			
#	Time	Action	Message		
18	12:00:04	INSERT INTO Employees VALUES (201, 'Nikhil', 'Finance', 25000)	Error Code: 1062. Duplicate entry '201' for key 'employees.PRIMARY'		
19	12:01:30	INSERT INTO Employees VALUES (201, 'Nikhil', 'Finance', 25000)	Error Code: 1062. Duplicate entry '201' for key 'employees.PRIMARY'		
20	12:04:59	INSERT INTO Employees VALUES (206, 'Nikhil', 'Finance', 25000)	1 row(s) affected		
21	12:04:59	INSERT INTO Employees VALUES (207, 'Sruthi', 'IT', 30000)	1 row(s) affected		
22	12:04:59	INSERT INTO Employees VALUES (208, 'Manoj', 'Finance', 27000)	1 row(s) affected		
23	12:09:35	CREATE PROCEDURE UpdateEmployeeBonus(IN dept_name VARCHAR(100), IN bonus_percent DECI...	0 row(s) affected		

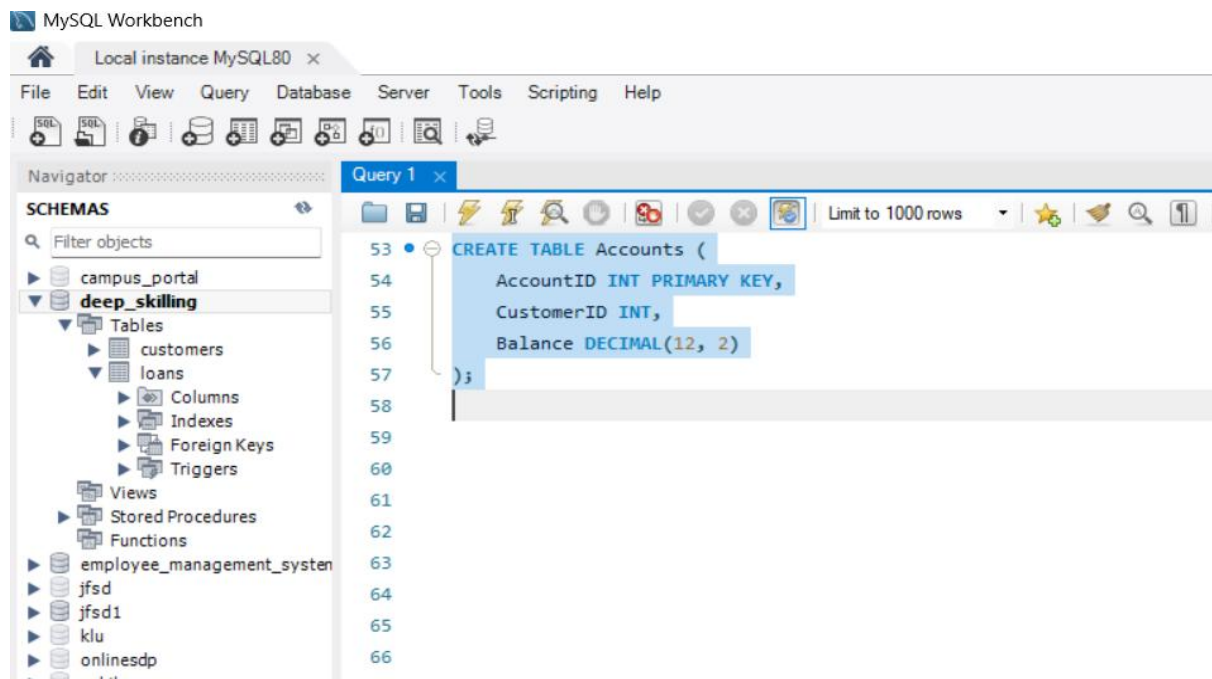
Account table

code:

```

CREATE TABLE Accounts (
    AccountID INT PRIMARY KEY,
    CustomerID INT,
    Balance DECIMAL(12, 2)
);

```

Output screen

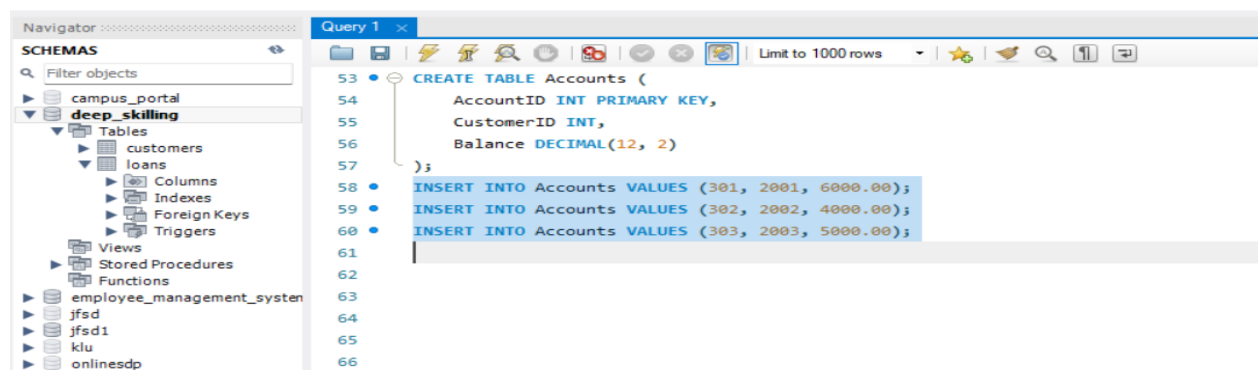
Schema: deep_skilling			
Output			
Action Output			
#	Time	Action	Message
19	12:01:30	INSERT INTO Employees VALUES (201, 'Nikhil', 'Finance', 25000)	Error Code: 1062. Duplicate entry
20	12:04:59	INSERT INTO Employees VALUES (206, 'Nikhil', 'Finance', 25000)	1 row(s) affected
21	12:04:59	INSERT INTO Employees VALUES (207, 'Sruthi', 'IT', 30000)	1 row(s) affected
22	12:04:59	INSERT INTO Employees VALUES (208, 'Manoj', 'Finance', 27000)	1 row(s) affected
23	12:09:35	CREATE PROCEDURE UpdateEmployeeBonus(IN dept_name VARCHAR(100), IN bonus_percent DECIMAL(5, 2))	0 row(s) affected
24	12:11:50	CREATE TABLE Accounts (AccountID INT PRIMARY KEY, CustomerID INT, Balance DECIMAL(12, 2))	0 row(s) affected

Insert the data in account table

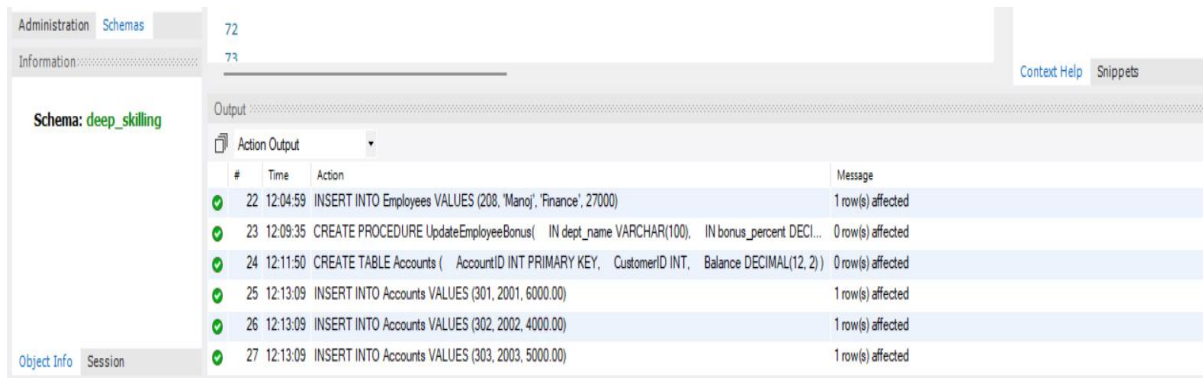
INSERT INTO Accounts VALUES (301, 2001, 6000.00);

INSERT INTO Accounts VALUES (302, 2002, 4000.00);

INSERT INTO Accounts VALUES (303, 2003, 5000.00);



Output screen:



The screenshot shows a database management tool interface. On the left, there's a sidebar with 'Administration' and 'Schemas' tabs. The 'Schemas' tab is active, showing a tree view with 'Schema: deep_skilling'. Below this, there are tabs for 'Object Info' and 'Session'. The main area is titled 'Output' and contains a table of execution results. The table has columns for '#', 'Time', 'Action', and 'Message'. The results show a series of SQL statements being executed, including an INSERT into Employees, a CREATE PROCEDURE statement, a CREATE TABLE statement, and several INSERT statements into the Accounts table. Each row is preceded by a green checkmark icon, indicating successful execution. The messages indicate the number of rows affected by each statement.

#	Time	Action	Message
22	12:04:59	INSERT INTO Employees VALUES (208, 'Manoj', 'Finance', 27000)	1 row(s) affected
23	12:09:35	CREATE PROCEDURE UpdateEmployeeBonus(IN dept_name VARCHAR(100), IN bonus_percent DECIMAL(12, 2))	0 row(s) affected
24	12:11:50	CREATE TABLE Accounts (AccountID INT PRIMARY KEY, CustomerID INT, Balance DECIMAL(12, 2))	0 row(s) affected
25	12:13:09	INSERT INTO Accounts VALUES (301, 2001, 6000.00)	1 row(s) affected
26	12:13:09	INSERT INTO Accounts VALUES (302, 2002, 4000.00)	1 row(s) affected
27	12:13:09	INSERT INTO Accounts VALUES (303, 2003, 5000.00)	1 row(s) affected

Stored procedures code:

DELIMITER //

CREATE PROCEDURE TransferFunds(

IN from_account INT,

IN to_account INT,

IN amount DECIMAL(12, 2)

)

BEGIN

DECLARE from_balance DECIMAL(12, 2);

SELECT Balance INTO from_balance

FROM Accounts

WHERE AccountID = from_account;

IF from_balance >= amount THEN

UPDATE Accounts

SET Balance = Balance - amount

```

WHERE AccountID = from_account;

UPDATE Accounts

SET Balance = Balance + amount

WHERE AccountID = to_account;

ELSE

    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Insufficient
Funds';

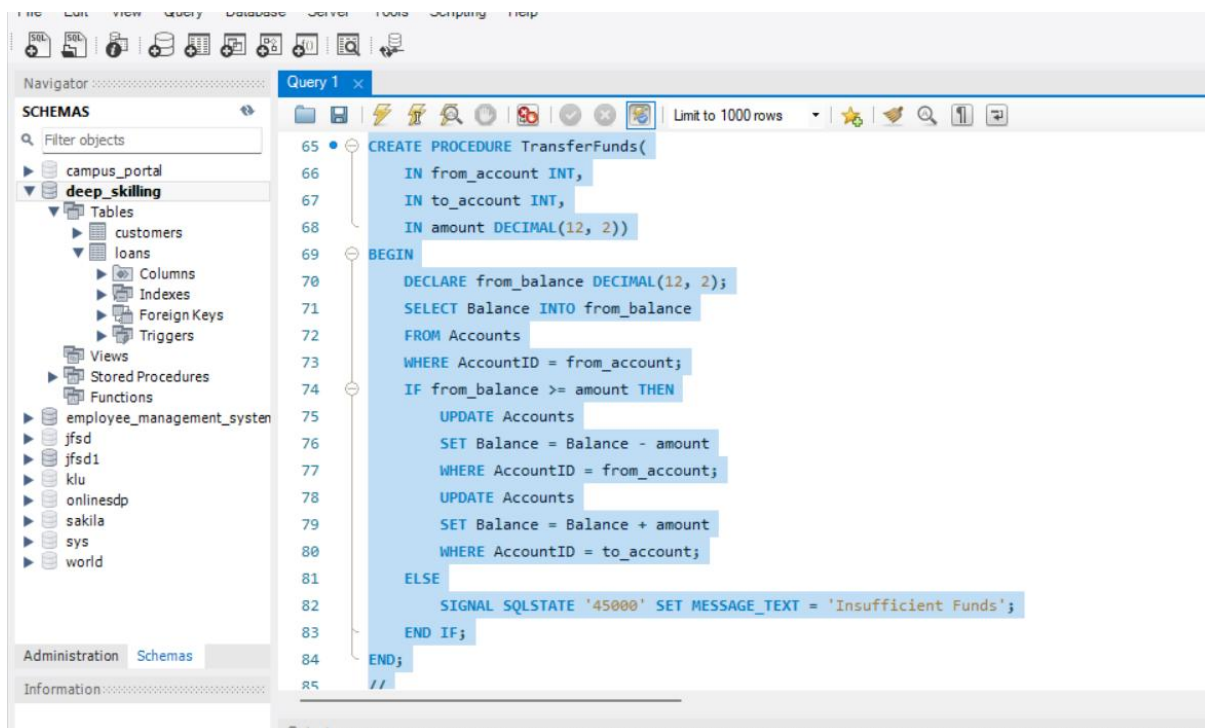
END IF;

END;

//

DELIMITER ;

```



Output screen:

Information	76	SET Balance = Balance + amount	Context
Schema: deep_skilling	Output		
	Action Output		
#	Time	Action	Message
23	12:09:35	CREATE PROCEDURE UpdateEmployeeBonus(IN dept_name VARCHAR(100), IN bonus_percent DECIMAL(4,2))	0 row(s) affected
24	12:11:50	CREATE TABLE Accounts (AccountID INT PRIMARY KEY, CustomerID INT, Balance DECIMAL(12,2))	0 row(s) affected
25	12:13:09	INSERT INTO Accounts VALUES (301, 2001, 6000.00)	1 row(s) affected
26	12:13:09	INSERT INTO Accounts VALUES (302, 2002, 4000.00)	1 row(s) affected
27	12:13:09	INSERT INTO Accounts VALUES (303, 2003, 5000.00)	1 row(s) affected
28	12:16:46	CREATE PROCEDURE TransferFunds(IN from_account INT, IN to_account INT, IN amount DECIMAL(12,2))	0 row(s) affected

Question: Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.

Code:

```
CALL ProcessMonthlyInterest();
```

```
SELECT * FROM SavingsAccounts;
```

Output Screen:

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

campus_portal

deep_skilling

Tables

customers

loans

Columns

Indexes

Foreign Keys

Triggers

Views

Stored Procedures

Functions

employee_management_system

jfsd

jfsd1

klu

onlinesdp

sakila

sys

world

Query 1

Limit to 1000 rows

```
89
90
91 • CALL ProcessMonthlyInterest();
92 • SELECT * FROM SavingsAccounts;
93
94
95
96
```

Result Grid

Filter Rows:

Edit

Export/Import

AccountID	CustomerID	Balance
11	1001	3030.00
12	1002	4545.00
13	1003	7070.00
NULL	NULL	NULL

Question: Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

A) Code:

```
CALL UpdateEmployeeBonus('Finance', 5);  
SELECT * FROM Employees;
```

Output Screen:

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

campus_portal

deep_skilling

Tables

customers

loans

Columns

Indexes

Foreign Keys

Triggers

Views

Stored Procedures

Functions

employee_management_system

jfsd

jfsd1

klu

onlinesdp

sakila

sys

world

Query 1

94

95 • CALL UpdateEmployeeBonus('Finance', 5);

96 • SELECT * FROM Employees;

97

98

99

100

101

Result Grid

Filter Rows:

Edit: Export/Import:

EmployeeID	Name	Department	Salary
201	Nikhil	Finance	26250.00
202	sruthi	IT	30000.00
206	Nikhil	Finance	26250.00
207	Sruthi	IT	30000.00
208	Manoj	Finance	28350.00
*	NULL	NULL	NULL

Question: Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

A) Code

```
CALL TransferFunds(301, 302, 1000);  
SELECT * FROM Accounts;
```

OUTPUT SCREEN:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'deep_skilling' expanded, showing tables like 'customers', 'loans', and 'Accounts'. The main query editor shows the following SQL code:

```
97  
98  
99 • CALL TransferFunds(301, 302, 1000);  
100 • SELECT * FROM Accounts;  
101  
102  
103
```

The 'Result Grid' at the bottom displays the output of the query, showing the state of the 'Accounts' table after the transfer. The table has three columns: AccountID, CustomerID, and Balance.

AccountID	CustomerID	Balance
301	2001	5000.00
302	2002	5000.00
303	2003	5000.00
NULL	NULL	NULL