PROMINEO TECH

Relational Databases with MySQL Week 4 Coding Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

Instructions: Using a text editor of your choice, write the queries that accomplishes the objectives listed below. Take screenshots of the queries and results and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document, with your Java project code, to the repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

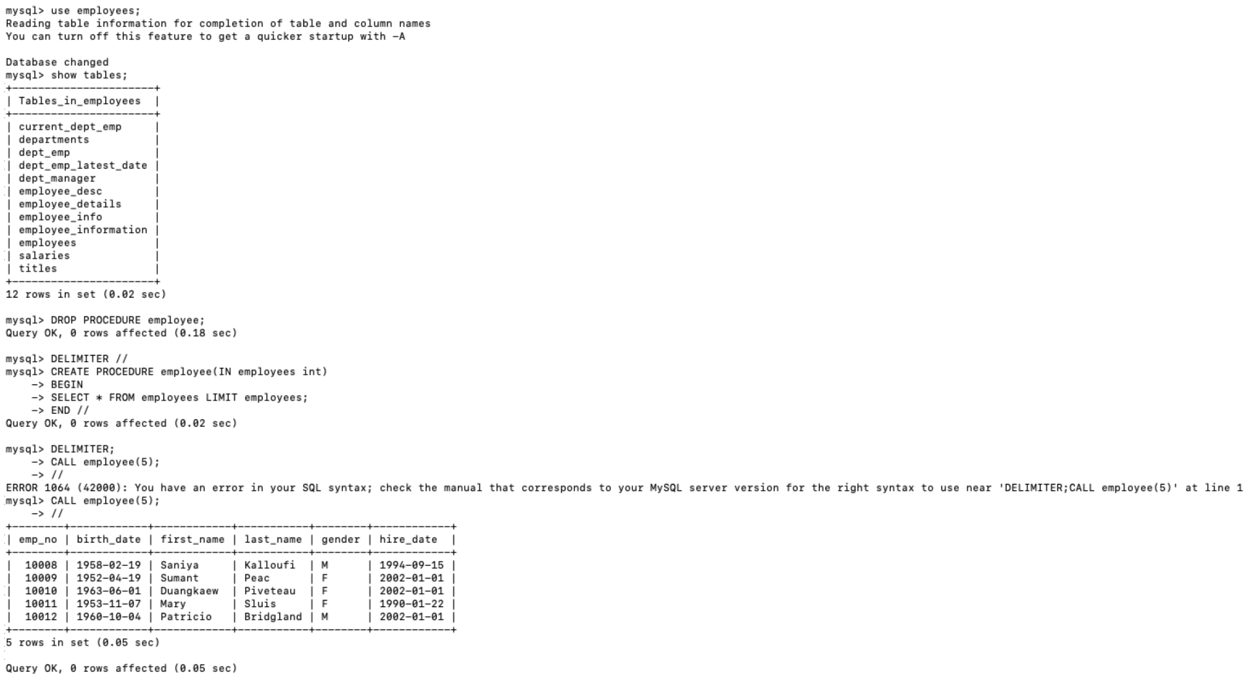
Coding Steps:

Write 5 stored procedures for the employees database.

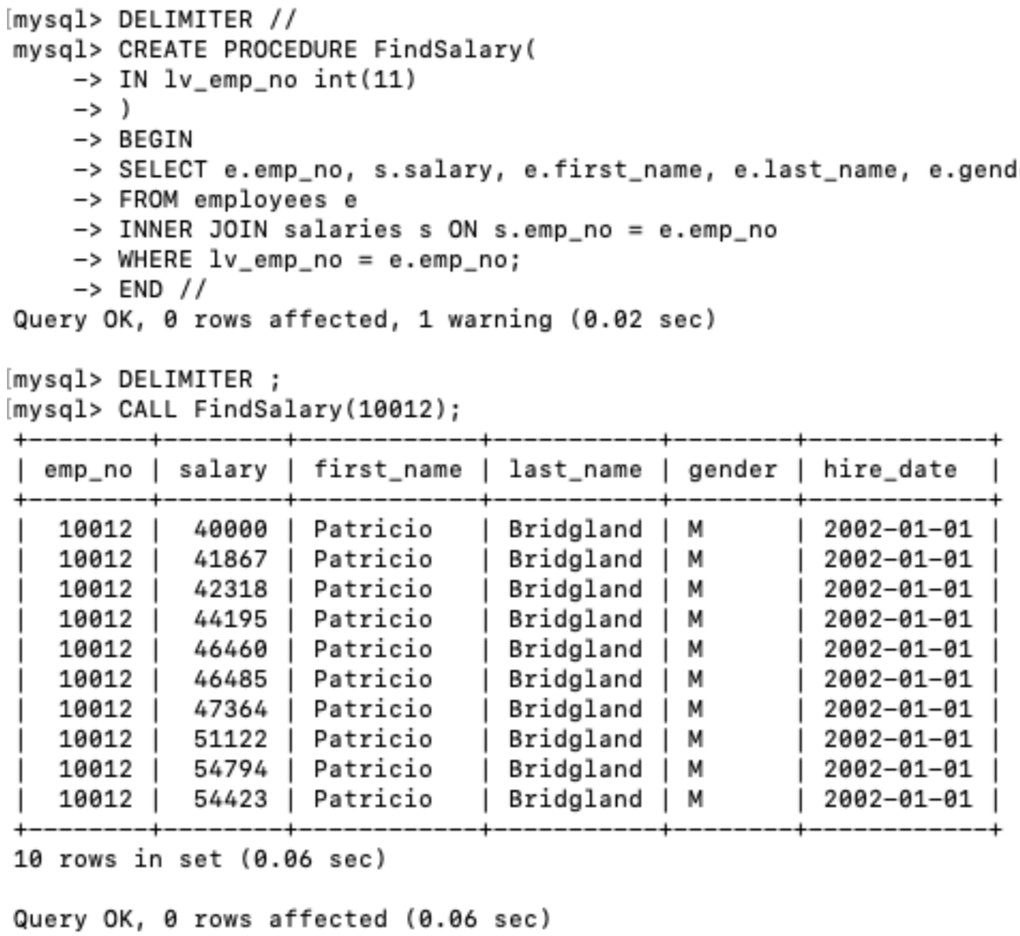
Write a description of what each stored procedure does and how to use it.

Procedures should use constructs you learned about from your research assignment and be more than just queries.

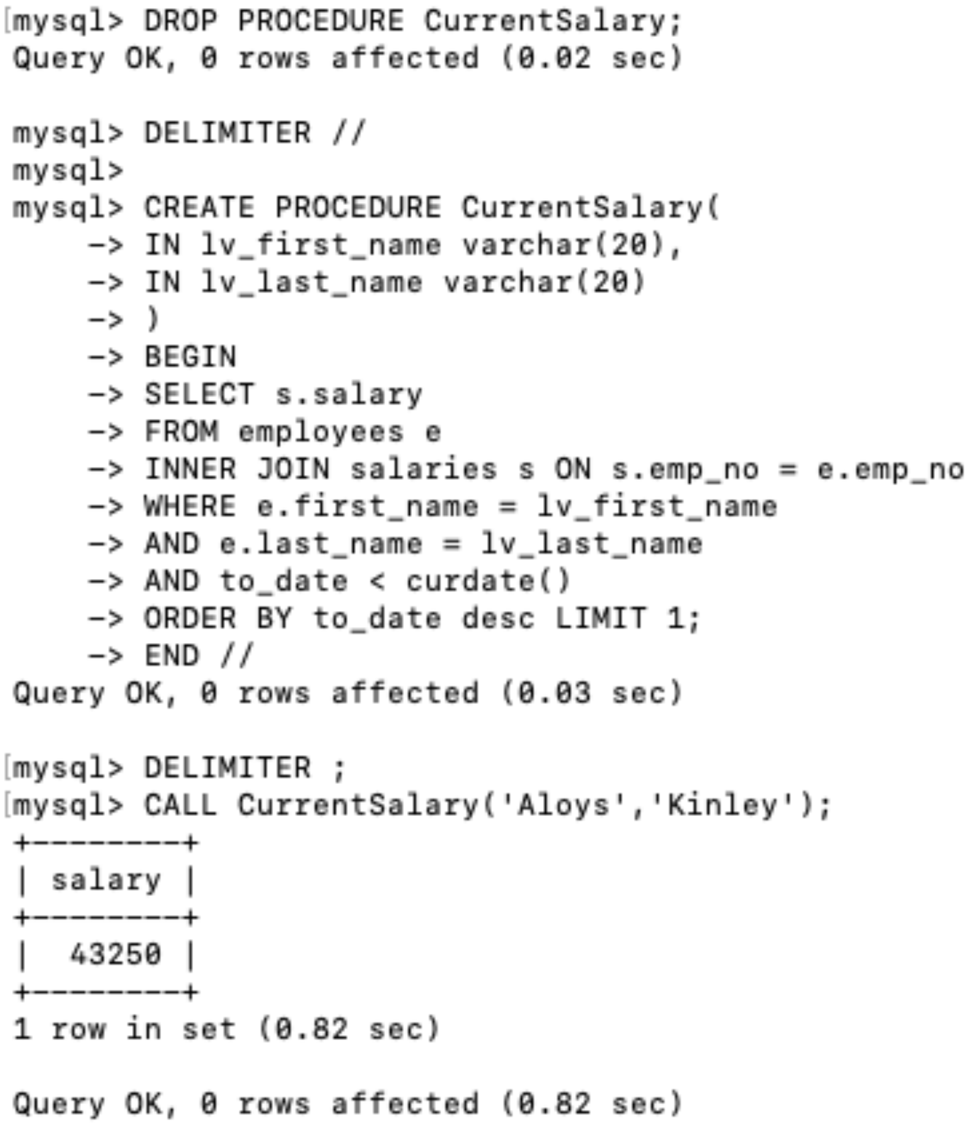
Screenshots:



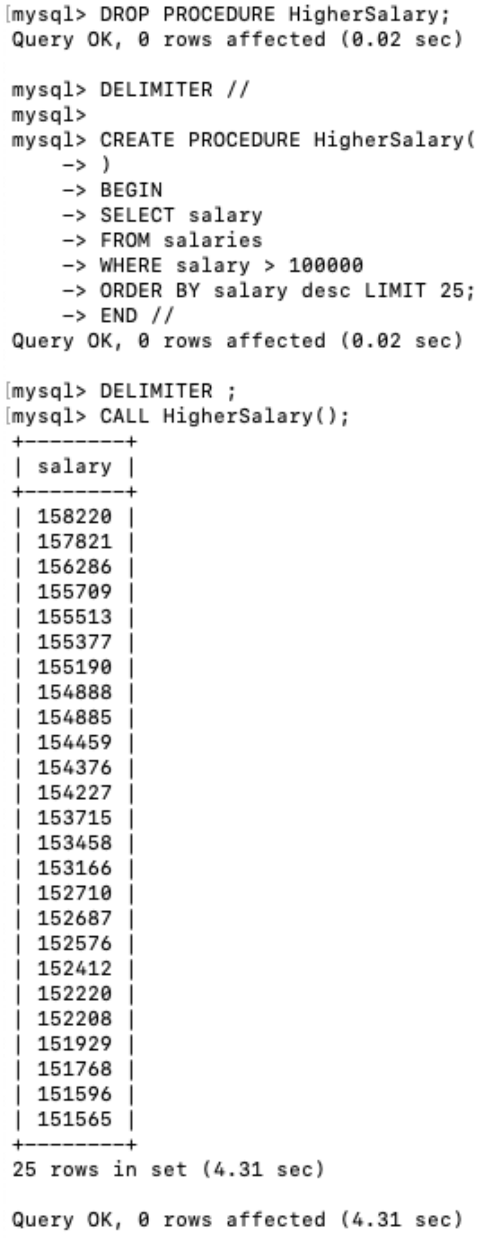
```
mysql> use employees;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+----------------------+
| Tables_in_employees  |
+----------------------+
| current_dept_emp     |
| departments          |
| dept_emp             |
| dept_emp_latest_date |
| dept_manager         |
| employee_desc        |
| employee_details     |
| employee_info        |
| employee_information |
| employees            |
| salaries             |
| titles               |
+----------------------+
12 rows in set (0.02 sec)

mysql> DROP PROCEDURE employee;
Query OK, 0 rows affected (0.18 sec)

mysql> DELIMITER //
mysql> CREATE PROCEDURE employee(IN employees int)
    -> BEGIN
    -> SELECT * FROM employees LIMIT employees;
    -> END //
Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER;
    -> CALL employee(5);
    -> //
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 'DELIMITER;CALL employee(5)' at line 1
mysql> CALL employee(5);
    -> //
+--------+------------+------------+-----------+--------+------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  |
+--------+------------+------------+-----------+--------+------------+
|  10008 | 1958-02-19 | Saniya     | Kalloufi  | M      | 1994-09-15 |
|  10009 | 1952-04-19 | Sumant     | Peac      | F      | 2002-01-01 |
|  10010 | 1963-06-01 | Duangkaew  | Piveteau  | F      | 2002-01-01 |
|  10011 | 1953-11-07 | Mary       | Sluis     | F      | 1990-01-22 |
|  10012 | 1960-10-04 | Patricio   | Bridgland | M      | 2002-01-01 |
+--------+------------+------------+-----------+--------+------------+
5 rows in set (0.05 sec)

Query OK, 0 rows affected (0.05 sec)
```



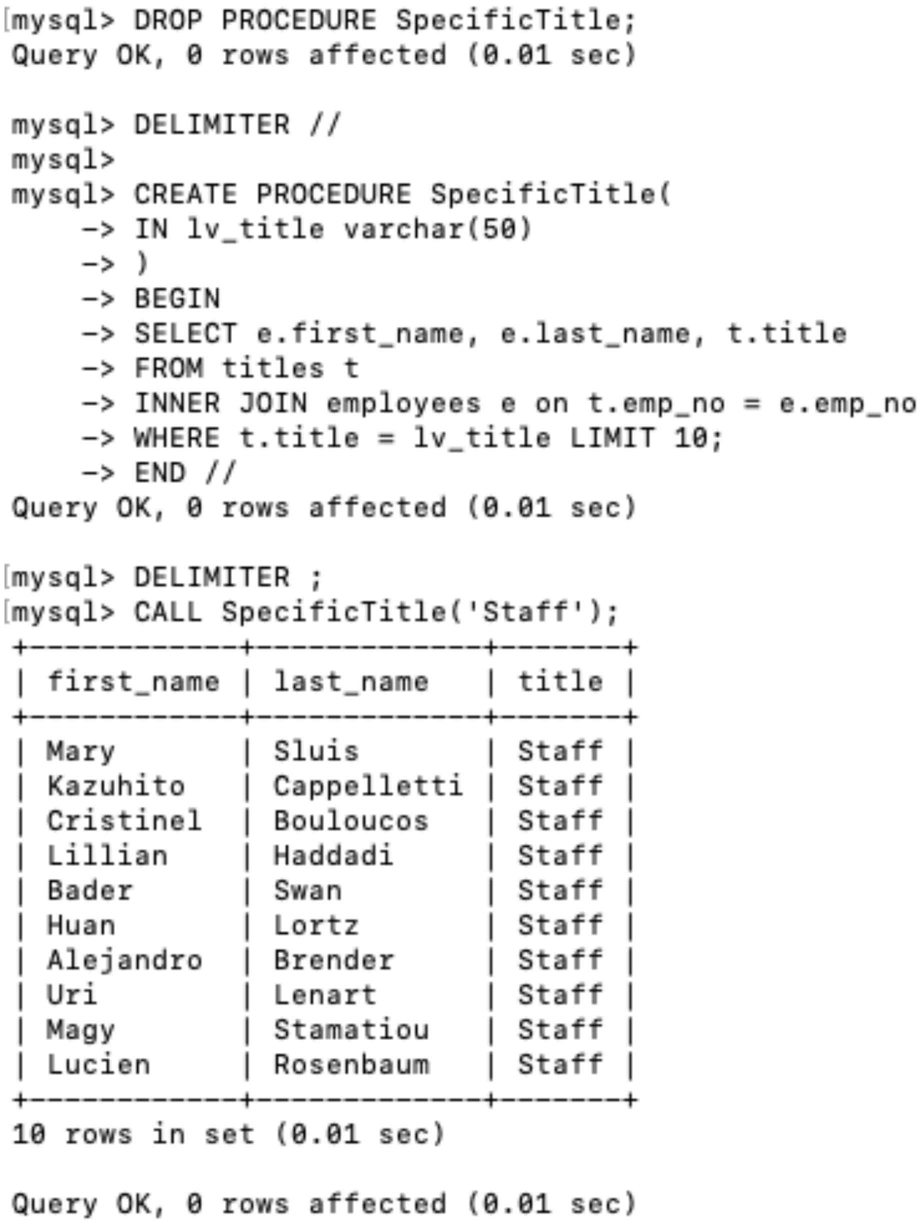
```
[mysql> DELIMITER //
 mysql> CREATE PROCEDURE FindSalary(
     -> IN lv_emp_no int(11)
     -> )
     -> BEGIN
     -> SELECT e.emp_no, s.salary, e.first_name, e.last_name, e.gender, e.hire_date
     -> FROM employees e
     -> INNER JOIN salaries s ON s.emp_no = e.emp_no
     -> WHERE lv_emp_no = e.emp_no;
     -> END //
 Query OK, 0 rows affected, 1 warning (0.02 sec)

[mysql> DELIMITER ;
[mysql> CALL FindSalary(10012);
 +--------+--------+------------+-----------+--------+------------+
 | emp_no | salary | first_name | last_name | gender | hire_date  |
 +--------+--------+------------+-----------+--------+------------+
 |  10012 |  40000 | Patricio   | Bridgland | M      | 2002-01-01 |
 |  10012 |  41867 | Patricio   | Bridgland | M      | 2002-01-01 |
 |  10012 |  42318 | Patricio   | Bridgland | M      | 2002-01-01 |
 |  10012 |  44195 | Patricio   | Bridgland | M      | 2002-01-01 |
 |  10012 |  46460 | Patricio   | Bridgland | M      | 2002-01-01 |
 |  10012 |  46485 | Patricio   | Bridgland | M      | 2002-01-01 |
 |  10012 |  47364 | Patricio   | Bridgland | M      | 2002-01-01 |
 |  10012 |  51122 | Patricio   | Bridgland | M      | 2002-01-01 |
 |  10012 |  54794 | Patricio   | Bridgland | M      | 2002-01-01 |
 |  10012 |  54423 | Patricio   | Bridgland | M      | 2002-01-01 |
 +--------+--------+------------+-----------+--------+------------+
 10 rows in set (0.06 sec)

 Query OK, 0 rows affected (0.06 sec)
```



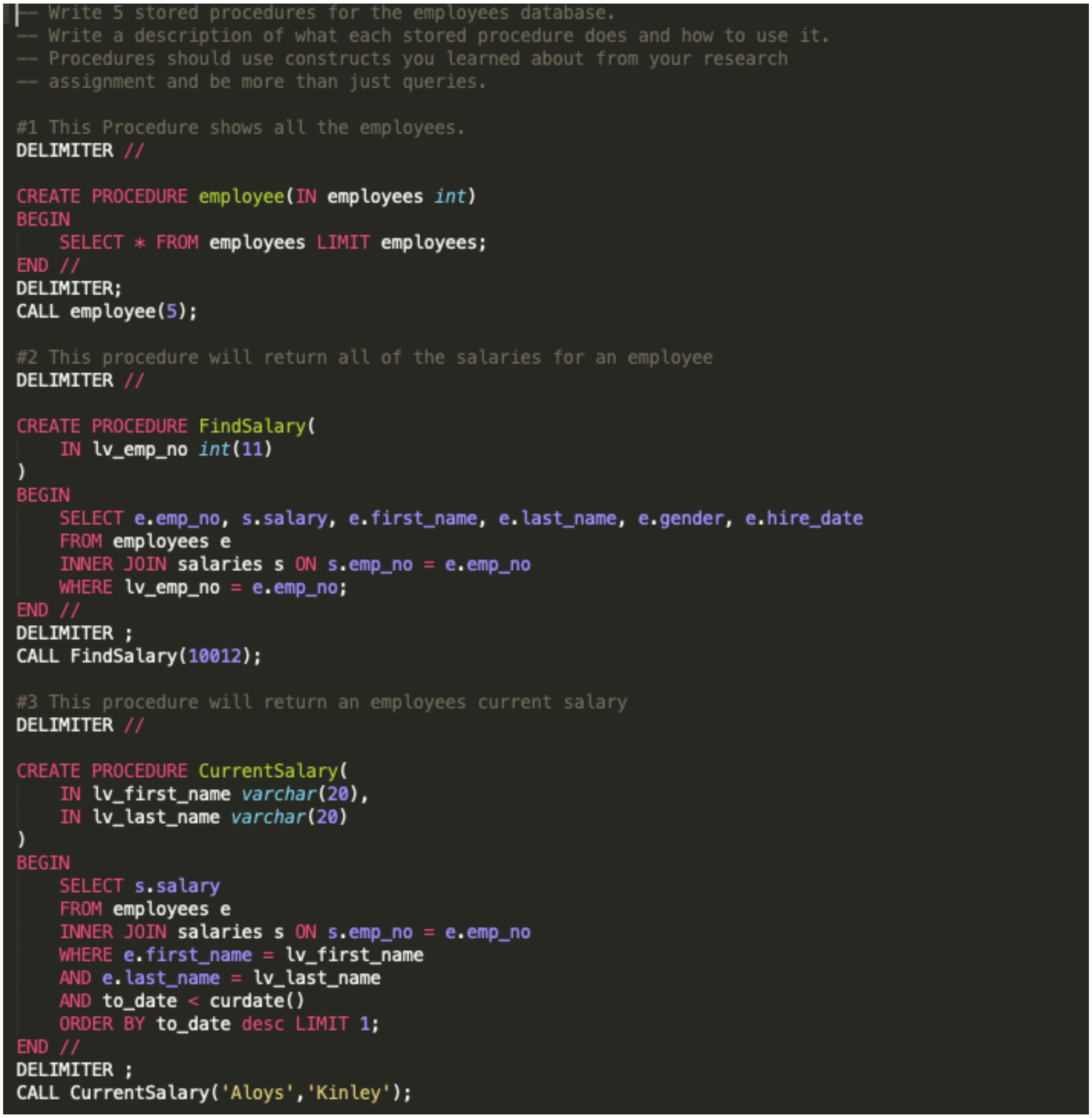
```
[mysql> DROP PROCEDURE CurrentSalary;
Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE CurrentSalary(
    -> IN lv_first_name varchar(20),
    -> IN lv_last_name varchar(20)
    -> )
    -> BEGIN
    -> SELECT s.salary
    -> FROM employees e
    -> INNER JOIN salaries s ON s.emp_no = e.emp_no
    -> WHERE e.first_name = lv_first_name
    -> AND e.last_name = lv_last_name
    -> AND to_date < curdate()
    -> ORDER BY to_date desc LIMIT 1;
    -> END //
Query OK, 0 rows affected (0.03 sec)

[mysql> DELIMITER ;
[mysql> CALL CurrentSalary('Aloys','Kinley');
+--------+
| salary |
+--------+
|  43250 |
+--------+
1 row in set (0.82 sec)

Query OK, 0 rows affected (0.82 sec)
```



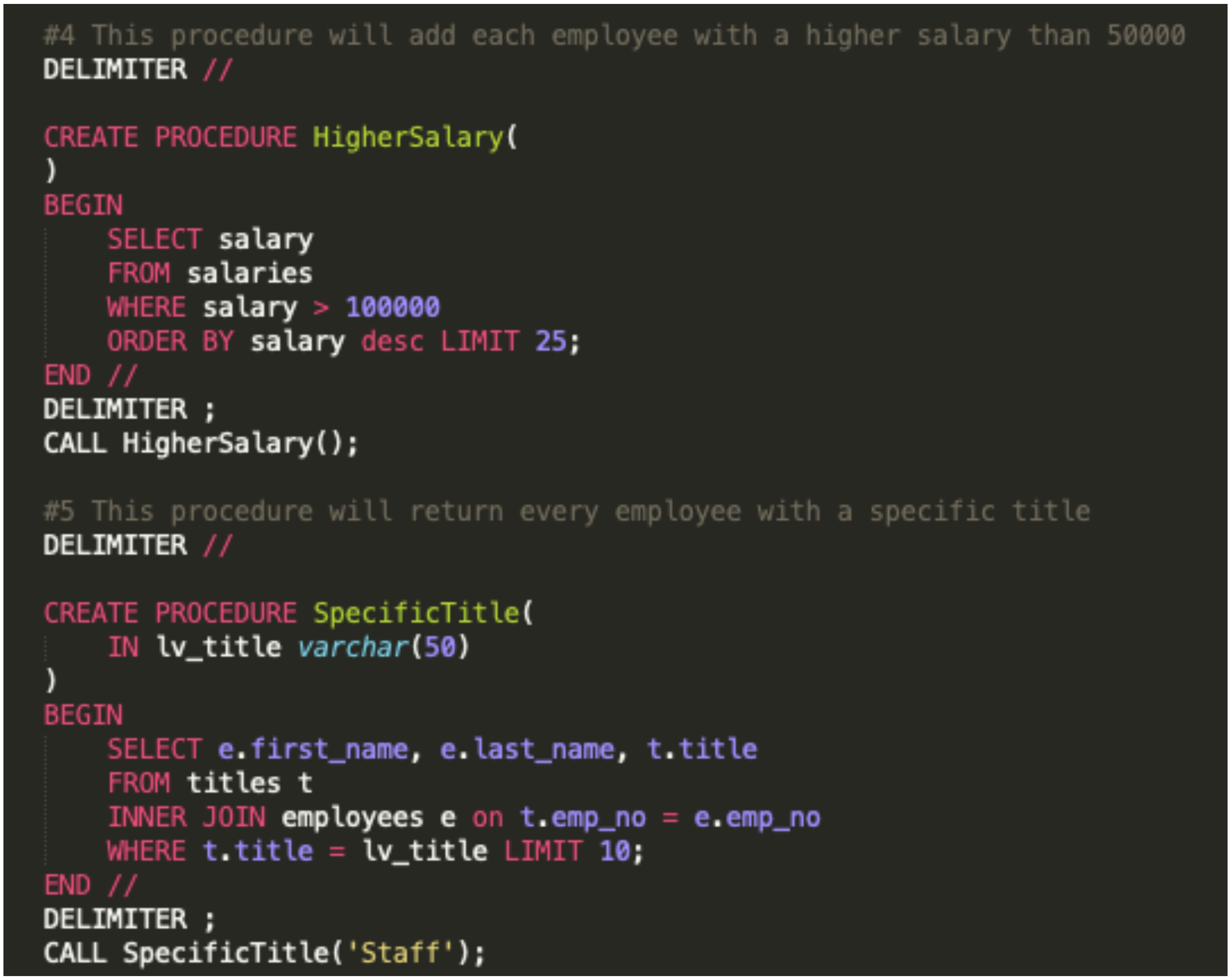
```
[mysql> DROP PROCEDURE HigherSalary;
 Query OK, 0 rows affected (0.02 sec)

 mysql> DELIMITER //
 mysql>
 mysql> CREATE PROCEDURE HigherSalary(
     -> )
     -> BEGIN
     -> SELECT salary
     -> FROM salaries
     -> WHERE salary > 100000
     -> ORDER BY salary desc LIMIT 25;
     -> END //
 Query OK, 0 rows affected (0.02 sec)

[mysql> DELIMITER ;
[mysql> CALL HigherSalary();
 +--------+
 | salary |
 +--------+
 | 158220 |
 | 157821 |
 | 156286 |
 | 155709 |
 | 155513 |
 | 155377 |
 | 155190 |
 | 154888 |
 | 154885 |
 | 154459 |
 | 154376 |
 | 154227 |
 | 153715 |
 | 153458 |
 | 153166 |
 | 152710 |
 | 152687 |
 | 152576 |
 | 152412 |
 | 152220 |
 | 152208 |
 | 151929 |
 | 151768 |
 | 151596 |
 | 151565 |
 +--------+
 25 rows in set (4.31 sec)

 Query OK, 0 rows affected (4.31 sec)
```



```
[mysql> DROP PROCEDURE SpecificTitle;
 Query OK, 0 rows affected (0.01 sec)

 mysql> DELIMITER //
 mysql>
 mysql> CREATE PROCEDURE SpecificTitle(
     -> IN lv_title varchar(50)
     -> )
     -> BEGIN
     -> SELECT e.first_name, e.last_name, t.title
     -> FROM titles t
     -> INNER JOIN employees e on t.emp_no = e.emp_no
     -> WHERE t.title = lv_title LIMIT 10;
     -> END //
 Query OK, 0 rows affected (0.01 sec)

[mysql> DELIMITER ;
[mysql> CALL SpecificTitle('Staff');
 +------------+-------------+-------+
 | first_name | last_name   | title |
 +------------+-------------+-------+
 | Mary       | Sluis       | Staff |
 | Kazuhito   | Cappelletti | Staff |
 | Cristinel  | Bouloucos   | Staff |
 | Lillian    | Haddadi     | Staff |
 | Bader      | Swan        | Staff |
 | Huan       | Lortz       | Staff |
 | Alejandro  | Brender     | Staff |
 | Uri        | Lenart      | Staff |
 | Magy       | Stamatiou   | Staff |
 | Lucien     | Rosenbaum   | Staff |
 +------------+-------------+-------+
 10 rows in set (0.01 sec)

 Query OK, 0 rows affected (0.01 sec)
```



```
-- Write 5 stored procedures for the employees database.
-- Write a description of what each stored procedure does and how to use it.
-- Procedures should use constructs you learned about from your research
-- assignment and be more than just queries.

#1 This Procedure shows all the employees.
DELIMITER //

CREATE PROCEDURE employee(IN employees int)
BEGIN
    SELECT * FROM employees LIMIT employees;
END //
DELIMITER;
CALL employee(5);

#2 This procedure will return all of the salaries for an employee
DELIMITER //

CREATE PROCEDURE FindSalary(
    IN lv_emp_no int(11)
)
BEGIN
    SELECT e.emp_no, s.salary, e.first_name, e.last_name, e.gender, e.hire_date
    FROM employees e
    INNER JOIN salaries s ON s.emp_no = e.emp_no
    WHERE lv_emp_no = e.emp_no;
END //
DELIMITER ;
CALL FindSalary(10012);

#3 This procedure will return an employees current salary
DELIMITER //

CREATE PROCEDURE CurrentSalary(
    IN lv_first_name varchar(20),
    IN lv_last_name varchar(20)
)
BEGIN
    SELECT s.salary
    FROM employees e
    INNER JOIN salaries s ON s.emp_no = e.emp_no
    WHERE e.first_name = lv_first_name
    AND e.last_name = lv_last_name
    AND to_date < curdate()
    ORDER BY to_date desc LIMIT 1;
END //
DELIMITER ;
CALL CurrentSalary('Aloys','Kinley');
```



```
#4 This procedure will add each employee with a higher salary than 50000
DELIMITER //

CREATE PROCEDURE HigherSalary(
)
BEGIN
    SELECT salary
    FROM salaries
    WHERE salary > 100000
    ORDER BY salary desc LIMIT 25;
END //
DELIMITER ;
CALL HigherSalary();

#5 This procedure will return every employee with a specific title
DELIMITER //

CREATE PROCEDURE SpecificTitle(
    IN lv_title varchar(50)
)
BEGIN
    SELECT e.first_name, e.last_name, t.title
    FROM titles t
    INNER JOIN employees e on t.emp_no = e.emp_no
    WHERE t.title = lv_title LIMIT 10;
END //
DELIMITER ;
CALL SpecificTitle('Staff');
```

URL to GitHub Repository: https://github.com/DesmondYo/MySQLWeek4