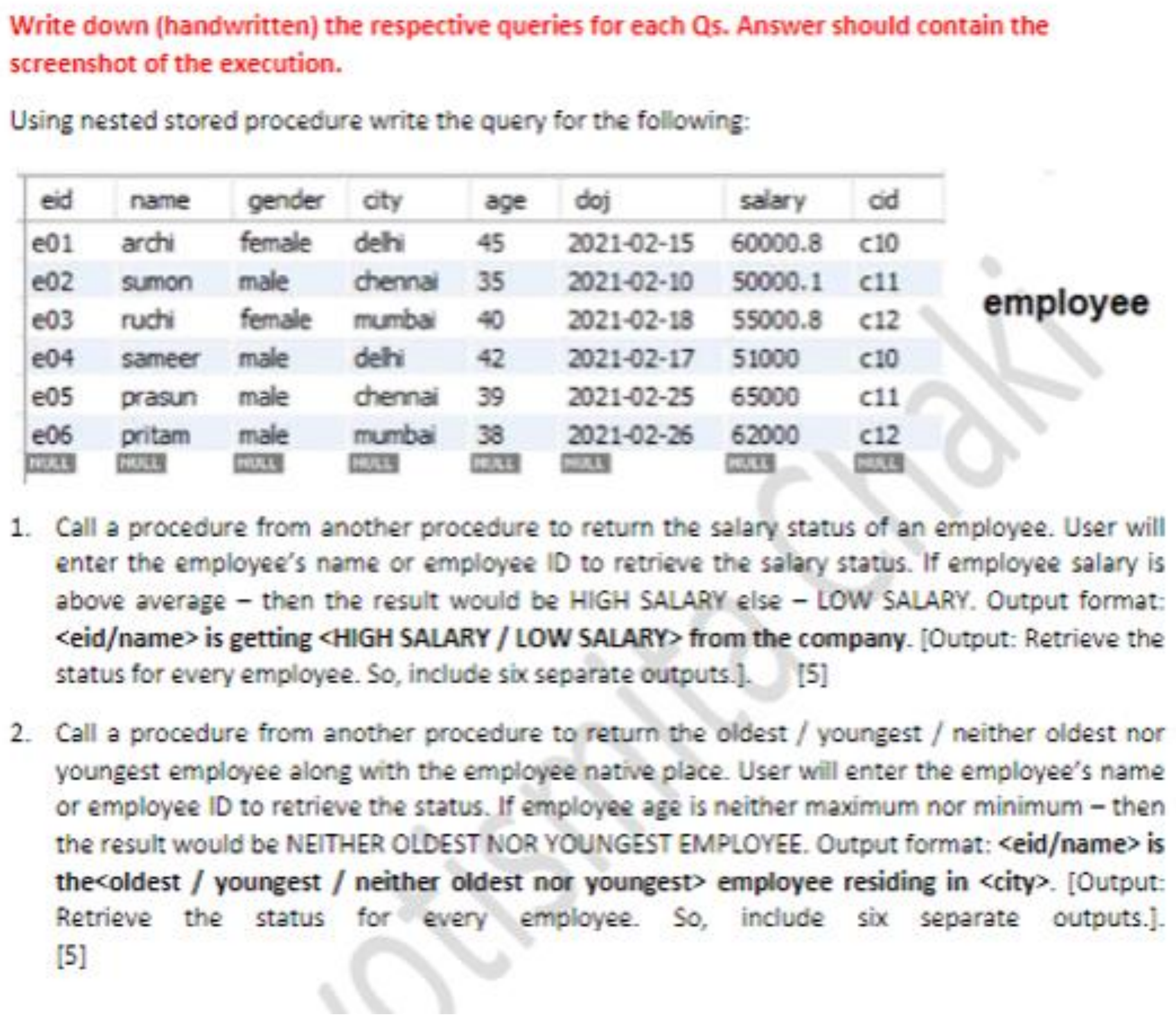
Course: CSE 2004

DBMS

LAB ASSIGNMENT 5

NAME: PRITHAK GAJUREL

REGISTRATION NUMBER: 20BCE2921


Write down (handwritten) the respective queries for each Qs. Answer should contain the screenshot of the execution.

Using nested stored procedure write the query for the following:

employee

eid	name	gender	city	age	doj	salary	cid
e01	archi	female	delhi	45	2021-02-15	60000.8	c10
e02	sumon	male	chennai	35	2021-02-10	50000.1	c11
e03	ruchi	female	mumbai	40	2021-02-18	55000.8	c12
e04	sameer	male	delhi	42	2021-02-17	51000	c10
e05	prasun	male	chennai	39	2021-02-25	65000	c11
e06	pritam	male	mumbai	38	2021-02-26	62000	c12
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

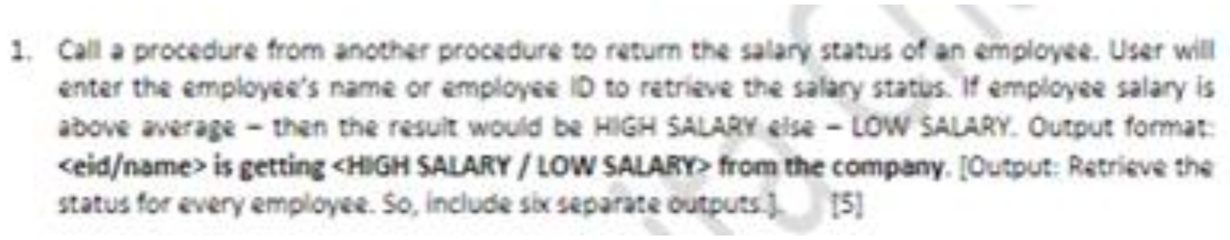
1. Call a procedure from another procedure to return the salary status of an employee. User will enter the employee's name or employee ID to retrieve the salary status. If employee salary is above average – then the result would be HIGH SALARY else – LOW SALARY. Output format: **<eid/name> is getting <HIGH SALARY / LOW SALARY> from the company**. [Output: Retrieve the status for every employee. So, include six separate outputs.]. [5]

2. Call a procedure from another procedure to return the oldest / youngest / neither oldest nor youngest employee along with the employee native place. User will enter the employee's name or employee ID to retrieve the status. If employee age is neither maximum nor minimum – then the result would be NEITHER OLDEST NOR YOUNGEST EMPLOYEE. Output format: **<eid/name> is the<oldest / youngest / neither oldest nor youngest> employee residing in <city>**. [Output: Retrieve the status for every employee. So, include six separate outputs.]. [5]

Creation of table:

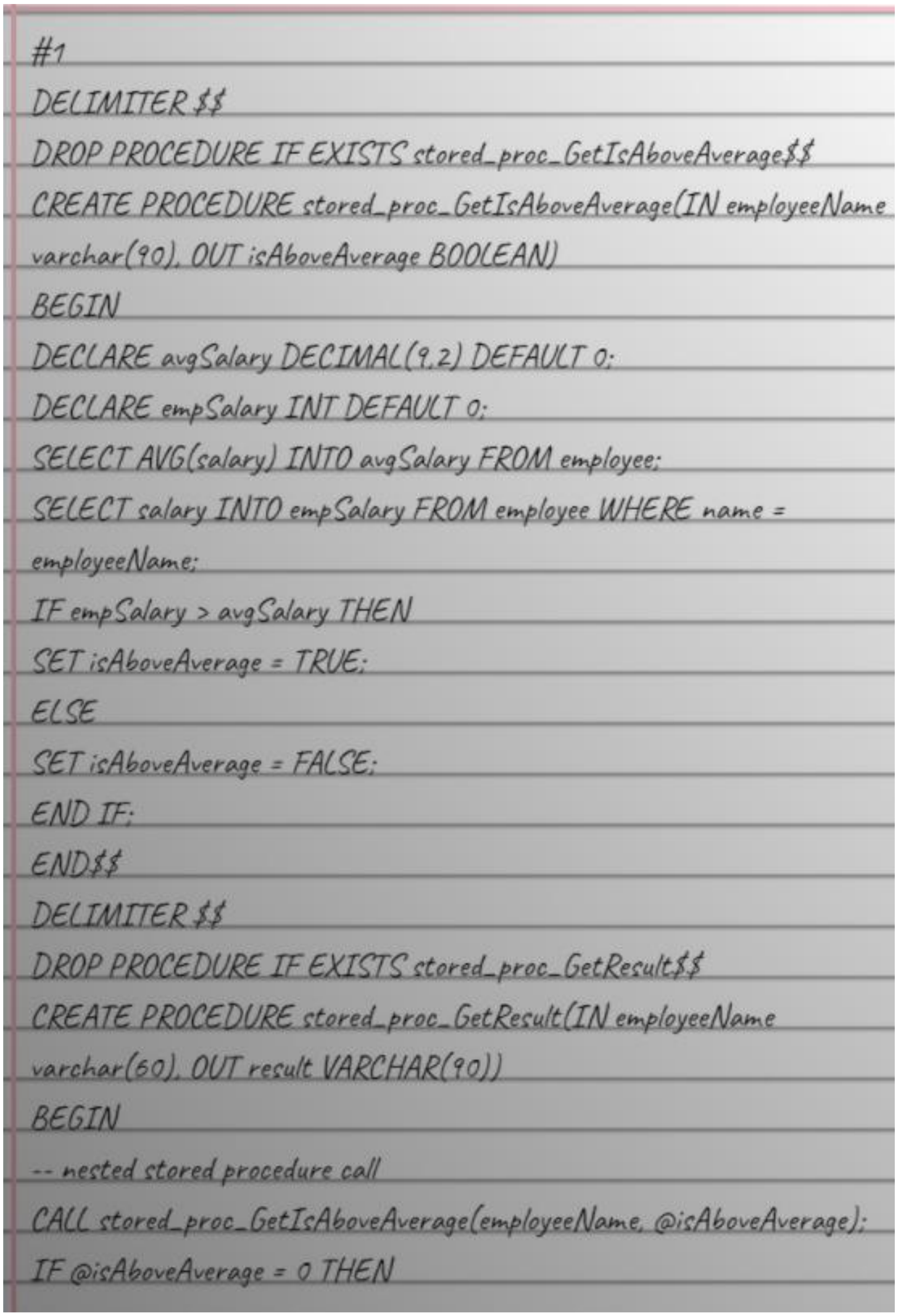
```
create table employee
(
eid varchar(200) primary key,
name varchar(200) not null,
gender varchar(200),
city varchar(200),
age int,
doj date,
salary float ,
cid varchar(200)
);
```

```sql
insert into employee
values('e01','archi','female','delhi','45','2021-02-15',60000.8,'c10');
insert into employee
values('e02','sumon','male','chennai','35','2021-02-10',50000.1,'c11');
insert into employee
values('e03','ruchi','female','mumbai','40','2021-02-18',55000.8,'c12');
insert into employee
values('e04','sameer','male','delhi','42','2021-02-17',51000,'c10');
insert into employee
values('e05','prasun','male','chennai','39','2021-02-25',65000,'c11');
insert into employee
values('e06','pritam','male','mumbai','38','2021-02-26',62000,'c12');
select * from employee;
```

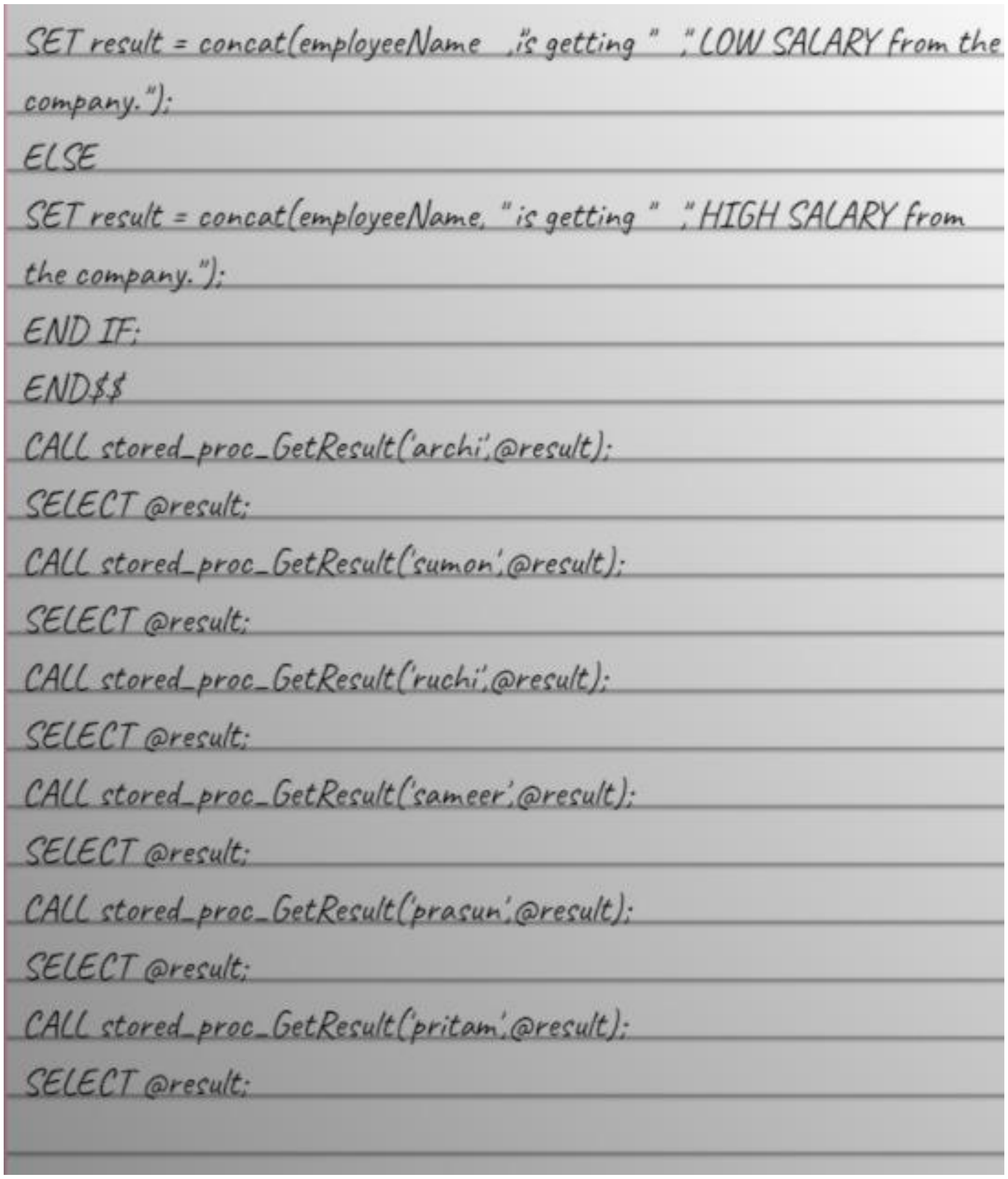


1. Call a procedure from another procedure to return the salary status of an employee. User will enter the employee's name or employee ID to retrieve the salary status. If employee salary is above average – then the result would be HIGH SALARY else – LOW SALARY. Output format: **<eid/name> is getting <HIGH SALARY / LOW SALARY> from the company**. [Output: Retrieve the status for every employee. So, include six separate outputs.] [5]

Handwritten code:

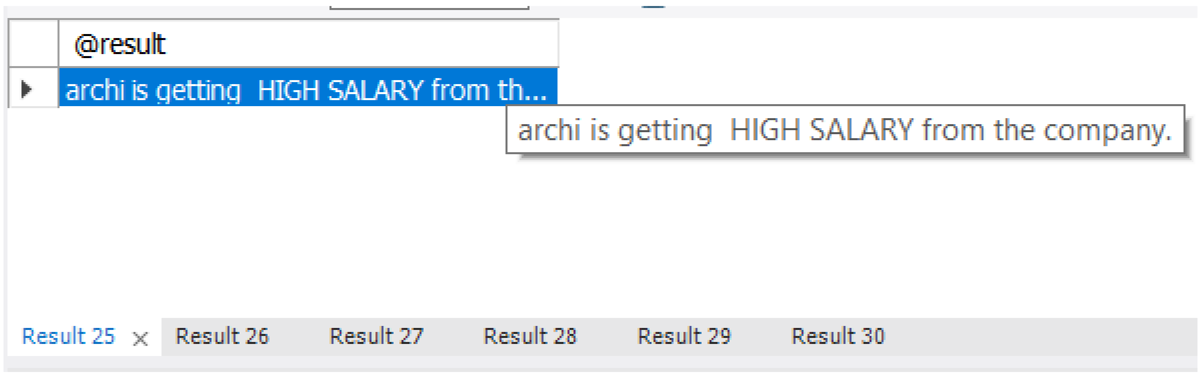


```
#1
DELIMITER $$
DROP PROCEDURE IF EXISTS stored_proc_GetIsAboveAverage$$
CREATE PROCEDURE stored_proc_GetIsAboveAverage(IN employeeName
varchar(90), OUT isAboveAverage BOOLEAN)
BEGIN
DECLARE avgSalary DECIMAL(9,2) DEFAULT 0;
DECLARE empSalary INT DEFAULT 0;
SELECT AVG(salary) INTO avgSalary FROM employee;
SELECT salary INTO empSalary FROM employee WHERE name =
employeeName;
IF empSalary > avgSalary THEN
SET isAboveAverage = TRUE;
ELSE
SET isAboveAverage = FALSE;
END IF;
END$$
DELIMITER $$
DROP PROCEDURE IF EXISTS stored_proc_GetResult$$
CREATE PROCEDURE stored_proc_GetResult(IN employeeName
varchar(60), OUT result VARCHAR(90))
BEGIN
-- nested stored procedure call
CALL stored_proc_GetIsAboveAverage(employeeName, @isAboveAverage);
IF @isAboveAverage = 0 THEN
```

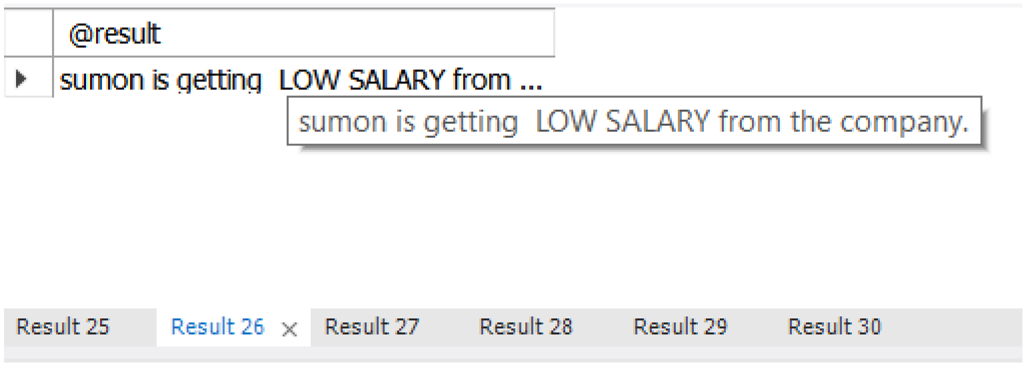


```
SET result = concat(employeeName ,"is getting " ,"LOW SALARY from the
company.");
ELSE
SET result = concat(employeeName, "is getting " ,"HIGH SALARY from
the company.");
END IF;
END$$
CALL stored_proc_GetResult('archi',@result);
SELECT @result;
CALL stored_proc_GetResult('sumon',@result);
SELECT @result;
CALL stored_proc_GetResult('ruchi',@result);
SELECT @result;
CALL stored_proc_GetResult('sameer',@result);
SELECT @result;
CALL stored_proc_GetResult('prasun',@result);
SELECT @result;
CALL stored_proc_GetResult('pritam',@result);
SELECT @result;
```

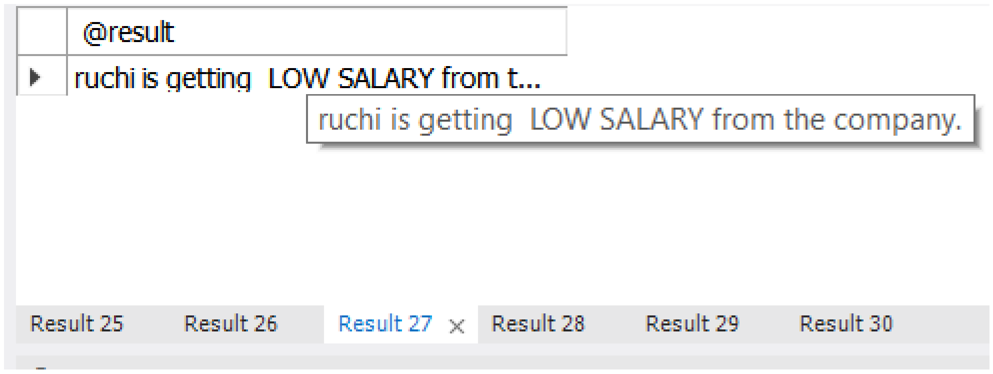
Output:



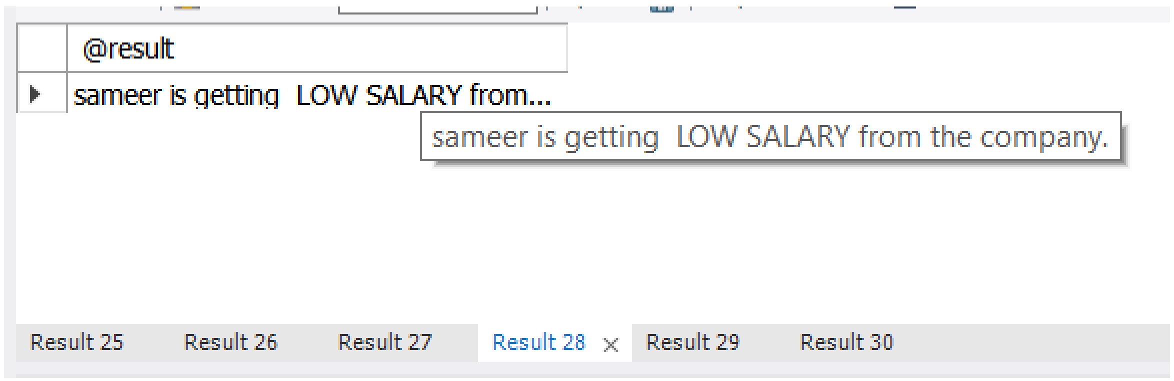
```
@result
archi is getting  HIGH SALARY from th...
archi is getting  HIGH SALARY from the company.
```



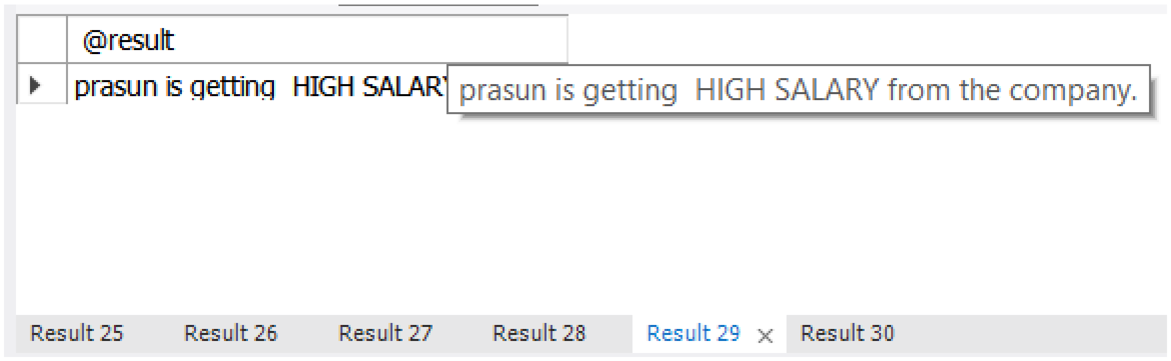
```
@result
sumon is getting  LOW SALARY from ...
sumon is getting  LOW SALARY from the company.
```



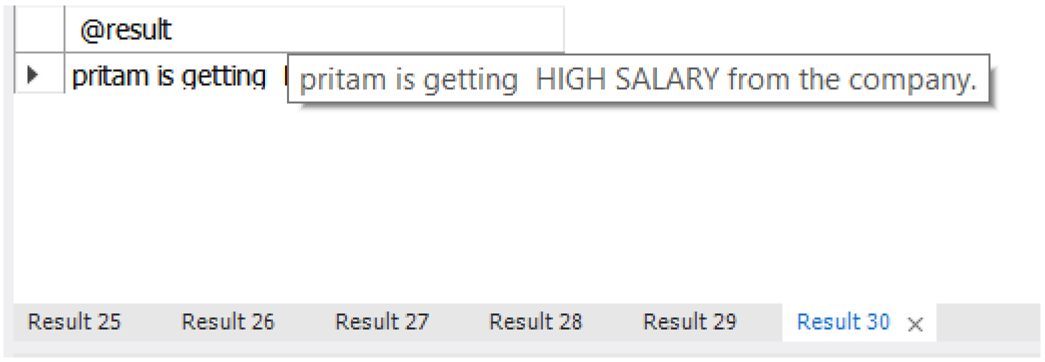
```
@result
ruchi is getting  LOW SALARY from t...
ruchi is getting  LOW SALARY from the company.
```



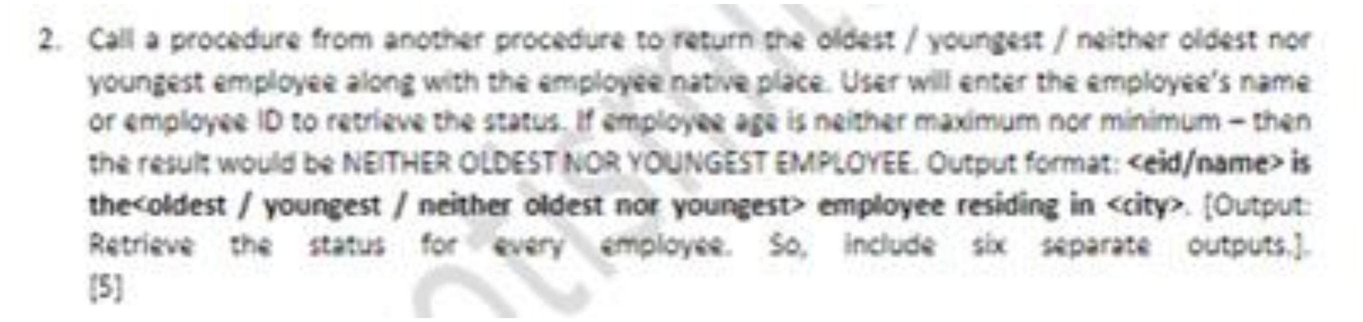
```
@result
sameer is getting  LOW SALARY from...
sameer is getting  LOW SALARY from the company.
```



```
@result
prasun is getting  HIGH SALAR
prasun is getting  HIGH SALARY from the company.
```

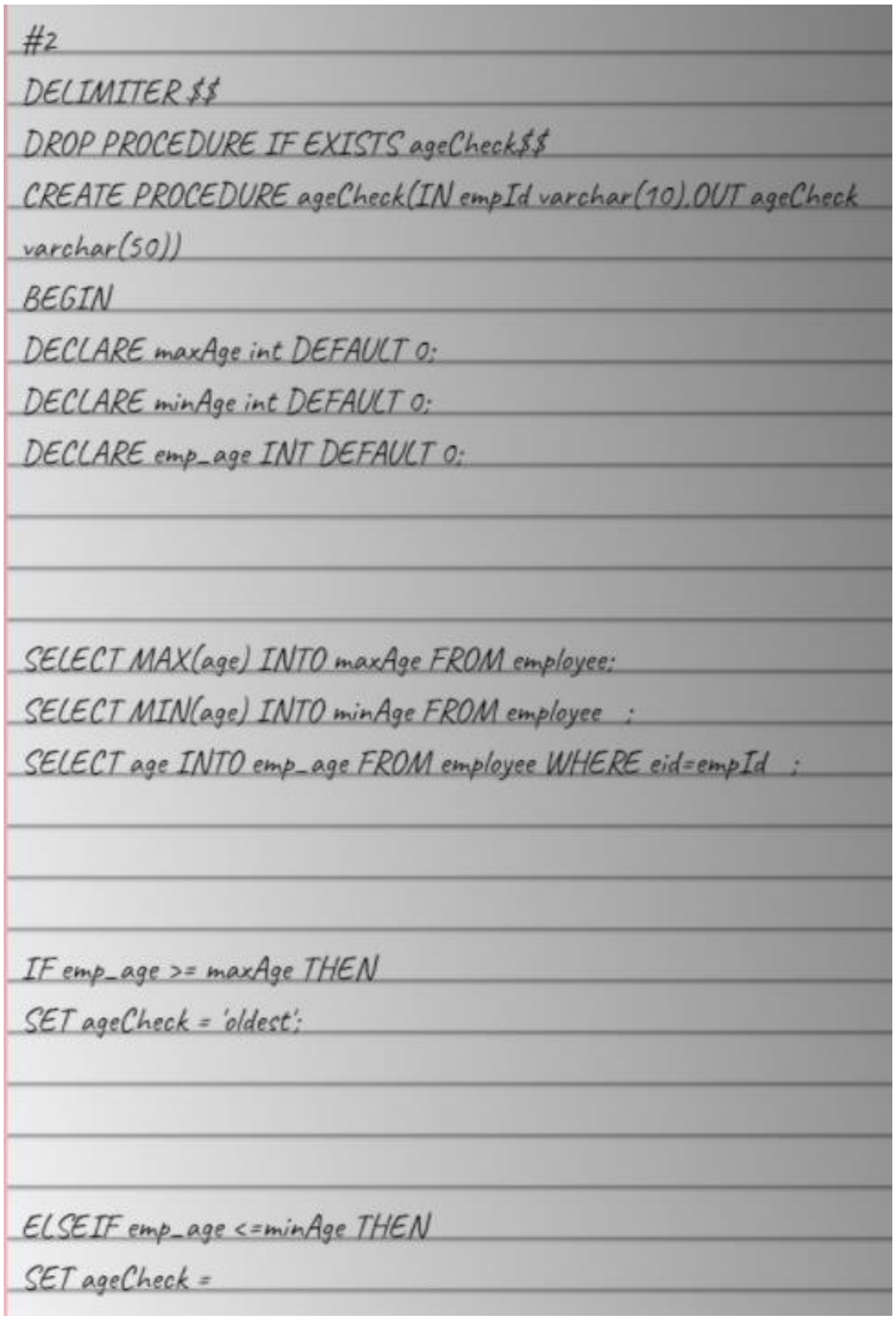


```
@result
pritam is getting  HIGH SALARY from the company.
```

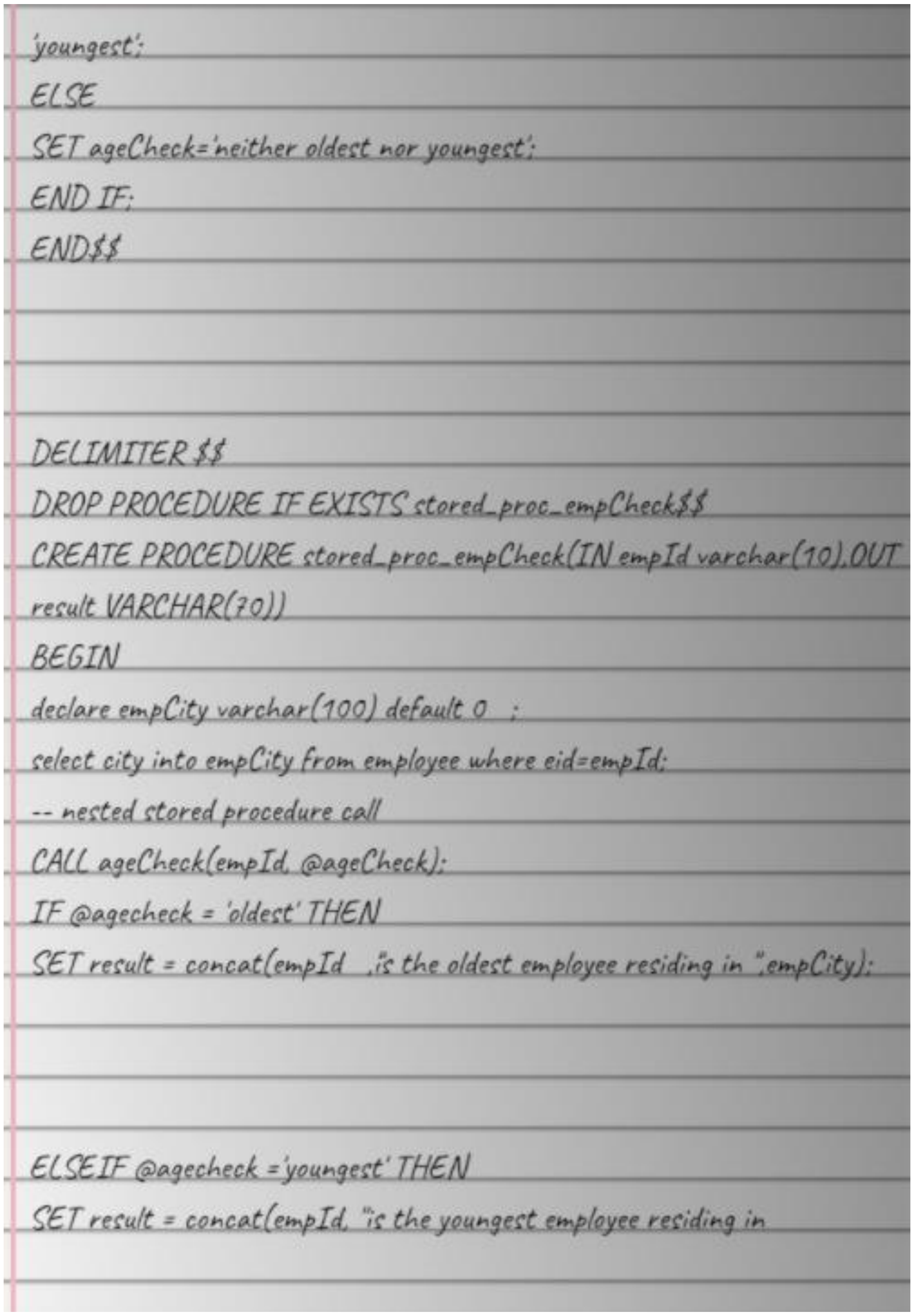


2. Call a procedure from another procedure to return the oldest / youngest / neither oldest nor youngest employee along with the employee native place. User will enter the employee's name or employee ID to retrieve the status. If employee age is neither maximum nor minimum – then the result would be NEITHER OLDEST NOR YOUNGEST EMPLOYEE. Output format: **<eid/name> is the<oldest / youngest / neither oldest nor youngest> employee residing in <city>**. [Output: Retrieve the status for every employee. So, include six separate outputs.]. [5]

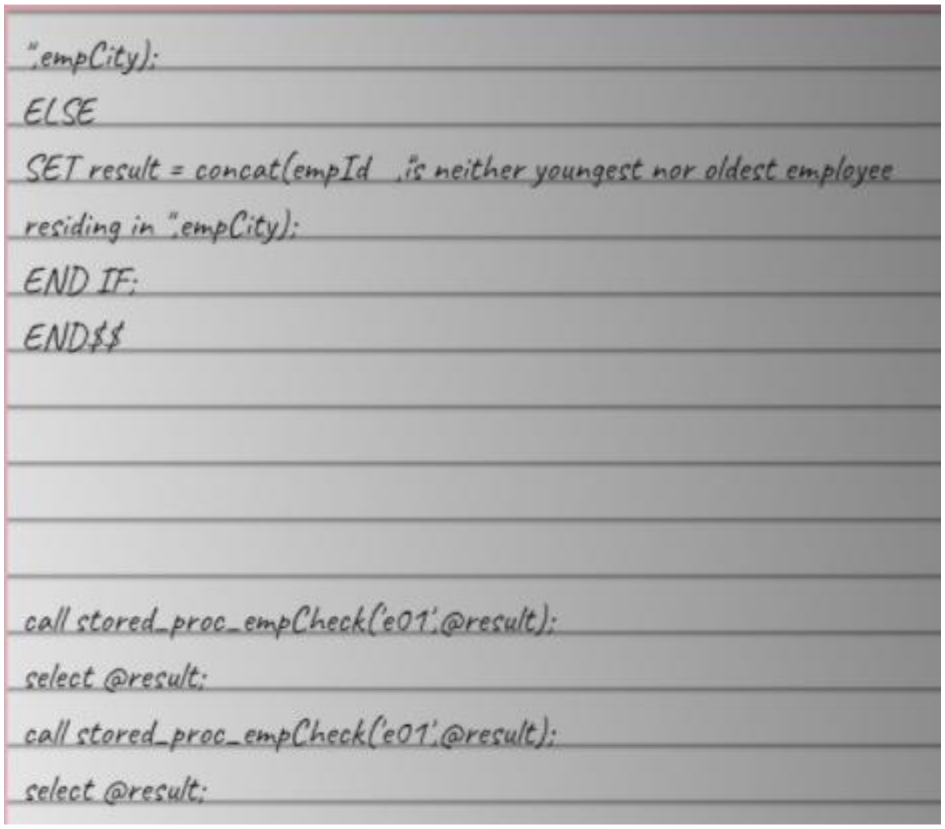
Handwritten code:



```
#2
DELIMITER $$
DROP PROCEDURE IF EXISTS ageCheck$$
CREATE PROCEDURE ageCheck(IN empId varchar(10),OUT ageCheck
varchar(50))
BEGIN
DECLARE maxAge int DEFAULT 0;
DECLARE minAge int DEFAULT 0;
DECLARE emp_age INT DEFAULT 0;



SELECT MAX(age) INTO maxAge FROM employee;
SELECT MIN(age) INTO minAge FROM employee   ;
SELECT age INTO emp_age FROM employee WHERE eid=empId   ;



IF emp_age >= maxAge THEN
SET ageCheck = 'oldest';



ELSEIF emp_age <=minAge THEN
SET ageCheck =
```

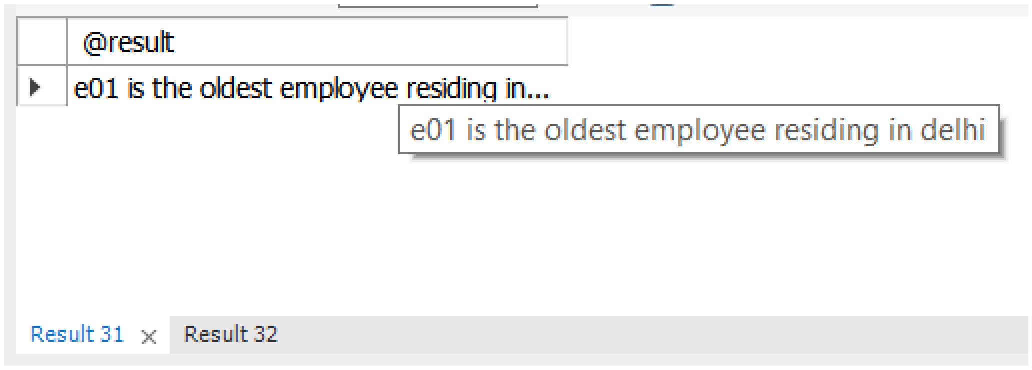


```
'youngest';
ELSE
SET ageCheck='neither oldest nor youngest';
END IF;
END$$



DELIMITER $$
DROP PROCEDURE IF EXISTS stored_proc_empCheck$$
CREATE PROCEDURE stored_proc_empCheck(IN empId varchar(10),OUT
result VARCHAR(70))
BEGIN
declare empCity varchar(100) default 0   ;
select city into empCity from employee where eid=empId;
-- nested stored procedure call
CALL ageCheck(empId, @ageCheck);
IF @agecheck = 'oldest' THEN
SET result = concat(empId   ,"is the oldest employee residing in ",empCity);



ELSEIF @agecheck ='youngest' THEN
SET result = concat(empId, "is the youngest employee residing in
```

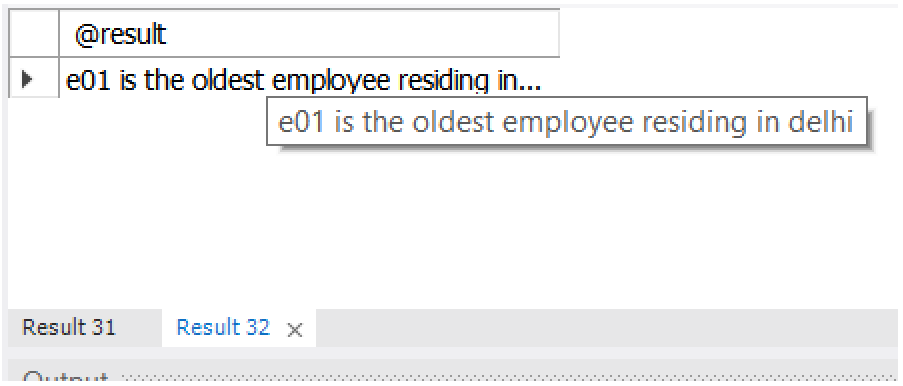


```
",empCity);
ELSE
SET result = concat(empId ,"is neither youngest nor oldest employee
residing in ",empCity);
END IF;
END$$




call stored_proc_empCheck('e01',@result);
select @result;
call stored_proc_empCheck('e01',@result);
select @result;
```

Output:



```
@result
e01 is the oldest employee residing in...
e01 is the oldest employee residing in delhi
```



```
@result
e01 is the oldest employee residing in...
e01 is the oldest employee residing in delhi
```