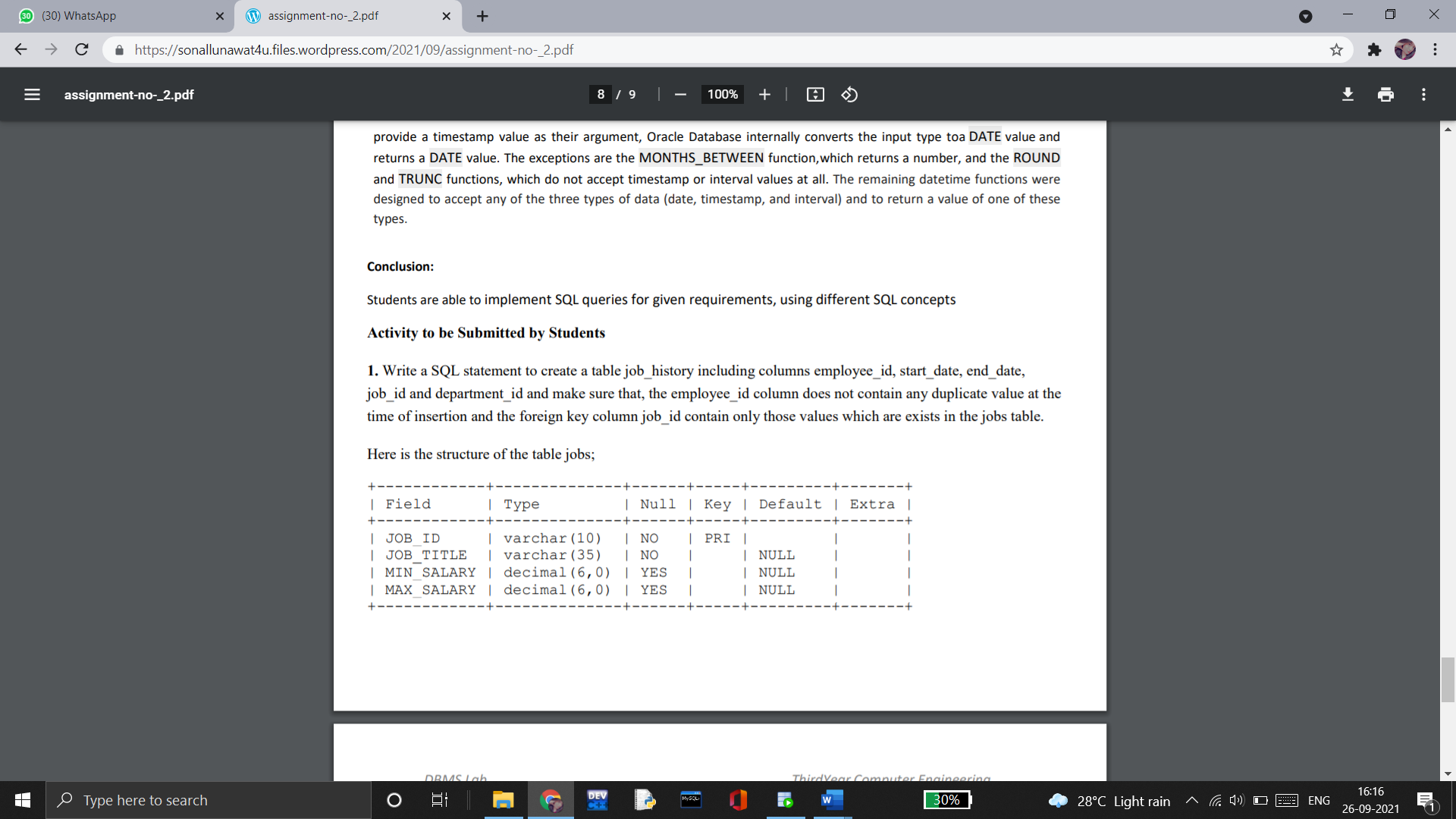
**Lab Assignment – 2**

**Name- Kaustubh Ulhas Patil**

**Roll No: - TECOMP-A68**

1. Write a SQL statement to create a table job\_history including columns employee\_id, start\_date, end\_date, job\_id and department\_id and make sure that, the employee\_id column does not contain any duplicate value at the time of insertion and the foreign key column job\_id contain only those values which are exists in the jobs table.

Here is the structure of the table jobs;



**Ans:**

CREATE TABLE IF NOT EXISTS jobs (

-> JOB\_ID varchar(10) NOT NULL UNIQUE,

-> JOB\_TITLE varchar(35) NOT NULL DEFAULT ' ',

-> MIN\_SALARY decimal(6,0) DEFAULT 8000,

-> MAX\_SALARY decimal(6,0) DEFAULT NULL

-> );

CREATE TABLE job\_history (

-> EMPLOYEE\_ID decimal(6,0) NOT NULL PRIMARY KEY,

-> START\_DATE date NOT NULL,

-> END\_DATE date NOT NULL,

-> JOB\_ID varchar(10) NOT NULL,

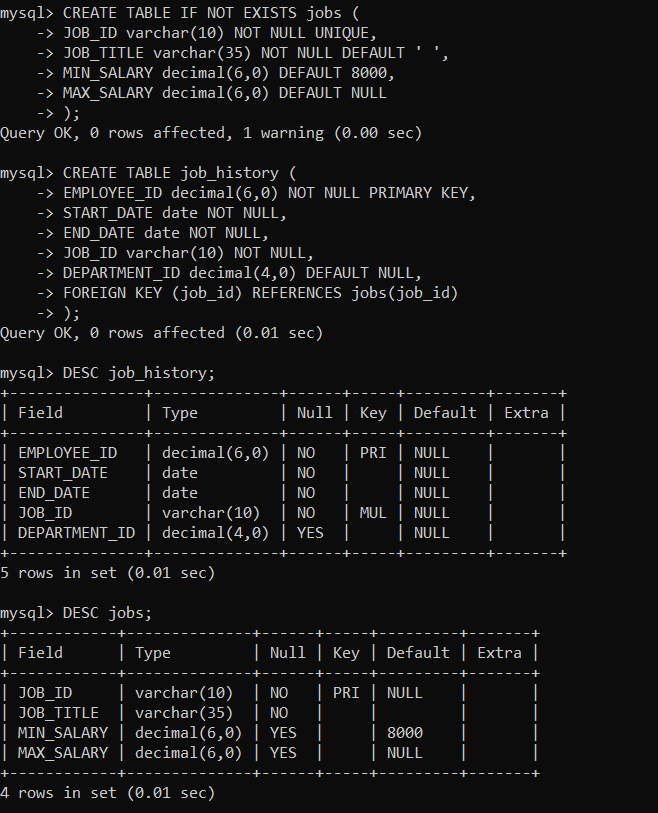
-> DEPARTMENT\_ID decimal(4,0) DEFAULT NULL,

-> FOREIGN KEY (job\_id) REFERENCES jobs(job\_id)

-> );

DESC job\_history;

**Output: -**



2. Write a SQL statement to create a table employees including columns employee\_id, first\_name, last\_name, job\_id, salary and make sure that, the employee\_id column does not contain any duplicate value at the time of insertion, and the foreign key column job\_id, referenced by the column job\_id of jobs table, can contain only those values which are exists in the jobs table. The InnoDB Engine have been used to create the tables. The specialty of the statement is that, The ON DELETE NO ACTION and the ON UPDATE NO ACTION actions will reject the deletion and any updates.

**Ans:**

CREATE TABLE IF NOT EXISTS employees (

-> EMPLOYEE\_ID decimal(6,0) NOT NULL PRIMARY KEY,

-> FIRST\_NAME varchar(20) DEFAULT NULL,

-> LAST\_NAME varchar(25) NOT NULL,

-> JOB\_ID INTEGER NOT NULL,

-> SALARY decimal(8,2) DEFAULT NULL,

-> FOREIGN KEY(JOB\_ID)

-> REFERENCES jobs(JOB\_ID)

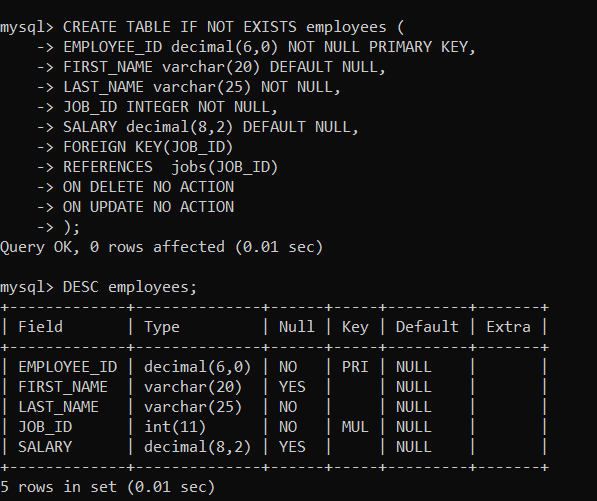
-> ON DELETE NO ACTION

-> ON UPDATE NO ACTION

-> );

DESC employees;

**Output: -**



3. Consider the following schema for a LibraryDatabase: BOOK (Book\_id, Title, Publisher\_Name, Pub\_Year) BOOK\_AUTHORS (Book\_id, Author\_Name) PUBLISHER (Name, Address, Phone) BOOK\_COPIES (Book\_id, Branch\_id, Noof\_Copies) BOOK\_LENDING (Book\_id, Branch\_id, Card\_No, Date\_Out, Due\_Date) LIBRARY\_BRANCH (Branch\_id, Branch\_Name, Address) Write SQL queries to

1. Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch,etc.

**Ans:**

SELECT B.BOOK\_ID, B.TITLE, B.PUBLISHER\_NAME, A.AUTHOR\_NAME C.NO\_OF\_COPIES, L.BRANCH\_ID

FROM BOOK B, BOOK\_AUTHOR A, BOOK\_COPIES C, LIBRARY\_BRANCH L WHERE B.BOOK\_ID = C.BOOK\_ID

AND L.BRANCH\_ID = C.BRANCH\_ID

2. Get the particulars of borrowers who have borrowed more than

**Ans:**

SELECT CARD\_NO FROM BOOK\_LEADING

WHERE DATE\_OUT BETWEEN ’01-JAN-2017’ AND

’01-JUL-2017’ GROUP BY CARD\_NO

HAVING COUNT (\*) >3;

3. Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.

**Ans:**

DELETE FROM BOOK WHERE BOOK\_ID= 3;

4. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.

**Ans:**

CREATE VIEW

V\_PUBLICATION AS

SELECT PUB\_YEAR

FROM BOOK;

5. Create a view of all books and its number of copies that are currently available in the Library.

**Ans:**

CREATE VIEW V\_BOOKS AS

SELECT B.BOOK\_ID, B.TITLE, C.NO\_OF\_COPIES

FROM BOOK B, BOOK\_COPIES C,

LIBRARY\_BRANCH L

WHERE B.BOOK\_ID = C.BOOK\_ID

**Lab Assignment – 3**

**Name: - Kaustubh Patil**

**Roll No: - TECOMP-A68**

**Q1.**

**Consider the schema for MovieDatabase:**

**ACTOR (Act\_id, Act\_Name, Act\_Gender)**

**DIRECTOR (Dir\_id, Dir\_Name, Dir\_Phone)**

**MOVIES (Mov\_id, Mov\_Title, Mov\_Year, Mov\_Lang, Dir\_id)**

**MOVIE\_CAST (Act\_id, Mov\_id, Role)**

**RATING (Mov\_id, Rev\_Stars)**

**Write SQL queries to**

**1. List the titles of all movies directed by‘Hitchcock’.**

**2. Find the movie names where one or more actors acted in two or moremovies.**

**3. List all actors who acted in a movie before 2000 and also in a DBMS Lab ThirdYear Computer Engineering movieafter 2015 (use JOINoperation).**

**4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.**

**5. Update rating of all movies directed by ‘Steven Spielberg’ to5**

**Ans –**

1. List the titles of all movies directed by ‘Hitchcock’.

SELECT MOV\_TITLE FROM MOVIES WHERE DIR\_ID = (SELECT DIR\_ID FROM DIRECTOR WHERE DIR\_NAME='HITCHCOCK');

1. Find the movie names where one or more actors acted in two or moremovies.

**Ans –**

SELECT MOV\_TITLE FROM MOVIES M, MOVIE\_CAST MC WHERE M.MOV\_ID=MC.MOV\_ID AND ACT\_ID IN (SELECT ACT\_ID FROM MOVIE\_CAST GROUP BY ACT\_ID HAVING COUNT(ACT\_ID)>1) GROUP BY MOV\_TITLE HAVING COUNT(\*)>1;

1. List all actors who acted in a movie before 2000 and also in a DBMS Lab ThirdYear Computer Engineering movie after 2015 (use JOIN operation).

**Ans –**

SELECT ACT\_NAME FROM ACTOR A JOIN MOVIE\_CAST C ON A.ACT\_ID=C.ACT\_ID JOIN MOVIES M ON C.MOV\_ID=M.MOV\_ID WHERE M.MOV\_YEAR NOT BETWEEN 2000 AND 2015;

1. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

**Ans –**

SELECT MOV\_TITLE,MAX(REV\_STARS)FROM MOVIES INNER JOIN RATING USING (MOV\_ID) GROUP BY MOV\_TITLE HAVING MAX(REV\_STARS)>0 ORDER BY MOV\_TITLE;

1. Update rating of all movies directed by ‘Steven Spielberg’ to 5

**Ans –**

UPDATE RATING SET REV\_STARS=5 WHERE MOV\_ID IN (SELECT MOV\_ID FROM MOVIES WHERE DIR\_ID IN (SELECT DIR\_ID FROM DIRECTOR WHERE DIR\_NAME='STEVEN SPIELBERG'));

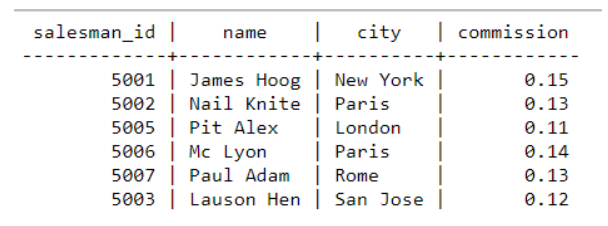
**Q2 . Apply Self Join to Student(Roll,name,address,branch,Class)**

**ANS -**

SELECT S1.Roll,S1.name,S2.address,S2.branch,S2.Class FROM Student S1,Student S2 WHERE S1.Roll=S2.Roll;

**Q3. From the following table, create a view for all salespersons. Return salesperson ID, name, and city.**

**Sample table: salesman**



**ANS -**

CREATE VIEW salespersons AS SELECT (salesperson ID, name, city) FROM salesman ;

**Assignment 11**

**Name:** Kaustubh Patil

**Roll Number:** TECOMP-A68

**Problem Statement:**

Implement Map reduces operation with a suitable example using MongoDB.

**Code:**

To add entries in database:

db.city.insertOne({city:"Rajkot", type:"Urban",state:"Gujarat", population: 1500000});

To create collections and view entries:

db.createCollection('city')

db.city.find().pretty()

Map Reduce commands:

1] Statewise Population

var map=function(){emit(this.state,this.population)}

var reduce=function(key,values){return Array.sum(values)};

db.city.mapReduce(map,reduce,{out:'statewise'});

db.statewise.find().pretty();

2] Citywise Population

var map=function(){emit(this.city,this.population)}

var reduce=function(key,values){return Array.sum(values)};

db.city.mapReduce(map,reduce,{out:'citywise'});

db.citywise.find().pretty();

3] Area Typewise Population

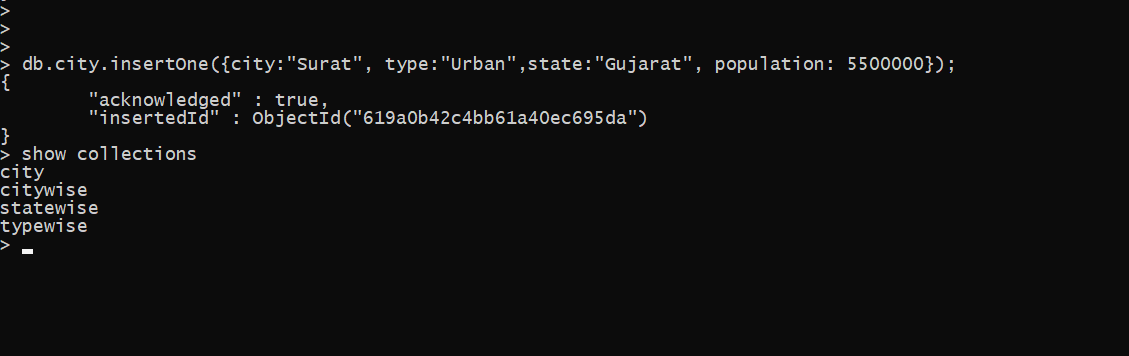
var map=function(){emit(this.type,this.population)}

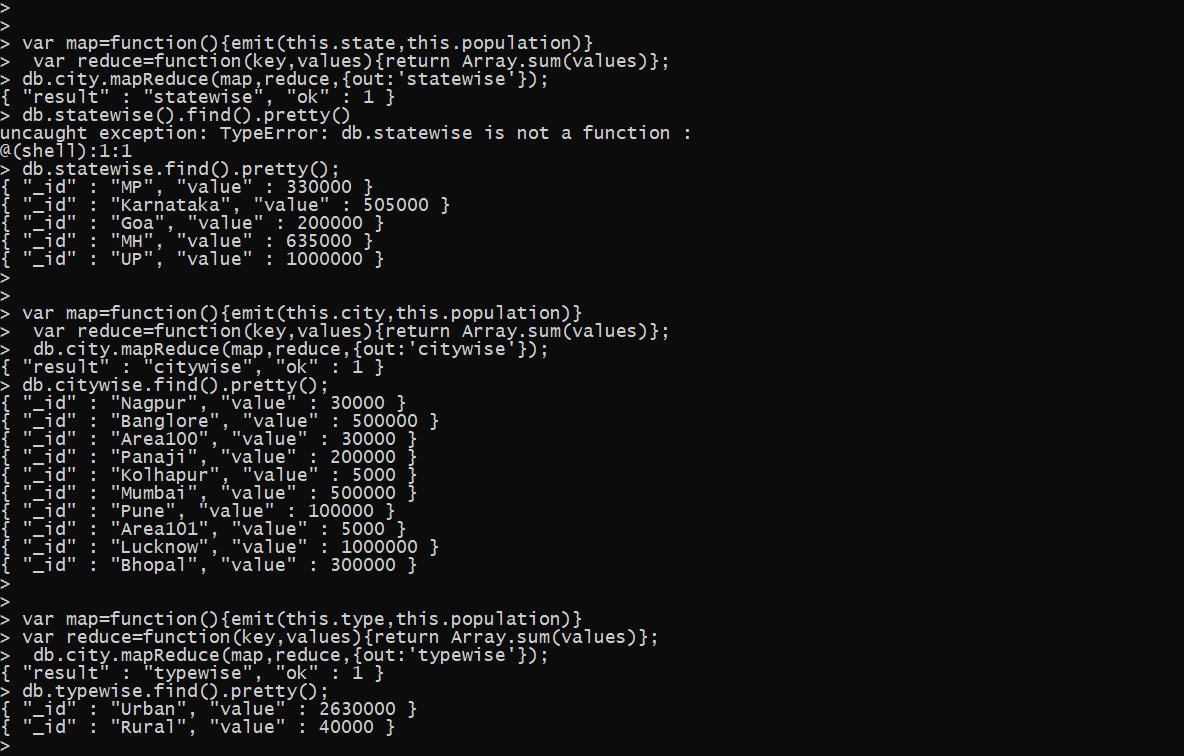
var reduce=function(key,values){return Array.sum(values)};

db.city.mapReduce(map,reduce,{out:'typewise'});

db.typewise.find().pretty();

**Output:**

****

****

**Lab Activity No.10**

**Name : Kaustubh Patil**

**Roll No. : TECOMP-A68**

**Q1 :** Perform aggregation and Indexing using mongodb on below database

**1. Create a database department**

> use Department

**2. Create a collection as teacher with fields as name , department ,experience and salary**

> db.Teacher.insert({name:"Swarali",department:"Computer",experience:3,salary:40000})

WriteResult({ "nInserted" : 1 })

> db.Teacher.insertMany([{name:"Ram",department:"Computer",experience:2,salary:38900},{name:"Manisha",department:"ENTC",salary:56000}])

{

"acknowledged" : true,

"insertedIds" : [

ObjectId("615a824529f11a2bde351654"),

ObjectId("615a824529f11a2bde351655")

]

}

> db.Teacher.insert({name:"Neha",department:"Civil",experience:4,salary:45000})

WriteResult({ "nInserted" : 1 })

**3. Display the department wise average salary.**

db.Teacher.aggregate([{$group:{\_id:"$department",average\_sal:{$avg:"$salary"}}}])

{ "\_id" : "ENTC", "average\_sal" : 56000 }

{ "\_id" : "Civil", "average\_sal" : 45000 }

{ "\_id" : "Computer", "average\_sal" : 39450 }

**4. Display the no. Of employees working in each department**.

> db.Teacher.aggregate({$group:{\_id:"$department",Number\_of\_Employees:{$sum:1}}})

{ "\_id" : "Computer", "Number\_of\_Employees" : 2 }

{ "\_id" : "ENTC", "Number\_of\_Employees" : 1 }

{ "\_id" : "Civil", "Number\_of\_Employees" : 1 }

**5. Display the department wise minimum salary. 6. Apply index and drop index**

> db.Teacher.createIndex({"name":1,"department":-1})

{

"numIndexesBefore" : 1,

"numIndexesAfter" : 2,

"createdCollectionAutomatically" : false,

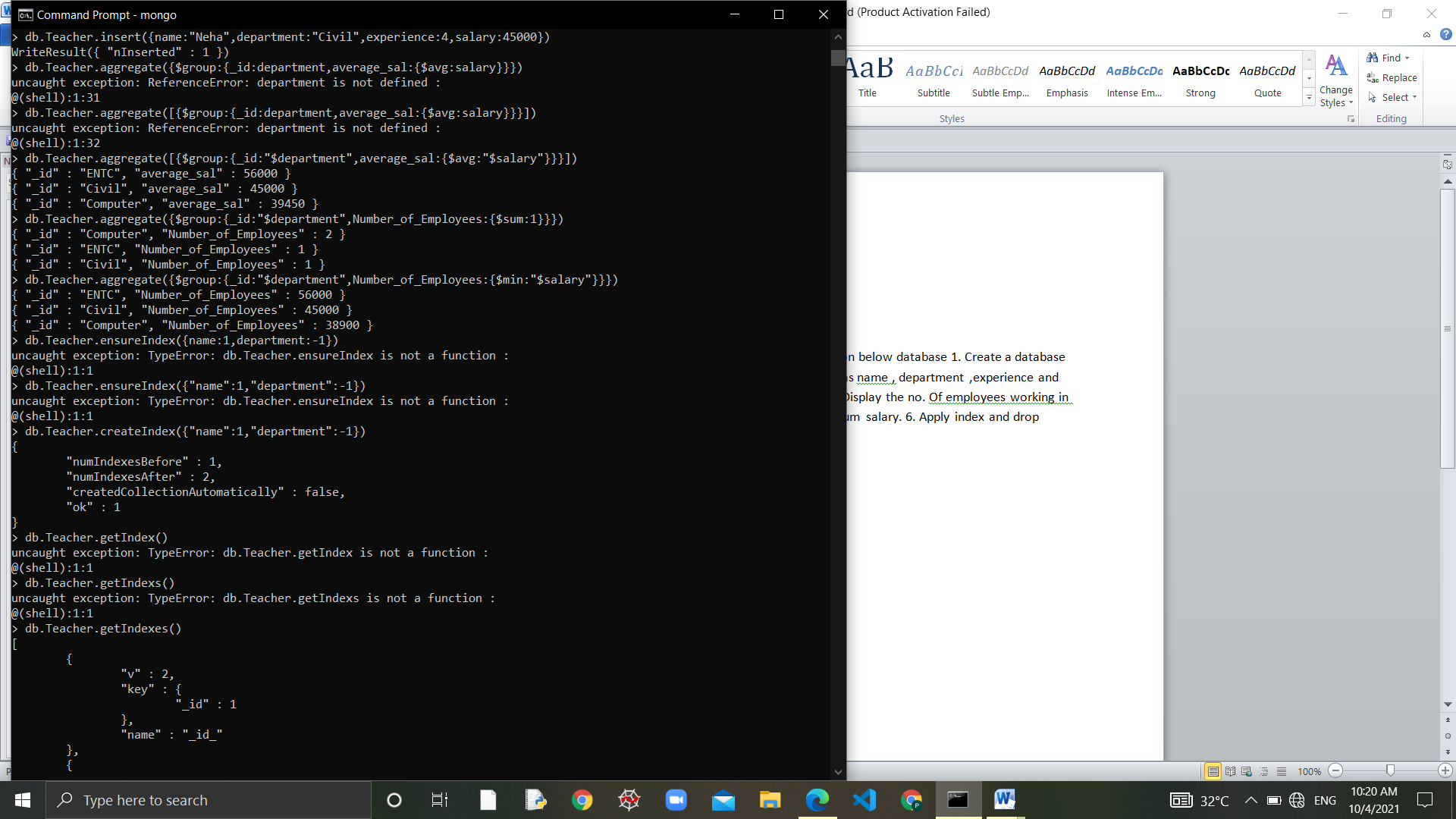
"ok" : 1

}

> db.Teacher.dropIndex({"name":1,"department":-1})

{ "nIndexesWas" : 2, "ok" : 1 }

**Output :**



**Java/PHP - Mysql Connectivity**

**Name :** Kaustubh Patil

**Roll No :** TECOMP-A68

**Program :**

**Config.php**

<?php

session\_start();

$host = "localhost";

$user = "root";

$password = "tiger";

$dbname = "login";

$con = mysqli\_connect($host, $user, $password,$dbname);

if (!$con) {

die("Connection failed: " . mysqli\_connect\_error());

}

**Login.html**

<div class="container">

<form method="post" action="my.php">

<div id="div\_login">

<h1>Login</h1>

<div>

<input type="text" class="textbox" id="txt\_uname" name="txt\_uname" placeholder="Username" />

</div> <div>

<input type="password" class="textbox" id="txt\_uname" name="txt\_pwd" placeholder="Password"/>

</div>

<div>

<input type="submit" value="Submit" name="but\_submit" id="but\_submit" />

</div>

</div>

</form>

</div>

**my.php**

<?php

include "config.php";

if(isset($\_POST['but\_submit'])){

$uname = mysqli\_real\_escape\_string($con,$\_POST['txt\_uname']);

$password = mysqli\_real\_escape\_string($con,$\_POST['txt\_pwd']);

if ($uname != "" && $password != ""){

$sql\_query = "select count(\*) as cntUser from user where uname='".$uname."' and pass='".$password."'";

$result = mysqli\_query($con,$sql\_query);

$row = mysqli\_fetch\_array($result);

$count = $row['cntUser'];

if($count > 0){

echo "Welcome ",$uname ," to PCCOER";

// header("Location:login.html");

}else{

echo "Invalid username and password";

}

}

}

**Lab Assignment 7**

**Name: Kaustubh Patil**

**Roll No.:** TECOMP-A68

**Q1. Write pl/sql code in Trigger not to accept the existing Empno (Unique no).**

**Program**

CREATE OR REPLACE TRIGGER checkExists BEFORE INSERT ON emp FOR EACH ROW

DECLARE

eid number;

CURSOR empc is SELECT id FROM emp;

BEGIN

open empc;

loop

fetch empc into eid;

EXIT WHEN empc%notfound;

if eid = :NEW.id then

raise\_application\_error(-20001, 'Id should be unique');

end if;

end loop;

close empc;

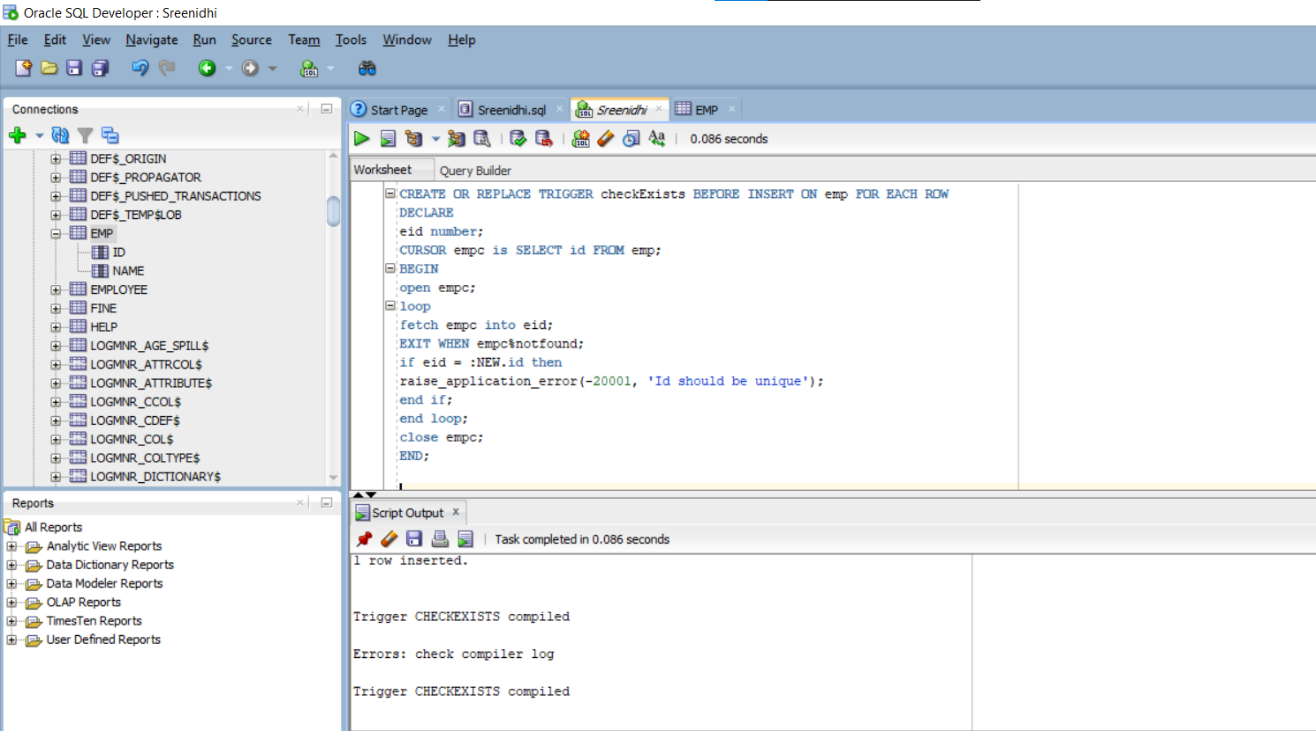
END;

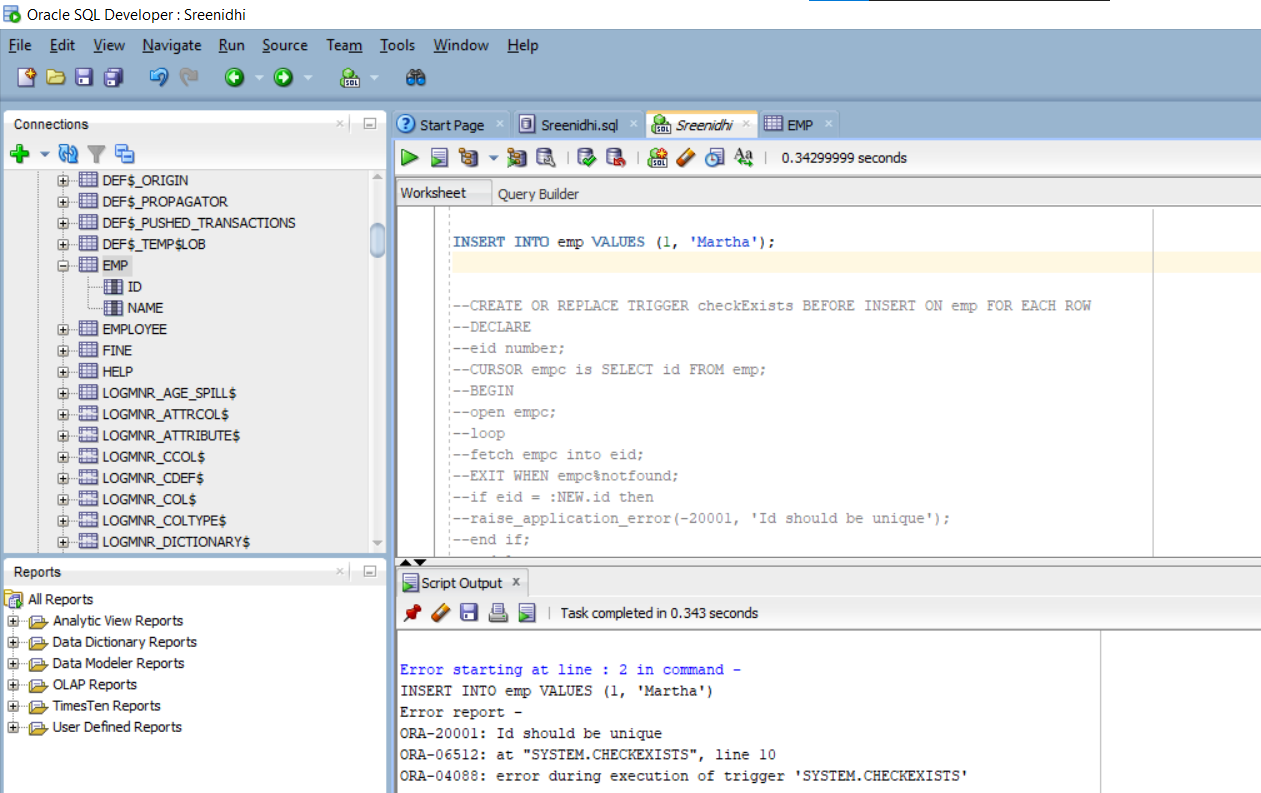
-------

INSERT INTO emp VALUES (1, 'Martha');

INSERT INTO emp VALUES (1, 'Marty');

**Output**

****

****

**Q2. Write pl/sql code using Trigger to salary with more than old salary**

**Program**

CREATE OR REPLACE TRIGGER updateSalary BEFORE UPDATE ON employee FOR EACH ROW WHEN (NEW.salary > OLD.salary)

DECLARE

BEGIN

dbms\_output.put\_line ('Prevoius salary: ' || :OLD.salary);

dbms\_output.put\_line ('Current salary: ' || :NEW.salary);

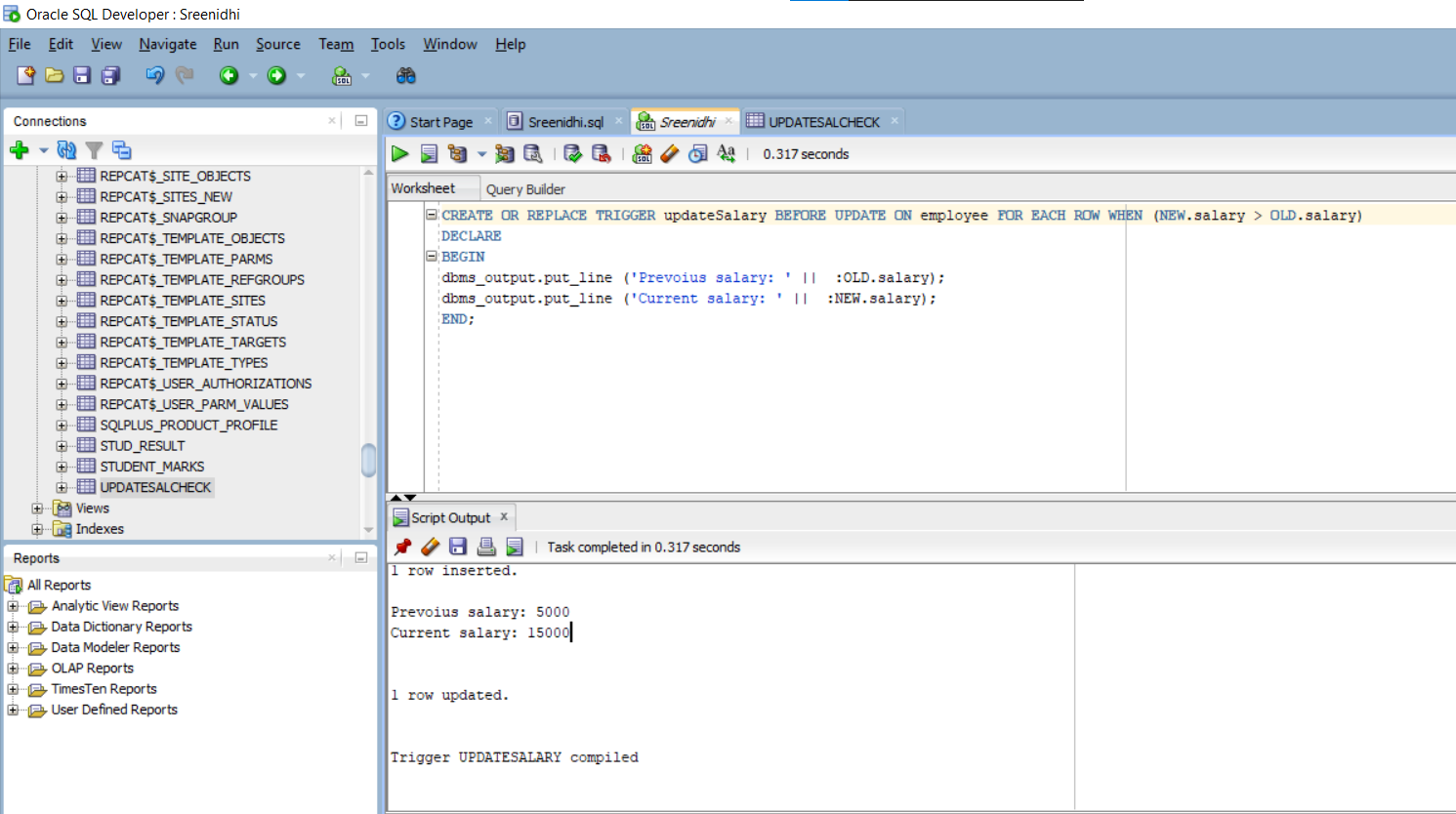
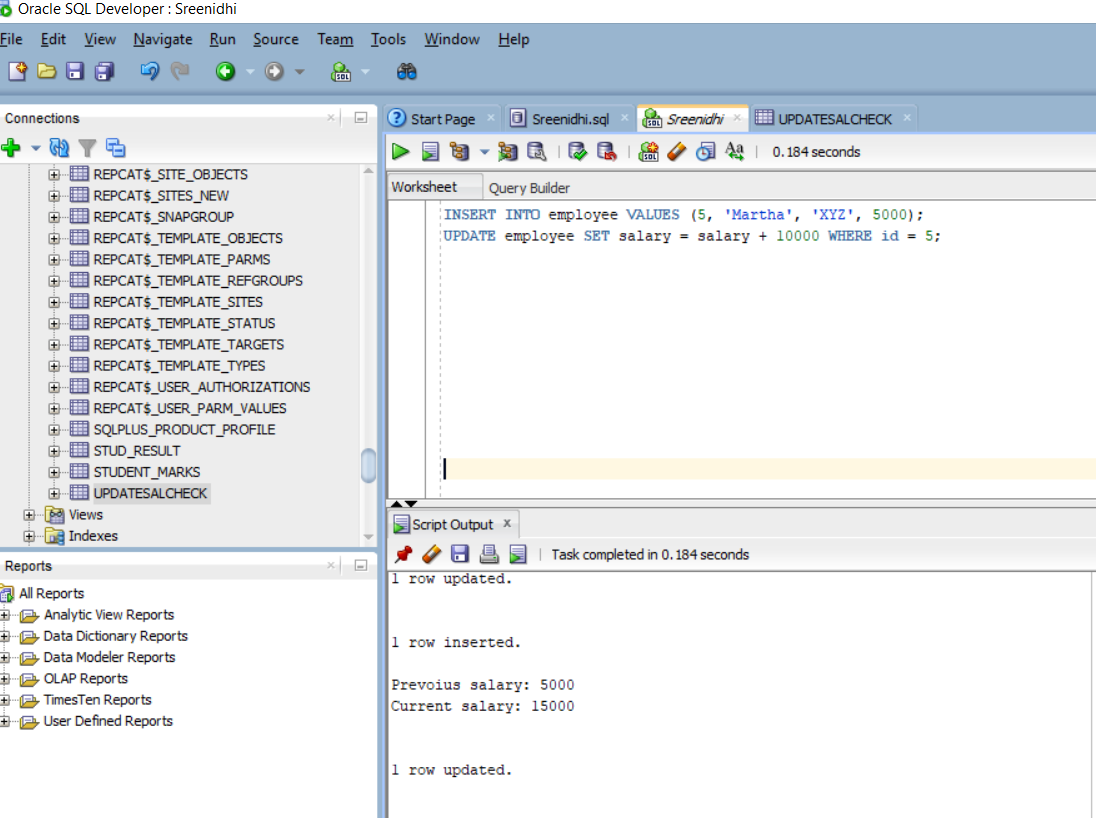
END;

-------

INSERT INTO employee VALUES (5, 'Martha', 'XYZ', 5000);

UPDATE employee SET salary = salary + 10000 WHERE id = 5;

**Output**

****

**Lab Assignment 6**

**Name: Kaustubh Patil**

**Roll No.:** TECOMP-A68

**Q1. Write PL/SQL code to display Employee details using Explicit Cursors Program:**

**Program**

set serveroutput on;

DECLARE

empid employee.id%type;

empName employee.name%type;

empAddr employee.address%type;

CURSOR emp is

SELECT id, name, address FROM employee;

BEGIN

OPEN emp;

LOOP

FETCH emp into empid, empName, empAddr;

EXIT WHEN emp%notfound;

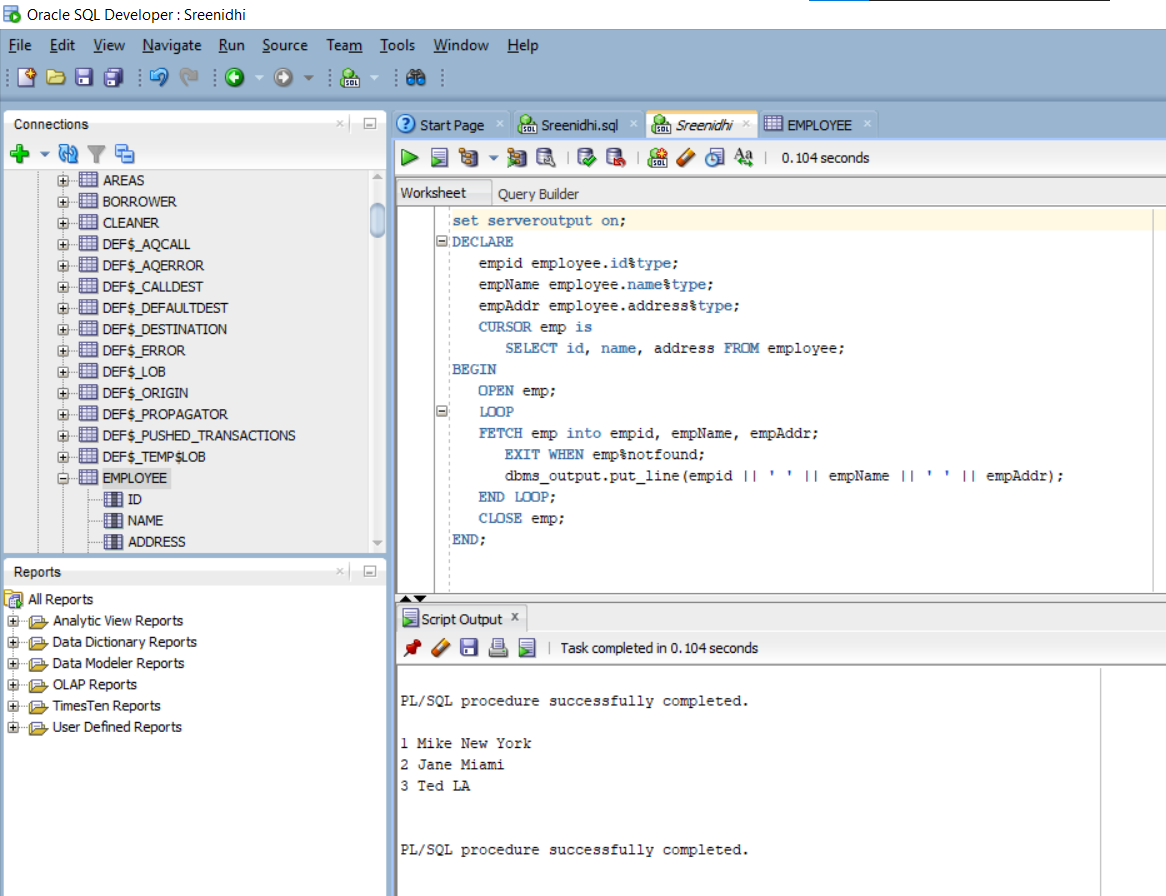
dbms\_output.put\_line(empid || ' ' || empName || ' ' || empAddr);

END LOOP;

CLOSE emp;

END;

**Output**



**Q2. Write PL/SQL code in Cursor to display employee names and salary.**

**Program**

set serveroutput on;

DECLARE

empid employee.id%type;

empName employee.name%type;

empSal employee.salary%type;

CURSOR emp is

SELECT id, name, salary FROM employee;

BEGIN

OPEN emp;

LOOP

FETCH emp into empid, empName, empSal;

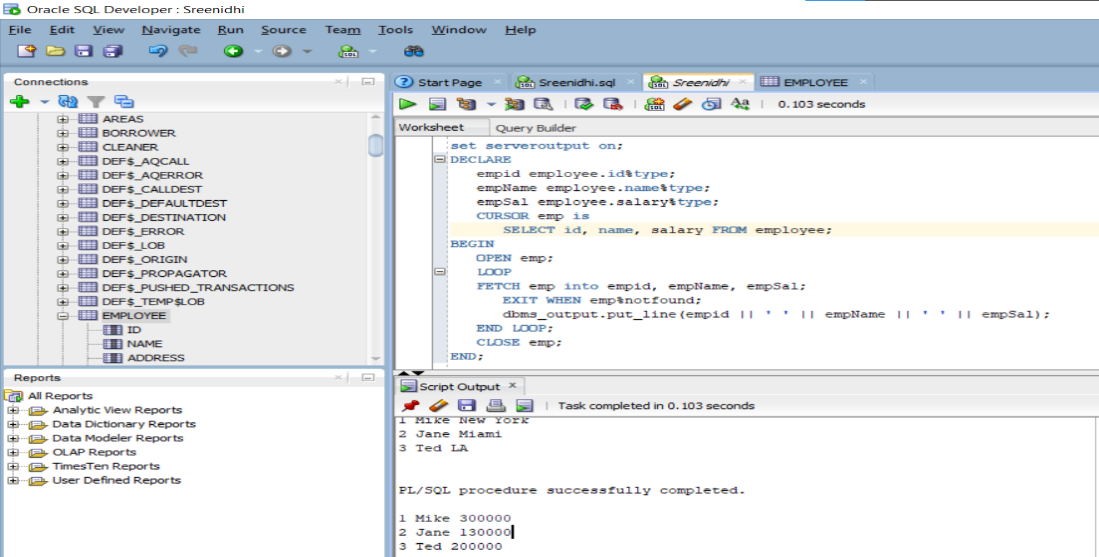
EXIT WHEN emp%notfound;

dbms\_output.put\_line(empid || ' ' || empName || ' ' || empSal);

END LOOP;

CLOSE emp;

END;



**Output**

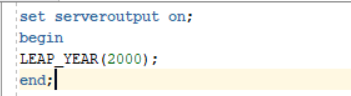
**Lab Assignment 5**

**Name:** Kaustubh Patil

**Roll No.:** TECOMP-A68

**Q1. Implement Stored Procedure to Check if a given year is a leap year. The condition is:- year should be (divisible by 4 and not divisible by 100) or (divisible by 4 and divisible by 400.) Display the output on the screen using dbms\_output.put\_line. The year should be input by the user.**

**Program:**



CREATE OR REPLACE PROCEDURE LEAP\_YEAR

(

YEAR IN VARCHAR2

) AS

result1 int;

result2 int;

result3 int;

BEGIN

result1:=mod(YEAR,100);

result2:=mod(YEAR,400);

result3:=mod(YEAR,4);

IF ((result3=0) AND not(result1=0) OR (result3=0) AND (result2=0))THEN

dbms\_output.put\_line(' '||YEAR||' is a leap year');

ELSE

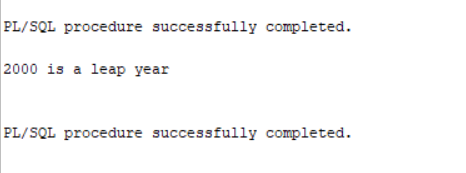
dbms\_output.put\_line(' '||YEAR||' is not a leap year');

END IF;

NULL;

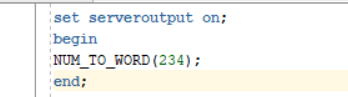
END LEAP\_YEAR;

**Output**



**Q2. Implement Stored Procedure or Function to read in a number and print it out digit by digit, as a series of words. For example, the number 523 would be printed as "five two three". Use decode function within a for loop. Display the results on the screen using dbms\_output.put\_line**

**Program**

****

CREATE OR REPLACE PROCEDURE NUM\_TO\_WORD (

NUM IN VARCHAR2

) AS

i varchar(1);

n varchar(5):=NUM;

c int:=length(num);

RESULT varchar(10);

BEGIN

dbms\_output.put\_line('ENTERED NO. IS '|| num);

for j in 1..c LOOP

i:=substr(n, 1,1);

n:=substr(n,2);

SELECT decode(i, 1, 'one', 2, 'two', 3, 'three', 4, 'four', 5, 'five', 6, 'six', 7, 'seven', 8, 'eight',9,'nine','zero') INTO RESULT FROM dual;

dbms\_output.put\_line(RESULT);

--exit WHEN c=1;

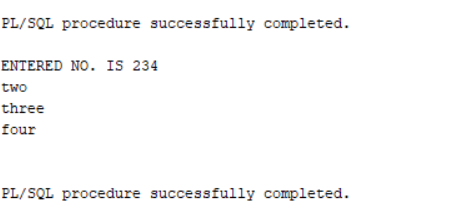
--c:=c-1;

end LOOP;

NULL;

END NUM\_TO\_WORD;

**Output**



**Lab Assignment -4**

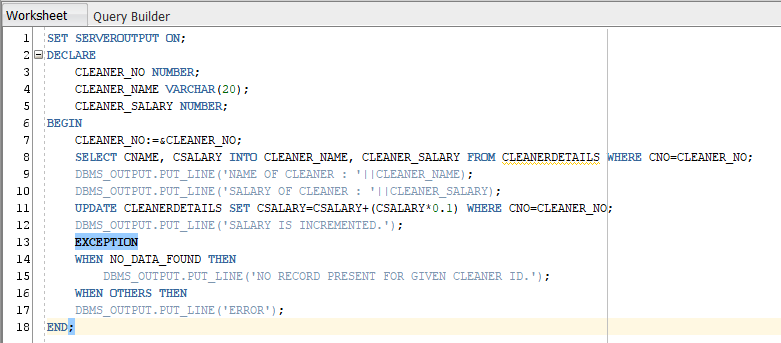
**Name: - Kaustubh Patil**

**Roll No: - TECOMP-A68**

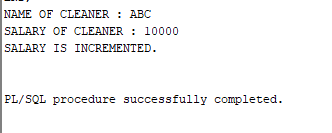
Q1. Write a PL/SQL block which accepts a cleaner number and returns the cleaners name and salary and update this cleaner’s details by increasing salary by 10%. Use Exception Handling

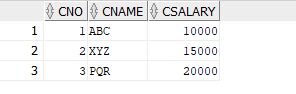
**ANS –**

**Code: -**



**Output: -**

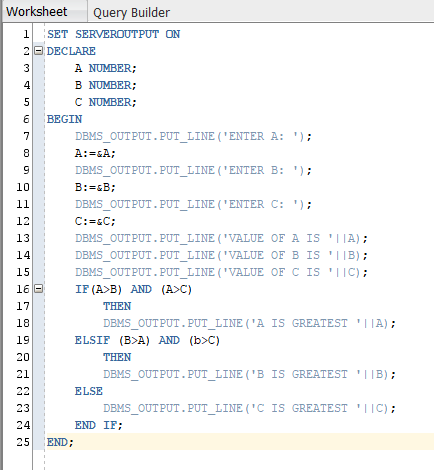


**CLEANERDETAILS Table: -**   


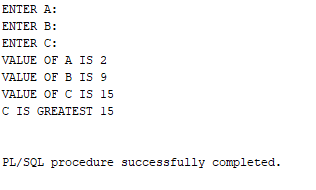
Q2. Write a PL/SQL block to find Largest of three number

**Ans –**

**Code: -**



**Output: -**



ASSIGNMENT 1:

Q.1 Draw an ER Diagram for an ATM System

NNNq

Savings

Current

IS A

Account

Holds

Customer

Uses

Enquiry

Withdraw

Deposit

Transfer

IS A

Transaction

Performs

Operator

Operates

Card

Inserted to

ATM MACHINE

Q.2 CREATE ER-DIAGRAM INTO TABLE

Table Conversion

Account(AC\_NO, Balance, b\_name)

Branch(b\_name, b\_city, Assets)

Loan(L\_no, Amt, b\_name)

Borrower(c\_name, L\_no)

Customer(c\_name, c\_city, c\_street)

Depositor(ac\_n0, c\_name)

Q.3 Covert Table to 3 NF

Conversion to 1NF

|  |  |  |  |
| --- | --- | --- | --- |
| FULL NAME | Physical Address | Movies Rented | Salutation |
| Janet Jones | First Street | Pirates of Caribean | Ms |
| Janet Jones | First Street | Clash of Titans | Ms |
| Robert Phil | 3rd Street | Forgetting Sarah marshal | Mr |
| Robert Phil | 3rd Street | Daddys little girl | Mr |
| Robert Phil | 5th avenue | Clash of titans | Mr |

Conversion to 2NF

2a:

|  |  |  |  |
| --- | --- | --- | --- |
| Memb\_id | Full name | Physical Address | Salutation |
| 1 | Janet Jones | 1st street | Ms |
| 2 | Robert Phil | 3rd street | Ms |
| 3 | Robert Phil | 5th avenue | Mr |

2b:

|  |  |
| --- | --- |
| Memb\_Id | Movies\_Rented |
| 1 | Pirates of Caribbean |
| 1 | Clash of Titans |
| 2 | Forgetting sarah marshal |
| 2 | Dadys little girl |
| 3 | Clash of titans |

Conversion to 3NF

3a:

|  |  |  |  |
| --- | --- | --- | --- |
| Memb\_id | Full name | Physical Address | Salutation id |
| 1 | Janet Jones | 1st street | 2 |
| 2 | Robert Phil | 3rd street | 1 |
| 3 | Robert Phil | 5th avenue | 1 |

3b:

|  |  |
| --- | --- |
| Memb\_Id | Movies\_Rented |
| 1 | Pirates of Caribbean |
| 1 | Clash of Titans |
| 2 | Forgetting sarah marshal |
| 2 | Dadys little girl |
| 3 | Clash of titans |

3c:

|  |  |
| --- | --- |
| Salutation id | salutation |
| 1 | Mr |
| 2 | Ms |
| 3 | Mrs |
| 4 | Dr |

Q.4 Consider following Relationship Entity diagram

Which of the following possible relation will not hold if above ERD is mapped into a relation model?

1. Person(Name, NID)
2. Exam(ExamID, ExamName)
3. Qualifiaction(NID, ExamID, QualifiedDate)
4. Exam(ExamID, ExamName, NID)

Exam(ExamID, ExamName, NID) this relation will not hold as it contains NID attribute of Person entity