

NAME:-ASHUTOSH RAI

ROLL NO.: -205119021

SUBJECT:-DATABASE MANAGEMENT SYSTEM LAB

DEPT:- COMPUTER APPLICATIONS

SEMESTER:-SECOND SEMESTER

LAB MANUAL

NATIONAL INSTITUTE OF TECHNOLOGY

# 1. Data Definition Language (DDL) commands in RDBMS.

**Problem 1.1: Create a table called EMP with the following structure.**

**Name Type**

-----  
EMPNO NUMBER(6)  
ENAME VARCHAR2(20)  
JOB VARCHAR2(10)  
MGR NUMBER(4)  
DEPTNO NUMBER(3)  
SAL NUMBER(7,2)

Allow NULL for all columns except ename and job.

=>

Create table emp(empno number(6),ename varchar(20) not null,job varchar(10) not null,mgr number(4),deptno number(3) not null,sal number(7,2),primary key (deptno));

**Problem 1.2: Add a column commission to the emp table**

**Commission numeric null allowed.**

=>

Alter table emp add commission number(10);

**Problem 1.3: Modify the column width of the job field of emp table.**

=>

Alter table emp modify column job varchar(30);

**Problem 1.4: Create dept table with the following structure.**

**Name Type**

-----  
DEPTNO NUMBER(2)  
DNAME VARCHAR2(10)  
LOC VARCHAR2(10)  
Deptno as the primarykey

=>

Create table dept(deptno number(3) not null,dname varchar(10),loc varchar(10),primary key (deptno));

**Problem 1.5: Add constraints to the emp table that empno as the primary key and deptno as the foreign key.**

=>

**Alter table emp add constraint fk\_emp foreign key (deptno) references emp(deptno);**

**Problem 1.6: Add constraints to the emp table to check the empno value while entering (i.e) empno > 100.**

**=>**

**Alter table emp add constraint ck\_empno check (empno>100);**

**Problem 1.7: Salary value by default is 5000, otherwise as entered values**

**=>**

**Alter table emp alter sal set default 5000;**

**Problem 1.8: Add columns Dob to the emp table.**

**=>**

**Alter table emp add dob date(10);**

## **2. Data Manipulation Language (DML) commands in RDBMS.**

**Problem 2.1: Insert 3 records into dept table.**

**=>**

**Insert into dept values(10 , 'MANAGEMENT ' , 'MAIN BLOCK');**

**'same command only by changing the value we can enter all other given values but sequence should be same as the sequence of column.'**

**Problem 2.2: Insert 10 records into emp table.**

**=>**

**Insert into emp values(7001, 'smith', 'clerk', 7566, 20, 800, 200, 17-dec-1975);**

**'same command only by changing the value we can enter all other given values but sequence should be same as the sequence of column.'**

**Problem 2.3: Update the emp table to set the default commission of all employees to Rs 1000/- who are working as managers.**

**=>**

**Update emp set commission=1000 where job='managers';**

**Problem 2.4: Create a pseudo table employee with the same structure as the table emp and insert rows into the table using select clauses.**

**=>**

**Create table employee as select \* from emp;**

**Problem 2.5: Delete only those who are working as supervisors.**

**=>**

**Delete from employee where job='supervisors';**

**Problem 2.6: Delete the rows whose empno is 7599.**

**=>**

**Delete from employee where empno=7599;**

**Problem 2.7: List the records in the emp table orderby salary in ascending order.**

**=>**

**Select \* from emp order by sal;**

**Problem 2.8: List the records in the emp table orderby salary in descending order.**

**=>**

**Select \* from emp order by sal desc;**

**Problem 2.9: Display only those employees whose deptno is 30.**

=>

Select \* from emp where deptno=30;

**Problem 2.10: Display deptno from the table employee avoiding the duplicated values.**

=>

Select distinct deptno from emp;

**Problem 2.11: List the records in sorted order of their employees.**

=>

Select \* from emp order by ename;

**Problem 2.12: create a manager table from the emp table which should hold details a only about the managers.**

=>

Create table manager as select \* from emp where job='manager';

**Problem 2.13: List the employee names whose commission is null.**

=>

Select ename from emp where commission=null;

**Problem 2.14: List the employee names and the department name in which they are working.**

=>

Select ename,dname from emp,dept where emp.deptno=dept.deptno;

### 3. In Built functions in RDBMS.

**Problem 3.1: Select all employees from department numbers 7369,7499.**

=>

**Select \* from emp where deptno in(7369,7499);**

**Problem 3.2: Display all the details of the records whose employee name starts with 'S'.**

=>

**Select \* from emp where ename like 's%';**

**Problem 3.3: Display all the details of the records whose employee name does not starts with 'S'.**

=>

**Select \* from emp where ename not like 's%';**

**Problem 3.4: Display the rows whose empno ranges from 7500 to 7600.**

=>

**Select \* from emp where empno between 7500 and 7600;**

**Problem 3.5: Display the rows whose empno not in range from 7500 to 7600.**

=>

**Select \* from emp where empno not between 7500 and 7600;**

**Problem 3.6: Calculate the square root of the salary of all employees.**

=>

**Select sqrt(sal) from emp;**

**Problem 3.7: Count the total records in the emp table.**

=>

**Select count(\*) from emp;**

**Problem 3.8: Calculate the total and average salary amount of the emtable.**

=>

**Select sum(sal),avg(sal) from emp;**

**Problem 3.9: Determine the max and min salary and rename the column as max\_salary and min\_salary.**

=>

**Select min(sal) "min\_sal",max(sal) "max\_sal" from emp;**

**Problem 3.10: Display total salary spent for employees.**

=>

**Select sum(sal) from emp;**

**Problem 3.11: Display total salary spent for each job category.**

=>

**Select job,sum(sal) from emp group by job;**

**Problem 3.12: Display the month name of date "14-jul-09" in full.**

**=>**

**Select to\_char(to\_date('14-jul-09'),'month') from dual;**

**Problem 3.13: Display the Dob of all employees in the format "dd-mm-yy".**

**=>**

**Select to\_date(dob,'dd-mm-yy') from emp;**

**Problem 3.14: Display the date two months after the Dob of employees.**

**=>**

**Select add\_months(dob,2) from emp;**

**Problem 3.15: Display the last date of that month in "05-Oct-09".**

**=>**

**Select last\_day('05-Oct-09') from dual';**

**Problem 3.16: Display the rounded date in the year format, month format, day format in the employees.**

**=>**

**Select round(to\_date(dob),'day') from emp;**

**Select round(to\_date(dob),'month') from emp;**

**Select round(to\_date(dob),'year') from emp;**

**Problem 3.17: Display the date 60 days before current date.**

**=>**

**Select (sysdate-60) from emp;**

**Problem 3.18: List all employee names , salary and 15% rise in salary.**

**=>**

**Select ename,sal,sal+0.15\*sal from emp;**

**Problem 3.19: List all employees which starts with either B or C.**

**=>**

**Select \* from emp where ename like'b%' or ename like 'c%';**

**Problem 3.20: Display lowest paid employee details under each manager.**

**=>**

**Select \* from emp where sal in(select min(sal) from emp where group by mgr);**

**Problem 3.21: Display number of employees working in each department and their department name.**

**=>**

**Select dname,count(ename) from dept,emp where dept.deptno=emp.deptno group by deptno;**

**Problem 3.22: Display the employee names whose name contains up to 5**

characters.

=>

Select ename from emp where length(ename)<=5;

**Problem 3.23: List all employee names and their manager whose manager is 7499 or 7566 Or 7611.**

=>

Select ename from emp where mgr in(7499,7566,7611);

**Problem3.24: Find how many job titles are available in employee table.**

=>

Select count(distinct job) from emp;

**Problem 3.25 : What is the difference between maximum and minimum salaries of employees in the organization?**

=>

Select max(sal)-min(sal) from emp;

**Problem 3.26: Find no.of dept in employee table.**

=>

Select count(distinct deptno) from emp;

**Problem 3.27: Display the names and dob of all employees who were born in Feburary.**

=>

Select ename,dob from emp where to\_char(dob,'mon')='feb';

**Problem 3.28: List out the employee names who will celebrate their birthdays during current month.**

=>

Select ename from emp where to\_char(dob,'mon') like to\_char(sysdate,'mon');

**Problem 3.29: List out the employee names whose names starts with s and ends with h.**

=>

Select ename from emp where ename like 's%' and ename like '%h';

**Problem 3.30: List out the employee names whose salary is greater than 5000,6000.**

=>

Select ename from emp where sal>5000;



## 4. Nested Queries & Joins in RDBMS

**Problem 4.1:** Select all employees from 'maintainance' and 'development' dept.

=>

Select \* from emp,dept where emp.deptno=dept.deptno and (dname='maintainance' or dname='development');

**Problem 4.2:** Display all employee names and salary whose salary is greater than minimum salary of the company and job title starts with 'M'.

=>

Select ename,sal from emp where job like 'm%' and sal>(select min(sal) from emp);

**Problem 4.3:** Issue a query to find all the employees who work in the same job as jones.

=>

Select \* from emp where job=(select job from emp where ename='jones');

**Problem 4.4:** Issue a query to display information about employees who earn more than any employee in dept 30.

=>

Select \* from emp where sal>(select max(sal) from emp where deptno=30);

**Problem 4.5:** Display the employees who have the same job as jones and whose salary >= fords.

=>

Select \* from emp where job=(select job from emp where ename='jones') and sal>=(select sal from emp where ename='fords');

**Problem 4.6:** Write a query to display the name and job of all employees in dept 20 who have a job that someone in the Management dept as well.

=>

Select ename,job from emp where job=(select job from emp x,dept y where x.deptno=y.deptno and (x.deptno=20 and y.dname='management');

**Problem 4.7:** Issue a query to list all the employees who salary is > the average salary of their own dept.

=>select \* from emp x where x.sal>(select avg(sal) from emp where deptno=x.deptno);

**Problem 4.8:** Write a query that would display the empname, job where each employee works and the name of their dept.

=>select ename,dname from emp,dept where emp.deptno=dept.deptno;

**Problem 4.9:** Write a query to list the employees having the same job as

employees located in 'mainblock'.(use multiple subquery)

=>select \* from emp x where job=(select job from emp where loc='mainblock' and job=x.job);

**Problem 4.10:** Write a query to list the employees in dept 10 with the same job as anyone in the development dept.

=>

SELECT \* FROM EMP WHERE JOB=(SELECT JOB FROM EMP WHERE EMPNAME='FORD') AND SAL=(SELECT SAL FROM EMP WHERE EMPNAME='FORD');

**Problem 4.11:** Write a query to list the employees with the same job and salary as 'ford'.

=>

SELECT DNAME FROM DEPT WHERE DEPTNO=ANY(SELECT DEPTNO FROM (SELECT COUNT(JOB) AS NO,DEPTNO FROM EMP WHERE JOB='SALESMAN' GROUP BY DEPTNO) WHERE NO>=2);

**Problem 4.12:** Write a query to list all depts. with at least 2 salesman.

=>

SELECT DNAME FROM DEPT WHERE DEPTNO=ANY(SELECT DEPTNO FROM (SELECT COUNT(JOB) AS NO,DEPTNO FROM EMP WHERE JOB='SALESMAN' GROUP BY DEPTNO) WHERE NO>=2);

**Problem 4.13:** Write a query to list the employees in dept 20 with the same job as anyone in dept 30.

=>

SELECT \* FROM EMP WHERE DEPTNO=20 AND JOB=ANY(SELECT JOB FROM EMP WHERE DEPTNO=30);

**Problem 4.14:** List out the employee names who get the salary greater than the maximum salaries of dept with dept no 20,30

=>

SELECT \* FROM EMP WHERE SAL>ANY(SELECT MAX(SAL) FROM EMP WHERE DEPTNO=20 OR DEPTNO=30 GROUP BY DEPTNO);

**Problem 4.15:** Display the maximum salaries of the departments whose maximum salary is greater than 9000.

=>

SELECT MAX(SAL) FROM EMP GROUP BY DEPTNO HAVING MAX(SAL)>9000;

**Problem 4.16:** Display the maximum salaries of the departments whose minimum salary is greater than 1000 and lesser than 5000.

=>

SELECT MAX(SAL) FROM EMP GROUP BY EMPNAME HAVING MIN(SAL)>1000 AND MIN(SAL)<5000;

## **JOINS**

### **EQUI-JOIN**

**Problem 4.17:** Display the departments that are accredited by the quality council.

=>

```
SELECT A.DNAME FROM DEPT D,ACCDEPT A WHERE D.DEPTNO=A.DEPTNO;
```

### **NON-EQUIJOIN**

**Problem 4.18:** Display the employees of departments which are not accredited by the quality council

=>

```
SELECT EMPNAME FROM EMP WHERE DEPTNO!=ANY(SELECT DEPTNO FROM ACCDEPT);
```

### **LEFTOUT-JOIN**

**Problem 4.19:** Display all the employees and the departments implementing a left outer join.

=>

```
SELECT * FROM EMP LEFT JOIN DEPT ON DEPT.DEPTNO=EMP.DEPTNO;
```

### **RIGHTOUTER-JOIN**

**Problem 4.20:** Display the employee name and department name in which they are working implementing a right outer join.

=>

```
SELECT * FROM EMP RIGHT JOIN DEPT ON DEPT.DEPTNO=EMP.DEPTNO;
```

### **FULLOUTER-JOIN**

**Problem 4.21:** Display the employee name and department name in which they are working implementing a full outer join.

=>

```
SELECT * FROM EMP FULL JOIN DEPT ON DEPT.DEPTNO=EMP.DEPTNO;
```

### **SELFJOIN**

**Problem 4.22:** Write a query to display their employee names and their managers name.

=>

```
SELECT E.EMPNAME,M.EMPNAME FROM EMP E,EMP M WHERE E.MGR=M.EMPNO;
```

**Problem 4.23:** Write a query to display their employee names and their managers salary for every employee .

=>

```
SELECT E.EMPNAME,M.SAL FROM EMP E,EMP M WHERE E.MGR=M.EMPNO;
```

**Problem 4.24:** Write a query to output the name , job, empno, deptname and location for each dept, even if there are no employees.

=>

```
SELECT E.EMPNAME,E.JOB,E.EMPNO,D.DNAME,D.DLOC FROM EMP E, DEPT D WHERE  
E.DEPTNO=E.DEPTNO AND D.DEPTNO=E.DEPTNO;
```

**Problem 4.25:** Find the name of the manager for each employee. Include  
The following in the output: empno, empname, job and his manager's name.

=>

```
SELECT E.EMPNO,E.EMPNAME,E.JOB,M.EMPNAME FROM EMP E,EMP M WHERE E.MGR=M.EMPNO;
```

**Problem 4.26:** Display the details of those who draw the same salary.

=>

```
SELECT E.EMPNAME,P.EMPNAME FROM EMP E,EMP P WHERE E.SAL=P.SAL AND  
E.EMPNAME!=P.EMPNAME;
```

## 5. Set operators & Views in RDBMS.

**Problem 5.1:** Display all the dept numbers available with the dept and accdept tables avoiding duplicates.

=>

```
SELECT DEPTNO FROM DEPT UNION SELECT DEPTNO FROM ACCDEPT;
```

**Problem 5.2:** Display all the dept numbers available with the dept and accdept

=>

```
SELECT DEPTNO FROM DEPT UNION ALL SELECT DEPTNO FROM ACCDEPT;
```

**Problem 5.3:** Display dept no available in both the dept and acc dept tables.

=>

```
SELECT DEPTNO FROM DEPT INTERSECT SELECT DEPTNO FROM ACCDEPT;
```

**Problem 5.4:** Display all the dept numbers available in dept and not in accdept Table.

=>

```
SELECT DEPTNO FROM DEPT MINUS SELECT DEPTNO FROM ACCDEPT;
```

### Views

**Problem 5.5:** The organization wants to display only the details of the employees those who are managers.( horizontal portioning)

=>

```
CREATE VIEW MANAGERS AS SELECT * FROM EMP WHERE JOB='MANAGER';  
SELECT * FROM MANAGERS;
```

**Problem 5.6:** The organization wants to display only the details like empno,empname,deptno,deptname of the employees .  
(vertical portioning)

=>

```
create view general as select empno,ename,emp.deptno,dname from emp,dept where  
emp.deptno=dept.deptno;  
select * from general;
```

**Problem 5.7:** The organization wants to display only the details like empno,empname,deptno,deptname of the all the employees except the HOD and CEO. (full portioning)

=>

```
CREATE VIEW EMP_ALL AS SELECT E.EMPNO,E.EMPNAME,D.DEPTNO,D.DNAME FROM EMP E, DEPT D  
WHERE E.DEPTNO=D.DEPTNO AND E.JOB NOT IN('HOD','CEO');  
SELECT * FROM EMP_ALL;
```

**Problem 5.8:** Display all the views generated.

=>

```
SELECT * FROM GENERAL;
```

```
SELECT * FROM EMP_ALL ;
```

**Problem 5.9: Execute the DML commands on the view created.**

**=>**

```
DROP VIEW EMP_ALL;
```

**Problem 5.10: Drop a view.**

**=>**

```
DROP VIEW EMP_ALL;
```

## 6. Control Structures

**Program 6.1: write a pl/sql program to swap two numbers with out taking third variable**

```
=>
declare
a number(10);
b number(10);
begin
a:=&a;
b:=&b;
dbms_output.put_line('THE PREV VALUES OF A AND B WERE');
dbms_output.put_line(a);
dbms_output.put_line(b);
a:=a+b;
b:=a-b;
a:=a-b;
dbms_output.put_line('THE VALUES OF A AND B ARE');
dbms_output.put_line(a);
dbms_output.put_line(b);
end;
/
```

**Program 6.2: write a pl/sql program to swap two numbers by taking third variable**

```
=>

declare
a number(10);
b number(10);
c number(10);
begin
a:=&a;
b:=&b;
dbms_output.put_line('THE PREV VALUES OF A AND B WERE');
dbms_output.put_line(a);
dbms_output.put_line(b);
c:=a;
a:=b;
b:=c;
dbms_output.put_line('THE VALUES OF A AND B ARE');
dbms_output.put_line(a);
dbms_output.put_line(b);
end;
/
```

**Program 6.3: Write a pl/sql program to find the largest of two numbers**

```
=>
declare
a number;
b number;
begin
a:=&a;
b:=&b;
```

```

if a=b then
dbms_output.put_line('BOTH ARE EQUAL');
elsif a>b then
dbms_output.put_line('A IS GREATER');
else
dbms_output.put_line('B IS GREATER');
end if;
end;
/

```

**Program 6.4:write a pl/sql program to find the total and average of 6 subjects and display the grade**

=>

```

declare
java number(10);
dbms number(10);
co number(10);
se number(10); es
number(10); ppl
number(10); total
number(10); avgs
number(10); per
number(10);
dbms_output.put_line('ENTER THE MARKS');
begin
java:=&java;
dbms:=&dbms;
co:=&co;
se:=&se;
es:=&es;
ppl:=&ppl;
total:=(java+dbms+co+se+es+ppl);
per:=(total/600)*100;
if java<40 or dbms<40 or co<40 or se<40 or es<40 or ppl<40 then
dbms_output.put_line('FAIL');
elsif per>75 then
dbms_output.put_line('GRADE A');
elsif per>65 and per<75 then
dbms_output.put_line('GRADE B');
elsif per>55 and per<65 then
dbms_output.put_line('GRADE C');
else
dbms_output.put_line('INVALID INPUT');
end if;
dbms_output.put_line('PERCENTAGE IS '||per);
dbms_output.put_line('TOTAL IS '||total);
end;
/

```

**Program 6.5:Write a pl/sql program to find the sum of digits in a given number**

=>

declare



```

a number;
d number:=0;
sum1 number:=0;
begin
a:=&a;
while a>0
loop
d:=mod(a,10);
sum1:=sum1+d;
a:=trunc(a/10);
end loop;
dbms_output.put_line('SUM = ' || sum1);
end;
/

```

**Program 6.6:write a pl/sql program to display the number in reverse order**

=>

```

declare
a number;
rev number;
d number;
begin
a:=&a;
rev:=0;
while a>0
loop
d:=mod(a,10);
rev:=(rev*10)+d;
a:=trunc(a/10);
end loop;
dbms_output.put_line('REVERSE NUMBER = ' || rev);
end;
/

```

**Program 6.7:Write a pl/sql program to check whether the given number is prime or not**

=>

```

declare
a number;
c number:=0;
i number;
begin
a:=&a;
for i in 1..a
loop
if mod(a,i)=0 then
c:=c+1;
end if;
end loop;
if c=2 then
dbms_output.put_line(a || ' is a prime number');
else
dbms_output.put_line(a || ' is not a prime number');
end if;

```

```
end;  
/
```

**Program 6.8: Write a pl/sql program to find the factorial of a given number**

```
=>  
declare  
n number;  
f number:=1;  
begin  
n:=&n;  
for i in 1..n  
loop  
f:=f*i;  
end loop;  
dbms_output.put_line('Factorial ' || n || ' is ' || f);  
end;  
/
```

**Program 6.9:write a pl/sql code block to calculate the area of a circle for a value of radius varying from 3 to 7.**

**Store the radius and the corresponding values of calculated area in an empty table named areas ,consisting of two columns radius & area**

**TABLE NAME:AREAS**

**RADIUS AREA**

```
=>  
create table areas(radius number(10),area number(6,2));
```

```
declare  
pi constant number(4,2):=3.14;  
radius number(5):=3;  
area number(6,2);  
begin  
while radius<7 loop  
area:=pi*power(radius,2);  
insert into areas values(radius,area);  
radius:=radius+1;  
end loop;  
end;  
/
```

**Program 6.10:write a pl/sql code block that will accept an account number from the user,check if the users balance is less than minimum balance,only then deduct rs.100/- from the balance.this process is fired on the acct table.**

```
=>  
create table acct(name varchar2(10),cur_bal number(10),acctno number(6,2));  
insert into stud values('&sname',&rollno,&marks);  
select * from acct;
```

```
declare  
mano number(5);
```

```
mcb number(6,2);
minibal constant number(7,2):=1000.00;
fine number(6,2):=100.00;
begin
mano:=&mano;
select cur_bal into mcb from acct where acctno=mano;
if mcb<minibal then
update acct set cur_bal=cur_bal-fine where acctno=mano;
end if;
end;
/
```

## 7. Procedures and Functions

7.1 Write a procedure to add an amount of Rs.1000 for the employees whose salaries is greater than 5000 and who belongs to the deptno passed as an argument.

=>

```
create or replace procedure salary(deptid number) as
begin
    update emp set sal=sal+1000 where sal>5000 AND deptno=deptid;
end;
```

7.2 Write a PL/SQL block to update the salary of the employee with a 10% increase whose empno is to be passed as an argument for the procedure.

=>

```
create or replace procedure salary1(empid number) as
begin
    update emp set sal=sal+sal*(0.1) where empno=empid;
end;
```

7.3 Write a function to find the salary of the employee who is working in the deptno 20(to be passed as an argument).

=>

```
create or replace procedure get_sal(dept number) as
begin
    for s in (select * from emp where deptno = dept)
    loop
        dbms_output.put_line(s.sal);
    end loop;
end;
```

7.4 Write a function to find the nature of job of the employee whose deptno is 20(to be passed as an argument)

=>

```
create or replace procedure get_nature(dept number) as
begin
    for s in (select * from emp where deptno = dept)
    loop
        dbms_output.put_line(s.job);
    end loop;
end;
```

7.5 Write a PL/SQL block to obtain the department name of the employee who works for deptno 30.

=>

```
create or replace procedure dep_name(deptid number) as
begin
    select dept.dname from dept,emp where emp.deptno=dept.deptno;
end;
```

```
create or replace procedure get_nature(dept number) as
begin
    for s in (select * from emp where deptno = dept)
    loop
        dbms_output.put_line(s.job);
    end loop;
end;
```

/

**exec get\_nature(30);**

## 8. Triggers

8.1 Write a Trigger to ensure that DEPT TABLE does not contain duplicate of null values in DEPTNO column.

=>

CREATE OR RELPLACE TRIGGER trig1 before insert on DEPT for each row DECLARE a number;

BEGIN

if(:new.DEPTNO is Null) then

raise\_application\_error(-20001,'error:: DEPTNO cannot be null');

else

select count(\*) into a from DEPT where DEPTNO =:new.DEPTNO;

if(a=1) then

raise\_application\_error(-20002,'error:: cannot have duplicate DEPTNo ');

end if;

end if;

END;

8.2 Write a Trigger to carry out the following action: on deleting a deptno from dept table , all the records with that deptno has to be deleted from the emp table

=>

CREATE [OR REPLACE] TRIGGER trig2 After delete on DEPT FOR EACH ROW

BEGIN

DELETE FROM emp WHERE emp.deptno=:new.deptno;

END

8.3 Write a Trigger to carry out the following action: on deleting any records from the emp table,the same values must be inserted into the log table.

=>

CREATE TRIGGER trig3 AFTER DELETE ON emp FOR EACH ROW

BEGIN

INSERT INTO log(val1, val2, ...) VALUES (old.val1, old.val2, ...);

END;

