|  |  |  |  |
| --- | --- | --- | --- |
| **ENAME** | **SAL** | **JOB** | **DEPTNO** |
| SCOTT | 3000 | CLERK | 10 |
| KING | 5000 | MANAGER | 20 |

**EMP TABLE**

**TEMPP TABLE**

|  |  |
| --- | --- |
| **FIR** | **SEC** |
|  |  |

**#1**

delimiter //

create procedure abc()

begin

declare x int ;

select sal into x from emp

where ename = 'KING' ;

/\* processing , e.g. set hra = x\*0.4, etc \*/

insert into tempp values (x, 'KING') ;

end; //

delimiter ;

Select columnname into variablename from …….

Where …… ;

* X variable should have same datatype and same width as EMP table SAL column

**#2**

delimiter //

create procedure abc(y char (15))

begin

declare x int ;

select sal into x from emp

where ename = y;

/\* processing , e.g. set hra = x\*0.4, etc \*/

insert into tempp values (x, y) ;

end; //

delimiter ;

call abc (‘KING’) ;

**#3**

delimiter //

create procedure abc(

begin

declare x int ;

declare y char(15) ;

select sal, job into x, y from emp

where ename = ‘KING’;

/\* processing , e.g. set hra = x\*0.4, set y = lower(y) etc \*/

insert into tempp values (x, y) ;

end; //

delimiter ;

call abc () ;

**Decision making using IF statement**

**EMP**

|  |  |
| --- | --- |
| **ENAME** | **SAL** |
| KING | 5000 |

**#4**

Delimiter //

Create procedure abc()

Begin

Declare x int;

Select sal into x from emp where ename = ‘KING’ ;

If x > 4000 then

Insert into tempp values (x, ‘High sal’) ;

End if;

End ; //

Delimiter ;

If <condition> then

………………………. ;

………………………. ;

End if ;

**========================================================**

**EMP**

|  |  |
| --- | --- |
| **ENAME** | **SAL** |
| KING | 3000 |

**#5**

Delimiter //

Create procedure abc()

Begin

Declare x int;

Select sal into x from emp where ename = ‘KING’ ;

If x > 4000 then

Insert into tempp values (x, ‘High sal’) ;

Else

Insert into tempp values (x, ‘Low sal’) ;

End if;

End ; //

Delimiter ;

**#6**

Delimiter //

Create procedure abc()

Begin

Declare x int;

Select sal into x from emp where ename = ‘KING’ ;

If x > 4000 then

Insert into tempp values (x, ‘High sal’) ;

Else

If x < 4000 then

Insert into tempp values (x, ‘Low sal’) ;

Else

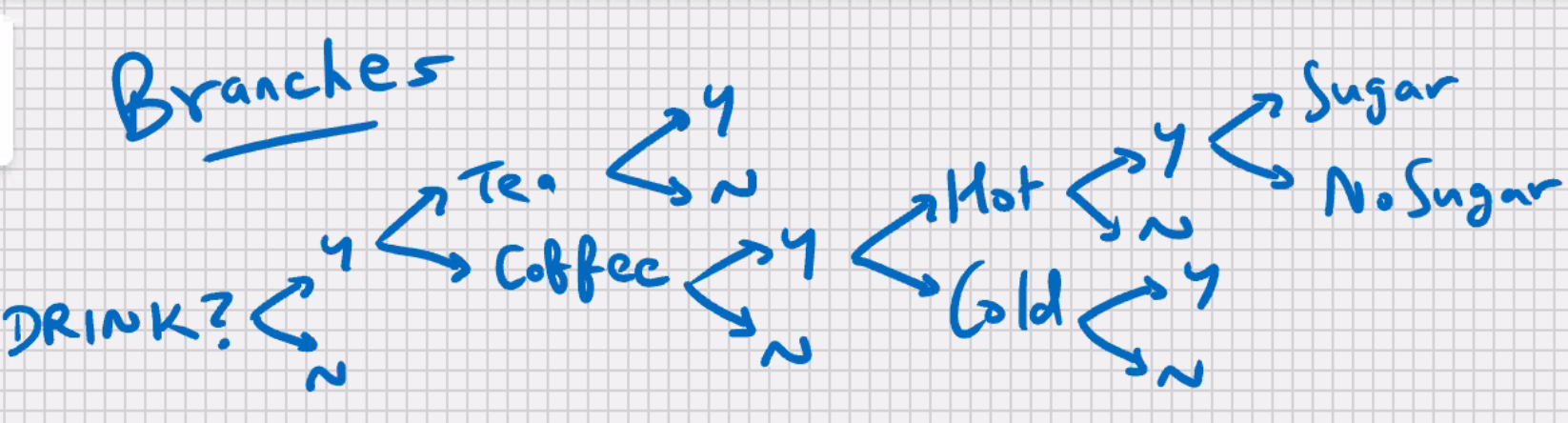
Insert into tempp values (x, ‘Medium sal’) ;

End if;

End if;

End ; //

Delimiter ;



**#7**

Delimiter //

Create procedure abc()

Begin

Declare x int;

Select sal into x from emp where ename = ‘KING’ ;

If x > 4000 then

Insert into tempp values (x, ‘High sal’) ;

Elseif x < 4000 then

Insert into tempp values (x, ‘Low sal’) ;

Else

Insert into tempp values (x, ‘Medium sal’) ;

End if;

End ; //

Delimiter ;

If x>4000 and x<5000 then

………………………. ;

………………………. ;

ElseIf x between 3000 and 4000 then

………………………. ;

………………………. ;

ElseIf round(x) < 3000 then

………………………. ;

………………………. ;

ElseIf <condition> then

………………………. ;

………………………. ;

End if;

**#8**

Delimiter //

Create procedure abc()

Begin

Declare x Boolean default TRUE;

If x then

Insert into tempp values (1, ‘Mumbai’) ;

End if;

End ; //

Delimiter ;

**#9**

Delimiter //

Create procedure abc()

Begin

Declare x Boolean default FALSE;

If not x then

Insert into tempp values (1, ‘Mumbai’) ;

End if;

End ; //

Delimiter ;

**MySQL-PL LOOPS**

* For repetitive /iterative processing

------------------------------------------------------------------------------------

**While loop**

* Check for some condition before entering the loop

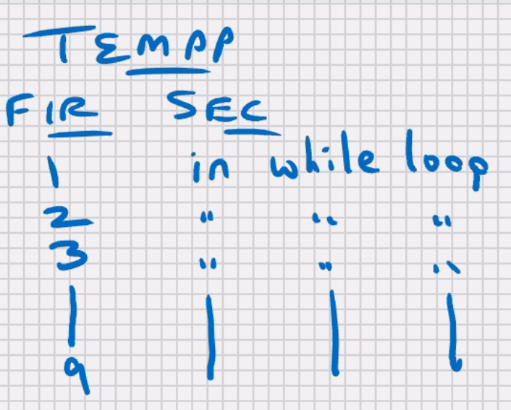
------------------------------------------------------------------------------------

While expression DO

……………………………. ;

……………………………. ;

End while ;

**#10**

Delimiter //

Create procedure abc ()

Begin

Declare x int default 1;

While x < 10 do

Insert into tempp values (x, ‘in while loop’);

Set x = x+1;

End while;

End ; //

Delimiter ;

**#11**

Delimiter //

Create procedure abc ()

Begin

Declare x int default 1;

While x < 10 do

Insert into tempp values (x, ‘in while loop’);

/\*Set x = x+1;\*/ ***infinite loop***

End while;

End ; //

Delimiter ;

**#12**

Delimiter //

Create procedure abc ()

Begin

Declare x int default 1;

Declare y int default 1;

While x < 10 do

While y < 10 do

Insert into tempp values (y, ‘in y loop’);

Set y = y+1;

End while;

Insert into tempp values (x, ‘in x loop’);

Set x = x+1;

End while;

End ; //

Delimiter ;

**#13**

Delimiter //

Create procedure abc ()

Begin

Declare x int default 1;

Declare y int default 1;

While x < 10 do

While **y < x** do

Insert into tempp values (y, ‘in y loop’);

Set y = y+1;

End while;

Insert into tempp values (x, ‘in x loop’);

Set x = x+1;

End while;

End ; //

Delimiter ;

Repeat Loop

* Similar to Do While loop
* There’s no coditon to enter the loop, but there is a condition to exit the loop
* It will execute at least once
* e.g., Outer join

repeat

………………….. ;

………………….. ;

Until expression

End repeat;

**#14**

Delimiter //

Create procedure abc ()

Begin

Declare x int default 1;

repeat

Insert into tempp values (x, ‘in loop’);

Set x = x+1;

Until x > 5

End repeat;

End ; //

Delimiter ;

Loop, Leave, and Iterate statement: -

* **Leave** statement allows you to exit the loop (similar to **break** statement of c programming)
* **Iterate** statement allows you skip the entire code under it and start a new iteration (similar to ‘**continue**’ statement of C programming)
* Loop statement allows you to execute a block of code repeatedly with an additional flexibility of using a loop label (you can give a name to the loop)

**#15**

Delimiter //

Create procedure abc ()

Begin

Declare x int default 1;

Pqr\_loop: loop

If x > 10 then

Leave pqr\_loop;

End if;

Set x = x+1;

If mod(x,2) != 0 then

Iterate pqr\_loop;

Else

Insert into tempp values (x, ’loop’) ;

End if;

End loop;

End ; //

Delimiter ;

**Global variables (session variables):-**

Set @x = 10;

Select @x from dual; -> 10

Set @x = @x + 1;

Select @x from dual; -> 11

**CURSORS (MOST IMPORTANT TOPIC)**

**EMP TABLE**

|  |  |  |  |
| --- | --- | --- | --- |
| **EMPNO** | **ENAME** | **SAL** | **DEPTNO** |
| **1** | **A** | **5000** | **1** |
| **2** | **B** | **6000** | **1** |
| **3** | **C** | **7000** | **1** |
| **4** | **D** | **9000** | **2** |
| **5** | **E** | **8000** | **2** |

**TEMPP TABLE**

|  |  |
| --- | --- |
| **FIR** | **SEC** |
|  |  |

* Present in all RDBMS, some DBMS and some of the programming languages and some front-ends also

Create table emp (

Empno int,

Ename varchar (15),

Sal int,

Deptno int

);

* Cursor is a type of a variable

Declare x int;

Declare hra flaot;

Select sal into x from emp where empno = 1;

Set hra = x\*0.4;

Insert into tempp values (hra, ‘HRA’);

Select sal into x from emp where empno = 2;

Set hra = x\*0.4;

Insert into tempp values (hra, ‘HRA’);

Select sal into x from emp where empno = 3;

Set hra = x\*0.4;

Insert into tempp values (hra, ‘HRA’);

Select sal into x from emp where empno = 4;

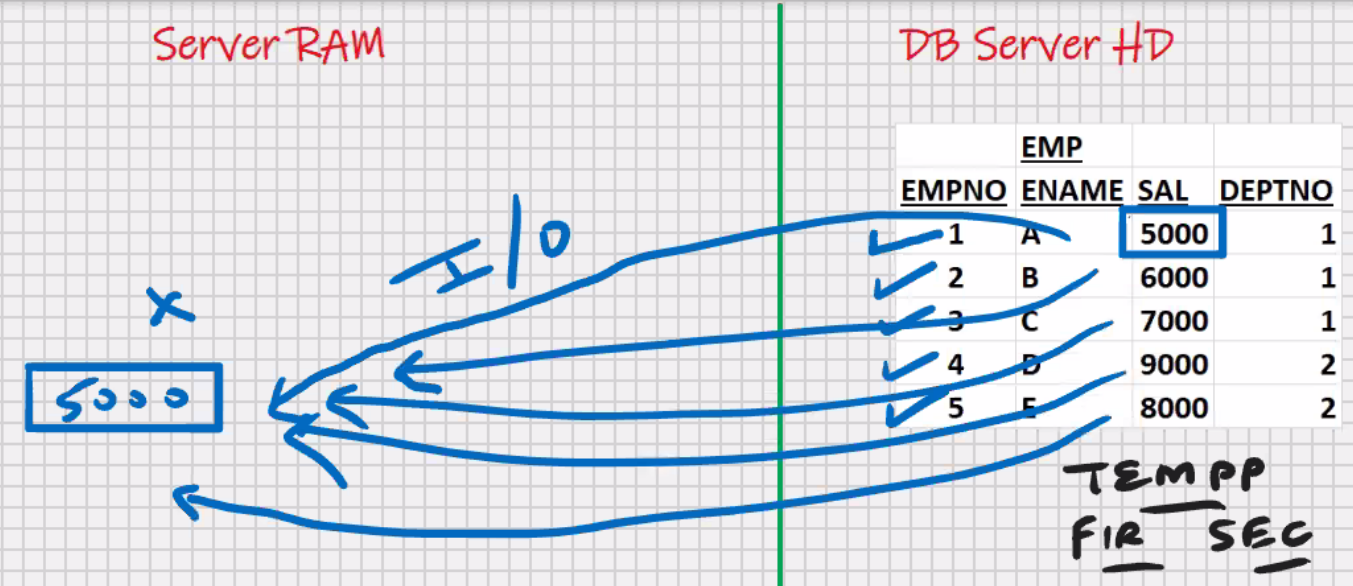
Set hra = x\*0.4;

Insert into tempp values (hra, ‘HRA’);

Select sal into x from emp where empno = 5;

Set hra = x\*0.4;

Insert into tempp values (hra, ‘HRA’);



This is slow process

Declare y int default 1;

Declare x int;

Declare hra flaot;

While y < 6

Select sal into x from emp where empno = y;

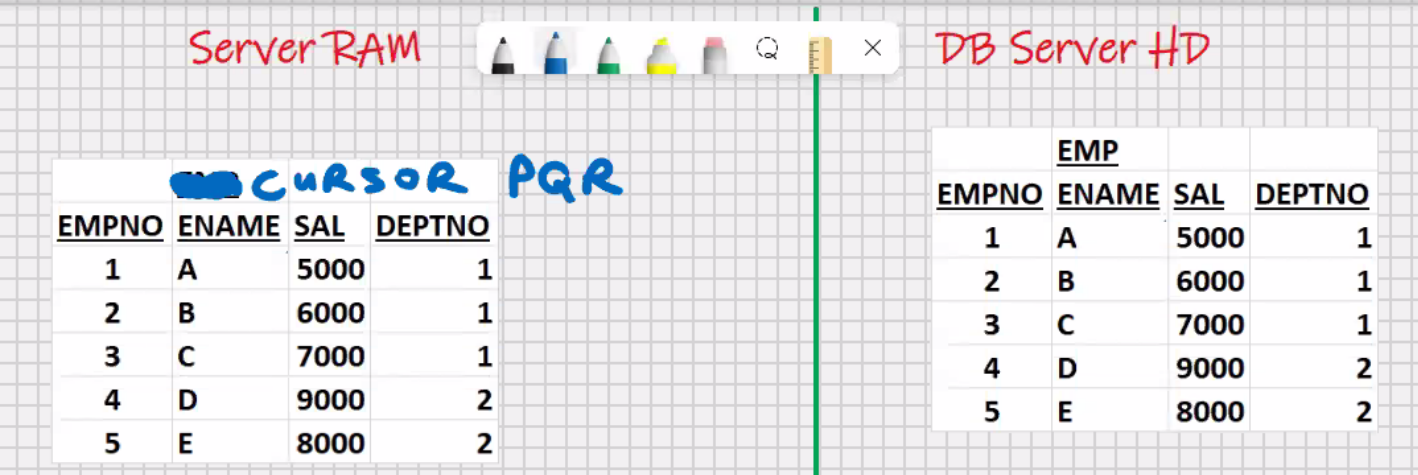
Set hra = x\*0.4;

Insert into tempp values (hra, ‘HRA’);

Set y = y+1

End while;

**Solution using Cursors**

****

* Used for storing **multiple rows**
* Used for processing storing multiple rows
* Used for handling multiple rows
* Used for storing the **data temporarily**
* Cursor is similar to a **2D array**
* Cursor is based on select statement

**Declare pqr cursor for select \* from emp;**

**Declare pqr cursor for select ename, sal from emp;**

**Declare pqr cursor for select ename, sal from emp where deptno=1;**

**Declare pqr cursor for select ename, sal from emp where deptno=1**

**Order by 1;**

**#16**

Delimiter //

Create procedure abc ()

Begin

Declare a int;

Declare b varchar(15);

Declare c int;

Declare d int;

Declare x int default 0;

Declare c1 cursor for select \* from emp; <- *cursor declaration/definition (at this point, the cursor does not contain any data)*

Open c1; <- *this will open the cursor c1, executes the select statement , and it will populate the cursor c1*

While x < 5 do

Fetch c1 into a, b, c, d; *<-Fetches the next row*

/\*processing, e.g. set hra = c\*0.4, etc \*/

Insert into tempp values (a, b);

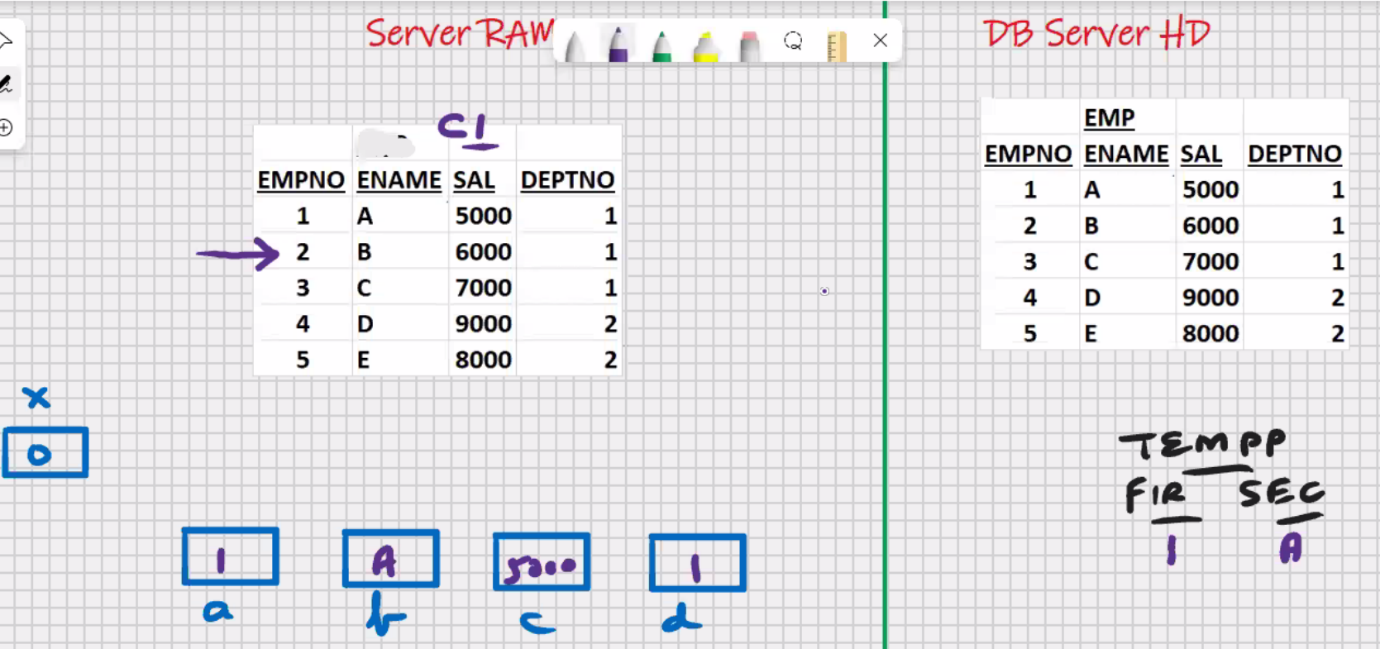
Set x = x+1;

End while;

Close c1; *<-this will close the cursor c1 and free the RAM*

End ; //

Delimiter ;

****

* Cursor is read only variable
* The data that is present inside the cursor, it cannot be manipulated
* You will have to fetch 1 row at a time into some intermediate ‘a’,’b’,’c’,’d’ variables, and do your processing with those variables
* You can only fetch sequentially (top to bottom)
* In MySQL you cannot fetch backwards in a oracle
* You can only fetch 1 row at a time

**#17**

Delimiter //

Create procedure abc ()

Begin

Declare a int;

Declare b varchar(15);

Declare c int;

Declare d int;

Declare x int default 0;

Declare c1 cursor for select \* from emp;

Open c1;

While **x < 3** do

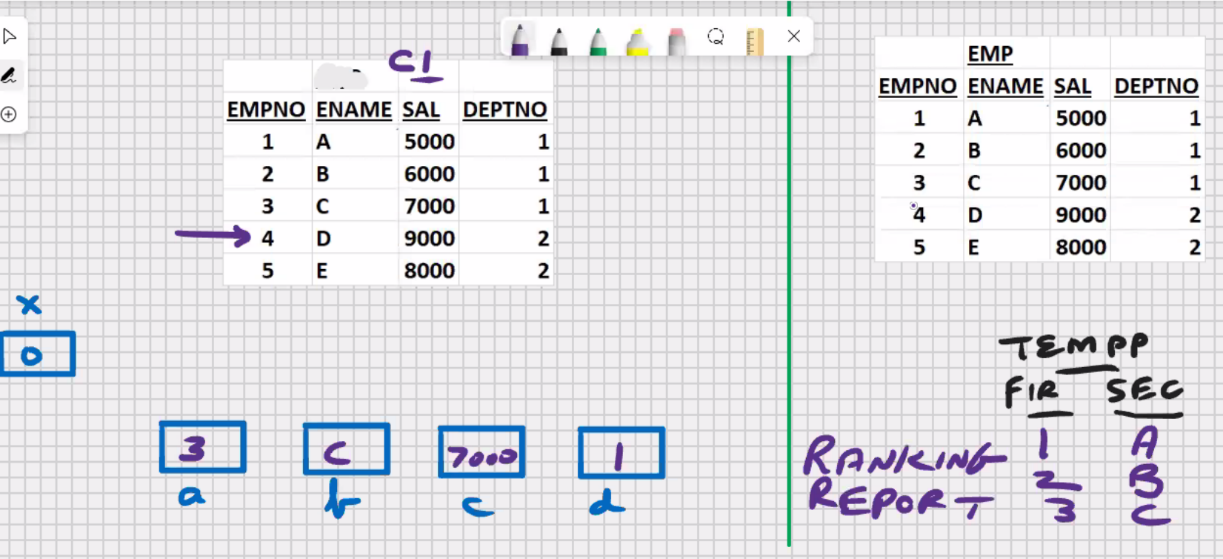
Fetch c1 into a, b, c, d;

Insert into tempp values (a, b);

Set x = x+1;

End while;

Close c1;

****End ; //

Delimiter ;

**#18**

Delimiter //

Create procedure abc ()

Begin

Declare a int;

Declare b varchar(15);

Declare c int;

Declare d int;

Declare x int default 0;

Declare c1 cursor for select \* from emp;

Open c1;

While **x < 10** do

Fetch c1 into a, b, c, d; -> ERROR when x=5

Insert into tempp values (a, b);

Set x = x+1;

End while;

Close c1;

End ; //

Delimiter ;

**#19**

Delimiter //

Create procedure abc ()

Begin

Declare a int;

Declare b varchar(15);

Declare c int;

Declare d int;

Declare x int default 0;

Declare y int;

Declare c1 cursor for select \* from emp order by sal desc;

Select count (\*) into y from emp;

Open c1;

While **x < y** do

Fetch c1 into a, b, c, d;

Insert into tempp values (a, b);

Set x = x+1;

End while;

Close c1;

End; //

Delimiter;

* Declare a continue handler for not found event (exception)
* Not found is a cursor attribute, it returns a Boolean true value if the last fetch was unsuccessful, and false value if the last fetch was successful

**#20**

Delimiter //

Create procedure abc ()

Begin

Declare a int;

Declare b varchar(15);

Declare c int;

Declare d int;

**Declare y int default 0**;

Declare c1 cursor for select \* from emp;

**Declare continue handler for not found set y = 1;**

Open c1;

Cursor\_c1\_loop: loop

Fetch c1 into a, b, c, d;

If y = 1 then

Leave cursor\_c1\_loop;

End if;

Insert into tempp values (a, b);

End loop cursor\_c1\_loop;

/\* Close c1; \*/ *<- closing is optional if it is last line*

End; //

Delimiter;

* You cannot reset the cursor pointer

Open c1;

Open c1;

* You cannot same twice, you will get an error that cursor is already open, you will have to close the cursor before you reopen it

**To reset the cursor pointer: -**

Close c1;

Open c1;

**#21**

Delimiter //

Create procedure abc ()

Begin

Declare a varchar(15);

Declare b int;

**Declare y int default 0**;

Declare c1 cursor for select ename, sal from emp;

**Declare continue handler for not found set y = 1;**

Open c1;

Cursor\_c1\_loop: loop

Fetch c1 into a, b;

If y = 1 then

Leave cursor\_c1\_loop;

End if;

Insert into tempp values (b, a);

End loop cursor\_c1\_loop;

Close c1;

End; //

Delimiter;

**#22**

Delimiter //

Create procedure abc ()

Begin

Declare a int ;

Declare b varchar(15);

Declare c int;

Declare d int;

Declare y int default 0;

Declare c1 cursor for select \* from emp where deptno = 1;

Declare continue handler for not found set y = 1;

Open c1;

Cursor\_c1\_loop: loop

Fetch c1 into a, b, c, d;

If y = 1 then

Leave cursor\_c1\_loop;

End if;

Insert into tempp values (c, b);

End loop cursor\_c1\_loop;

Close c1;

End; //

Delimiter;

**#23**

Delimiter //

Create procedure abc (**dd int**)

Begin

Declare a int ;

Declare b varchar(15);

Declare c int;

Declare d int;

Declare y int default 0;

Declare c1 cursor for select \* from emp where deptno = **dd**;

Declare continue handler for not found set y = 1;

Open c1;

Cursor\_c1\_loop: loop

Fetch c1 into a, b, c, d;

If y = 1 then

Leave cursor\_c1\_loop;

End if;

Insert into tempp values (c, b);

End loop cursor\_c1\_loop;

Close c1;

End; //

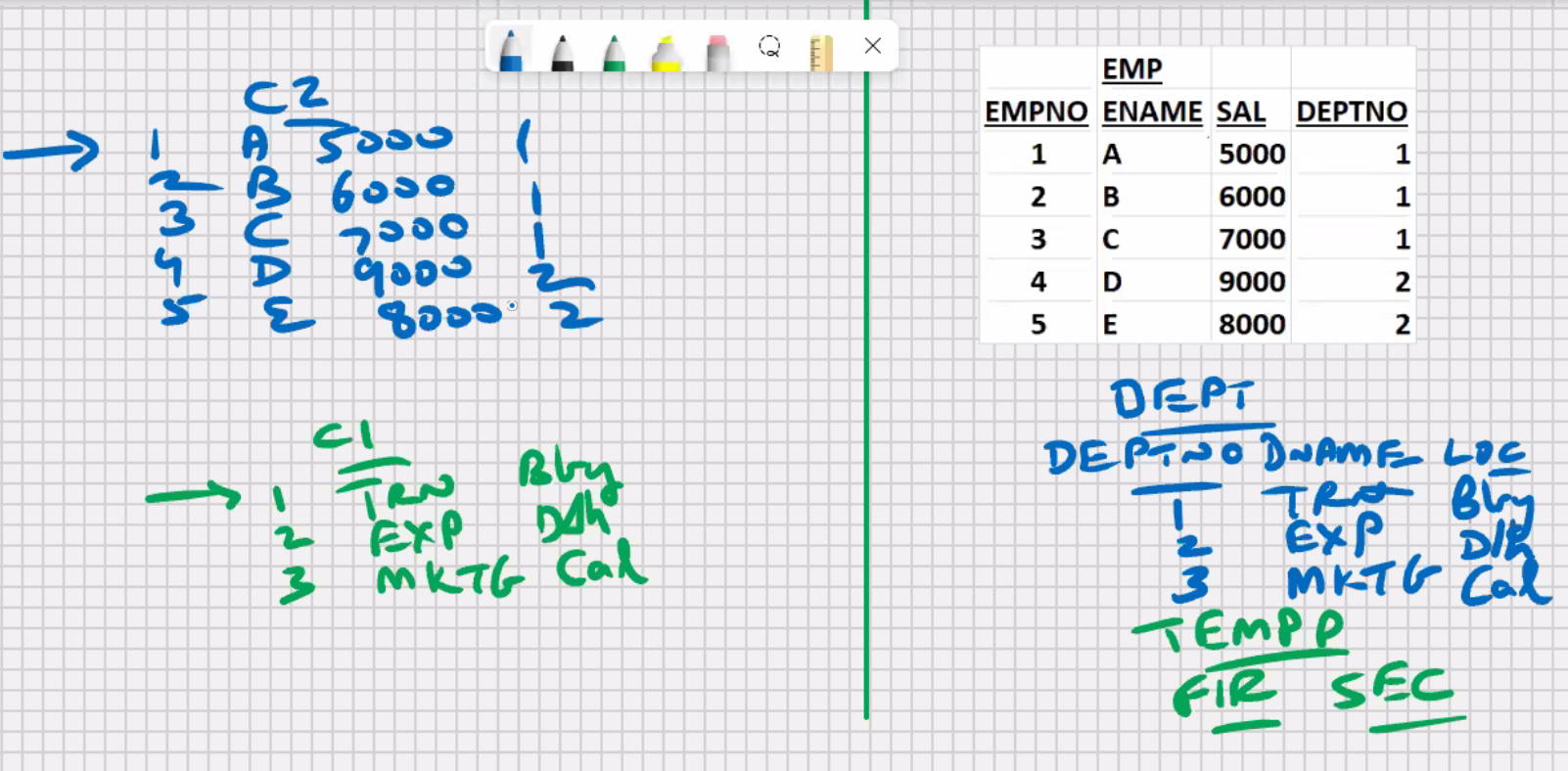
Delimiter;

Call abc (1);

Call abc (2);

**DEPT TABLE**

|  |  |  |
| --- | --- | --- |
| DEPTNO | DNAME | LOC |
| 1 | TRN | MUMBAI |
| 2 | EXP | DELHI |
| 3 | MRKG | CALCUTTA |



**#24**

Delimiter //

Create procedure abc ()

Begin

Declare a int ;

Declare b varchar(15);

Declare c int;

Declare d int;

Declare y int default 0;

Declare c1 cursor for select \* from dept;

Declare c2 cursor for select \* from emp;

Declare continue handler for not found set y = 1;

…………………………………. ;

Open c1;

Open c2;

…………………………………. ;

Close c1;

Close c2;

End; //

Delimiter;

* No upper limit on the number of cursors that you can open at a time
* The only restriction would be size of server RAM (it should be large enough to manage so much data)

**#25**

Delimiter //

Create procedure abc ()

Begin

Declare a int;

Declare b varchar (15);

Declare c int;

Declare d int;

Declare y int default 0;

Declare c1 cursor for select empno, dname from dept emp, dept where dept.deptno = emp.deptno ;

…………………………………. ;

Open c1;

…………………………………. ;

Close c1;

End; //

Delimiter;