## Hierarchical Model

***INTRODUCTION***

## Database Management System

This model is like a hierarchical tree structure, used to construct a hierarchy of records in the form of nodes and branches. The data elements present in the structure have Parent-Child relationship. Closely related information in the parent-child structure is stored together as a logical unit. A parent unit may have many child units, but a child is restricted to have only one parent.

## The drawbacks of this model are:

The hierarchical structure is not flexible to represent all the relationship proportions, which occur in the real world.

It cannot demonstrate the overall data model for the enterprise because of the non-availability of actual data at the time of designing the data model.

It cannot represent the Many-to-Many relationship.

## Network Model

It supports the One-To-One and One-To-Many types only. The basic objects in this model are Data Items, Data Aggregates, Records and Sets.

It is an improvement on the Hierarchical Model. Here multiple parent-child relationships are used. Rapid and easy access to data is possible in this model due to multiple access paths to the data elements.

## Relational Model

Does not maintain physical connection between relations Data is organized in terms of rows and columns in a table

The position of a row and/or column in a table is of no importance The intersection of a row and column must give a single value

## Features of an RDBMS

The ability to create multiple relations and enter data into them an attractive query language

Retrieval of information stored in more than one table

An RDBMS product has to satisfy at least seven of the 12 rules of Codd to be accepted as a full- fledged RDBMS.

## Relational Database Management System

RDBMS is acronym for Relation Database Management System. Dr. E. F. Codd first introduced the Relational Database Model in 1970. The Relational model allows data to be represented in a simple row- column. Each data field is considered as a column and each record is considered as a row. Relational Database is more or less similar to Database Management System. In relational model there is relation between their data elements. Data is stored in tables. Tables have columns, rows and names. Tables can be related to each other if each has a column with a common type of information. The most famous RDBMS packages are Oracle, Sybase and Informix.

Simple example of Relational model is as follows:

## Student Details Table

|  |  |  |  |
| --- | --- | --- | --- |
| Roll\_no | Sname | S\_Address | |
| 1 | Rahul | Satelite |  |
| 2 | Sachin | Ambawadi | |
| 3 | Saurav | Naranpura | |
| **Student Marksheet Table** | | | |
| Rollno | Sub1 | Sub2 | Sub3 |
| 1 | 78 | 89 | 94 |
| 2 | 54 | 65 | 77 |
| 3 | 23 | 78 | 46 |

Here, both tables are based on student’s details. Common field in both tables is Rollno. So we can say both tables are related with each other through Rollno column.

## Degree of Relationship

One to One (1:1)

One to Many or Many to One (1:M / M: 1) Many to Many (M: M)

The Degree of Relationship indicates the link between two entities for a specified occurrence of each.

## One to One Relationship: (1:1) 1 1

**Student Has Roll No.**

One student has only one Rollno. For one occurrence of the first entity, there can be, at the most one related occurrence of the second entity, and vice-versa.

## One to Many or Many to One Relationship: (1:M/M: 1) 1 M

**Course Contains Students**

As per the Institutions Norm, One student can enroll in one course at a time however, in one course, there can be more than one student.

For one occurrence of the first entity there can exist many related occurrences of the second entity and for every occurrence of the second entity there exists only one associated occurrence of the first.

## Many to Many Relationship: (M:M) M M

**Students Appears Tests**

The major disadvantage of the relational model is that a clear-cut interface cannot be determined. Reusability of a structure is not possible. The Relational Database now accepted model on which major database system are built.

Oracle has introduced added functionality to this by incorporated object-oriented capabilities. Now it is known is as Object Relational Database Management System (ORDBMS). Object- oriented concept is added in Oracle8.

Some basic rules have to be followed for a DBMS to be relational. They are known as Codd’s rules, designed in such a way that when the database is ready for use it encapsulates the relational theory to its full potential. These twelve rules are as follows.

## E.F. Codd Rules

1. **The Information Rule**

All information must be store in table as data values.

## The Rule of Guaranteed Access

Every item in a table must be logically addressable with the help of a table name.

## The Systematic Treatment of Null Values

The RDBMS must be taken care of null values to represent missing or inapplicable information.

## The Database Description Rule

A description of database is maintained using the same logical structures with which data was defined by the RDBMS.

## Comprehensive Data Sub Language

According to the rule the system must support data definition, view definition, data manipulation, integrity constraints, authorization and transaction management operations.

## The View Updating Rule

All views that are theoretically updateable are also updateable by the system.

## The Insert and Update Rule

This rule indicates that all the data manipulation commands must be operational on sets of rows having a relation rather than on a single row.

## The Physical Independence Rule

Application programs must remain unimpaired when any changes are made in storage representation or access methods.

## The Logical Data Independence Rule

The changes that are made should not affect the user’s ability to work with the data. The change can be splitting table into many more tables.

## The Integrity Independence Rule

The integrity constraints should store in the system catalog or in the database.

## The Distribution Rule

The system must be access or manipulate the data that is distributed in other system.

## The Non-subversion Rule

If a RDBMS supports a lower level language then it should not bypass any integrity constraints defined in the higher level.

* 1. **Database Schema for a customer-sale scenario** Customer(**Cust\_id : integer**, cust\_name : string) Item(**item\_id : integer**, item\_name : string, price : integer)

Sale(**bill\_no : integer**, bill\_date : date, **cust\_id : integer, item\_id: integer**, qty\_sold : integer)

For the above schema, perform the following

* + 1. Create the tables with the appropriate integrity constraints

SQL> create table customer(cust\_id int primary key, cust\_name varchar(20)); Table created.

SQL> create table item(item\_id int primary key, item\_name varchar(20), price int); Table created.

SQL>create table sale(bill\_no int primary key, bill\_date date,cust\_id int references customer, item\_id int references item, qty\_sold int);

Table created.

* + 1. Insert around 10 records in each of the tables SQL> insert into customer values ( 1201 , ’tara’ ); 1 row created.

SQL> insert into customer values ( 1202 , ’sanyu’ ); 1 row created.

SQL> insert into customer values ( 1203 , ’aadarsh’ ); 1 row created.

SQL> insert into customer values ( 1204 , ’randhir’ ); 1 row created.

SQL> insert into customer values ( 1205 , ’kaif’ ); 1 row created.

SQL> insert into customer values ( 1206 , ’pari’ );

1 row created.

SQL> insert into customer values ( 1207 , ’rahul’ ); 1 row created.

SQL> insert into customer values ( 1208 , ’tushar’ ); 1 row created.

SQL> insert into customer values ( 1209 , ’shakthi’ ); 1 row created.

SQL> insert into customer values ( 1210 , ’diya’ ); 1 row created.

SQL>select \* from customer; CUST\_ID CUST\_NAME

|  |  |
| --- | --- |
| 1201 | tara |
| 1202 | sanyu |
| 1203 | aadarsh |
| 1204 | randhir |
| 1205 | kaif |
| 1206 | pari |
| 1207 | rahul |
| 1208 | tushar |
| 1209 | shakthi |
| 1210 | diya |

SQL> insert into item values ( 1301, ‘silk’, 60); 1 row created.

SQL> insert into item values ( 1302, ‘cakes’, 30); 1 row created.

SQL> insert into item values ( 1303, ‘apples’, 120); 1 row created.

SQL> insert into item values ( 1304, ‘dal’, 250); 1 row created.

SQL> insert into item values ( 1305, ‘celebrations’, 500); 1 row created.

SQL> insert into item values ( 1306, ‘rin’, 240); 1 row created.

SQL> insert into item values ( 1307, ‘soap’, 90); 1 row created.

SQL> insert into item values ( 1308, ‘creams’, 160); 1 row created.

SQL> insert into item values ( 1309, ‘milk’, 140); 1 row created.

SQL> insert into item values ( 1310, ‘maggie’, 250); 1 row created.

SQL> select \* from item;

ITEM\_ID ITEM\_NAME PRICE

|  |  |  |
| --- | --- | --- |
| 1301 | silk | 60 |
| 1302 | cakes | 30 |
| 1303 | apples | 120 |
| 1304 | dal | 250 |
| 1305 | celebrations | 500 |
| 1306 | rin | 240 |
| 1307 | soap | 90 |

|  |  |  |
| --- | --- | --- |
| 1308 | creams | 160 |
| 1309 | milk | 140 |
| 1310 | maggie | 250 |

SQL> insert into sale values ( 1401, ’2017-jan-01’, 1201, 1301, 3);

1 row created.

SQL> insert into sale values ( 1402, ’2017-jan-04’, 1202, 1302, 5);

1 row created.

SQL> insert into sale values ( 1403, ’2017-jan-01’, 1203, 1303, 2);

1 row created.

SQL> insert into sale values ( 1404, ’2017-jan-01’, 1204, 1304, 1);

1 row created.

SQL> insert into sale values ( 1405, ’2017-jan-11’, 1205, 1305, 4);

1 row created.

SQL> insert into sale values ( 1406, ’2017-jan-18’, 1206, 1306, 1);

1 row created.

SQL> insert into sale values ( 1407, ’2017-jan-10’, 1207, 1307, 3);

1 row created.

SQL> insert into sale values ( 1408, ’2017-jan-21’, 1208, 1308, 1);

1 row created.

SQL> insert into sale values ( 1409, ’2017-jan-17’, 1209, 1309, 5);

1 row created.

SQL> insert into sale values ( 1410, ’2017-jan-14’, 1210, 1310, 2);

1 row created.

BILL\_NO BILL\_DATE CUST\_ID ITEM\_ID QTY\_SOLD 1401 2017-JAN-01 1201 1301 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1402 | 2017-JAN-04 | 1202 | 1302 | 5 |
| 1403 | 2017-JAN-01 | 1203 | 1303 | 2 |
| 1404 | 2017-JAN-10 | 1204 | 1304 | 1 |
| 1405 | 2017-JAN-11 | 1205 | 1305 | 4 |
| 1406 | 2017-JAN-18 | 1206 | 1306 | 1 |
| 1407 | 2017-JAN-10 | 1207 | 1307 | 3 |
| 1408 | 2017-JAN-21 | 1208 | 1308 | 1 |
| 1409 | 2017-JAN-17 | 1209 | 1309 | 5 |
| 1410 | 2017-JAN-14 | 1210 | 1310 | 2 |

* + 1. List all the bills for the current date with the customer names and item numbers

SQL>select cust\_name, item\_id, bill\_no from customer c,sale s where s.cust\_id= c.cust\_id and bill\_date= ’2017-Jan-21’;

CUST\_NAME ITEM\_ID BILL\_NO

Tushar 1308 1408

* + 1. List the total Bill details with the quantity sold,price of the item and the final amount

SQL> select qty\_sold, price, i.price\*qty\_sold as total from item i, sale s where i.item\_id= s.item\_id; QTY\_SOLD PRICE TOTAL

|  |  |  |
| --- | --- | --- |
| 3 | 60 | 180 |
| 5 | 30 | 150 |
| 2 | 120 | 240 |
| 1 | 250 | 250 |
| 4 | 500 | 2000 |
| 1 | 240 | 240 |
| 3 | 90 | 270 |
| 1 | 160 | 160 |

|  |  |  |
| --- | --- | --- |
| 5 | 140 | 700 |
| 2 | 250 | 500 |

1. List the details of the customer who have bought a product which has a price>200

SQL> select c.cust\_id, cust\_name, bill\_no from customer c, sale s, item i where s.cust\_id= c.cust\_id and i.item\_id= s.item\_id and price>200;

CUST\_ID CUST\_NAME BILL\_NO

|  |  |  |
| --- | --- | --- |
| 4 | randhir | 1404 |
| 5 | kaif | 1405 |
| 6 | pari | 1406 |
| 10 | diya | 1410 |

1. Give a count of how many products have been bought by each customer

SQL> select count(i.item\_id) from item I,customer c, sales s where s.cust\_id= c.cust\_id and i.item\_id= s.item\_id group by cust\_name;

COUNT(I.ITEM\_ID) 1

1

1

1

1

1

1

1

1

1

1. Give a list of products bought by a customer having cust\_id as 5

SQL> select i.item\_name from customer c, sale s, item i where s.cust\_id= c.cust\_id and i.item\_id= s.item\_id and c.cust\_id=1205;

ITEM\_NAME

Celebrations

1. List the item details which are sold as of today

SQL> select i.item\_id, i.item\_nme from item I, sale s where i.item\_id=s.item\_id and bill\_date= ’2017-jan-21’;

ITEM\_ID ITEM\_NAME

1308 creams

1. Create a view which lists out the bill\_no, bill\_date, cust\_id, item\_id, price, qty\_sold, amount

SQL> create view v\_cus as select s.bill\_no, s.bill\_date, c.cust\_id, i.item\_id, i.price, s.qty\_sold, i.price\*qty\_sold as amount from item I, customer c, sales s where s.cust\_id= c.cust\_id and i.item\_id= s.item\_id;

View created.

SQL> select \* from v\_cus;

BILL\_NO BILL\_DATE CUST\_ID ITEM\_ID PRICE QTY\_SOLD AMOUNT

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1401 | 01-JAN-17 | 1201 | 1301 | 3 | 60 | 180 |
| 1402 | 04-JAN-17 | 1202 | 1302 | 5 | 30 | 150 |
| 1403 | 01-JAN-17 | 1203 | 1303 | 2 | 120 | 240 |
| 1404 | 10-JAN-17 | 1204 | 1304 | 1 | 250 | 250 |
| 1405 | 11-JAN-17 | 1205 | 1305 | 4 | 500 | 2000 |
| 1406 | 18-JAN-17 | 1206 | 1306 | 1 | 240 | 240 |
| 1407 | 10-JAN-17 | 1207 | 1307 | 3 | 90 | 270 |
| 1408 | 21-JAN-17 | 1208 | 1308 | 1 | 160 | 160 |
| 1409 | 17-JAN-17 | 1209 | 1309 | 5 | 140 | 700 |
| 1410 | 14-JAN-17 | 1210 | 1310 | 2 | 250 | 500 |

1. Create a view which lists the daily sales date wise for the last one week

SQL>crete view v\_date as select s.bill\_no, s.bill\_date, c.cust\_id, i.item\_id, i.price, s.qty\_sold, i.price\*qty\_sold as amount from item I, customer c, sales s where s.cust\_id= c.cust\_id and i.item\_id= s.item\_id and bill\_date between ’2017-jan-15’ and ’2017-jan-21’;

BILL\_NO BILL\_DATE CUST\_ID ITEM\_ID PRICE QTY\_SOLD AMOUNT

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1406 | 2017-JAN-18 | 1206 | 1306 | 1 | 240 | 240 |
| 1408 | 2017-JAN-21 | 1208 | 1308 | 1 | 160 | 160 |
| 1409 | 2017-JAN-17 | 1209 | 1309 | 5 | 140 | 700 |

## Database Schema for a Student Library scenario

**Student**(**Stud\_no : integer,** Stud\_name: string) **Membership**(**Mem\_no: integer**, **Stud\_no: integer**) **Book**(**book\_no: integer**, book\_name:string, author: string) **Iss\_rec**(**iss\_no:integer**, iss\_date: date, **Mem\_no: integer, book\_no: integer**)

For the above schema, perform the following—

* + 1. Create the tables with the appropriate integrity constraints
    2. Insert around 10 records in each of the tables
    3. List all the student names with their membership numbers
    4. List all the issues for the current date with student and Book names
    5. List the details of students who borrowed book whose author is CJDATE
    6. Give a count of how many books have been bought by each student
    7. Give a list of books taken by student with stud\_no as 5
    8. List the book details which are issued as of today
    9. Create a view which lists out the iss\_no, iss \_date, stud\_name, book name
    10. Create a view which lists the daily issues-date wise for the last one week

SQL>create table student(stud\_no int primary key,stud\_name varchar(15)); SQL>desc student;

Name Null? Type

……………………………………………………………………………………….

. STUD\_NO NOT NULL INT STUD\_NAME VARCAHR(15)

Valid Test Data:

SQL>insert into student values(&stud\_no,’&stud\_name’); SQL>select \* from student;

STUD\_NO STUD\_NAME

....................................................................

|  |  |
| --- | --- |
| 508 | HARISH |
| 513 | BALAJI |
| 518 | RAKESH |
| 524 | PAVAN |
| 534 | JOYCE |

SQL>create table membership(mem\_no int primary key,stud\_no int references student); SQL>desc membership;

Name Null? Type

……………………………………………………………………………………………………. MEM\_NO NOT NULL INT

STUD\_NO INT

SQL>insert into membership values(&mem\_no,&stud\_no); Enter value for mem\_no:5440

Enter value for stud\_no:510 SQL>select \* from membership;

MEM\_NO STUD\_NO

………………………………………………………………………..

|  |  |
| --- | --- |
| 5440 | 513 |
| 5441 | 508 |
| 5442 | 518 |
| 5443 | 534 |
| 5444 | 524 |

SQL>create table book(book\_no int primary key,book\_name varchar(20),author varchar(20)); SQL>desc book;

Name Null? Type

……………………………………………………………………………………….. BOOK\_NO NOT NULL INT

BOOK\_NAME VARCHAR(20)

AUTHOR VARCHAR(20)

SQL>insert into book values(&book\_no,’&book\_name’,’&author’); SQL>select \* from book;

BOOK\_NO BOOK\_NAME AUTHOR

………………………………………………………………………………………………..

|  |  |  |
| --- | --- | --- |
| 9123 | DBMS | Rama Krishna |
| 2342 | JAVA | Robett wilkins |
| 4523 | Fearless tales | Alfred |
| 8723 | my ambition | Harish |
| 7821 | Harry Potter | JK Rowling |

SQL>create table lss\_rec(iss\_no int primary key,iss\_date date,mem\_no int references membership,book\_no int references book);

SQL>desc iss\_rec;

Name Null? Type

………………………………………………………………………………………………………

|  |  |  |
| --- | --- | --- |
| ISS\_NO | NOT NULL | INT |
| ISS\_DATE |  | DATE |
| MEM\_NO |  | INT |
| BOOK\_NO  SQL>select \* from iss\_rec; |  | INT |

ISS\_NO ISS\_DATE MEM\_NO BOOK\_NO

…………………………………………………………………………………………………

|  |  |  |  |
| --- | --- | --- | --- |
| 43 | 2006-JAN-05 | 5443 | 4523 |
| 81 | 2005-DEC-28 | 5441 | 8723 |
| 22 | 2005-DEC-08 | 5440 | 7821 |
| 53 | 2006-JAN-07 | 5442 | 9123 |
| 35 | 2006-JAN-06 | 5444 | 2342 |

1. List all the student names with their membership numbers

SQL> select s.studname, m.memno from student s, membership m where m.studno=s.studno; STUDNAME MEMNO

|  |  |
| --- | --- |
| Abhijeet | 1001 |
| arun | 1002 |
| arvind | 1003 |

ashish 1004

ashwin 1005

1. List all the issues for the current date with student and Book names

SQL> select i.issno, s.studname, b.bookname from iss\_rec I, membership m, student s, book b where i.memno=m.memno and m.studno=s.studno and i.issdate=to\_char(sysdate);

ISSNO STUDNAME BOOKNAME

13 arvind P&S

1. List the details of students who borrowed book whose author is CJDATE

SQL> select \* from student where studno in(select studno from membership where memno in (select memno from iss\_rec where bookno in(select bookno from book where author=’CJDATE’)));

STUDNO STUDNAME

505 ashwin

1. Give a count of how many books have been bought by each student

SQL> select s.studno, count(i.bookno) from student s.membership m, book b, iss\_rec iwhere s.studno=m.studno and b.bookno=i.bookno group by s.studno;

STUDNO COUNT(I.BOOKNO)

|  |  |
| --- | --- |
| 501 | 5 |
| 502 | 5 |
| 503 | 5 |
| 504 | 5 |
| 505 | 5 |

1. Give a list of books taken by student with stud\_no as 5

SQL> select bookname from book where bookno in (select bookno from iss\_rec where memno in(select memno from membership where studno in(select studno from student where studno=5)));

BOOKNAME

NT

1. List the book details which are issued as of today

SQL>select b.bookno,b.bookname,b.author from book b,iss\_rec i where b.bookno=i.bookno and iss\_date=’2006-jan-17’;

BOOKNO BOOKNAME AUTHOR

9123 DBMS Ramakrishna

1. Create a view which lists out the iss\_no, iss \_date, stud\_name, book name

Create view d1 as(select i.iss\_no,i.iss\_date,s.studname,b.bookname from issrec i,book b,membership m,student s where s.studno=m.studno and .bookno=i.booknoo and m.memno=i.memno);

View created Select \* from d1;

ISS\_NO ISS\_DATE STUDNAME BOOKNAME

|  |  |  |  |
| --- | --- | --- | --- |
| 43 | 2006-JAN-05 | pavan | fearless tales |
| 81 | 2005-DEC-28 | harish | my ambition |
| 22 | 2005-DEC-08 | balaji | harry potter |
| 53 | 2006-JAN-07 | rakesh | DBMS |
| 35 | 2006-JAN-06 | joyice | JAVA |

(j) create view d2 as(select i.iss\_no,i.iss\_date,i.memno,i.bookno from iss\_rec i where iss\_date>’2005-dec- 08’;

ISS\_NO ISS\_DATE MEM\_NO BOOK\_NO

…………………………………………………………………………………………………

|  |  |  |  |
| --- | --- | --- | --- |
| 43 | 2006-JAN-05 | 5443 | 4523 |
| 81 | 2005-DEC-28 | 5441 | 8723 |
| 22 | 2005-DEC-08 | 5440 | 7821 |
| 53 | 2006-JAN-07 | 5442 | 9123 |
| 35 | 2006-JAN-06 | 5444 | 2342 |

* 1. **Database schema for employee-pay scenario** Employee (**empid: integer**, emp\_name: string) Department (**dept\_id: integer**, dept\_name: string)

Paydetails (**emp\_id: integer, dept\_id: integer**, basic: integer, deductions: integer, additions: integer, DOJ: date)

Payroll (**emp\_id: integer**, pay\_date: date)

* + 1. Create the tables with appropriate integrity constraints.

Create table employee (emp\_id int primary key, emp\_name varchar(20)); Table created

Create table department (dept\_id int primary key, dept\_name varchar(20)); Table created

Create table paydetails (emp\_id int references employee, dept\_id int references department, basic int, deductions int, additions int, DOJ date);

Table created

Create table payroll (emp\_id int references employee, pay\_date date);

Table created

* + 1. Insert around 10 records in each of the tables. Insert into employee values(1201,’rao’);

1 row created

Insert into employee values(1202,’prasad’); 1 row created

Insert into employee values(1203,’priya’); 1 row created

Insert into employee values(1204,’shilpa’); 1 row created

Insert into employee values(1205,’smitha’); 1 row created

Insert into employee values(1206,’ajay’); 1 row created

Insert into employee values(1207,’rahul’); 1 row created

Insert into employee values(1208,’tara’); 1 row created

Insert into employee values(1209,’akash’); 1 row created

Insert into employee values(1210,’sruthi’); 1 row created

Select \* from employee;

EMP\_ID EMP\_NAME

|  |  |
| --- | --- |
| 1201 | RAO |
| 1202 | PRASAD |
| 1203 | PRIYA |
| 1204 | SHILPA |

1205 SMITHA

1206 AJAY

1207 RAHUL

1208 TARA

1209 AKASH

1210 SRUTHI

10 rows selected

Insert into department values(1301,’it’); 1 row created

Insert into department values(1302,’exe’);

1 row created

Insert into department values(1303,’sales’); 1 row created

Insert into department values(1304,’pro’); 1 row created

Insert into department values(1305,’tech’); 1 row created

Insert into department values(1306,’arch’); 1 row created

Insert into department values(1307,’mech’); 1 row created

Insert into department values(1308,’ece’); 1 row created

Insert into department values(1309,’lab’); 1 row created

Insert into department values(1310,’mag’); 1 row created

Select \* from department;

DEPT\_ID DEPT\_NAME

|  |  |  |
| --- | --- | --- |
|  | 1301 | IT |
| 1302 | EXE |
| 1303 | SALES |
| 1304 | PRO |
| 1305 | TECH |
| 1306 | ARCH |
| 1307 | MECH |
| 1308 | ECE |
| 1309 | LAB |
| 1310 | MAG |
| 10 rows selected |  |  |

Insert into paydetails values (1201, 1303, 48960, 2380, 4699,’1998-jan-10’);

1 row created

Insert into paydetails values (1202, 1301, 98900, 4500, 6800,’1998-aug-20’);

1 row created

Insert into paydetails values (1203, 1308, 15000, 8000,’1999-jun-15’);

1 row created

Insert into paydetails values (1204, 1305, 65000, 2000, 3000,’2010-oct-18’);

1 row created

Insert into paydetails values (1205, 1306, 79000, 1000, 4000,’1998-jan-10’);

1 row created

Insert into paydetails values (1206, 1302, 85000, 3500, 1500,’19-may-12’);

1 row created

Insert into paydetails values (1207, 1304, 198000, 9000, 2000,’02-jan-97’);

1 row created

Insert into paydetails values (1208, 1309, 140000, 8000, 3500,’24-apr-98’);

1 row created

Insert into paydetails values (1209, 1307, 59000, 1300, 4000, ’01-jun-96’);

1 row created

Insert into paydetails values (1210, 1310, 260000, 12000, 5000, ’28-sep-93’);

1 row created

Select \* from paydetails;

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EMPID | DEPT\_ID | BASIC | DEDUCTIONS | ADDITIONS | DOJ |
| 1201 | 1303 | 48960 | 2380 | 4699 | 10-JAN-98 |
| 1202 | 1301 | 98900 | 4500 | 6800 | 20-AUG-98 |
| 1203 | 1308 | 15000 | 8000 | 2000 | 15-JUN-99 |
| 1204 | 1305 | 65000 | 2000 | 3000 | 18-OCT-10 |
| 1205 | 1306 | 79000 | 1000 | 4000 | 10-JAN-98 |
| 1206 | 1302 | 85000 | 3500 | 1500 | 19-MAY-12 |
| 1207 | 1304 | 198000 | 9000 | 2000 | 2-JAN-97 |
| 1208 | 1309 | 140000 | 8000 | 3500 | 24-APR-98 |
| 1209 | 1307 | 59000 | 1300 | 4000 | 1-JUN-96 |
| 1210 | 1310 | 260000 | 12000 | 5000 | 28-SEP-93 |

10 rows selected

Insert into payroll values(1201,’01-jan-17’);

1 row created

Insert into payroll values(1202,’02-jan-17’); 1 row created

Insert into payroll values(1203,’03-jan-17’); 1 row created

Insert into payroll values(1204,’07-jan-17’); 1 row created

Insert into payroll values(1205,’04-jan-17’); 1 row created

Insert into payroll values(1206,’02-jan-17’); 1 row created

Insert into payroll values(1207,’01-jan-17’); 1 row created

Insert into payroll values(1208,’01-jan-17’); 1 row created

Insert into payroll values(1209,’03-jan-17’); 1 row created

Insert into payroll values(1210,’10-jan-17’); 1 row created

Select \* from payroll;

EMP\_ID PAY\_DATE

|  |  |
| --- | --- |
| 1201 | 1-JAN-17 |
| 1202 | 2-JAN-17 |
| 1203 | 3-JAN-17 |
| 1204 | 7-JAN-17 |

|  |  |
| --- | --- |
| 1205 | 4-JAN-17 |
| 1206 | 2-JAN-17 |
| 1207 | 1-JAN-17 |
| 1208 | 1-JAN-17 |
| 1209 | 3-JAN-17 |
| 1210 | 10-JAN-17 |

* + 1. List all the employee details department wise

Select e.emp\_id, e.emp\_name, d.dept\_id, d.dept\_name, p.basic, p.deductions, p.additions, p.DOJ from employee e, department d, paydetails p where p.emp\_id=e.emp\_id and d.dept\_id=p.dept\_id;

EMPID EMPNAME DEPT\_ID DEPT\_NAME BASIC DEDUCTIONS ADDITIONS DOJ

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1205 | smitha | 1301 | it | 79000 | 1000 | 4000 | 20-nov-11 |
| 1204 | shilpa | 1302 | exe | 65000 | 2000 | 3000 | 18-oct-10 |
| 1203 | priya | 1303 | sales | 15000 | 8000 | 2000 | 15-jun-99 |
| 1204 | prasad | 1304 | pro | 98900 | 4500 | 68000 | 28-aug-02 |
| 1205 | rao | 1305 | tech | 48960 | 2380 | 4699 | 10-jan-98 |

* + 1. List all the employee names who joined after particular date

select emp\_name from employee e, paydetails p where e.emp\_id=p.emp\_id and DOJ>’1999-jun-15’; EMP\_NAME

Prasad Shilpa Smitha Ajay

* + 1. List the details of employees whose basic salary is between 10,000 and 20,000.

Select e.emp\_id, e.emp\_name, p.basic from employee e, paydetails p where e.emp\_id=p.emp\_id and basic between 10000 and 20000;

|  |  |  |
| --- | --- | --- |
| EMP\_ID | EMP\_NAME | BASIC |
| 1203 | priya | 15000 |

* + 1. Give a count of how many employees are working in each department.

Select d. dept\_name, count (e.emp\_id) from employee e, department d, paydetails p where e. emp\_id= p.emp\_id and d.dept\_id= p.dept\_id group by dept\_name;

DEPT\_NAME COUNT(E.EMP\_ID)

It 1

Exe 1

Sales 1

Pro 1

Tech 1

Arch 1

Mech 1

Ece 1

Lab 1

Mag 1

* + 1. give a names of the employees whose netsalary> 10,000.

Select e.emp\_name from employee e, paydetails p where e.emp\_id= p.emp\_id and p.basic>150000; EMP\_NAME

Rahul Sruthi

* + 1. List the details for an employee\_id= 5.

Select e.emp\_id, e.emp\_name, d.dept\_id, p.basic, p.DOJ, v. Pay\_date from employee e, department d, paydetails p, payroll v where p.emp\_id=e.emp\_id and d.dept\_id=p.dept\_id, e.emp\_id=v.emp\_id and e.emp\_id=1205;

EMPID EMPNAME DEPTID BASIC DOJ PAYDATE

1205 smitha 1306 79000 2011-nov-20 4-jan-17

* + 1. Create a view which lists out the emp\_name, department\_name, basic, deductions

create view v\_emp1 as select e.emp\_name, d.dept\_name, p.basic, p.deductions, p.basic-p.deductions as netsalary from employee e, department d, paydetails p where e.emp\_id=p.emp\_id and d.dept\_id=p.dept\_id;

view created

select \* from v\_emp1;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EMPNAME | DEPTNAME | BASIC | DEDUCTIONS | NETSALARY |
| RAO | SALES | 48960 | 2380 | 46580 |
| PRASAD | IT | 98900 | 4500 | 94400 |
| PRIYA | ECE | 15000 | 8000 | 14200 |
| SHILPA | TECH | 65000 | 2000 | 63000 |
| SMITHA | ARCH | 79000 | 1000 | 78000 |
| AJAY | EXE | 85000 | 3500 | 8500 |
| RAHUL | PRO | 198000 | 9000 | 189000 |
| TARA | LAB | 140000 | 8000 | 132000 |
| AKASH | MECH | 59000 | 1300 | 57700 |
| SRUTHI | MAG | 260000 | 12000 | 248000 |

* + 1. Create a view which lists the emp\_name and his netsalary.

create view v\_emp2 as select e.emp\_name, p.basic-p.deductions as netsalary from employee e, paydetails p where e.emp\_id=p.emp\_id;

view created

select \* from v\_emp2;

EMP\_NAME NETSALARY RAO 46580

PRASAD 94400

|  |  |
| --- | --- |
| PRIYA | 14200 |
| SHILPA | 63000 |
| SMITHA | 78000 |
| AJAY | 8500 |
| RAHUL | 189000 |
| TARA | 132000 |
| AKASH | 57700 |
| SRUTHI | 248000 |

* 1. **Database schema for video library scenario** Customer(**custno:integer**, custname:varchar[20]) Membership(**memno:integer**, **custno:integer**) Cassette(**cassno:integer**, cassname:string,language:string)

Iss\_rec(issno :integer primary key,issdate:date,**memno:integer,cassno: integer**)

1. Create the tables with appropriate integrity constraints. Create table customer(custno int primary key ,custname char(20));

Table created.

Create table membership(memno int primary key , custno int references customer); Table created.

Create table cassette(cassno int primary key ,cassname char(20) ,language char(20)); Table created.

Create table iss\_rec(issno int primary key, issdate date,memno int references membership, cassno int references cassette);

Table created.

1. Insert around 10 records in each tables .

->customer table

insert into table customer values(1,’payal’); 1 row created.

Insert into table customer(2,’simran’); 1 row created.

Insert into table customer(3,’vaish’); 1 row created.

Insert into table customer(4,’priyanka’); 1 row created.

Insert into table customer(5,’vidhi’); 1 row created.

Select \* from customer;

Custno custname

* 1. payal
  2. simran
  3. vaish
  4. priyanka
  5. vidhi

->membership table

Insert into table membership values(1,1); 1 row created.

Insert into table membership values(2,2); 1 row created.

Insert into table membership values(3,3); 1 row created.

Insert into table membership values(4,4); 1 row created.

Insert into table membership values(5,5);

1 row created.

Select \* from membership;

Memno custno

|  |  |
| --- | --- |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |

->cassette table

Insert into cassette values(1,’remix’,’hindi’); 1 row created.

Insert into cassette values(2,’classic’,’hindi’); 1 row created.

Insert into cassette values(3,’classic’,’urdu’); 1 row created.

Insert into cassette values(4,’gazal’,’urdu’); 1 row created.

Insert into cassette values(5,’slow’,’english’); 1 row created.

Select \* from cassette; Cassno cassname language

|  |  |  |
| --- | --- | --- |
| 1 | remix | hindi |
| 2 | classic | hindi |
| 3 | classic | urdu |

|  |  |  |
| --- | --- | --- |
| 4 | gazal | urdu |
| 5 | slow | English |

->iss\_rec table

Insert into iss\_rec values(12,’1999-june-30’,1,1); 1 row created.

Insert into iss\_rec values(13,’1999-june-20’2,2); 1 row created.

Insert into iss\_rec values(14,’1999-june-20’,2,2); 1 row created.

Insert into iss\_rec values(15,’1999-june-22’,4,3); 1 row created.

Insert into iss\_rec values(16,’1999-june-21’,4,4); 1 row created.

Select \* from iss\_rec;

Issno issdate memno cassno

12 1999-jun-30 1 1

13 1999-jun-20 2 2

14 1999-jun-20 2 2

15 1999-jun-22 4 3

16 1999-jun-21 4 4

1. list all the customer names with membership numbers.

Select custname,memno from customer c,membership m where c.custno=m.custno;

Custname memno

|  |  |
| --- | --- |
| Payal | 1 |
| Simran | 2 |
| Vaish | 3 |
| Priyanka | 4 |
| Vidhi | 5 |

1. list all the issues for the current date with customer names and cassette names.

Select custname,cassname,issdate from cassette c,iss\_rec i,customer co,membership m where co.custno=m.custno and c.cassno=i.cassno and m.memno=i.memno and issdate=’1999-june30’);

Custname cassname issdate

Payal remix 1999-jun-30

1. list the details of customer who has bought the cassette whose title is ”remix” Custno custname

1 payal

(F) give a count of how manycassettes have been borrowed by each customer

select c.custno,count(d.cassno) from membership m,customer c,iss\_rec I,cassette d where c.custno=m.custno and m.memno=i.memno and d.cassno=i.cassno group by c.custno;

Custno count(d.cassno)

|  |  |
| --- | --- |
| 1 | 1 |
| 2 | 2 |
| 4 | 2 |

1. Give a list of book which has been taken by the student with memno=4

Select d.cassname from iss\_rec I,membership m,cassette d,customer c,where i.memno=4 and m.memno=i.memno and c.custno=m.custno and d.cassno=i.cassno;

Cassname

Classic Gazal

1. List the cassette issues for today

Select d.cassno,d.cassname,d.language from iss\_rec i,customer c,cassette d,membership m where m.memno=i.memno and c.custno=m.custno and d.cassno=i.cassno and i.issdate ‘1999-june-22’;

Cassno cassname language

2 classic hindi

1. Create view which lists out issno,issdate,custname,cassname

Create view vidhi as(select i.issno,i.issdate,c.custname,d.cassname from iss\_rec i,membership m,customer c,cassette d where m.memno=i.memno and c.custno=m.custno and d.cassno=i.cassno);

View created

Select \* from vidhi;

Issno issdate custname cassname

|  |  |  |  |
| --- | --- | --- | --- |
| 12 | 1999-june-30 | payal | remix |
| 13 | 1999-june-20 | simran | classic |
| 14 | 1999-june-20 | simran | classic |
| 15 | 1999-june-22 | priyanka | classic |
| 16 | 1999-june-21 | priyanka | gazal |

1. create a view which lists issues date wise from 12june19 to 30june19

Create view vid5 as(select i.issno,i.issdate,i.memno,i.cassno from iss\_rec I where i.issdate between ‘1999-june-12’ and ’1999-june-30’);

View created Select \* from vid5;

Issno issdate memno cassno

|  |  |  |  |
| --- | --- | --- | --- |
| 12 | 1999-june-30 | 1 | 1 |
| 13 | 1999-june-20 | 2 | 2 |
| 14 | 1999-june-20 | 2 | 2 |
| 15 | 1999-june-22 | 4 | 3 |
| 16 | 1999-june-21 | 4 | 4 |

* 1. **Database Schema for a student-Lab scenario** Stu( **stud\_no: integer**,stud\_name:string,**class:string**) Class(**class: string,descript:string**)

Lab(**mach\_no: integer**,Lab\_no:integer,description:String) Allotment(**stud\_no: integer, mach\_no: integer, dayofweek: string)**

## From all above schema, perform the following

* Create the tables with appropriate integrity constraints

create table stu(stud\_no int primary key,stud\_name varchar(20) references class);

**o/p:-** Table created

create table class(class varchar(20) primary key,descript varchar(20));

**o/p:-** Table created

create table Lab(mach\_no int primary key, Lab\_no int, description varchar(20));

**o/p:-** Table created

create table allotment(stud\_no int references stu(stud\_no),mach\_no int references Lab(mach\_no), dayofweek varchar(20));

**o/p:-** Table created

/\* insert values into tables \*/ insert into Lab values(10,1,’heat’); 1 row created

insert into Lab values(20,1,’heat’);

1 row created

insert into Lab values(30,1,’heat’); 1 row created

insert into Lab values(40,1,’heat’); 1 row created

insert into Lab values(50,1,’heat’); 1 row created

insert into Lab values(60,1,’heat’); 1 row created

Select \* from Lab;

**Output**

MACH\_NO LAB\_NO DESCRIPTIONS

|  |  |  |
| --- | --- | --- |
| 10 | 1 | HEAT |
| 20 | 1 | HEAT |
| 30 | 1 | HEAT |
| 40 | 1 | HEAT |
| 50 | 1 | HEAT |
| 60 | 1 | HEAT |
| 70 | 1 | HEAT |
| 80 | 1 | HEAT |
| 90 | 1 | HEAT |
| 100 | 1 | HEAT |

/\* INSERT INTO CLASS \*/

Insert into class values(‘btech’,’it’); 1 row created

Insert into class values(‘bcom’,’it’); 1 row created

Insert into class values(‘bsc’,’it’); 1 row created

Insert into class values(‘be’,’it’); 1 row created

Insert into class values(‘bd’,’it’); 1 row created

Select \* from class;

## Class Descrip

BTECH IT

BCOM IT

BSC IT

BE IT

BD IT

## /\* insert into stu values \*/

Insert into stu values(33,’psai’,’btech’); 1 row created

Insert into stu values(25,’adi’,’bcom’); 1 row created

Insert into stu values(23,’chotu’,’bsc’); 1 row created

Insert into stu values(42,’nunny’,’be’); 1 row created

Insert into stu values(17,’esr’,’bd’); 1 row created

|  |  |  |
| --- | --- | --- |
| select \* from stu; |  | |
| STUD\_NO | STUD\_NAME | CLASS |
| 33 | PSAI | btech |
| 25 | adi | bcom |
| 23 | chotu | bsc |
| 42 | nunny | be |
| 17 | esr | bd |
| 05 | venkat | btech |
| 15 | mahesh | bcom |
| 28 | ranjith | bsc |
| 21 | chandra | be |
| 48 | sagar | bd |

## /\* insert into allotment \*/

Insert into allotment values(33,10,’mon’); 1 row created

Insert into allotment values(25,20,’tue’); 1 row created

Insert into allotment values(23,30,’wed’); 1 row created

Insert into allotment values(42,40,’thu’); 1 row created

Insert into allotment values(44,50,’fri’); 1 row created

|  |  |  |
| --- | --- | --- |
| Select \* from allotment; |  | |
| STUD\_NO | MACH\_NO | DAYOFWEEK |
| 33 | 10 | mon |
| 25 | 20 | tue |
| 23 | 30 | wed |
| 42 | 40 | thu |
| 17 | 50 | fri |
| 05 | 60 | mon |
| 15 | 70 | tue |
| 28 | 80 | wed |
| 21 | 90 | thu |
| 48 | 100 | fri |

1. list all the machine allotments with the student names,lab and machine numbers Select s.stud\_name, l.lab\_no, l.mach\_no from stu s, class c, lab l, allotment a where

s.stud\_no=a.stud\_no and c.class=s.class and l.mach\_no=a.mach\_no;

|  |  |  |  |
| --- | --- | --- | --- |
| STUD\_NAME | LAB\_NO |  | MACH\_NO |
| Psai | 1 |  | 10 |
| Adi | 1 |  | 20 |
| Chotu | 1 |  | 30 |
| Nunny | 1 |  | 40 |
| Esr | 1 |  | 50 |
| Venkat | 1 |  | 60 |
| Mahesh |  | 1 | 70 |

|  |  |  |  |
| --- | --- | --- | --- |
| Ranjith | 1 | 80 |  |
| Chandra | 1 |  | 90 |
| Sagar | 1 |  | 100 |

## list the total number of lab allotments day wise

select

## c) Give a count of how many machines have been allocated to the ‘btech’ class

select c.class,count(l.mach\_no) from stu s, class c, lab l, allotment a where s.class = c.class and s.stud\_no=a.stud\_no and l.mach\_no=a.mach\_no and c.class=’btech’ group by c.class;

## class count(L.mach\_no)

btech 1

## Give a mavhine allotment details of the stud\_no=33 with his personal and class details

Select s.stud\_no,l.mach\_no,c.class from stu s,class c,lab l,allotment a where s.stud\_no=a.stud\_no and s.class=c.class and l.mach\_no=a.mach\_no and s.stud\_no=33;

|  |  |  |
| --- | --- | --- |
| STUD\_NO | MACH\_NO | CLASS |
| 33 | 10 | btech |

## count for how many machines have been allocated in lab\_no 25 for the day of the week as “Monday”

Select count(l.mach\_no) from lab l, allotment a where lab\_no=25 and dayofweek=’mon’ and l.mach\_no=a.mach\_no group by lab\_no;

No rows selected

## how many students in the class wise have allocated machines in the labs

Select c.class,count(s.stud\_no) from stu s, lab l, class c, allotment a where s.stud\_no=a.stud\_no and s.class=c.class and l.mach\_no=a.mach\_no group by c.class;

|  |  |
| --- | --- |
| **CLASS** | **COUNT(S.STUD\_NO)** |
| bsc | 1 |
| be | 1 |
| bcom | 2 |
| btech | 1 |
| bd | 1 |

## create a view which list out the stud\_no, stud\_name, mach\_no, lab\_no, dayofweek

Create view stu\_lab as select s.stud\_no, s.stud\_name, l.mach\_no, l.lab\_no, a.dayofweek from stu s, class c,lab l, allotment a where s.stud\_no=a.stud\_no and s.class=c.class and l.mach\_no=a.mach\_no;

View created;

Select stud\_no,stud\_name from stu\_lab;

STUD\_NO STUD\_NAME MACH\_NO LAB\_NO DAYOFWEEK

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 33 | psai | 10 | 1 | mon |
| 25 | adi | 20 | 1 | tue |
| 23 | chotu | 30 | 1 | wed |
| 42 | nunny | 40 | 1 | thu |
| 17 | esr | 50 | 1 | fri |
| 05 | venkat | 60 | 1 | mon |

## create a view which list the machine allotment details for ‘Thursday’

Create view sce5 as select l.mach\_no,dayofweek from lab l,allotment a where a.dayofweek=’thu’ and l.mach\_no=a.mach\_no;

View created Select \* from sce5;

MACH\_NO DAYOFWEEK

40 THU

90 THU

# CREATE CURSOR WHICH DISPLAY ALL EMPLOYEE NUMBERS AND NAMES FROM THE EMP TABLE

DELIMITER $$;

CREATE PROCEDURE procemp()

begin

declare vename varchar(20); declare vsalary int;

declare vfinished integer default 0;

declare c1 CURSOR for select ename,salary from employee; declare continue handler for not found set vfinished=1; OPEN c1;

getemp:LOOP

FETCH c1 into vename,vsalary; if vfinished=1 then

leave getemp; end if;

select concat(vename,vsalary); END LOOP getemp;

CLOSE c1;

END $$

call procemp();

+ +

| concat(vename,vsalary) |

+ +

| abc11111 |

+ +

1 row in set (0.00 sec)

+ +

| concat(vename,vsalary) |

+ +

| def20000 |

+ +

1 row in set (0.02 sec)

+ +

| concat(vename,vsalary) |

+ +

| an25000 |

+ +

1 row in set (0.03 sec)

+ +

| concat(vename,vsalary) |

+ +

| ant60000 |

+ +

1 row in set (0.05 sec)

# CREATE A CURSOR WHICH UPDATE THE SALARIES OF ALL EMPLOYEES AS PER THE GIVEN DATA

DELIMITER $$;

CREATE PROCEDURE updatesal() begin

declare veid int(10); declare vsalary int(10);

declare vfinished integer default 0;

declare c2 CURSOR for select eid,salary from employee; declare continue handler for not found set vfinished=1; OPEN c2;

getemp:LOOP

FETCH c2 into veid,vsalary; if vfinished=1 then

leave getemp; end if;

update employee set salary=salary+200 where eid=veid; END LOOP getemp;

CLOSE c2;

End$$

call updatesal();

select \* from employee;

+ + + + +

| eid | ename | salary | mobilno |

+ + + + +

| 101 | abc | 11311 | 12345 |

| 102 | def | 20200 | 9999 |

| 103 | an | 25200 | 5555 |

| 104 | ant | 60200 | 66666 |

+ + + + +

4 rows in set (0.00 sec)

# CREATE A CURSOR WHICH DISPLAYS NAME OF EMPLOYEES HAVING SALARY >

## 10000

DELIMITER $$;

CREATE PROCEDURE salgreat() begin

declare vename varchar(20); declare veid int(10);

declare vfinished integer default 0;

declare c4 cursor for select eid,ename from employee where salary>20000; declare continue handler for not found set vfinished=1;

OPEN c4;

getemp:loop

FETCH c4 into veid,vename; if vfinished=1 then

leave getemp; end if;

select concat(veid,vename); end loop getemp;

CLOSE c4;

end $$

call salgreat();

+ +

| concat(veid,vename) |

+ +

| 102def |

+ +

1 row in set (0.00 sec)

+ +

| concat(veid,vename) |

+ +

| 103an |

+ +

1 row in set (0.02 sec)

+ +

| concat(veid,vename) |

+ +

| 104ant |

+ +

# CREATE A PROCEDURE TO FIND REVERSE OF A GIVEN NUMBER

CREATE PROCEDURE p1(p INT)

begin

declare num1,num2,rev int; set rev=0;

set num1:=p; while num1>0 do

set num2:=num1 mod 10; set rev:=num2+(rev\*10); set num1:=floor(num1/10);

end while; Select num1; Select num2; Select rev; select p;

end;

//

call p1(123)//

# CREATE A PROCEDURE TO UPPDATE THE SALARIES OF ALL EMPLLOYEES AS PER THE GIVEN DATA

CREATE PROCEDURE p2(IN id INT)

begin

declare empid int; set empid=id;

if empid>0 then

update coll set sal=sal+5 ; else

select 'employee id is WRONG' empid; end if ;

end

$$

call p2(1)// DELIMITER ;

select \* from coll;

DELIMITER $$;

CREATE PROCEDURE updatesal() begin

declare veid int(10); declare vsalary int(10);

declare vfinished integer default 0;

declare c2 CURSOR for select eid,salary from employee;

declare continue handler for not found set vfinished=1; OPEN c2;

getemp:LOOP

FETCH c2 into veid,vsalary; if vfinished=1 then

leave getemp; end if;

update employee set salary=salary+200 where eid=veid; END LOOP getemp;

CLOSE c2;

End$$

call updatesal();

select \* from employee;

+ + + + +

| eid | ename | salary | mobilno |

+ + + + +

| 101 | abc | 11311 | 12345 |

| 102 | def | 20200 | 9999 |

| 103 | an | 25200 | 5555 |

| 104 | ant | 60200 | 66666 |

+ + + + +

4 rows in set (0.00 sec)

# CREATE A PROCEDURE TO DEMONSTRATE IN, OUT AND INOUT PARAMETERS

1. DELIMITER $$

CREATE PROCEDURE `pro\_IN` (IN var1 INT) BEGIN

SELECT var1 + 2 AS result;

END$$

call pro\_IN(7);

1. mysql> create procedure p11(out cd int)

-> select count(\*) into cd from coll where sal>1000; Query OK, 0 rows affected (0.00 sec)

mysql> call p11(@a);

Query OK, 0 rows affected (0.00 sec)

mysql> select @a;

+ +

| @a |

+ +

| 4 |

+ +

1 row in set (0.00 sec)

1. mysql> create procedure p21(inout a int)

-> begin

-> declare p int;

-> set p:=1;

-> repeat

-> set p:=p\*a;

-> set a:=a-1;

-> until a<=1

-> end repeat;

-> set a:=p;

-> select a;

-> end

->$$

Query OK, 0 rows affected (0.00 sec)

mysql> set @a=5; //

Query OK, 0 rows affected (0.00 sec)

mysql> call p21(@a); //

+ +

| a |

+ +

| 120 |

+ +

1 row in set (0.00 sec)

1. mysql> CREATE PROCEDURE sal\_inc (INOUT p\_sal INT, IN p\_inc INT, OUT p\_pct\_inc DECIMAL (5,2))

-> BEGIN DECLARE p\_sal\_new INT;

-> SET p\_sal\_new := p\_sal + p\_inc;

-> SET p\_pct\_inc := (p\_sal\_new-p\_sal) / p\_sal \* 100;

-> SET p\_sal:=p\_sal\_new;

-> select p\_sal;

-> select p\_pct\_inc;

-> END$$

Query OK, 0 rows affected (0.00 sec) mysql>

mysql> set @p\_sal=19000; //

Query OK, 0 rows affected (0.00 sec) mysql> set @p\_inc=2000; //

Query OK, 0 rows affected (0.00 sec)

mysql> call sal\_inc(@p\_sal,@p\_inc,@p\_pct\_inc); //

+ +

| p\_sal |

+ +

| 21000 |

+ +

1 row in set (0.00 sec)

# CREATE A FUNCTION TO CHECK WHETHER GIVEV STRING IS PALINDROME OR NOR

mysql> create function ft13(a char(10))

-> returns char(10)

-> begin

-> declare re,r char(10);

-> set r:=reverse(a);

-> if strcmp(a,r)=0 then

-> set re='yes';

-> else

-> set re='no';

-> end if;

-> return re;

-> end$$

Query OK, 0 rows affected (0.00 sec)

mysql> select ft13('abc');

-> //

+ +

| ft13('abc') |

+ +

| no |

+ +

1 row in set (0.02 sec)

mysql> select ft13('aba');//

+ +

| ft13('aba') |

+ +

| yes |

+ +

1 row in set (0.00 sec)

# CREATE A FUNCTION TO FIND SUM OF SALARIES OF ALL EMPLOYEES AS PER THE GIVEN DATA

CREATE OR REPLACE FUNCTION TOTALSALARY RETURN NUMBER

IS

TOTAL\_SAL NUMBER:=0; BEGIN

SELECT SUM(SAL)INTO TOTAL\_SAL FROM EMP WHERE DEPTNO=10;

RETURN TOTAL\_SAL; END;

/

Function created.

DECLARE

X NUMBER(10):=0; BEGIN X:=TOTALSALARY;

DBMS\_OUTPUT.PUT\_LINE('THE TOTAL SALARY OF DEPT NO 10 IS : '||X) END;

/

PL/SQL procedure successfully completed.

SELECT TOTALSALARY FROM DUAL; OUTPUT :-

TOTALSALARY

8750

# 14 CREATE A TRIGGER BEFORE/AFTER UPDATE ON EMPLOYEE TABLE FOR EACH ROW

CREATE OR REPLACE TRIGGER DISPLAYCHANGEINSALARY BEFORE UPDATE ON EMP

FOR EACH ROW DECLARE

Z NUMBER(10);

BEGIN

Z := :NEW.SAL - :OLD.SAL; DBMS\_OUTPUT.PUT\_LINE('OLD SALARY : '||' '|| :OLD.SAL);

DBMS\_OUTPUT.PUT\_LINE('NEW SALARY : '||' '|| :NEW.SAL); DBMS\_OUTPUT.PUT\_LINE('DIFFERENCE SALARY : '||' '|| Z); END;

/

Trigger created.

OUTPUT :-

UPDATE EMP SET SAL=SAL+1000 WHERE ENAME= 'SCOTT';

OLD SALARY : 11450 NEW SALARY : 12450

DIFFERENCE SALARY : 1000

1 row updated.

/\* CREATE A TRIGGER AFTER UPDATE ON EMPLOYEE TABLE FOR EACH ROW \*/ CREATE OR REPLACE TRIGGER DISPLAYCHANGEINSALARY

AFTER UPDATE ON EMP FOR EACH ROW

DECLARE

Z NUMBER(10);

BEGIN

Z := :NEW.SAL - :OLD.SAL; DBMS\_OUTPUT.PUT\_LINE('OLD SALARY : '||' '|| :OLD.SAL);

DBMS\_OUTPUT.PUT\_LINE('NEW SALARY : '||' '|| :NEW.SAL); DBMS\_OUTPUT.PUT\_LINE('DIFFERENCE SALARY : '||' '|| Z); END;

/

Trigger created.

OUTPUT :-

UPDATE EMP SET SAL=SAL-100 WHERE ENAME='SCOTT';

OLD SALARY : 12450 NEW SALARY : 12350

DIFFERENCE SALARY : -100

1 row updated.

# 15 CREATE A TRIGGER BEFORE/AFTER DELETE ON EMPLOYEE TABLE FOR EACH ROW

CREATE OR REPLACE TRIGGER TRDEL AFTER DELETE ON DEPARTMENTS FOR EACH ROW

BEGIN

DELETE FROM EMPLOYEES WHERE DEPARTMENT\_ID = :OLD.DEPARTMENT\_ID; END;

/

Trigger created.

OUTPUT :-

DELETE FROM DEPARTMENTS WHERE DEPARTMENT\_ID=270;

1 row deleted.

SELECT \* FROM DEPARTMENTS;

DEPARTMENT\_ID DEPARTMENT\_NAME MANAGER\_ID LOCATION\_ID

|  |  |  |  |
| --- | --- | --- | --- |
| 10 | Administration | 200 | 1700 |
| 20 | Marketing | 201 | 1800 |
| 30 | Purchasing | 114 | 1700 |
| 40 | Human Resources | 203 | 2400 |
| 50 | Shipping | 121 | 1500 |
| 60 | IT | 103 | 1400 |
| 70 | Public Relations | 204 | 2700 |
| 80 | Sales | 145 | 2500 |
| 90 | Executive | 100 | 1700 |
| 100 | Finance | 108 | 1700 |
| 110 | Accounting | 205 | 1700 |

11 rows selected.

/\* CREATE A TRIGGER BEFORE DELETE ON EMPLOYEE TABLE FOR EACH ROW \*/ CREATE OR REPLACE TRIGGER TRDEL1

BEFORE DELETE ON DEPARTMENTS

FOR EACH ROW BEGIN

DELETE FROM EMPLOYEES WHERE DEPARTMENT\_ID = :OLD.DEPARTMENT\_ID; END;

/

Trigger created.

OUTPUT :-

DELETE FROM DEPARTMENTS WHERE DEPARTMENT\_ID=260;

1 row deleted.

SELECT \* FROM DEPARTMENTS;

DEPARTMENT\_ID DEPARTMENT\_NAME MANAGER\_ID LOCATION\_ID

|  |  |  |  |
| --- | --- | --- | --- |
| 10 | Administration | 200 | 1700 |
| 20 | Marketing | 201 | 1800 |
| 30 | Purchasing | 114 | 1700 |
| 40 | Human Resources | 203 | 2400 |
| 50 | Shipping | 121 | 1500 |
| 60 | IT | 103 | 1400 |
| 70 | Public Relations | 204 | 2700 |
| 80 | Sales | 145 | 2500 |
| 90 | Executive | 100 | 1700 |
| 100 | Finance | 108 | 1700 |
| 110 | Accounting | 205 | 1700 |

11 rows selected.

# 16 CREATE A TRIGGER BEFORE INSERT ON EMPLOYEES TABLE FOR EACH ROW

CREATE OR REPLACE TRIGGER TRG1 BEFORE INSERT ON EMPLOYEES FOR EACH ROW

BEGIN

IF :NEW.SALARY <5000 THEN

RAISE\_APPLICATION\_ERROR(-20001,'SALARY SHOULD BE ABOVE 5000'); END IF;

END;

/

Trigger created.

OUTPUT :-

INSERT INTO EMPLOYEES(EMPLOYEE\_ID,FIRST\_NAME,LAST\_NAME,EMAIL,PHONE\_NUMBER,HIRE\_DAT E,JOB\_ID,SALARY) VALUES (222,'RITESH','SINGH','RSINGH',2345679001,'15/MAR/2015','AC\_MGR',8000);

1 row created.

INSERT INTO EMPLOYEES(EMPLOYEE\_ID,FIRST\_NAME,LAST\_NAME,EMAIL,PHONE\_NUMBER,HIRE\_DAT E,JOB\_ID,SALARY) VALUES (223,'RITESH','GUPTA','RGUPTA',2345679001,'15/MAR/2015','AC\_MGR',3000);

INSERT INTO EMPLOYEES(EMPLOYEE\_ID,FIRST\_NAME,LAST\_NAME,EMAIL,PHONE\_NUMBER,HIRE\_DAT E,JOB\_ID,SALARY) VALUES (223,'RITESH','GUPTA','RGUPTA',2345679001,'15/MAR/2015','AC\_MGR',3000)

\*

ERROR at line 1:

ORA-20001: SALARY SHOULD BE ABOVE 5000 ORA-06512: at "HR.TRG1", line 3

ORA-04088: error during execution of trigger 'HR.TRG1'

/\* CREATE A TRIGGER AFTER INSERT ON EMPLOYEES TABLE FOR EACH ROW \*/ CREATE OR REPLACE TRIGGER TRG2

AFTER INSERT ON EMPLOYEES FOR EACH ROW

BEGIN

IF :NEW.SALARY <5000 THEN

RAISE\_APPLICATION\_ERROR(-20001,'SALARY SHOULD BE ABOVE 5000'); END IF;

END;

/

Trigger created.

OUTPUT :-

INSERT INTO EMPLOYEES(EMPLOYEE\_ID,FIRST\_NAME,LAST\_NAME,EMAIL,PHONE\_NUMBER,HIRE\_DAT E,JOB\_ID,SALARY) VALUES (227,'AJAY','KUMAR','AKUMAR',2345679341,'16/MAR/2015','AC\_MGR',4000);

INSERT INTO EMPLOYEES(EMPLOYEE\_ID,FIRST\_NAME,LAST\_NAME,EMAIL,PHONE\_NUMBER,HIRE\_DAT E,JOB\_ID,SALARY) VALUES

(225,'SURESH','GUPTA','SGUPTA',2345679341,'16/MAR/2015','AC\_MGR',8000)

\*

ERROR at line 1:

ORA-04091: table HR.EMPLOYEES is mutating, trigger/function may not see it ORA-06512: at "HR.TRG2", line 4

ORA-04088: error during execution of trigger 'HR.TRG2'

INSERT INTO EMPLOYEES(EMPLOYEE\_ID,FIRST\_NAME,LAST\_NAME,EMAIL,PHONE\_NUMBER,HIRE\_DAT E,JOB\_ID,SALARY) VALUES (227,'AJAY','KUMAR','AKUMAR',2345679341,'16/MAR/2015','AC\_MGR',4000);

INSERT INTO EMPLOYEES(EMPLOYEE\_ID,FIRST\_NAME,LAST\_NAME,EMAIL,PHONE\_NUMBER,HIRE\_DAT E,JOB\_ID,SALARY) VALUES

(227,'AJAY','KUMAR','AKUMAR',2345679341,'16/MAR/2015','AC\_MGR',4000)

\*

ERROR at line 1:

ORA-20001: SALARY SHOULD BE ABOVE 5000 ORA-06512: at "HR.TRG2", line 3

ORA-04088: error during execution of trigger 'HR.TRG2'