**Introduction to DBMS and SQL**

**Database:**

A **database** is a collection of related data. By data, we mean known facts that can be

recorded and that have implicit meaning.

**Database Management System:**

A **database management system (DBMS)** is a collection of programs that enables users to create and maintain a database. The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications.

**SQL**

SQL stands for Structured Query Language. It is an ANSI (American National Standards Institute) standard. SQL is a standard language for storing, manipulating and retrieving data in databases. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

SQL is widely popular because it offers the following advantages:

* Allows users to define the data in a database and manipulate that data.
* Allows to embed within other languages using SQL modules, libraries & pre-

compilers.

* Allows users to create and drop databases and tables.
* Allows users to create view, stored procedure, functions in a database.
* Allows users to set permissions on tables, procedures and views.

**SQL Commands**

* The standard SQL commands to interact with relational databases are classified into the following groups based on their nature

1. DDL - Data Definition Language

2. DML - Data Manipulation Language

3. DCL - Data Control Language

1. **DDL - Data Definition Language**

* Used to define, change and drop the structure of a table

|  |  |
| --- | --- |
| **Sl. No.** | **Command & Description** |
| 1 | **CREATE**  Creates a new table, a view of a table, or other object in the database. |
| 2 | **ALTER**  Modifies an existing database object, such as a table. |
| 3 | **DROP**  Deletes an entire table, a view of a table or other objects in the database. |

1. **DML - Data Manipulation Language**

* Used to insert, modify, delete and retrieve data from a table

|  |  |
| --- | --- |
| **Sl.No.** | **Command & Description** |
| **1** | **SELECT**  **Retrieves certain records from one or more tables.** |
| **2** | **INSERT**  **Creates a record.** |
| **3** | **UPDATE**  **Modifies records.** |
| **4** | **DELETE**  **Deletes records.** |

**3. DCL - Data Control Language**

* used to provide control over the data in a database

|  |  |
| --- | --- |
| **Sl.No.** | **Command & Description** |
| 1 | **GRANT**  Gives a privilege to user. |
| 2 | **REVOKE**  Takes back privileges granted from user. |

**SQL Data Types**

* **Numeric data types**
* integer numbers of various sizes (INTEGER or INT, and SMALLINT)
* floating-point (real) numbers of various precision (FLOAT or REAL, and DOUBLE PRECISION).
* Formatted numbers can be declared by using DECIMAL(i,j)—or DEC(i,j) or NUMERIC(i,j)—where

i -precision, total number of decimal digits

j - scale, number of digits after the decimal point.

* **Character-string data types**
* fixed length—CHAR(n) or CHARACTER(n), where n is the number of characters
* varying length—VARCHAR(n) or CHAR VARYING(n) or CHARACTER VARYING(n), where n is the maximum number of characters
* When specifying a literal string value, it is placed between single quotation marks (apostrophes), and it is case sensitive
* A **Boolean** data type has the traditional values of TRUE or FALSE
* In SQL, because of the presence of NULL values, a three-valued logic is used, so a third possible value for a Boolean data type is UNKNOWN
* The **DATE** data type has ten positions, and its components are YEAR, MONTH, and DAY in the form YYYY-MM-DD
* The **TIME** data type has at least eight positions, with the components HOUR, MINUTE, and SECOND in the form HH:MM:SS.

**Specifying Key and Referential Integrity Constraints**

* Constraints are the rules enforced on data columns on a table
* Primary key: uniquely identifies each row/record in a database table
* Foreign key: uniquely identifies a row/record in another table
* NOT NULL: ensures that a column cannot have a NULL value

**CREATE Table**

* used to create a new table.
* Syntax:

CREATE TABLE table\_name (column1 datatype,column2 datatype,column3

datatype, .. ., columnN datatype, PRIMARY KEY( one or more columns ) );

**INSERT**

* used to add new rows of data to a table in the database.
* Syntax:

INSERT INTO TABLE\_NAME (column1, column2, column3, ...columnN) VALUES

(value1, value2, value3,...valueN);

- Here, column1, column2, column3,...columnN are the names of the columns in the table

into which you want to insert the data.

**SELECT**

* used to fetch the data from a database table which returns this data in the form of a result table.
* Syntax:

SELECT column1, column2, columnN FROM table\_name;

Here, column1, column2... are the fields of a table whose values you want to fetch.

* If you want to fetch all the fields available in the field, then you can use the following syntax.

SELECT \* FROM table\_name;

**WHERE**

* clause is used to specify a condition while fetching the data from a single table or by joining with multiple tables.
* Syntax

SELECT column1, column2, columnN FROM table\_name WHERE [condition]

**UPDATE**

* used to modify the existing records in a table. You can use the WHERE clause with the UPDATE query to update the selected rows, otherwise all the rows would be affected.
* Syntax

UPDATE table\_name SET column1 = value1, column2 = value2...., columnN = valueN

WHERE [condition];

* You can combine N number of conditions using the AND or the OR operators.

**DROP TABLE**

* used to remove a table definition and all the data, indexes, triggers, constraints and permission specifications for that table.
* Syntax:

DROP TABLE table\_name;

**Lab Programs**

1. **Consider the following schema for a Library Database:**

BOOK (BOOK\_ID, TITLE, PUBLISHER\_NAME, PUB\_YEAR)

BOOK\_AUTHORS (BOOK\_ID, AUTHOR\_NAME)

PUBLISHER (NAME, ADDRESS, PHONE)

BOOK\_COPIES (BOOK\_ID, BRANCH\_ID, NO\_OF\_COPIES)

BOOK\_LENDING (BOOK\_ID, BRANCH\_ID, CARD\_NO, DATE\_OUT, DUE\_DATE)

LIBRARY\_BRANCH (BRANCH\_ID, BRANCH\_NAME, ADDRESS)

CARD (CARD\_NO)

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.
2. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017.
3. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.
4. Create a view of all books and its number of copies that are currently available in the library**.**
5. Delete a book in the BOOK table. Update the contents of other tables to reflect this data manipulation operation.

**Table Creation**

CREATE TABLE PUBLISHER (NAME VARCHAR2 (20) PRIMARY KEY, PHONE INTEGER, ADDRESS VARCHAR2 (20));

CREATE TABLE BOOK (BOOK\_ID INTEGER PRIMARY KEY, TITLE VARCHAR2 (20), PUB\_YEAR VARCHAR2 (20), PUBLISHER\_NAME REFERENCES PUBLISHER (NAME) ON DELETE CASCADE);

CREATE TABLE BOOK\_AUTHORS (AUTHOR\_NAME VARCHAR2 (20),

BOOK\_ID REFERENCES BOOK (BOOK\_ID) ON DELETE CASCADE, PRIMARY KEY (BOOK\_ID, AUTHOR\_NAME));

CREATE TABLE LIBRARY\_BRANCH (BRANCH\_ID INTEGER PRIMARY KEY, BRANCH\_NAME VARCHAR2 (50), ADDRESS VARCHAR2 (50));

CREATE TABLE BOOK\_COPIES (NO\_OF\_COPIES INTEGER,

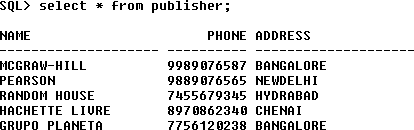
BOOK\_ID REFERENCES BOOK (BOOK\_ID) ON DELETE CASCADE,

BRANCH\_ID REFERENCES LIBRARY\_BRANCH (BRANCH\_ID) ON DELETE CASCADE, PRIMARY KEY (BOOK\_ID, BRANCH\_ID));

CREATE TABLE CARD (CARD\_NO INTEGER PRIMARY KEY);

CREATE TABLE BOOK\_LENDING (DATE\_OUT DATE, DUE\_DATE DATE,

BOOK\_ID REFERENCES BOOK (BOOK\_ID) ON DELETE CASCADE, BRANCH\_IDREFERENCES LIBRARY\_BRANCH (BRANCH\_ID) ON DELETE CASCADE, CARD\_NO REFERENCES CARD (CARD\_NO) ON DELETE CASCADE, PRIMARY KEY (BOOK\_ID, BRANCH\_ID, CARD\_NO));



Text

Description automatically generated

A picture containing text

Description automatically generated

Table

Description automatically generated with medium confidence

Table

Description automatically generated

Table

Description automatically generated

Table

Description automatically generated with medium confidence

**Queries:**

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

SELECT B.BOOK\_ID, B.TITLE, B.PUBLISHER\_NAME, A.AUTHOR\_NAME, C.NO\_OF\_COPIES, L.BRANCH\_ID FROM BOOK B, BOOK\_AUTHORS A, BOOK\_COPIES C, LIBRARY\_BRANCH L WHERE B.BOOK\_ID=A.BOOK\_ID AND B.BOOK\_ID=C.BOOK\_ID AND L.BRANCH\_ID=C.BRANCH\_ID;

Table

Description automatically generated with medium confidence

# Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017.

SELECT CARD\_NO FROM BOOK\_LENDING

WHERE DATE\_OUT BETWEEN ‘01-JAN-2017’ AND ‘01-JUL-2017’ GROUP BY CARD\_NO

HAVING COUNT (\*)>3;



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

Table

Description automatically generatedDELETE FROM BOOK WHERE BOOK\_ID=3;

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

CREATE VIEW V\_PUBLICATION AS SELECT PUB\_YEAR

FROM BOOK;

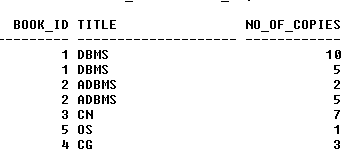


# Create a view of all books and its number of copies that are currently available in the Library.

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

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

1. **Consider the following schema for Order Database:**

SALESMAN (SALESMAN\_ID, NAME, CITY, COMMISSION)

CUSTOMER (CUSTOMER\_ID, CUST\_NAME, CITY, GRADE, SALESMAN\_ID)

ORDERS (ORD\_NO, PURCHASE\_AMT, ORD\_DATE, CUSTOMER\_ID, SALESMAN\_ID)

Write SQL queries to

a) Count the customers with grades above Bangalore’s average.

b) Find the name and numbers of all salesmen who had more than one customer.

c) List all the salesman and indicate those who have and do not have customers in their cities (Use

UNION operation.)

d) Create a view that finds the salesman who has the customer with the highest order of a day.

e) Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be

deleted.

**Table Creation**

create table Salesman(

Salesman\_ID int primary key,

Name varchar(20),

City varchar(20),

Commission varchar(20));

insert into Salesman values(&Salesman\_ID,'&Name','&City','&Commission');

SALESMAN\_ID NAME CITY COMMISSION

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

1000 John Bangalore 25 %

2000 Ravi Bangalore 20 %

3000 Kumar Mysuru 15 %

4000 Smith Delhi 30 %

5000 Harsha Hyderabad 15 %

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

Create table Customer(Customer\_ID int primary key,

Cust\_Name varchar(20),

City varchar(20),

Grade int,

Salesman\_ID references Salesman(Salesman\_ID) on delete set null);

insert into Customer values(&Customer\_ID,'&Cust\_Name','&City',&Grade,&Salesman\_ID);

CUSTOMER\_ID CUST\_NAME CITY GRADE SALESMAN\_ID

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

10 Preethi Bangalore 100 1000

11 Vivek Mangalore 300 1000

12 Bhaskar Chennai 400 2000

13 Chethan Bangalore 200 2000

14 Mamatha Bangalore 400 3000

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

create table Orders(Ord\_No int primary key,

Purchase\_Amt Number,

Ord\_Date date,

Customer\_ID references Customer(Customer\_ID) on delete set null,

Salesman\_ID references Salesman(Salesman\_ID) on delete set null);

insert into Orders values(&Ord\_No,&Purchase,'&Ord\_Date',&Customer\_ID,&Salesman\_ID);

ORD\_NO PURCHASE\_AMT ORD\_DATE CUSTOMER\_ID SALESMAN\_ID

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

50 5000 04-MAY-17 10 1000

51 450 20-JAN-17 10 2000

52 1000 20-JAN-17 13 2000

53 3500 13-APR-17 14 3000

54 550 9-MAR-17 12 2000

55 1000 9-MAR-17 14 1000

**Queries :**

a) Count the customers with grades above Bangalore’s average.

**select count(\*) from Customer where Grade > (select avg(Grade) from Customer where city = 'Bangalore');**

COUNT(\*)

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

3

b. Find the name and numbers of all salesmen who had more than one customer.

**select S.Salesman\_ID, Name from Salesman S having**

**(select count(\*) from Customer where Salesman\_ID=S.Salesman\_ID) > 1 group by Name,**

**S.Salesman\_ID;**

SALESMAN\_ID NAME

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

1000 John

1. Ravi

c. List all the salesman and indicate those who have and do not have customers in their cities (Use UNION operation.)

**select Salesman.Salesman\_ID,Name,Cust\_Name from Salesman ,Customer**

**where Salesman.City=Customer.City**

**Union**

**select Salesman.Salesman\_ID,Name,'No-Match'**

**from Salesman where not City=any (select City from Customer);**

SALESMAN\_ID NAME CUST\_NAME

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

1000 John Chethan

1000 John Mamatha

1000 John Preethi

2000 Ravi Chethan

2000 Ravi Mamatha

2000 Ravi Preethi

3000 Kumar No-Match

4000 Smith No-Match

1. Harsha No-Match

**d. Create a view that finds the salesman who has the customer with the highest order of a day.**

**create view salesman\_order as select b.Ord\_Date, a.Salesman\_ID,Name**

**from Salesman a, Orders B where A.Salesman\_ID=b.Salesman\_ID and b.Purchase\_Amt=**

**(select max(Purchase\_Amt) from Orders C where C.Ord\_Date=b.Ord\_Date);**

ORD\_DATE SALESMAN\_ID NAME

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

04-MAY-17 1000 John

20-JAN-17 2000 Ravi

13-APR-17 3000 Kumar

09-MAR-17 2000 Ravi

1. **Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be**

**deleted.**

**delete from Salesman where Salesman\_ID=1000;**

SALESMAN\_ID NAME CITY COMMISSION

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

2000 Ravi Bangalore 20 %

3000 Kumar Mysuru 15 %

4000 Smith Delhi 30 %

5000 Harsha Hyderabad 15 %

CUSTOMER\_ID CUST\_NAME CITY GRADE SALESMAN\_ID

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

10 Preethi Bangalore 100

11 Vivek Mangalore 300

12 Bhaskar Chennai 400 2000

13 Chethan Bangalore 200 2000

14 Mamatha Bangalore 400 3000

Text

Description automatically generated

1. **Consider the schema for Company Database: Exercise**

EMPLOYEE(SSN, NAME, ADDRESS, SEX, SALARY, SUPERSSN, DNO)

DEPARTMENT(DNO, DNAME, MGRSSN, MGRSTARTDATE)

DLOCATION(DNO,DLOC)

PROJECT(PNO, PNAME, PLOCATION, DNO)

WORKS\_ON(SSN, PNO, HOURS)

Write SQL queries to

a) Make a list of all project numbers for projects that involve an employee whose last name is ‘Scott’,

either as a worker or as a manager of the department that Controls the project.

b) Show the resulting salaries if every employee working on the ‘IoT’ project is Given a 10 percent

raise.

c) Find the sum of the salaries of all employees of the ‘Accounts’ department, as well as the

maximum salary, the minimum salary, and the average salary in this department

d) Retrieve the name of each employee who works on the entire projects controlled by department

number 5.

e) For each department that has more than five employees, retrieve the department number and

the number of its employees who are making more than Rs. 6,00,000.

Create table Depts (Dname varchar(20),Dno int Primary Key,Mgrstrdate date);

Alter table Depts add Mgrssn references Emp(SSN) on delete set NULL;

Update depts set mgrssn=111 where ssn=111;

A picture containing text

Description automatically generated

Create table Emp(SSN int Primary Key,

FName varchar(20),LName varchar(20),

Address varchar(20),

Sex char(1),

Salary int,superssn references emp(ssn) on delete cascade,Dno references Depts(Dno) on delete cascade);

Alter table Depts add Superssn references Emp(SSN) on delete set NULL;

Update emp set superssn= 111 where ssn=1;

**A picture containing table

Description automatically generated**

Create table Deptloc ( Dno references Depts(dno) ,

Dloc varchar(20));

insert into Deptloc values(&dno , '&dloc');

DNO DLOC

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

1 Banglore

2 Mysore

3 Hubli

4 Punjab

5 Guwahati

Create table Prj (Pno int Primary Key , Pname varchar(20), Ploc varchar(20), Dno references Depts(dno));

insert into Prj values(&pno, '&pname' , '&ploc', &dno);

**Table

Description automatically generated**

Create table Work(SSN references Emp(SSN), Pno references Prj(pno), hours int);

insert into work values(&SSN,&Pno, &hours);

Table

Description automatically generated with low confidence

**Queries:**

**a) Make a list of all project numbers for projects that involve an employee whose last name is ‘Scott’,**

**either as a worker or as a manager of the department that Controls the project.**

**Select Pno from EMP E, Work W where E.ssn = W.ssn and Lname=’scott’**

**Union**

**Select pno from EMP E, Department D , Project P where E.ssn=D.Mgrssn and D.Dno=P.Dno and Lname=’scott’;**

PNO

------

60

10

**b) Show the resulting salaries if every employee working on the ‘IoT’ project is Given a 10 percent**

**raise.**

**Select e.ssn , e.fname,e.lname, 1.1\*e.salary as raisedsal from emp e , prj p, work w where p.pname = 'IOT' and p.pno = w.pno and e.ssn = w.ssn;**

Text

Description automatically generated with medium confidence

**c) Find the sum of the salaries of all employees of the ‘Accounts’ department, as well as the**

**maximum salary, the minimum salary, and the average salary in this department**

**Select sum(salary) as sumsal, avg(salary) as avgsal, min(salary) as minsal, max(salary) as maxsal from Emp e, Depts d where e.dno = d.dno and d.dname='Accounts';**

A picture containing shape

Description automatically generated

**d) Retrieve the name of each employee who works on the entire projects controlled by department number 5.**

**select e.fname , e.lname from Emp e where not exists ((select pno from Prj where dno='5') minus (select pno from WORK where e.ssn=ssn));**

**or**

**select e.fname,e.lname from emp e, work w , prj p where e.ssn=w.ssn and w.pno=p.pno and p.dno=5;**

Shape

Description automatically generated

**e) For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs. 6,00,000.**

**select d.dno,d.dname,e.fname,e.lname ,count(\*) as noofemp from Emp e, Depts d where e.dno=d.dno and e.salary>600000 and d.dno in(select e1.dno from Emp e1 group by e1.dno having count(\*)>5) group by d.dno,d.dname,e.fname,e.lname;**

DNO DNAME FNAME LNAME NO OF EMP

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

1. research Kavya Bind 1

**4. The following relations keep track of airline flight information: Exercise**

**FLIGHTS (FLNO: INTEGER, SOURCE: STRING, DESTINATION: STRING, DISTANCE: INTEGER,**

**DEPARTS:TIME, ARRIVES: TIME, PRICE: INTEGER)**

**AIRCRAFT (AID: INTEGER, ANAME: STRING, CRUISINGRANGE: INTEGER)**

**CERTIFIED (EID: INTEGER, AID: INTEGER)**

**EMPLOYEES (EID: INTEGER, ENAME: STRING, SALARY: INTEGER)**

**Note that the Employees relation describes pilots and other kinds of employees as well; every pilot is certified**

**for some aircraft, and only pilots are certified to fly.**

**Write SQL queries to**

**a) Find the names of aircraft such that all pilots certified to operate them earn more than $80,000.**

**b) For each pilot who is certified for more than three aircraft, find the eid and the maximum cruisingrange**

**of the aircraft for which she or he is certified.**

**c) For all aircraft with cruisingrange over 1000 miles, find the name of the aircraft and the average salary**

**of all pilots certified for this aircraft.**

**d) Print the enames of pilots who can operate planes with cruising range greater than 3000 miles but are**

**not certified on any Boeing aircraft.**

**e) Print the name and salary of every nonpilot whose salary is more than the average salary for pilots.**

create table Flights(Flno int primary key, Source varchar(20), Destination varchar(20),Distance

int,DepartTime varchar(10),ArrivalTime varchar(10), Price int);

insert into Flights values(&Flno, '&Source', '&Destination', &Distance, '&DepartTime', '&ArrivalTime', &Price); **A picture containing graphical user interface

Description automatically generated**

**create table AirCraft(Aid int primary key, Aname varchar(20), CrusingRange int);**

**insert into AirCraft values(&Aid,'&AName',&CruisingRange);**

**A picture containing shape

Description automatically generated**

create table Employees(Eid int primary key, Ename varchar(20), Salary int);

insert into Employees values(&Eid,'&Ename',&Salary);

A picture containing graphical user interface

Description automatically generated

**create table Certified(Eid references Employees(Eid), Aid references AirCraft(Aid));**

**insert into Certified values(&Eid,&Aid);**

**EID AID**

**---------- ----------**

**11 3003**

**11 1001**

**11 2002**

**13 4004**

**12 2002**

**Queries:**

**A. Find the names of aircraft such that all pilots certified to operate them earn more than $80,000.**

* 1. **select Aname from AirCraft A, Employees E, Certified C where A.Aid = C.Aid and E.Eid = C.Eid and E.Salary > 80000;**

ANAME

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

Dishan

KingFisher

Deccan

GoIbibo

**B. For each pilot who is certified for more than three aircraft, find the eid and the maximum cruisingrange**

**of the aircraft for which she or he is certified.**

* 1. **select E.Eid, Max(CrusingRange) from Employees E, Certified C, AirCraft A where E.Eid=C.Eid and A.Aid = C.Aid group by E.Eid having count(\*) > 3;**

EID MAX(CRUSINGRANGE)

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

11 3000

**C. For all aircraft with cruisingrange over 1000 miles, find the name of the aircraft and the average salary**

**of all pilots certified for this aircraft.**

* 1. **select A.Aname ,avg(Salary) from Employees E, Certified C , Aircraft A where C.Eid = E.Eid and A.Aid = C.Aid and A.CrusingRange>1000 group by A.Aname;**

ANAME AVG(SALARY)

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

Deccan 112500

KingFisher 90000

Dishan 150000

**D. Print the enames of pilots who can operate planes with cruising range greater than 3000 miles but are**

**not certified on any Boeing aircraft.**

**d.**

**select distinct e.ename**

**from Aircraft a join Certified c**

**on a.aid=c.aid**

**join Employees e**

**on c.eid=e.eid**

**where a.cruisingrange>3000**

**and c.aid not IN**

**(select distinct aid from Aircarft where aname = 'Boeing' );**

**E. Print the name and salary of every nonpilot whose salary is more than the average salary for pilots.**

1. **Select Ename, Salary from Employees where salary > (select avg(salary) from Employees) and Eid not in ( select Eid from Certified);**

5. **Consider the following relations Structured Enquiry**

STUDENT(SNUM: INTEGER, SNAME: STRING, MAJOR: STRING, LEVEL: STRING, AGE: INTEGER)

CLASS(CNAME: STRING, MEETS AT: STRING, ROOM: STRING, FID: INTEGER)

ENROLLED(SNUM: INTEGER, CNAME: STRING)

FACULTY(FID: INTEGER, FNAME: STRING, DEPTID: INTEGER)

The meaning of these relations is straightforward; for example, Enrolled has one record per student-class

pair such that the student is enrolled in the class.

Write the following queries in SQL. No duplicates should be printed in any of the answers.

a) Find the names of all Juniors (level = JR) who are enrolled in a class taught by Rakesh.

b) Find the age of the oldest student who is either a history major or enrolled in a course taught by Ravi.

c) Find the names of all students who are enrolled in two classes that meet at the same time.

d) For each faculty member that has taught classes only in room R128, print the faculty member’s name

and the total number of classes she or he has taught.

e)Create a view that contains the details of students along with the name of the courses enrolled.

Create table Student (Snum int Primary Key,Sname varchar(20),Major varchar(10), Lev varchar(20),age int);

Insert into Student values(&Snum,'&Sname','&Major','&Lev',&age);

A picture containing table

Description automatically generated

ADD Age column to the above table.

Create table Faculty(Fid int Primary Key,Fname varchar(20),Depid int);

Insert into Faculty values(&Fid , '&Fname',&Depid);

Shape

Description automatically generated

Create table Class(Cname varchar(20) Primary Key, Meets\_at varchar(20),Room varchar(20),Fid references Faculty(Fid) on delete set NULL);

Insert into Class values ('&Cname' , '&Meets\_at' ,'&Room' , &Fid);

Table

Description automatically generated

Create table Enrolled (Snum references Student(Snum),Cname references Class(Cname));

Insert into Enrolled values(&Snum ,'&Cname');

Text

Description automatically generated

drop table Class cascade constraints;

drop table Faculty cascade constraints;

**A. Find the names of all Juniors (level = JR) who are enrolled in a class taught by Rakesh.**

* 1. **Select Distinct S.Sname from Student S,Class C, Enrolled E,Faculty F where S.Snum = E.Snum and E.Cname = C.Cname and C.Fid = F.Fid and F.Fname = 'Rakesh' and S.Lev = 'JR' ;**

**order by Snum;**

A picture containing text

Description automatically generated

**B. Find the age of the oldest student who is either a history major or enrolled in a course taught by Ravi.**

b**. Select MAX(S.age) as Age from Student S where (S.Major = 'History') OR S.Snum in (Select E.Snum from Class C , Enrolled E, Faculty F where E.cname = C.cname and C.Fid = F.Fid and F.Fname = 'Ravi');**

AGE

------

26

**C. Find the names of all students who are enrolled in two classes that meet at the same time.**

c. **Select Distinct S.Sname from Student S where S.Snum in (Select E1.Snum from Enrolled E1,Enrolled E2, Class C1, Class C2 where E1.snum = E2.snum and E1.cname <> E2.cname and E1.cname = C1.cname and E2.cname = C2.cname and C1.meets\_at = C2.meets\_at);**

SNAME

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

Helen

Zola

**D. For each faculty member that has taught classes only in room R128, print the faculty member’s name**

**and the total number of classes she or he has taught.**

d. **Select Distinct F.Fname , count(\*) as CourseCount from CLass C , Faculty F where C.FID not in (Select Fid from Class where Room IN (Select Room from Class where Room!= 'R128')) AND C.Fid = F.Fid group by F.Fname;**

FNAME COURSECOUNT

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

Kera 2

Jane 2

e)Create a view that contains the details of students along with the name of the courses enrolled.

**Create view as Student\_course as**

**Select s.snum, s.sname, s.major,s.level,s.age,e.cname**

**from Student S, Enrolled E where s.snum=e.snum;**

**Viva Questions**

1. **What is SQL?**

Structured Query Language

# What is database?

A database is a logically coherent collection of data with some inherent meaning, representing some aspect of real world and which is designed, built and populated with data for a specific purpose.

# What is DBMS?

It is a collection of programs that enables user to create and maintain a database. In other words it is general-purpose software that provides the users with the processes of defining, constructing and manipulating the database for various applications.

# What is a Database system?

The database and DBMS software together is called as Database system.

# Advantages of DBMS?

* + Redundancy is controlled.
  + Unauthorized access is restricted.
  + Providing multiple user interfaces.
  + Enforcing integrity constraints.
  + Providing backup and recovery.

# Disadvantage in File Processing System?

* + Data redundancy & inconsistency.
  + Difficult in accessing data.
  + Data isolation.
  + Data integrity.
  + Concurrent access is not possible.
  + Security Problems.

# Describe the three levels of data abstraction?

There are three levels of abstraction:

* + Physical level: The lowest level of abstraction describes how data are stored.
  + Logical level*:* The next higher level of abstraction, describes what data are stored in database and what relationship among those data.
  + View level:The highest level of abstraction describes only part of entire database.

# Define the "integrity rules"

There are two Integrity rules.

* + Entity Integrity:States that ―Primary key cannot have NULL value‖
  + Referential Integrity:States that ―Foreign Key can be either a NULL value or should be Primary Key value of other relation.

# What is extension and intension?

Extension - It is the number of tuples present in a table at any instance. This is time dependent.

Intension -It is a constant value that gives the name, structure of table and the constraints laid on it.

# What is Data Independence?

Data independence means that ―the application is independent of the storage structure and access strategy of data‖. In other words, The ability to modify the schema definition in one level should not affect the schema definition in the next higher level.

Two types of Data Independence:

* + Physical Data Independence: Modification in physical level should not affect the logical level.
  + Logical Data Independence: Modification in logical level should affect the view level.

NOTE: Logical Data Independence is more difficult to achieve.

# What is a view? How it is related to data independence?

A view may be thought of as a virtual table, that is, a table that does not really exist in its own right but is instead derived from one or more underlying base table. In other words, there is no stored file that direct represents the view instead a definition of view is stored in data dictionary.

Growth and restructuring of base tables is not reflected in views. Thus the view can insulate users from the effects of restructuring and growth in the database. Hence accounts for logical data independence.

# What is Data Model?

# A collection of conceptual tools for describing data, data relationships data semantics and constraints.

# What is E-R model?

This data model is based on real world that consists of basic objects called entities and of relationship among these objects. Entities are described in a database by a set of attributes.

# What is Object Oriented model?

This model is based on collection of objects. An object contains values stored in instance variables within the object. An object also contains bodies of code that operate on the object. These bodies of code are called methods. Objects that contain same types of values and the same methods are grouped together into classes.

# What is an Entity?

It is an 'object' in the real world with an independent existence.

# What is an Entity type?

It is a collection (set) of entities that have same attributes.

# What is an Entity set?

It is a collection of all entities of particular entity type in the database.

# What is an Extension of entity type?

The collections of entities of a particular entity type are grouped together into an entity

set.

# What is an attribute?

It is a particular property, which describes the entity.

# What is a Relation Schema and a Relation?

A relation Schema denoted by R(A1, A2, …, An) is made up of the relation name R and the list of attributes Ai that it contains. A relation is defined as a set of tuples. Let r be the relation which contains set tuples (t1, t2, t3, ...,tn). Each tuple is an ordered list of n- values t=(v1,v2, ..., vn).

# What is degree of a Relation?

It is the number of attribute of its relation schema.

# What is Relationship?

It is an association among two or more entities.

# What is Relationship set?

The collection (or set) of similar relationships.

1. ***What is Relationship type?***

Relationship type defines a set of associations or a relationship set among a given set of entity types.

# What is degree of Relationship type?

It is the number of entity type participating.

# What is DDL (Data Definition Language)?

A data base schema is specified by a set of definitions expressed by a special language called DDL.

# What is VDL (View Definition Language)?

It specifies user views and their mappings to the conceptual schema.

# What is SDL (Storage Definition Language)?

This language is to specify the internal schema. This language may specify the mapping between two schemas.

# What is Data Storage - Definition Language?

The storage structures and access methods used by database system are specified by a set of definition in a special type of DDL called data storage- definition language.

# What is DML (Data Manipulation Language)?

This language that enable user to access or manipulate data as organized by appropriate data model.

* + Procedural DML or Low level: DML requires a user to specify what data are needed and how to get those data.
  + Non-Procedural DML or High level: DML requires a user to specify what data are needed without specifying how to get those data.

# What is DML Compiler?

It translates DML statements in a query language into low-level instruction that the query evaluation engine can understand.

1. **What is Relational Algebra?**

It is a procedural query language. It consists of a set of operations that take one or two relations as input and produce a new relation.

# What is Relational Calculus?

It is an applied predicate calculus specifically tailored for relational databases proposed by E.F. Codd. E.g. of languages based on it are DSL, ALPHA, QUEL.

# What is normalization?

It is a process of analyzing the given relation schemas based on their Functional Dependencies (FDs) and primary key to achieve the properties

* + Minimizing redundancy
  + Minimizing insertion, deletion and update anomalies.

# What is Functional Dependency?

A Functional dependency is denoted by X Y between two sets of attributes X and Y that are subsets of R specifies a constraint on the possible tuple that can form a relation state r of

R. The constraint is for any two tuples t1 and t2 in r if t1[X] = t2[X] then they have t1[Y] = t2[Y]. This means the value of X component of a tuple uniquely determines the value of component Y.

# When is a functional dependency F said to be minimal?

* + Every dependency in F has a single attribute for its right hand side.
  + We cannot replace any dependency X A in F with a dependency Y A where Y is a proper subset of X and still have a set of dependency that is equivalent to F.
  + We cannot remove any dependency from F and still have set of dependency that is equivalent to F.

# What is Multivalued dependency?

Multivalued dependency denoted by X Y specified on relation schema R, where X and Y are both subsets of R, specifies the following constraint on any relation r of R: if two tuples t1 and t2 exist in r such that t1[X] = t2[X] then t3 and t4 should also exist in r with the following properties

* t3[x] = t4[X] = t1[X] = t2[X]
* t3[Y] = t1[Y] and t4[Y] = t2[Y]
* t3[Z] = t2[Z] and t4[Z] = t1[Z]

where [Z = (R-(X U Y)) ]

# What is Lossless join property?

It guarantees that the spurious tuple generation does not occur with respect to relation schemas after decomposition.

# What is 1 NF (Normal Form)?

The domain of attribute must include only atomic (simple, indivisible) values.

# What is Fully Functional dependency?

It is based on concept of full functional dependency. A functional dependency X Y is fully functional dependency if removal of any attribute A from X means that the dependency does not hold any more.

# What is 2NF?

A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on primary key.

# What is 3NF?

A relation schema R is in 3NF if it is in 2NF and for every FD X A either of the following is true

* X is a Super-key of R.
* A is a prime attribute of R.

In other words, if every non prime attribute is non-transitively dependent on primary key.

# What is BCNF (Boyce-Codd Normal Form)?

A relation schema R is in BCNF if it is in 3NF and satisfies additional constraints that for every FD X A, X must be a candidate key.

# What is 4NF?

A relation schema R is said to be in 4NF if for every Multivalued dependency X Y that holds over R, one of following is true

* X is subset or equal to (or) XY = R.
* X is a super key.

# What is 5NF?

A Relation schema R is said to be 5NF if for every join dependency {R1, R2, ...,Rn} that holds R, one the following is true

* Ri = R for some i.
* The join dependency is implied by the set of FD, over R in which the left side is key of R.

# What is Domain-Key Normal Form?

A relation is said to be in DKNF if all constraints and dependencies that should hold on the constraint can be enforced by simply enforcing the domain constraint and key constraint on the relation.