UNIT ⇒ '3'

Relational Data Modal, Implementation & Manipulation.

Abhay Kumar Mishra
Lecturer, Dept. of B.C.A.
Maharaja College, Ara

Relational Database Management System

Relational Database Management System (RDBMS) is a S/w package used to store & retrive data that is organised in the form of tables.

In the other word RDBMS stores information in rows & column.

- **Concepts of Attributes & Domains & Tuples :-**

Entity Employee

Attributes → Emp-Code	Name	Year
21130	Jyoti	1
21140	Jayanti	3
21150	Khushi	2
21160	Puja	3
21170	Shweta	6

Domain

Key
(the primary or composite)

- Row-wise information is called Tuple.
- Column-wise information is called attribute.

Abhay Kumar Mishra
Lecturer, Dept. of B.C.A.
Maithanta College, Ara (V.K.S.U.)

Each Tuple or row has a set of permitted values for attributes is called Domain.

For example :- Domain of attributes Name is the set of all alphabetic string of finite length.

Domain of attribute Year is the set of all whole no. as shown in figure.

* Introduction to CODD's Rules

There are twelve (12) rules formulated by codes for RDBMS in 1970, If an RDBMS satisfies all the twelve rules, then full benefits of the relational base result can be obtained.

The twelve rules are having the following main po

a) Information Representation.

b) Guaranteed Access.

c) Systematic Treatment of NULL value.

d) Database Description Rule.

e) Comprehensive data sub-language.

View updating.

High-level update, Insert, Delete.

physical Data Independence.

Logical Data Independence.

The Distribution Rule.

Non-subversion Rule.

Integrity Rule.

Information Representation :-

In a Relational Data Model all information should be always in the form of table data stored in Data Dictionary should also be in the form of table.

Guaranteed Access :-

Every value of data items must be logically addressable by using a combination of table-name, primary-key value & column-name. There it is to be stored.

Abhay Kumar Mishra
Lecturer, Deptt. of B.C.A.
Mahatma Phule... (V.K.S.U.)

3) Systematic Treatment of NULL Value :-)

In RDBMS NULL value should be supported for the representation of missing & in applicable information only. The DBMS must has a consistent method for representing NULL value.

For example :->

NULL values of numeric data must be distinct from zero any other non-numeric value & character data it must be different from string of blank.

4) Data Base Description Rule :-)

The description of data base is stored & maintained in the form of tables. That mean RDBMS is constructed out of tables & & / or view that can examined using the SQL.

Comprehensive Data sub-language :-)

The RDBMS must be completely manageable through it's own extension of SQL. The SQL should support Data of View, Data manipulation etc.

6) View Updating :-
Any view that can be defined using combination of base tables is theoretical updatable, must also be capable of being updated by the DBMS.

7) High level update, Insert & Delete :-
It must be possible to insert, delete & update from from the relation set.

8) Physical Data Independence :-
Change made to physical storage representation are omitted do not changes to be made to the application used to manipulate data in a table.

9) Logical data Independence :-
Application program should be affected by the changes made to the base tables. Changes made to tables should not required changes to be made application program, operating on the table.

10) The Distribution Rule :-

A RDBMS package must have distributed independent. Thus RDBMS package built on relation will for today's client / server database design.

i.e. Multiple computer.

11) Non-Subversion :-

If the RDBMS support facilities allowing application programs to operate on table a row at time.
(एक बार में एक Row Operate हो).

12) Integrity Rule :-

Integrity constraint specific to a particular relational database must be definable in SQL.

* Relational Algebra :-

Relational Algebra is a collection of operations to Manipulate Relations.

Relational algebraic operation can be divided into basic set-oriented operation like Union, Difference, Intersection, Cartesian product etc. Relational oriented operation like:- Join, Selection, Projection & Division etc.

We shall understand various operation of relational algebra one-by-one.

Set oriented Operation :-

Set union :-

The result of union operation is denoted by the symbol ∪.

Suppose taking X & Y are two relation.

X holding the details of employee working on project J1.

Y holding those who are working.

The two table are :-

X:

Emp-No.	Emp-name
101	Rita
103	Rita
104	Jyoti
106	Leela
107	Neema

Y:

Emp-no.	Emp-name
101	Rita
105	Isha
107	Neema
112	Aditi

Abhay Kumar N
Lecturer, Dept. of E

$X \cup Y$

Emp-no.	Emp-name
101	Rita
103	Ritu
104	Jyoti
105	Isha
106	Leila
107	Neema
112	Aditi

- **Set Intersection :-**
  - * Denoted by $\cap$
    i.e. $X \cap Y$
  - * It include all tuples that are in both X & Y.

$X \cap Y$

Emp-no.	Emp-name
101	Rita
107	Neema

Note :- There are two emp working project J_1 & also J_2.

Set Difference :-

* Denoted by X-Y.
* To find tuples that are in one relation but not in another relation.

X-Y

Emp-no.	Emp-name
103	Rita.
104	Jyoti
106	Leila

Y-X

Emp-no	Emp-name
105	Isha
112	Aditi

- **Cross Product** :- (Cartesian Product)

~~X~~ X x Y

X:

Emp-no.	Emp-name
101	Rita
103	Reeta
104	Jyoti
106	Leila
107	Neema

Y:

Project
AX10
AX11

Emp-no.	Emp-name	Project
101	Rita	AX10
101	Rita	AX11
103	Reeta	AX10
103	Reeta	AX11
104	Jyoti	AX10
104	Jyoti	AX11
106	Leila	AX10
106	Leila	AX11
107	Neema	AX10
107	Neema	AX11

Abhay Kumar Mishra
Lecturer, Dept. of B.C.A.

Relational Oriented Operation :-

Select (σ) :-

The select operation extract (निकालें) specific (विशेष) tuples from a relation.

It is denoted by Sigma (σ)

Example1 :-

Suppose you want to see tuples which have Branch-name = "Janakpuri" from Deposit table.

We can select like this :-

σ Branch-name = "Janakpuri" (Deposit)

In general select operation is denoted

σ <select-condition> (<relation name>)

We can use this

(=, ≠, <, ≤, >, ≥)

Use logical connection :-

And (∧) OR (∨) & NOT (¬)

Example 2 :-

Suppose we want tuples which contain the faculty name " " who teaches RDBMS.

From Relation (table) Teach.

σ name = Pooja $\wedge$ course = "RDBMS" (Teach).

Example 3 :-

Find tuples in which the balance amount is any more than ₹ 7,000 /=

σ Balance > 7000 (Deposit)

Project (π)

* The project operation is a unary operation which extracts attributes (columns) from a relation.

* It is denoted by pi (π)

* The general form of project operation is

$$\pi_{<\text{attribute list}>}(R)$$

Example :-

Suppose we want to know the customer name. 3
the Bank Balance amount from deposit relation

π (customer_name, Balance) (Deposit).

Abhay Kumar
Lecturer, Dept. of
Maharaja College, Ara

III Join ($\bowtie$) :-

The join operator is the name suggests allow the combining of two relation to form a single new relation.

The join operation allows the processing of relations existing between the operand relation.

Example :-

Given the Employee & Salary relation of figure. If we have to find the salary of employee by name to we join the relation Emp with in salary.

Such that the value of the attributes Id in Emp is same as their Salary.

Emp-id	Name
101	Jon
102	Mon

Id	Salary
101	5000
102	6000

Emp $\bowtie$ Salary =

π (Name, Salary) (Emp $\bowtie$ Salay)

Emp-id	Name	Salary
101	Jon	5000
102	Mon	6000

Division – 1

A	B
a_0	b_1
a_1	b_2
a_2	b_1
a_1	b_1
a_0	b_2

Design an E-R diagram for airline reservation system consisting of flights, aircrafts, airports, fares, reservation ticket, pilot clearly highlight the entities relation

Give an example of each of the following relationships.

a) One to one.

b) One to many.

c) Many to Many.

What do you mean by specialization? Explain giving example.

What are drawbacks of relation DBMS.

Classification of Data Model.

Abhay Kumar Sh
Lecturer, Dept. of B
Ara (M

- **Super Key** :-
A super key of an entity is a set of one or mo attribute whose combined value uniquely identifies entity in the entity set.

(emp-name, add)

- **Primary Key** :-
Uniquly identifies each record in a table.

- **Candidate Key** :-
A candidate is an attribute or set of attribute that uniquly identifies a record runaming alternate key.

Composite Key :-
We will have tables that will use more than column as part of the primary key.

Secondary Key :-
It is an attribute or combination of attribute that may not be a candidate key.

Exampl
Let
no

Example :-

Let us take another ex. suppose we want to print names & addresses of those faculty member who teaching "RDBMS" course.

Then we write query.

π name , address (σ course = "RDBMS")$^{(Teach)}$

Relational Calculus :-

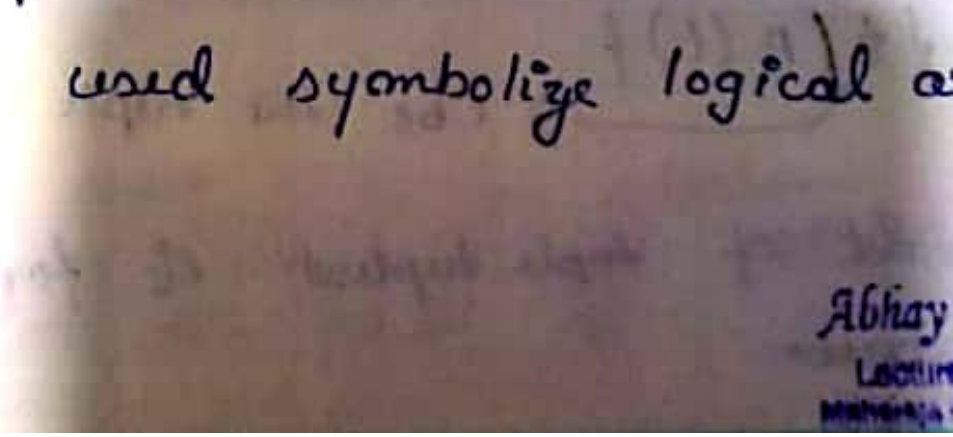
Relational Calculus is of two (2) types :-

1) Tuple Relational Calculus.

2) Domain Relational Calculus.

In Relational Calculus a query is expressed formula consisting of no. of variables & an ion involving these variables.

Relational Calculus wh calculating with predicates calculus. The latter formal language used symbolize logical a in mathematics.



In formal logic like:

If P & R are proposition we can say not P, P or R, P and R and so on.

Predicate:

Consider that statement

WXY is a company

i.e. is a company (WXY)

Ram is taller than Shyam.

i.e. taller than (Ram, Shyam).

2) Tuple Calculus (TC)

i) The tuple calculus is a non-procedural (The relational algebra was procedural).

ii) A query in tuple relation calculus is as variable.

$\{t \mid P(t)\}$

t be the tuple of P

* i.e. Set of tuple tuplest t for which pre true.

iii) We also use

* $t[a]$ to indicate the value of tuple t on attribute a

* $t \in t$ to show that t is in relation.

lational Calculus :-)

The tuple calculus is a non-procedural language (The relational algebra was procedural).

A query in tuple relational calculus is exp ssed as variable.

$$\{ t \mid P(t) \}$$

- t be the tuple
- Lower bar namely that t be tuple of relation emp.
- for which predicate P true.

We can also use $t(a)$ to indicate the va f tuple t on attribute a.

$\in R$ to show that t is in relation.

ondition is in the form of

$x=y$, $x<>y$, $x<=y$ $x>y$, $x>=y$ etc.

* * We can use

* The tuple relational calculus is based on specifying a no. of tuple variables.

Example :-

Consider the following relation :-

emp (emp-code , emp-name , Design , Doj , salary
Dept-code).

Dept (Dept-code , Dep-name , location).

1*) To find all employees whose salary is above ₹ 5500.

$\{t \mid emp(t)$ and $t.salary > 5500\}$

The ranges relation of table variable t is emp.

Condition.

2*) To find the selected emp-name & salary > 5500.

$\{t.emp\text{-}name , t.salary \mid emp(t)$ and $t.salary > 550$

3*) To retrieve date of joining & designation of the employee whose name is "Jyoti" is given by.

$\{ t.Doj, t.Desig \mid emp(t) \text{ and } t.emp\_name = \text{"Jyoti"} \}$

4*) To find the emp-name & emp-code of all employee who work for the "Research" departm

$\{ t.emp\_name, t.emp\_code \mid emp(t) \text{ and}$

$(\exists d)$ department (d) and Dept_name $=$ "Research" and $d.dept\_code = t.dept.code.$

existance Quantifier — Tuple of dept record.

Domain Calculus

* An expression is of the form

$\{ x \mid f(x) \}$

$\{ \langle x_1, x_2, x_3, x_4 \ldots x_n \rangle \mid f(x_1, x_2 \ldots x_n) \}$

$\{ x_i$

$x_i \mid$

where x is represent domain variables

A in a formula on x.

* The comparism operator used → $\{ =, \neq, <, \leq, >$
* We can use logical operator → $(\wedge, \vee, \neg$
* An atom in the domain calculus is in following form:-

1) $x \in R$ where R is a relation.
   or
   $\langle x_1 \text{---} x_n \rangle \in R$ where x is attribute.

2) $X \theta Y$ or $X \theta C$ where θ is on the co
   on operator $\{ =, \neq, <, \leq, >, \geq \}$

* Formulas are built from atom using fol
  rules.

* An atom is a formula.
* If f & g are formula then,

$\neg f$, $(+)$, $f \vee g$, $f \wedge g$, $f \rightarrow g$ are also formula.

B_3 If $f(x)$ is a formula where x is then $\exists x\, f(x)$ and $\forall$ also formula

$\forall$ → universal quantifier.

$\exists$ → quantifier

Consider the following Query :-

* Find branch name, Loan no., customer-name amount for Loans are $ 1200.

$$\{ < b, l, c, a > / < b, l, c, a > \in \text{borrow} \wedge a > 1200$$

* Find all customer who loan fo amount > than $1200.

$$\{ < c > / \exists\, b, l, a\ (< b, l, c, a > \in \text{borrow} \wedge$$

← Project (project #, project_name,

← Emp (Emp #, emp-name)

← Assigned_To (project #, Emp #)

Abhay Kum

$\neg f$, $(+)$, $f \vee g$, $f \wedge g$, $f \rightarrow g$ are also formula.

B_3 If $f(x)$ is a formula where x is then $\exists x f(x)$ and $\forall$ also formula

$\forall$ → universal quantifier.

$\exists$ → quantifier

Consider the following Query :-

* Find branch name, loan no., customer-name amount for loans are $ 1200.

$$\{ < b, l, c, a > / < b, l, c, a > \in borrow \wedge a > 1200$$

* Find all customer who loan fo amount > than $1200.

$$\{ < c > / \exists\ b, l, a\ (< b, l, c, a > \in borrow \wedge$$

← Project (project ≠ , project_name,

← Emp (Emp ≠ , emp-name)

← Assigned_To (project ≠ , Emp ≠)

Abhay Kum

Get employee_no for employee working on project COMP 353.

$$\{ c \mid \exists P \ (\langle c, P \rangle \in \text{Assigned\_To} \wedge P = \text{comp'353'}) \}$$

↓

It used for known value of P.

The quantitative can be dropped

$$\{ c \mid \langle c, P \rangle \in \text{Assigned\_to} \wedge P \ \text{comp'353'} \}$$

Compile the details of employee working on a database project can be

$$\{ e, m \mid \exists P_1, e_1, P_2, n_2 \ (\langle P_1, e_1 \rangle \in \text{Assigned to} \wedge \{e, m\} \ \text{employee} \wedge \langle P_2, n_1, n_2 \rangle \in \text{project} \wedge e_1 = e \wedge P_1 = P_2 \wedge n_2 =$$

SQL → (Structured Query Language)

SQL is the standard language for making queries in relational database management packages such as:- Oracle, Sybase, SQL server, Ingres.

The standard language for accessing client / server database is also SQL.

SQL language was 1st commercially implemented by Oracle corporating in the year 1979, there was no official standard until 1986.

In 1986, SQL was jointly published by ANSI (American National Standard Institute) & ISO (International Standards Organization).

SQL-92 was also developed in with more standard.

Advantages of SQL :-

SQL has several simple to use commands for data processing activities.

We can use SQL commands for different following purpose.

a) Creating (defining), deleting, structure.

b) Defining relationship between two or more table.

c) Inserting data into tables.

d) Updating data.

e) Controlling a database.

* SQL in Action

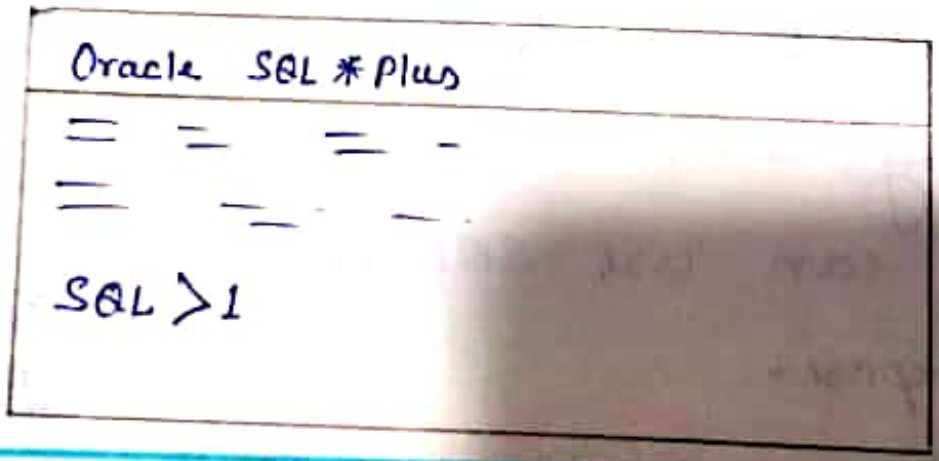
SQL *plus tool provided by oracle corporation USA.

To start SQL *plus do the following:-

1) Click start menu → Oracle → Application Development → SQL plus.

2) Enter user name & password Log on then click

3) SQL *plus window shown.



a) DDL (Data Definition Language Commands).
b) DML (Data Manipulation " ").
c) DCL (Data Control " ").

DDL :-

Data Def: In SQL is via the create statement.
The statement is used to create a table, indu
etc.

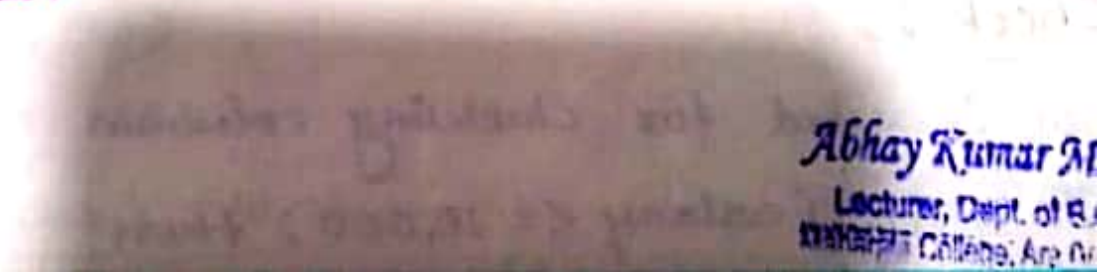
Integrity Constraint :-

While creating a table, you can place certain li
tation on the values stored in the table. Differin
constraints applicable to SQL are:-

Not NULL → The Not NULL constraints ensures that the user
ays type the data for that column.

NIQUE → This is a constraint for the column in a table.
The Primary key constraint ensures that the colu
cannot be & that the values in the column
unique.

mary Key.



d) Check → The check constraints ensures that colum entered the data in the column is limite fic values.

e) Default.

f) Reference.

a) Not NULL :-)

Not NULL constraints prevents a column fro ng NULL value. If you try to insert a NULL such a column it will be reject.

NULL mean Not zero means absence of any d at column.

b) UNIQUE :-)

Values entered into a column are all differe unique. A column with this constraints will duplicate value.

c) Primary Key :-)

It is used to declare Primary Key.

d) Check :-) c

It is used for checking column

It check (salary <= 10,000) then

Default :-

It used to assign default values to a column befor any value to be assigned

For example :-

Salary default 0 will assign default valu salary 0.0.

(यदि उस कॉलम में value नहीं देंगा तो वहाँ zero हो जाएगा)

References :-

A foreign key has value which for the primary k in another table.

- Dept_code.

Create table <Relation> <attributi>

attribute list = <attribute name> (<data table>)
(not NULL)
[<attribute list$_2$>

Create table employee.
(
Emp-no. integer not NULL,
Name char (20),
Skill char (25),
Pay_Rat Number (10, 2),
);

SQL)

```
Creati table Department.
(
 Dep-code  number(2)  primary key,
Dep-name  VarCHAR 2 (10) NOT NULL.
Floor   number (1)  Not NULL,
);
```

SQL)

```
Creati table Employee.
(
Emp-code number(4) Primary key.
Emp-Name  VARCHAR 2 (25) NOT NULL
DESIG  CHAR (10)  NOT NULL
Head  Number (4)
```

Alter Table Command :-

```
Alter Table <table_name>
[options];
```

```
Alter Table employee
Add phone char(7);
```

One can change the difinition of a table even aft
creating it. For this Alter Table command is us

This command can be used to ad
columns to the table, change their sizes, ch
their datatypes.

Add options

Add keyboard is used to add column or const
nts to the table.

Ex:- Alter Table employee
Add Phone char(7)

Modify Options :-

The modify option can change the data type,
width & constraints of an existing column.

Ex. Alter Table employee
Modify Emp-name varchar2(30);

Abhay Kum
Lecturer, Dep

create table <relation> (<attribute list>);

Where the <attribute list> is specified as:-

<attribute list> = <attribute name> (<data type>) [not NULL]

[, <attribute list>].

Not zero

Integrity Constraint.

(certain limitation for

Create table Employee.

Datatype in SQL :-

CHAR :-

It defined character size 1 to maximum size is 255.
Such as:- A.....Z , a,b.....z)
Character like @, #, &, etc.

VARCHAR 2 (size)

It is similar to char size

Number (P,S)

- P → Total no. of digit.
- S → Total no. of digit possible to right the decimal part.

Date :- (DD-MM-YY)

Long :- Variable length character string containing 2Gb.

SQL

```
Create table Employee.
(
emp_code number(4) primary key.
emp_name character (25) Not NULL
CHR (10) Not NULL.
```

DOJ Date NOT NULL ;

Number (4)

);

Table Created.

- SQL >

SQL > describe Employee

Desc.

Name	Null ?	type
Emp-code	Not Null	
Emp-name	Not Null	Vchar
	Not Null	
Doj	Not Null	
Head		

Insert into emp values (&emp, '&emp-no', '&DATE')

6) Data Manipulation : SQL.

We present the idata manipulation statement sup ted in SQL.

SQL provide the following basic data manip tion statements.

Select, update, delete, & Insert.

Select Statement :-

The select statement, the only data retrieval in SQL.

DML :- (Data Manipulation Language.)

We present the data manipulation statement supported in SQL.

SQL provide the following basic data manipulation statement.

1) Select 2) Update 3) Delete 4) Insert.

Select Statement :-

It is very powerful every command in SQL.

The select statement, the only data retrieval in SQL, specifies the method of selecting the tuples of the relation (s).

```
Select [ distinct ] < target list >
from < relation list >
[where < predicate >]
```

use operator
(>, ≥, <, ≤, and, or, not, =, ≠)

Distinct :-

It is used in the select statement to eliminati

Abhay Kumar Mi

From, where → Clause.

* Select name
  from employee;

* Select * (details)
  from department
  ↓

(Suppose you want to retrieve all the data sto
department table, then type the following).

To display emp-code, emp-name, Desig
from all employee.

Select emp-code, emp-name, Desig, Basic f
loyee.

Example:-

To display as your where Desig = Salesman

Select Emp code, Emp-name
Desig, Basic
From Employee,
where Desig = 'Salesman';

Scanned with CamScanner

Insert into Dept
values (10, 'Account', 1);

Specified the method of selecting the duplis of the rel. (2):

Select [distinct] <target list>
from <relation list>
[where <predicate>]

Distinct :- It used in the select to eliminate duple in the result from, where — clause.

Select Name
from Employee

or

Select Distinct Name
from employee.

3) Update :-

It is used to modify one or more records in specified relation.

Update <relation> set <attribute_value_list>
where <predicate>.

```
Update Employee
Set pay-rate = 7.85
where      name = 'RON';
```

4) **Delete** :-

It is used to delete one or more record from a relation.

```
delete <relation> [where <predicates>].
```

Example:-

```
delete Employee
where Name = 'RON'
```

Select

```
Oracle Select *
SQL> Select * from emp;

emp-no.   emp.name
  10        Ram
  20        Ron
  30        Mon
  40        Dom
SQL
```

Insert

```
SQL> Insert into
     Dept
```

- Suppose we want to see details of only those emp whose designation is 'Salesman':

```
Select Emp_code, Emp_name, Basic, from
where Design = 'Salesman'
```

```
Select <column_name(s)>
from <table name>
where <column name> <operator> value;
```

```
Insert into table
values (<value list>)
```

Update

```
Update <table name>
SET <column name = new value>
[where <condition>];
```

Aggrate Function :-

SQL aggrigate function such as:-

a) SUM
b) AVG
c) MAX
d) MIN
e) COUNT

Produce a single value for the entire group of table entities.

Aggregate or column function can also be along with the select command.

SUM :-

Sum function calculates the arithmetic sum of selected value of a column.

For example:-

Find the sum of Basic Salary from the e

```
Select sum(Basic) from emp;
```

b) Min / Max :-)

MIN function calculates the smallest of all se
ted values of a given column

MAX is highest.

For example :-)

```
Select min (Basic ) from emp;
Select Max (Basic from emp;
```

c) Count :-)

This function counts the number of rows in
output table.

For example :-)

To find how many emp. are in the emp.

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

Find out no. of department in the emp. table.

```
Select count (Dept. code ) from emp;
```

oins :-)

The syntax for select statement, where we join t
tables

Select <column name (s)>
from <table1>, <table 2> ..... <table N>
where <table 1. col1> = <table . col1>
And <table 3. col 2> : <table 2. col 2>

se example:-)

If we wish to retrieve emp-code, emp-name & the Dept-name of all the employees we will have to use two tables, namely:-

Employee & Department.

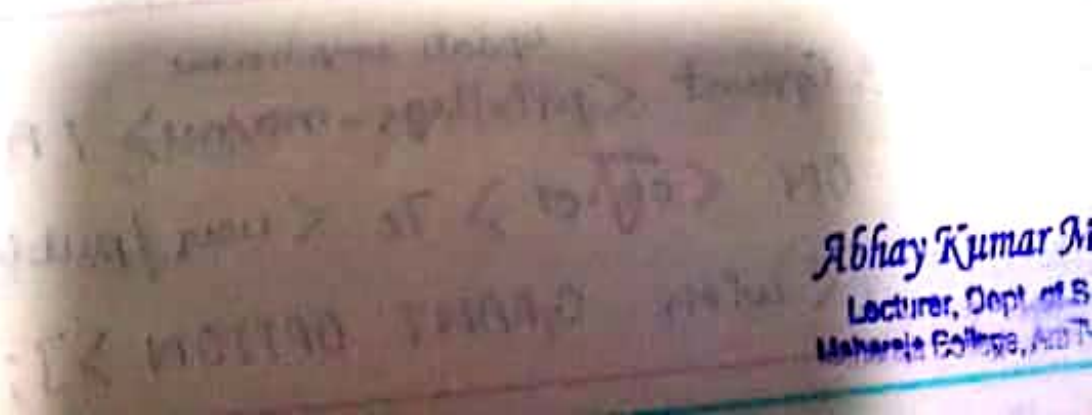
This is because

Emp (Emp-no., Emp-name) Dep (Dept-name, Dept-co

elect emp code, emp name, Dept name from

Emp, Dept.

where, Emp. Dept-code = Dept. Dept-code;



c) DCL :- (Data Control Language)

DCL is an another portion of SQL which allo definition of a security mechanism or schema protecting data from unauthorized users.

DCL consists of features that determ whether a user is permitted to perform a p cular action.

It contain command like :-

* Grant.
* Revoke etc.

These are also known as Transaction control ands.

* Grant Command :-

The Grant command is used to permit us access to the database.

Syntax :-

```
Grant <privilege-name> / All
ON <object> To <user/public>
[<with GRANT OPTION>];
```

* Privilege name, specifies the action such as Insert, delete, update, select etc.
* The user can perform on the object.
* ALL - Clause.
* <User> the name the user to whom the privileges has been granted.
* To grant the privileges to all the user public option can be use.
* [with Grant option] is optional.

Example:-

A query to provide select permission on the Departm ent table to user 'mini' & allow to give the furthe grants will be as follow.

Grant Select on Dept.
To mini with Grant option;

We can also grant privileges on individua Column in a table.

Abhay Kumar Mi
Lecturer, Dept. of B C
Kishanganj College Aur Al

For example:-

To give user 'mini' the privilege to update column .emp-name.

```
Grant update empname
ON Emp to Mini;
```

* Revoke Command :-

This command is used to cancel database privileges from user(s).

Syntax:-

```
REVOKE <privilege-name> / ALL
ON <object> from <user / public>;
```

Example :-

A query to revoke the privilege we gave the user 'mini' will be as follows:-

```
REVOKE Select on Dept
From Mini;
```

Introduction to QUEL :-

INGRESS → (INTERACTIVE Graphics & RETrieval System).
is a RDBMS developed at the University of California at Berkeley.

This project ran almost concurrently with the system R project at IBM'S San Jose Research Center.

- QUEL (QUERY Language), the data manipulation language for INGRESS, is based on relational tuple calculus.

Unlike, SQL it does not support relational algebraic.

The original version of INGRESS, is used extensively in the academic & rum-under UNIX on VAX systems; is well as workstations based on the MC68000 of Micro-processor.

* Data Definition :-

The basic statement used to define relations & access aids in QUEL are :-

create, range, index, destroy & Modify.

- **Create Statement :-**

The create statement is used to create a new relation. Its syntax is :-

- **Syntax :-**

> Create <relation name> (<attribute list>)

where <attribute list> is defined as :-

<attribute list> ::= <attribute name> : <format>

Example :- The Statement <attribute

Create NEW_DUTY_ALLOCATION (Position_No = i
Empl_NO = i, shift : i,

New relation called NEW_DUTY_ALLOCATION
attributes position_No., Emp_no., Shift, & Day
all the attributes defined as Integer.

- **Range Statement :-**

Tuple Variable (known as range variable in
although we will continue to refer to
by the familiar term) are defined
range declaration statement it's usage is

range of <tuple variable e> is <relation name>

Example :-

range of d is DUTY_ALLOCATION

range of e is Employee.

The tuple variable d & e, at any given time refer to a tuple in the duty-allocation & Employee relation respectively.

* Data Manipulation : QUEL

The basic data retrieval statement in QUEL is retrieve. It is used for both projection & selection.

retrieve unique target list

Example :-

1) Project the Duty-allocation relation on the position No. & Emp-No attributes. Range of d is DUTY-allocation retrieve (d.position no, d.emp no)

2) Range of d is DUTY-allocation
retrieve (d.all) where d.emp-no = 1234
!= , <= , >= instead of

* **QBE** → (Query - By - Example).

* It is developed by M.M. Zloof at IBM's Yor Height Research Laboratory.

* It has since been marketed by various rela system form IBM as part of their QM (Query Mangt. facility).

* In QMF to QBE queries is translating equivalent SQL Queries.

* RDBMS such as DBASE - IV, INGRESS & have some form of ex. or form based system.

* It is based on domain calculus & two dimension at syntax. (Horizontal &

* The queries are written in two of a table.

* Queries are formed by entering an a possible answering skeleton (empty)

* The Skeleton Table does not have
  • The 1st column is used for relat