order by (sal, doj);

wire centre.

Query (DQL) → select

Emp (Eno, Ename, dept, salary, doj)
(Eno → 1° key)

→ extract out relevant info.

Emp (Eno, Ename, dept, salary, doj)

select Eno, " " " " from Emp;

" dept, sal, eno, ename, doj from Emp;

columnwise — select * from Emp;

reporting change { Select Eno as "Employee Number", Ename as "Employee name" from emp;

Select eno, ename from emp

rowwise — (where) salary > 10000 and/or dept like 'Network';

(not like → for string/textual data)

<
>=
<=
<>

select * from emp where salary in (5000, 10000, 25000);

" " " " " " between (5000, 25000)
→ "range"

select distinct salary from emp.

select * from emp where dept is NULL;

is NOT NULL;

" " " " " ename like 'A%'; → starts w/ A

'%A'; (string finishes w/ A)

'%A%'; (A in middle)

select * from emp order by salary;

→ all data rearranged in ascending order.

order by salary desc; → descending

DBMS - lab

SQL Plus

student/student@172.16.1.210:1521/oracle

SQL>

Emp

Eno	Number
Ename	varchar2
Job	char
Mgr	Number
Hiredate	Date
Sal	Number
Comm	Number
Dept No	Number

Dept

DNO	Number
Dname	Varchar
Location	varchar

2) Add a new attribute phone to the table emp
3) Change the datatype of job from char to varchar
4) Remove the attribute phone
5) Describe the table
6) Insert atleast 5-7 records per table

- Drop - to delete the table (but table must have contents)

  drop table table_name

- Alter - modify str. of existing table

  alter table table_name
  add phone number(10);
  modify
  rename
  drop/delete

```
insert into Student values(
    &r, '&n', '&d', '&do');
    Enter new record for roll:-
```

to delete all records:-

```
delete from Student
```

to delete particular record:-

```
delete from student where roll=01, name like 'Ansari';
                          roll
```

to update:-

```
update emp
   set salary = 5000

update emp
   set salary = salary + 5000
   where empno = 14;
```

~~check (CLIENTNO~~

```
check(client_No LIKE 'C%'));
```

```
SELECT COUNT(CITY) - COUNT(DISTINCT(CITY)) FROM STATION;

LEN(string)

order by column1, column2, ... ASC|DESC;

top 1
```

Salespeople (snum, sname, city, comm)
Customers (cnum, cname, city, Rating, snum)
Orders (onum, Amount, Odate, cnum, snum)

Foreign key - normal key in one table referring to primary key of other table.

domain
data size
" width
} must be same

create table salespeople (snum number(4) primary key,
sname varchar2(30)
city " (15),
comm number(7,2));

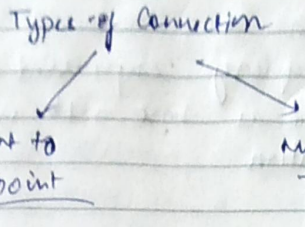
create table customer (cnum number(4) primary key,
cname varchar2(30),
city " (15),
Rating number(4),
snum number(4) references salespeople(snum));

create table orders (onum number(4) primary key,
amount number(7,2),
Odate date,
cnum number(4) references customer(cnum)
snum " " " salesperson(snum));

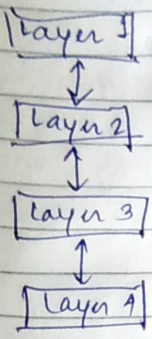
Aggregate func → works on single col → single value result

sum
avg
max
min
count
}

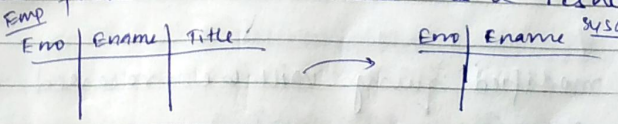
select count(*) from tablename;
select max(amt) "Maximum Amount"
from Orders;



* Layered architecture.



Database view is a subset of a database and it is derived from base relations as a result of a query



View provides security from unauthorised access.

CREATE VIEW SYSAN (ENO, Ename) AS SELECT ENO, Ename from Emp;

View in distributed DBMS

It may be derived from fragmented relⁿ stored at diff. sites.

When a view is defined its name and its retrieval query are stored in a catalogue.

View definitions can be centralised and at one side, partially replicated or fully replicated.

However evaluating views in distributed systems can be costly. In centralised system views can be modified as a base relⁿ unlike in distributed relⁿ

Query Modification

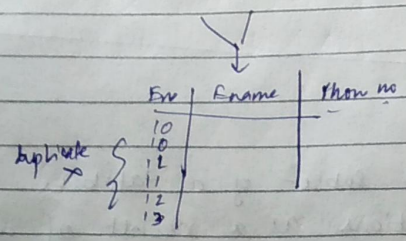
Mapping the query expressed on views into a query expressed on base rel^n. is done using query modification

Select Ename, Eno, RESP from
Emp, Asg
where Emp.Eno = Asg.Eno

↳ join query

Eno	Ename
10	
11	
12	

PNo	Phone no.
10	-
11	
12	-
13	-



and the modified query will be processed by the query processor

For distributed DBMS also we use query modification for mapping. The qualification defining a views is found in the distributed database catalogue and then merged w/ the query to provide query on base rel^n.
Such a query is called distributed query.
To avoid view derivation actual versions of the view are maintained called materialized views.
It stored the tuples of the views like other database tuples through indices which makes the access to this materialized views faster.

This materialized views can involve aggregate func, grouping etc.

Maintenance of materialized views:-

As materialized views are a copy it must be consistent with the base data. They are more complex as it involves complex queries.

~~Refreshing the data~~

refreshing can be done ~~by~~ in diff times

1. periodically
2. forcedly
3. lazily