'any' or 'all' we use operators but we don't use operator 'in' → 'in' & 'not in', → between → not between

Row - Tuple
column → Fields or Attribute

→ Database Management system Vs File system
→ Database instance/schema
→ Data Models
→ 3-tier Architecture / Database Abstraction / 3 level schema / Data Independence

Ques Write sql queries for
supplier (sid, sname, city)
pares (pid, pname, color)
orders (sid, pid, quantity).

(i) Find the name of suppliers who belongs to lucknow, kanpur & varanasi.
select sname from supplier where city in ('Lucknow', 'kanpur', 'varanasi');

(ii) Find the name of supplier who have ordered red color parts.
select sname from supplier, parts, orders where supplier.sid = orders.sid and parts.pid = orders.pid and color = 'red';

(iii) Find the name of city from where more-than 10 suppliers belongs.
select city from supplier group by city having count(city) > 10;

(iv) Find the name of supplier who have ordered all parts.

also max(), min(), avg()

(v) Find the name of supplier who belongs to lucknow and orders parts in quantity more than 5000.
select sname from suppliers, orders where supplier.sid = orders.sid and city = 'Lucknow' and quantity > 5000;

(vi) Find the name of supplier who have ordered P1.
select sname from suppliers, orders where suppliers.sid = orders.sid and pid = 'P1';

Note → When we write select * from S1, S2 then it performs cross join or cartesian product.

→ To avoid duplicacy in any table and to get only one time, then we use distinct keyword.
e.g select distinct name from student.

will display Ajay once only

Roll	Name	Branch
101	Ajay	CS
102	Ajay	ME
		CS

→ To concatenate two or more columns we use concat.
eg. Aqsa Khan, given in two columns.
select concat('fname', 'lname') as name from student;

→ order by: to get info in ascending or descending order. (by default ascending).
e.g: select * from student order by marks asc/dsc;

→ limit: to get only a given no. of result.
e.g. select * from student order by marks desc limit 5;
will display top 5 students.

for derived values. if aggregate function use having instead of where. if we get the value straight from the table then we use where keyword.

e.g select * from student order by marks desc limit 5, 2;
will leave top 5 and display the next two.

Nested query:
select name from student where marks = (select

Group by: will give result of top entry. e.g.
select * from student group by Branch.
will display only one students entry from each branch.

select Branch, count(Branch) from student group by Branch;

to get this output →

Branch	count
CS	2
ME	2
EC	1

Branch
CS
ME

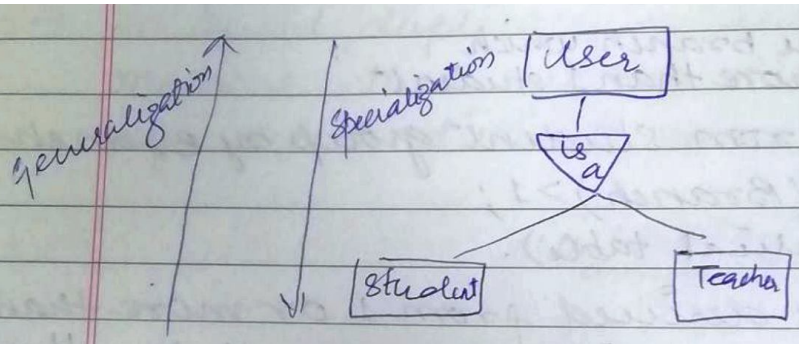
to get the branch which have more than 1 student

select Branch from student group by Branch having count(Branch) > 1;

View: (it is a virtual table).
a table which is derived from 1 or more than 1 tables is known as virtual table. It doesn't occupy space. Insertion & deletion is difficult in it.

create view result as select S1.Roll, S1.Name, M1, M2, M3 from S1, S2, S3 where S1.ROll = S2.ROll and S2.ROll = S3.ROll;

View can be used only for updation
(See purpose of view).



Relational Algebra.

It is a procedural language.

Operator

Basic
- π projection
- σ selection
- X cross product
- ∪ union
- − set difference
- ρ Rename

Distinct
- ⋈ join
- ∩ intersection
- / division.

See the syntax.

SQL & Relational Algebra

to select names of students.

select sname from supplier

to select sid & sname

π_{sname} (supplier)

$\pi_{sid, sname}$ (supplier)

supplier (sid, sname, city)
parts (pid, pname, color)
orders (sid, pid, quantity

π points to columns.

$\sigma_{city='lucknow'}$ (supplier) ↑ same as select * from supplier where city="lucknow"

σ points to columns.

to select name of those who belongs to city lucknow

π_{sname} ($\sigma_{city='lucknow'}$ (supplier))

→ select orders.sid from supplier, orders where supplier.sid = orders.sid and quantity > 5000

$\pi_{orders.sid}$ ($\sigma_{supplier.sid = orders.sid \text{ and } quantity > 5000}$ (supplier × orders))

Imp/ Find the sid of suppliers who have ordered no parts.

- π_{sid} (supplier) − π_{sid} (orders).
- select sid from supplier where sid Not In (select sid from orders)

select supplier.sid from supplier left outer join orders where supplier.sid = orders.sid and orders.sid is Null;

Join

inner (matching) → Natural join

outer → left, right, full.

Ques Find the name of suppliers who have ordered all products.

1 mark Diff b/w RA & SQL

Set difference and divide in RA.

SQL is a structured query language
Relational algebra is a procedural language

imp TO get the name of suppliers who have ordered all parts

$\pi_{sid,pid}(orders) / \pi_{pid}(parts)$

orders

sid	pid
S1	P1
S1	P2
S2	P1
S2	P2
S1	P3

parts

Pid
P1
P2
P3

~~Difference between Relational Algebra and SQL~~

RA

$\pi_{sid}(orderss)$

↑ it by default gives distinct values

SQL

select distinct sid from orderss.

↑ here we have to provide the distinct keyword to do so.

Suppliers who have ordered the maximum no. of products.

select sname from suppliers where sid = (select sid from orders where quantity = (select max (quantity) from orders));

To get ~~the~~ how many ~~no of~~ suppliers ~~from~~ belongs to a city

select city, count(city) from supplier group by city;

Name of city from where more than 7 suppliers belongs.

select city from supplier group by city having count(city) > 7;

max, min, avg, count, sum } aggregate function & we use having in place of where.

count does not count the null value.

select output

select * from supplier group by city;

Functional Dependency

$A \rightarrow B$

It follows Armstrong rule.

(i) Reflexive: If $B \subseteq A$ then $A \rightarrow B$

(ii) Augmentation: If $A \rightarrow B$ then $AC \rightarrow BC$

(iii) Transitive: If $A \rightarrow B$ & $B \rightarrow C$ then $A \rightarrow C$.

There are three additional rules also.

(i) Union: If $A \rightarrow B$ and $A \rightarrow C$, then $A \rightarrow BC$
or If $A \rightarrow B$ and $C \rightarrow D$, then $AC \rightarrow BD$.

(ii) Decomposition: If $A \rightarrow BC$ then $A \rightarrow B$ and $A \rightarrow C$.
and if $AC \rightarrow BD$ then $AC \rightarrow B$ and $AC \rightarrow D$.

(iii) Pseudo Transitivity: means false
If $A \rightarrow \alpha B$, $\alpha B \rightarrow \alpha C$ (added α on both sides.)
then $A \rightarrow \alpha C$

Q $R(A, B, C, D)$, FD's $A \rightarrow B$, $B \rightarrow CD$
$R_1(A, B)$ $R_2(A, C, D)$
$A \rightarrow AB$ or $A \rightarrow ACD$.
So, this condition satisfies $A \rightarrow B$, so, it is.

Ques $R(X, Y, Z, W)$, FD's $(X \rightarrow Y, Z \rightarrow W)$
$R_1(X, Y)$, $R_2(Z, W)$
It is lossy or lossless?
It is lossy as the tables do not have common
But the dependency is preserved here as X, Y are both together in R_1 and Z, W are together in R_2.

tuple = rows.

σ = row select

π = column project.

Remember this rule in lossy or lossless.

$R_1 \cap R_2 \cap \ldots R_n \longrightarrow R_1$

or

$R_1 \cap R_2 \cap \ldots R_n \longrightarrow R_2$

$\vdots$

$R_1 \cap R_2 \cap \ldots R_n \longrightarrow R_n$.

Normalization

(1) Functional dependency

$X \longrightarrow Y$

If $t_1(x) = t_2(x)$

then $t_1(Y) = t_2(Y)$

Ques find lossy or lossless. Dependency preserved? Find key also

$R(A,B,C,D,E,F)$, $R_1(A,B,C)$, $R_2(B,C,D)$

$R_3(B,E,F)$

FD's : $A \to BE$, $E \to F$, $C \to D$, $D \to BC$.

$B \to ABC$ or $B \to BCD$ or $B \to BEF$.

Since, B is not on the left hand side in above FD's so it is a lossy.

since all the FD's are not satisfying above R's, so, it is ~~no~~dependency is not preserved.

$A \to B$

$\downarrow$

$E \to F$

$C \rightleftarrows D$

AC or AD can be key.

Ques find key $R(A,B,C,D,E,F)$, FD's $A \to BC$, $C \to AE$, $E \to D$

$A \to B$

$\downarrow\uparrow$

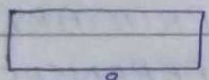
$C \to E \to D$

F

So, AF or BCF can be the key.

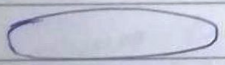
(i) ER Model
(ii) EER (Extended ER Model)
(iii) Integrity Constraints
- → Domain constraints
- → Entity Integrity
- → Referential Integrity ← Insert into child table / Delete from main table
- → Key constraints

ER Model



Entity

Relationship



Attribute

- → Specialization
- → Generalization
- → Aggregation

Diff b/w Drop, delete & truncate
Referential Integrity

Student (Roll, sname, f-name, city, marks)
enrollment (enroll, sname, 12th roll, 12thmarks, Roll)

create table student (Roll int(8), sname varchar(30), f-name varchar(30), city varchar(20), marks decimal (4,2), PRIMARY KEY (ROLL));

Create table enrollment (enroll int(10), sname varchar(30), 12th roll int(15), 12th marks decimal (4,2), Roll int(8), PRIMARY KEY (enroll), FOREIGN KEY (ROLL) REFERENCES Student (ROLL));

ERdiagram
Hospital management system.
College library management system

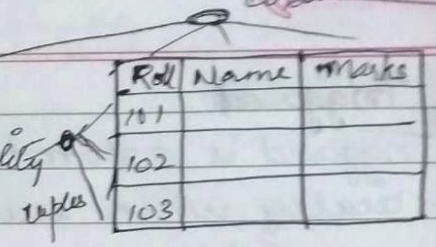
Find the detail of student who got more than 80% in their 12th.
Select * from student, enrollment where student.Roll = enrollment.Roll and 12thmarks > 80;

To increase 2 marks of each student.
update student set marks = marks + 2;

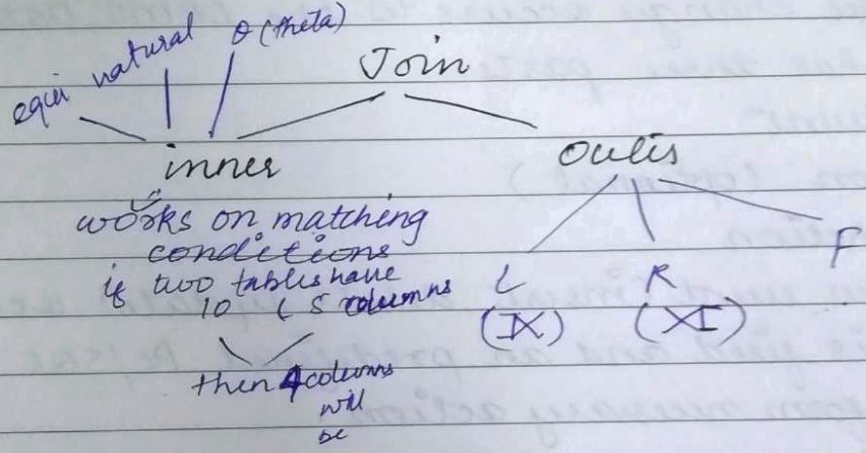
Find the name of student who have scored lowest marks.
Select sname from student where marks = (select min(marks) from student);

Aggregation: Relationship over relationship
in, not in, order by, group by

Degree refers to no. of columns or fields or attributes. Cardinality refers to no. of tuples or rows.



Library Management System.
Entity : Students, Books, Issuebook.



$\rho(R)$ (rho)
creates a copy of R.

Trigger

own: Triggered is a stored procedure which executes automatically when events like (insert, delete, update) are executed. It can execute before the execution of events or after it.

site: Trigger is a procedure that starts automatically if specific changes occurs to the DBMS. Database Trigger has three parts.

- Trigger Event
- condition (optional)
- Trigger Action

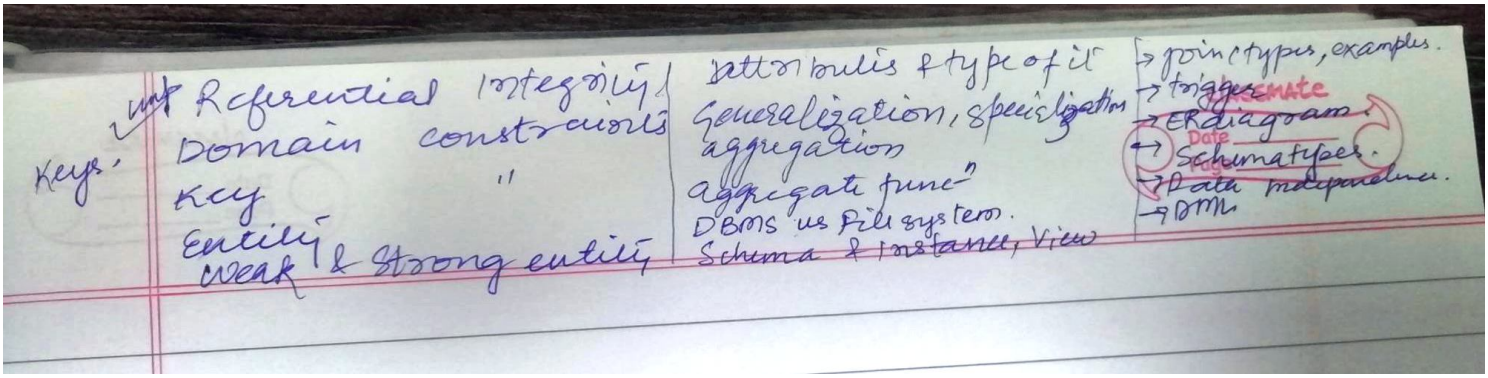
When an event (insert, delete, update) occurs, DB trigger is fired and an predefined PL/SQL block will perform necessary action.

Syntax:

```
Create [or Replace] Trigger trigger-name [before/After]
triggerEvent on tablename [for each row]
[when condition]
Declare
        declaration stmt;
Begin
        executable stmt;
exception
        exception-handling stmt;
END;
```

PL/SQL (not case sensitive)

PL/SQL : Procedural Extension Language for SQL. It is not case sensitive.


Keys:
Referential Integrity
Domain constraints
Key
Entity
Weak & Strong entity
Attributes & type of it
Generalization, specialization
aggregation
aggregate func"
DBMS vs File system.
Schema & Instance, View
→ join types, examples.
→ triggers
→ ER diagram.
→ Schema types.
→ Data independence.
→ DML
Date