Joins

cross product + condition
↓
select statement

1) Cross Join / Product
2) Natural join
3) Conditional join
4) Equi join
5) Outer join → Left
→ Full
→ Right

Natural Join → Put common Attributes equal

mxn

E-no	E-name	Address
1	Ram	Delhi
2	Vaan	Chd
3	Ram	Chd
4	Ansh	Delhi

Employee

Deptno	Name	Eno
D1	HR	1
D2	IT	2
D3	MKT	4

Dept

Q# Find the Emp who's working in a dept?
Names

```
Select E_name from Emp, Dept
where emp.eno = dept.e.no;
```

Sol. Select E_name from Emp
Natural join Dept;

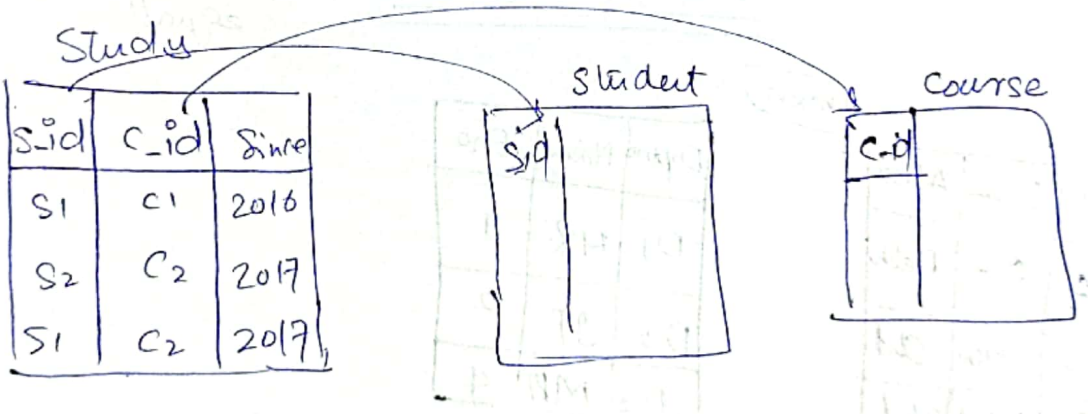
Select * emp_name
from emp INNER JOIN Dept
on emp.e_no = dept.e_no;

Diff b/w Inner join & Natural Join?

Inner join → display common column of both tables.

Natural join → display one copy of common column.

Self Join



Pk = S_id + C_id

Q# find Student_id who is enrolled in atleast 2 courses.

Ans: Select T.sid from Study as T_1, Study T_2
where T_1.sid = T_2.sid and
T_1.cid <> T_2.cid;

, HTTP

SQL	VS	NO SQL
1) Relational Database Management System (RDMS)		1) Non-relational or distributed database system. → Graph / document / column oriented.
2) Static / Fixed schema		2) Dynamic schema
3) Not suited for heirearichal data storage.		3) Suited for hierearichal data storage
4) Vertically Scalable → Adding floors to building (SSD, RAM)		4) Horizontally Scalable (SERVER) → Adding buildings to neighbourhood.
5) Follow ACID Properties		5) Follows CAP (consistency, Availability, Partition Tolerance).

Query: Name of employees not working on any Project?

Emp

eid	Name

Pro

pid	Project	eid

Select Emp.Name from Emp left join Pro P on (emp.eid = Pro.eid) AND P.pid is Null.

Query: All employees Born during 1950s. (inclusive operator between)

Select name from employee where bd between #01-01-50# and #31-12-50#;

Query: Name of employees working in same dept

emp

eid	dept	Name
1	gt	a
2	gsl	b
3	gt	c
4	Chem	d
5	Phy	e

Select emp1.name, emp1.dept, emp2.name from emp1 where emp1.eid = emp2.eid AND emp1.dept = emp2.dept group by emp1.dept.

emp	emp
a	a

↳ Select * from emp where dept IN
(Select dept from emp group by dept
having count (*) > 1);

~~emp~~ id	ename	Deptname	Sal
1	Ram	HR	10,000
2	Amrit	MRKT	20,000
3	Ravi	HR	30,000
4	Nitin	MRKT	40,000
5	Varun	IT	50,000

Query: Name of employees and dept belonging to dept having less than 2 Employee.

Select ename, ~~dept~~name from emp
where ~~count (deptname) < 2~~;

dname ~~dept~~ IN (Select dname from Emp
group by dname having count (*) < 2);

emp

id	Sal	Name	dept-id
1	34000	a	~~000~~ US
2	33000	b	Back
3	36000	c	Sach
4	3600	d	US
5	3700	e	UI

Select max(Sal), dept_id, from
emp group by dept_id;

Query: Highest Sal of each dept;

Query: find Nth max sal from table.

```
Select max(sal) from emp;
Select sal from emp order by sal desc
Limit n-1, 1;
```

4th

```
Select sal from emp order by sal desc
limit 3,1;
```

3 ↓ skip 3 rows, return 1 row

Coalesce

Query: Select even rows?

```
Select * from emp where mod(empid, 2)=0
```

odd rows:

```
Select * from emp where mod(empid, 2)=1
```

Query: Select All customers that are from same ~~company~~ country as Supplier.

```
Select *country from Customers where
country In ( Select country from Supplies
```

75 | 92

DBMS | 1000+

1) Indexing? Advantages / Disadvantages → Detail Implementation
2) Normalization / De-normalization Adv / Dis Adv.?
3) Db Schema Designing?
4) Transaction? Acidity
5) Draw Schema & implement fruit table in dB. Schema in such case?
   Write query which brings all funds of user 'Ali'.
6) How does server processes a request, generated from browser?
7) Poor table structure? ✓
8) Why we limit normalization b/w 3nf & bcnf?
9) Triggers
10) SQL injection.
11) Drop Column, Insert name = 'b')
12) update name 'a' to 'b'.
13) 2 joins in a query.

↳ TCP, HTTP
Q. Implement que using stack.
Q. most Repeating digit?

Database System RDBMS

Collection of related Data ≈ Database

eg: Railways

DBMS → Insert, update, delete → operations performing system

① Oracle
② MySQL
③ SQL Server

Database
- Unstructured: Webpages
- Structured: RDBMS → Relations → Tables

File vs DBMS

File
- User store & Access
- eg. Folderson Drive
  (1) 25 GB File
  (2) Metadata Required, Location

DBMS → Client-Server Architecture
- → centralized Data
  (1) 1 KB data search
  (2) Concurrency; many people accessing data at same time
  (3) Security eg: CMS Role based Access control
  (4) Redundancy; constraints

Schema → logical Representation of Data
Entity & Attributes

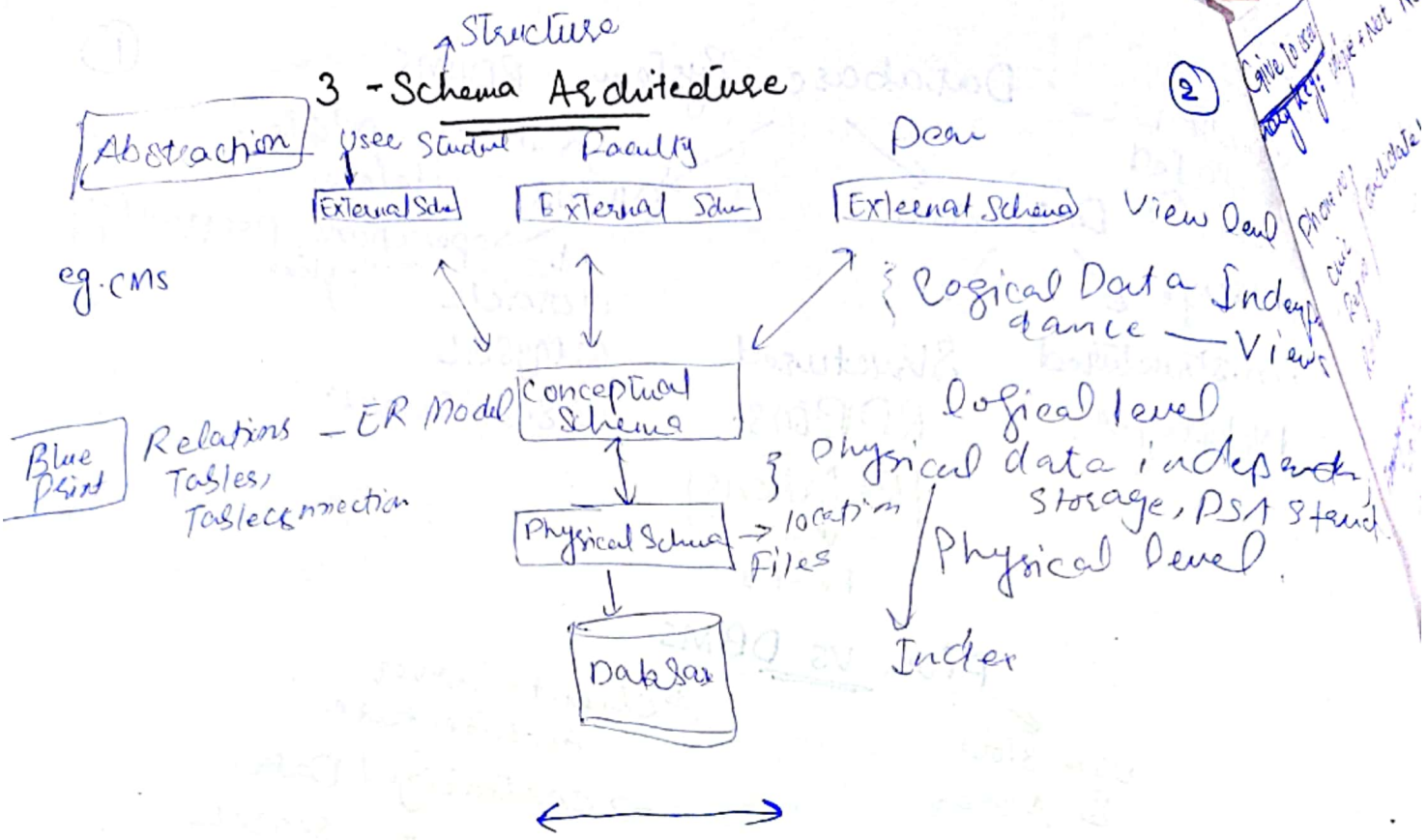
Student

Rollno	Name	Ph.no

Course

C_id	C_name	Duration

DDL Data Defined Lang. in SQL to implement schema



Key → uniquely Identify any 2 tuples in table.

Attribute

Roll no	Name	City	Age
1	x	a	3
2	y	b	2
3	x	a	3

Student Table

1) Civic
2) Roll no
3) Reg. No
4) Lisence No
5) Votee_id
6) Phone num
7) Email

Candidate:

collection of all keys is called candidate key.

1) Primary key

other → Alternative key.

Give to users

Primary key: unique + Not Null

Phone no
Cnic
Reg no
Roll no
} candidate keys → unique, can be null

Foreign key:

Attribute or set of attributes that references to primary key of same or another table.

→ maintains Referencial Integrity → same value.

Student

eg. marketPrice

RK Student			Course			FK	
Rollno	Name	Address	c-id	c-name		Rollno	Referencing Table

Referenced/Base Table

```
Create table Cource (
    cid varchar(10),
    c-name varchar(20),
    Roll no int refrences Student (Roll no).
    );
Alter table Cource Add constraint fk
    foreign key (Roll no) reference Student
    (Roll no);
```

Reference table	Referencing table
① Insert - No violation	① Insertion may cause Voilation
② Delete - may cause	② Deletion - No violation
③ update - may cause voilation	③ update — May cause Voilation

- on Delete Cascade
- on Delete set Null
- on Delete No Action

Superkey: set of all possible Attributes uniquely identify

↳ candidatekey + Attribute

CK = Rollno + Name Rollno + Name + Age

SK SK

Superset of candidatekey.

$n-1$

$A_1, A_2, A_3 \dots A_n$

CK 2^{n-1} Possible S.K

$A_1 A_2$

CK $2^{n-1} + 2^{n-1} - 2^{n-2} \rightarrow SK$

ER - Model → schema

logical / conceptual view

Entity

Any object having physical existence.

Attribute → characteristics of Entity.

Entity

Relationship

Attribute - Type

Types of Attributes

1) Single Vs Multivalued

Reg No

Ph-no → Students can have more than 1 Ph-no

2) Simple Vs Composite / Compound

↳ can't be divided e.g. Age

↳ Name: FN + MN + LN

3) Stored vs Derived.

DOB Age

4) key vs Non-key Attribute

unique

Reg

5) Required vs Optional

mandatory

6) Complex → composite + Multivalued

Degree of Relationship (Cardinality)

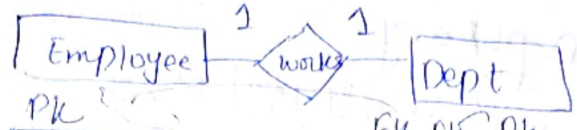
Association b/w Entities

1 - 1

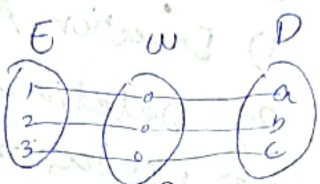
1 - M

M - 1

M - N



PK			FK	FK	PK		
E-id	E_name	Age	E_id	D_id	D-id	D_name	Loc
			E1	D1			
			E3	D2			
			E2	D3			



⇒ PK = E_id D/D_id

↳ Reduces to 2 tables

1 to Many

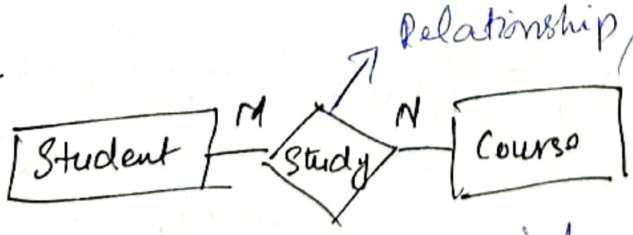
Relationship Attribute

Discriptive Attribute → Date

(1) P.K = M O-no

(2) Reduces to 2 tables.

M-M



(1) P.K = Rollno + cosid = compsite key

(2) Can't be reduced.

2nd Nov

Normalization:-

Technique to remove/reduce redundancy from table.

2 Types of Duplicacy:

1) Row level → P.K P

2) Column level

↓

Few column's values are same.

Solution

Divide table

3 Types of anamolies. Problem occur at special event

1) Insertion Anamoly

2) Deletion

3) Updation

SID	Sname	Cid	C-name	Fid	Fname	Fsal
1	a	C1	Db	f1	Joh	30
	b	C2	Db	f2		
2	c	C1	Db	f1	John	30
3						

First Normal Form

Table shouldn't contain any multivalued Attribute.

Roll no	Name	Cource
1	a	C/C++
		Java
2	b	O/DBMS
3	d	

PK = Rollno + Course

① Rollno | Name | C1 | C2

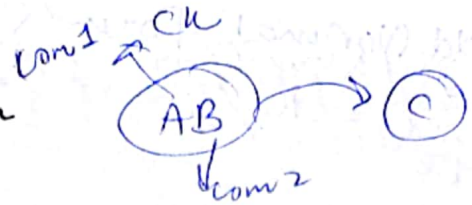
② Rollno | Name/course

③ Rollno | Name (PK)

Rollno | Course (FK)

PK = Rollno + Course

2nd Normal Form



(1) Table must be in 1st NF

(2) All non-prime Attributes should be fully functional dependant on CK

→ Non-Prime Attributes → Not in forming CK

→ Fully Functional dependancy → No partial dependancy

→ No partial dependancy.

Not in 2nd NF	Cust-id	Store-id	Loc
	1	1	Delhi
	2	3	Mum
	2	1	Delhi

P.A = Cust-id, Store-id

CK = Cust-id Store Id

N.P.A = Loc.

Sol — Divide table

3rd Normal Form:

→ L.H.S = CA/SK of FD or R.H.S → P.A

(1) Table should be in 2nd NF

(2) No transitive dependency in a table.

→ No Non-Prime Attribute can determine NP Attribute

→ P.A determine NP Attribute

CK – Rollno (P.A)

F.D: Roll → State (N.PA)

State → City (NPA → NPA)

Rollno	State	City
1	Punjab	Mohali
2		
3		
4		
5		

ABCD

F.D: AB → C, C → D

CK = AB

PA; A, B

NPA: C, D

BCNF : Boyce codd Normal Form / Special case of 3rd NF

→ Table in 3rd Normal form
L.H.S of F.D = CK / SK
must be

CK = {Roll no, Voterid}

Roll no → name
Voterid → age

4th NF

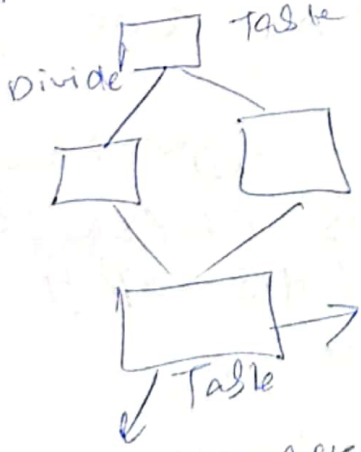
1. In BCNF
2. No - multivalued Dependency

eg. Ali — 3 Ph#, 3 email

1NF : No multivalued Attribute

5th NF

4 NF + Lossless Decomposition



Extra tuples may be → lossy

Common Attribute → C.K → Lossless Decomposition

ACID properties of Transaction: (eg. ATM)

1) Atomicity → Either All or Null

2) Consistency: Before Transaction starts / After Transaction ends, sum of money = same

3) Isolation

4) Durability

Rollback T1

R(A)
A = A - 50
W(A)
commit

(9) Failed Transaction always restart / not resumed

A = 2000
B = 3000
5000

T1
R(A) = 2000
A = A - 1000
W(A) (1000)
R(B) = 3000
B = B + 1000
W(B) = 4000

A = 1000
B = 4000
5000

(3) Isolation:

- Parallel Transactions

T1	T2	T3
R(A)	R(B)	R(A)

T1 ⇄ T2

T1 → T2

T2 → T1

⇒ convert Parallel Transaction into serial Transaction; consistent Transaction

(4) Durability: All changes made are permanent.

A = 1000
B = 4000

Joins

cross product + condition

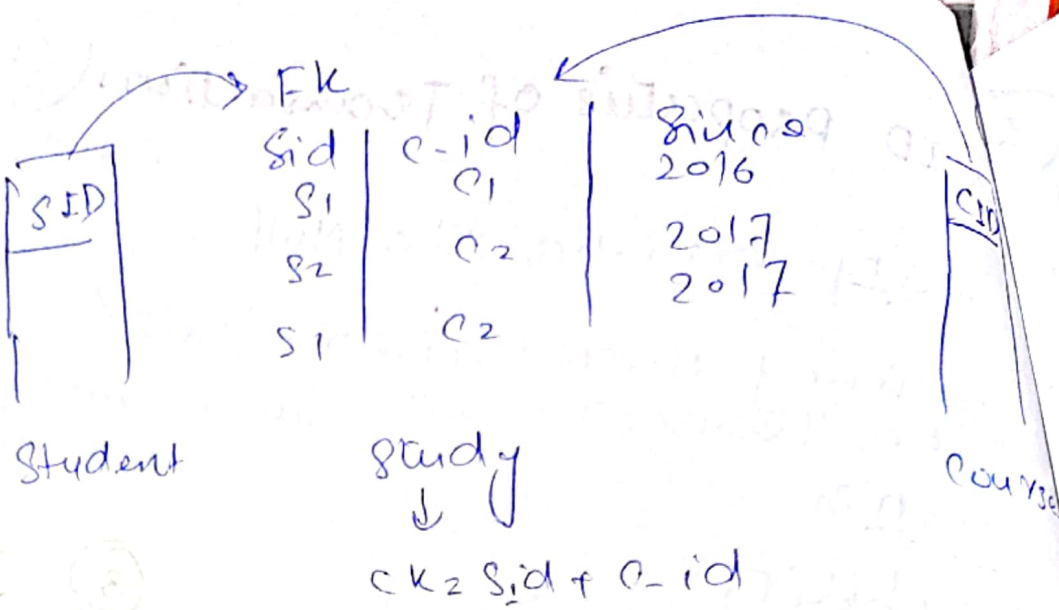
1) Natural Join

↳ common Attributes value equalize.

① Select Ename from Emp Natural Join Dept.

② select Ename from Emp, Dept where Emp.no = Dept.emp.no

Self Join



ck2 Sid + C-id

↳ Find Student enrolled in atleast 2 courses?

Select T1.Sid from Study as T1 , Study as T2
where T1.Sid = T1.Sid and T1.cid <> T1.cid

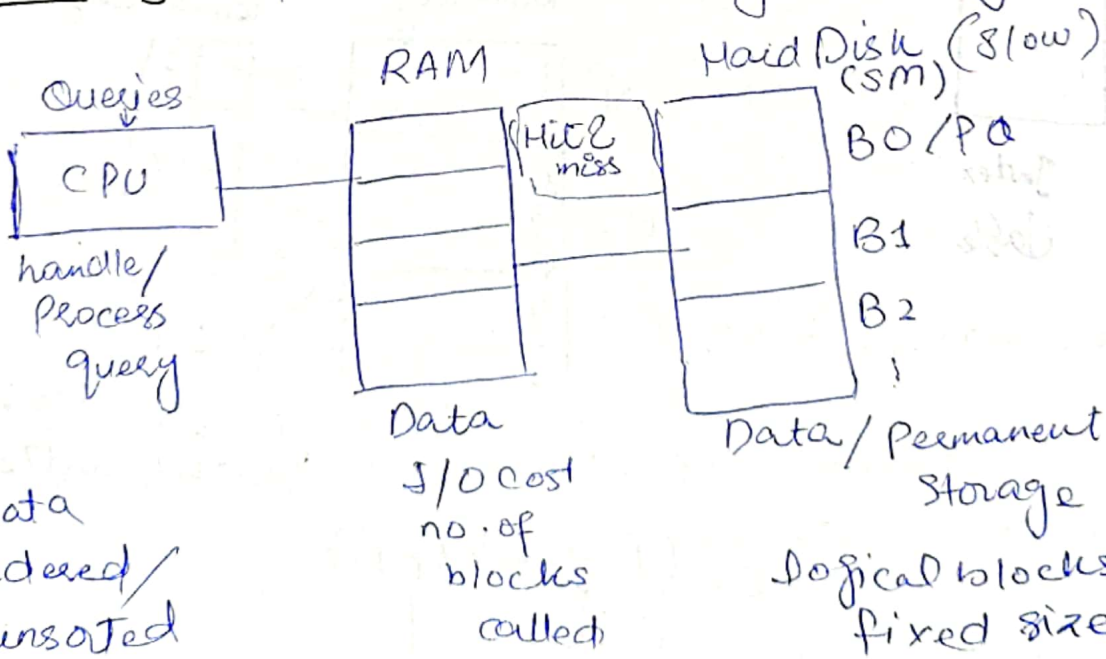
Equi Join

Constraints in SQL

Conditions on Columns ; Attributes.

1. Unique ; no duplicacy
2. Not NULL ; mendatory
3. PK → Unique + Not NULL
4. Check int Age ,-10 x
   check(age > 18)
   Adress (Delhi, chd, Mumbai)
5. FK
6. Default

Indexing : I/O cost Reduced by Indexing.



↳ Reduce no. of blocks to be called.

Types of Indexes

1) Primary
2) Clustered
3) Secondary

$TCC = (\log_2 N + 1)$

No. of entries in IT = No. of blocks in HD

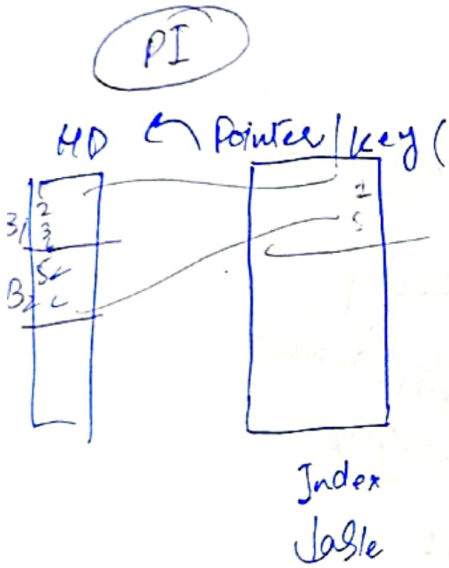
sparse

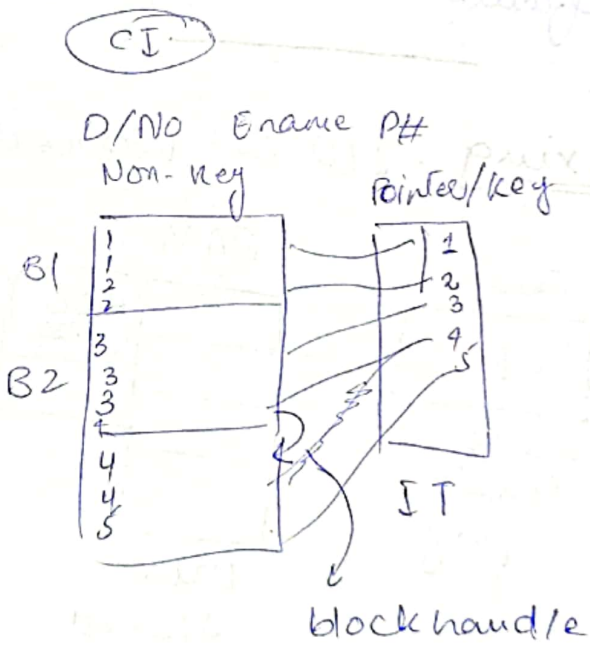
Ordered (asc/desc sorted)	Primary Index	Clustered Index
Un-ordered	Secondary Index	Second Index

At most 1

Key → unique

Non Key → non-unique





$TC = (\log_2 N + 1 + 1)$

sparse

Secondary Index

unordered Data

PAN (key, PK) — Name (Non key)

No. of indexes in Index = No. of record in Table.

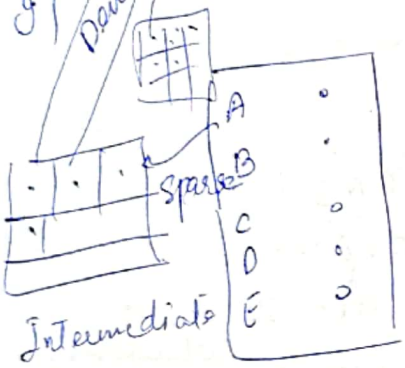
$T(C) = \log_2 N + 1$

Eid | Name | PAN | Pay

Query: Search Data on basis of PAN or name.

PK fails → Primary key

OT



Intermediate Layer (Block & Record Pointer)

$T(C) = \log_2 N + 1 + 1$

looks like table / not a table

Physical space in HD

Views

↳ virtual table; create table, Attributes

↳ Results of stored query; create view V1 as select id from emp;

create view V1 as select id from Student;

Base table → view taken

but not store result

Require no space

view → Read only

→ updateable views

* Materialized View :

↓

take space

Crash data to local server from Client-Server Architecture.

→ Multiple data in view from Tables.

Advantages.

1) Restrict Data Access ; security — CMS — 20 columns & Access to 4 columns.
2) Make complex queries easy — 2016, 2017, 2018
3) Provide Data independance ; particular Access
4) Present diff. views of same table. — CMS

Joins

Equi Join '=' b/w any operator Natural Join '=' in Common Attributes

Emp (Empno, Ename, Address) Dept (Deptno, Loc, Empno)

Question Find Emp.name who worked in a dept having location same as their address

Select Ename from Emp, Dept where

~~emp.Adress = Dept.loc~~

emp.no = dept.empno and emp.Address = dept.loc;

Left outer join
Right outer join — Natural join + sth
Full outer join

Left outer join:

gives matching rows + rows in left table.

Emp

Emp_no	Ename	Deptno
E1		D1
E2		D2
E3		D1
E4		

Dept

Deptno	Dname	Loc
D1		
D2		
D3		

Select Empno, ename, dname, loc from emp ~~Emp~~ left outer join dept on (emp.deptno = dept.deptno)

Right outer join

Transaction → From ATM

↓

represent change in database

set of operations used to perform a work.

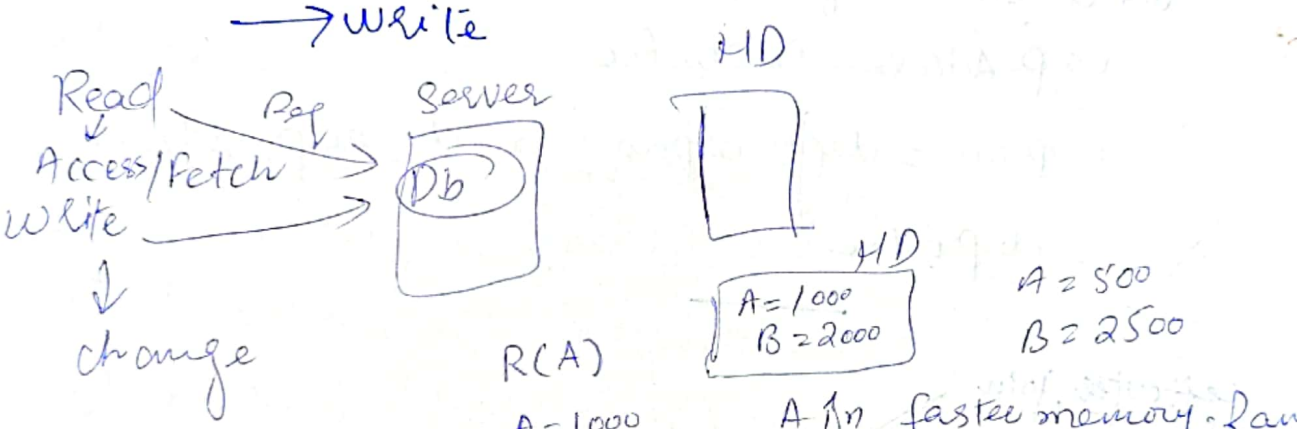
↓

logical unit of

2 Major operations

→ Read

→ Write



R(A)

A=1000

A=A−500

W(A)

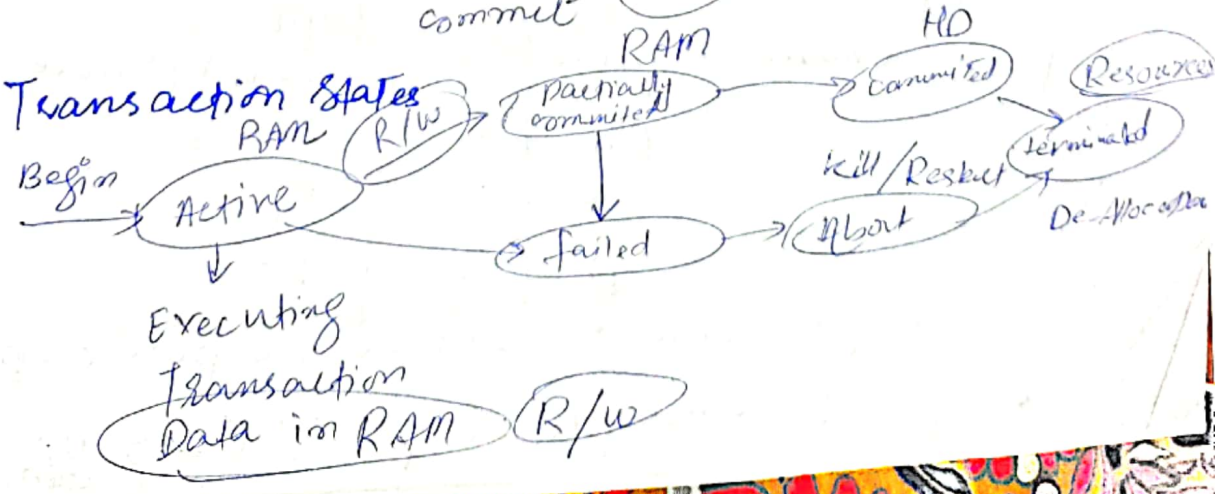
R(B)

B= B+500

W(B)

commit

A in faster memory; Ram



Triggers & Stored procedures

NULL functions

1) IfNULL(); Select Name, IFNULL(Price, 0) from Products
2) ISNULL()
3) NVL()
4) Coalesce() Select PName, Coalesce(Price, 0) from Products

- Stored procedure : prepared SQL code, you can save to be reused.

⇒ Create procedure SelectAll AS
    select * from Employees
    Go;
⇒ Exec SelectAll;

Comments (--) /* */

- Triggers

Benefits
↳ Security
↳ Referential Integrity
↳ Synchronous replication of table
↳ Preventing invalid transactions
↳ Auditing
→ storing proc.
→ Event logging ↓ who accessed table.

creating table ⇒ some value changes in table

Create Trigger T
Before/After
Insert/update/Delete
on Table
Declare
Declaration
Begin
Ex. Statem
END

create Trigger Sal-ch
Before
Insert or deleter
on emp
for each row
Dellare
Sal-dif number
Begin
Sal diff := :newsal..

SQL injection : Type of web-Attack
↓
Attacker is trying to insert sth
to access database
→ Steal valueable data
→ destroy db.

Delete Truncate Drop
↙ ↓ ↘
DML DDL DDL
↓ ↓ ↓
Row Delete all Table
↓ rows
can be restored using Rollback command
↓ faster
can't be restore dl
using RollBack comm