Data - collection of interrelated data.
DBMS - " & set of program to access data.

- DBMS is interface b/w Database & Application

Limitation of flat files

1) Complex to retrieve data
2) Security issue
3) Loss of data
4) Data Redundancy / Inconsistency

- Low cost

DBMS Architecture

1) Single Tier 2) Two Tier (Client-server) 3) Three Tier

Data Integrity - accuracy & consistency of data.

↳ entity Integrity - PRIMARY KEY
↳ Domain " - DATATYPES, CONSTRAINTS
↳ Referencial " - FOREIGN KEY

Rows / Tupples / Records
Table Name
cardinality - no. of tupples
degree - no. of attributes

classmate
Date
Page

Candidate Key
- min set of attribute that can be used to uniquely identify a single tuple

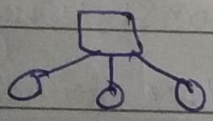
Primary Key
- candidate key that is selected to uniquely identify single tuple

Composite Primary Key – when 2 or more attribute are PK

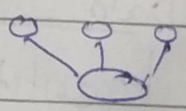
Foreign Key – child table can have NULL value. these are PK in another table

---

Generalization
- bottom up approach



Specialization
- top down approach



- Weak entity set does not have PK, it depends on strong entity set

Structured Query Language

DDL	DML	DCL	TCL
Data Definition language	Data Manipulation language	Data control language	Transaction control language
- create - alter - drop - truncate - rename	- Insert - update - Delete - Select - merge	- Grant - Revoke	- Commit - Rollback

Data Types

SMALLINT, INTEGER

NUMBER(3, 1)

Precision, Scale

1
99.9
3

Logical operator

AND, OR, NOT

BETWEEN <> and <>

IN ('A', 'B', 'C')

LIKE 'S%'

Functions

UPPER, LOWER, SUM, ABS, ROUND, CEIL, FLOOR,

SUBSTR('NAME', 2, 3);

MIN, MAX, COUNT, AVG, DISTINCT

NVL(val1, val2)

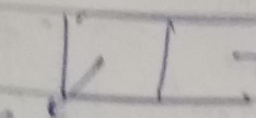
SYSDATE, SYSTIMESTAMP, ADD_MONTHS(date, n);

MONTHS_BETWEEN(date1, date2)

TO_CHAR, TO_DATE, TO_NUMBER

Indexing

- inc database performance



- Dense Indexing
- Sparse Indexing

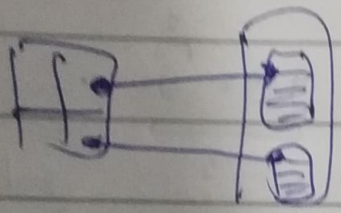
Single level

Multi-level

i.e B tree

[coz single level can't handle large size of indexes]

① Primary ~~Index~~



- Index entry for 1st record of each block

② Secondary Index

- non key attribute, unordered db

③ Cluster Index

- non-key attribute, ordered db.

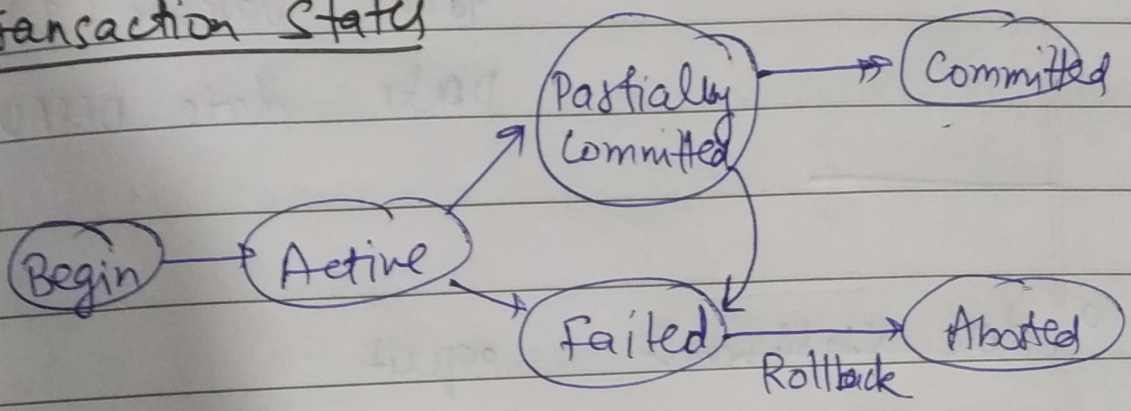
Atomicity (Transaction)

- done completely or none at all
- undo if failed in b/w

SET, COMMTT, ROLLBACK

SET AUTOCOMMIT (ON / OFF) | SHOW AUTOCOMMIT

Transaction States



A - Atomicity
C - Consistency
I - Isolation
D - Durability

SQL

- CREATE DATABESE < >;
  use < >;

- CREATE TABLE < > (rollno integer primary key,
  name char(10) not null,
  marks numeric (8, 2),
  grade varchar (1),
  DOB date DEFAUL sysdate);

- desc <table name> oquit

- DROP Table < >
  TRUNCATE TABLE < >
  RENAME < > TO/AS < >

- ALTER TABLE < > RENAME TO < >

  ALTER TABLE < > ADD (Age Number (2));

  ALTER TABLE < > CHAGE
  COLUMN < > TO < >

- CREATE VIEW <>
  AS SELECT ___

- INSERT INTO <> VALUES (3, 'Ann', 65, 'A');
  INSERT INTO <> (rno, name) VALUES (6, "sam");

- UPDATE <> SET mark = mark+2 WHERE rno=2;
  (HAVING, group By)

- SELECT * from <> order by <> desc;
  ↓
  rno, name

- SELECT DISTINCT rno FROM <>

- SELECT min (salary) FROM <> WHERE email is not null
  ↓
  (max, avg)

- SELECT Salary * 12 AS "annual salary"

- SELECT * from <> WHERE DEPT not IN (20, 30)

  WHERE dept = 20 or dept = 30

  WHERE name like "_M%"

- SELECT A. code, S. name
  from Schools, Admin A
  WHERE Admin. code = School. code

Group By - Aggregate, having

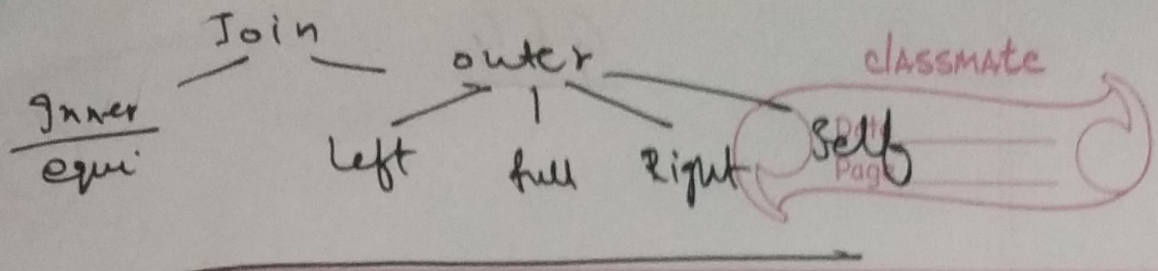
- SELECT DEPT, sum(salary)
  FROM <>
  Group by DEPT

Dept	Salary
A	1
A	2
B	5
B	5
C	5
C	5

⇒

Dept	Salary
A	3
B	-
C	-

- SELECT count (Subject)
  FROM <>
  Groupby Subject
  HAVING ___



- **Join** - fetch data from multiple table in single query

- **Union** - combine result from 2 or more select statements.
  - compatible (same no. of attributes)

SELECT ---- UNION SELECT ----

UnionAll

Union Distinct (default)

GFG

① 2nd highest Salary

```
SELECT max(sal) from emp
where sal not in (select max(sal)
                  from emp);
```
(not in ≈ <)

② Dept wise highest salary.

```
Select max(sal), deptno
from dept group by deptno
```

③ Display Alternate Records

```
SELET * from
(select empno, sal, rownum as rn from emp
 order by rn)
where mod(rn, 2) != 0;
```

④ Display duplicate ~~frequency~~ rows by ename

```
Selet ename, count(*)
from emp
group by ename
having count(*) > 1
```

classmate
Date
Page

(6) Display Kth row

```
select * from
(select rownum r , ename from emp)
where r=4

select * from emp where rownum <=4
minus
select * from emp whe rownum <=3
```

---

(7) 3rd Highest Salary

```
SELECT TOP 1 salary
FROM ( SELECT TOP 3 salary from <>
       ORDER BY sal DESC ) AS emp
order by salary ASC ;
```

---

(8) Substring

```
SELECT SUBSTRING ( name, 1, 5 ) as student_name
from <>
```

classmate
Date
Page

Functional Dependency

-fully, partially, Transitive

① If $t_1[x] = t_2[x]$
then $t_1[y] = t_2[y]$

② If $x \rightarrow y$
↓
unique always true.

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_2	c_1	d_2
a_2	b_2	c_2	d_3
a_2	b_3	c_2	d_4
a_3	b_3	c_2	d_5

$D \rightarrow A$ } fully
$D \rightarrow B$ }
$D \rightarrow C$ }

$A \rightarrow C$ (partial)

$C \rightarrow A$ (x)

$AB \rightarrow C$ (Transitive)

Normalization

- reorganizing data in db, to remove redundancy

1 Nf

- Remove multi-valued attributes

a	1,2
b	3

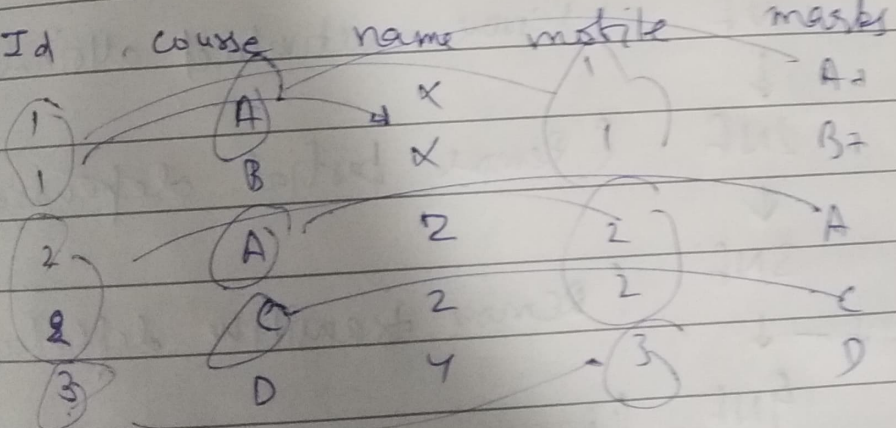
⇒

a	1
a	2
b	3

2 Nf

1 Nf &

- Remove partial dependency b/w non-key attribute & key attribute
  - P → NP ✗
  - PA ✓
  - NP → NP ✓



Id	mobile	name
1	x	1
2	y	2
3	z	3

Id	course	mark
1	A	A
1	B	B
2	A	A
2	C	C
3	D	D

↑ 8

- 3NF Remove Transitive dependency

X → Y

SK or PA

NP → NP (X)

- BCNF (Boyce Codd Normal form)

X → Y

Should be SK or CK

handling this case

X → P/NP P

- 3NF is better than BCNF

(lossless / dependency preservation)

(cost↑) have null values

- 1NF (Remove multivalued attribute)
  ↓
  2NF (Remove partial dependency)
  ↓
  3NF (Remove transitive dependency)
  ↓
  BCNF
  4NF (Remove multivalued dependency)
  ↓
  5NF (Remove Join dependency)

classmate
Date
Page

Remove Redundancy

Q(2)

	1NF	2NF	3NF
A → D	A → D	A → D	A → D
BC → AD	BC → A	C → A	C → A
C → B	BC → D	C → D ✗	
E → A	C → B	C → B	C → B
E → D	E → A	E → A	E → A
	E → D	E → D ✗	

BC → A except C

except B

BD

C D B

C → A

except BC → A

BC → D

B

BD A

C

CB AD

except BC → D

(2) R (A B C D)

$(AB)^+ = ABCD$

AB → CK

(PA) A, B → Prime Attribute

(NP) C, D → non-prime "

Partial dependency

B → C

P → NP 2NF

PA

make B Primary key on another table

A → D
B → D
B → C

R1 (A B D)

CK

entirely on PA

R2 (BC)

(3) R (A B C)

A → C
B → C
C → B

(A)+ → x
(AB)+ = ABC
(AC)+ = ABC
≥ 2 CK

AB → C
C → B
P → NP ✗
2 Nf ✓

3 Nf ✓

BCNf
(X) → Y
SK or CK

classmate
Date
Page

4) R(A B C D)

(AB)+ = ABCD

A, B → PA

AB → CK

C, D → NP

AB → C

C → D

N·P → N·P 3Nf

coz if c is null, as it is not PA, then we can't find P

(Transitive)

2Nf AB → C

PA → NP ✓

C → D

N·P → N·P ✓

3Nf

R1 (A B C)

R2 (C D)

≠

then it is PA

make C PA of diff table

1Nf
2Nf
3
BCNf ↑

classmate
Date
Page

(1) AB → C
A → DE
B → F
F → GH
(AB) CK

BCNf ✗
3NF ✗ F → GH
2Nf ✗ P → NPA A → DE
1Nf ✓

---

(2) CE → D
D → B
C → A
(CE)

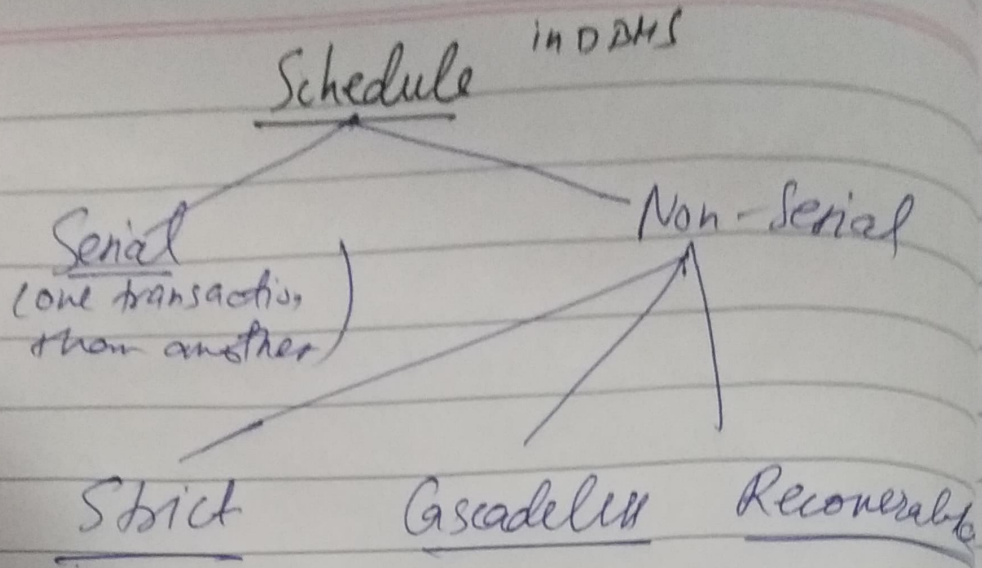
BCNf ✗
3Nf ✗
2Nf ✗

---

(3) AB → C
DC → AE
E → F
(A BD) (BCD)

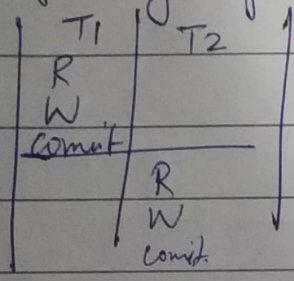
BCNf ✗
3Nf ✗
2Nf ✗

Read (R) | Write (W)



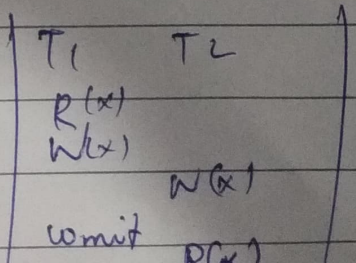
Strict Schedule

If 'write' operation proceeds a conflicting (Read / write operation), then commit or Abort of such transaction should also precede conflicting operation of another transaction



Cascadeless

If Transaction is going to perform read operation, it has to wait until above write transaction is committed



classmate
Date
Page

Recoverable

T1	T2
R(x)	
W(x)	
	R(x)
	W(x)
	R(x)
commit	
	commit

then only it can be recoverable

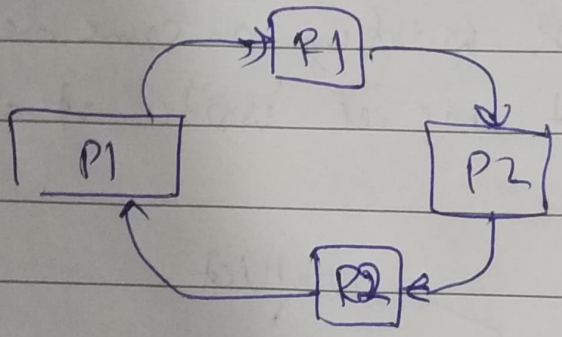
Conflicting operation

1. different transaction (both)
2. both are working on same data item
3. Atleast one is 'write' operation

Serializability
- conflict
- view

Deadlock

- where 2 or more task are waiting for each other, in order to be finished.
but none of the task is willing to give resources that other task needs.
- no task gets finished & they stay in waiting state forever.



resource Scheduler
- detection
& - Prevention

- Ostrich Algorithm - Windows & Unix
- let deadlock happen.
- manually terminate task
- ex windows hang & you have to reboot it to get it working.

classmate
Date
Page

Deadlock Prevention - Slow process, make system slow

1. Remove Mutual Exclusion
2. Remove hold & wait cond^n
3. Avoid circular wait cond^n
4. Preemption of resources

Deadlock Avoidance (timestamp)

Wait & Die

older process - waits

younger process - Dies

Wound & wait

younger process - dies and restarted after old is completed