Syllabus:

→ Integrity constraints and ER model } 2 marks

→ Normalization } 2 marks

→ Queries { RA, SQL, TRC } 4 marks

→ File organization and indexing } 2-4 marks.

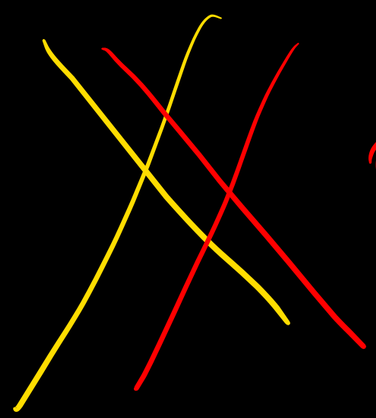
→ Transactions and Concurrency control } 2-4 marks.

Books:

~~DMS~~ DBMS by Raghu ramama Krishna
DBMS Navathe
DBMS Korth.

} Standard book.

DBMS → favorite ✓



Dont read books.
my notes is sufficient.
You will get full marks by my notes.

DBMS → I dont have notes as I keep adding new concepts.

TOC → Alka's notes has been uploaded.

DBMS → Alka's notes after DBMS will be uploaded:

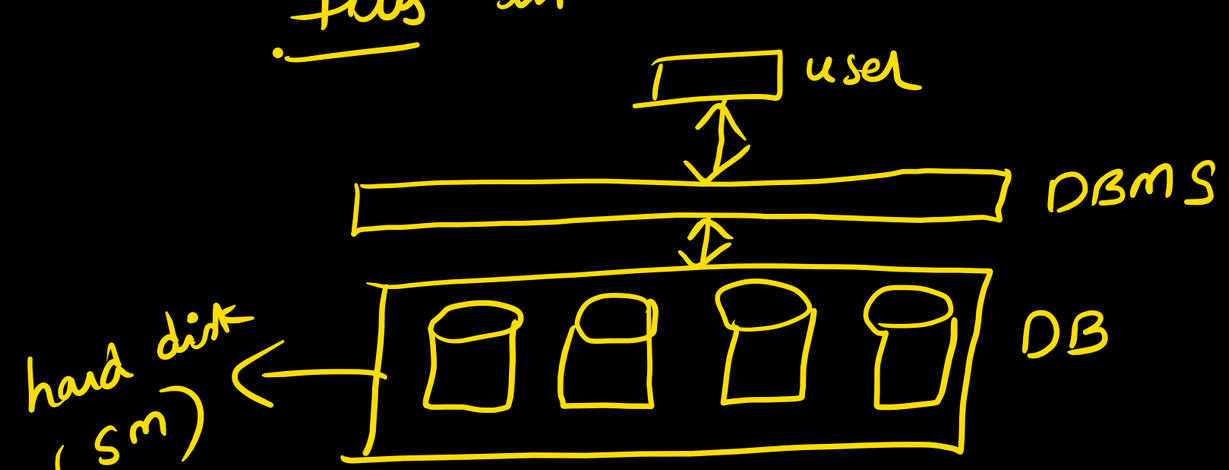
DPP →

DBMS:

Data base: Collection of reletid data.

Ex: Students info

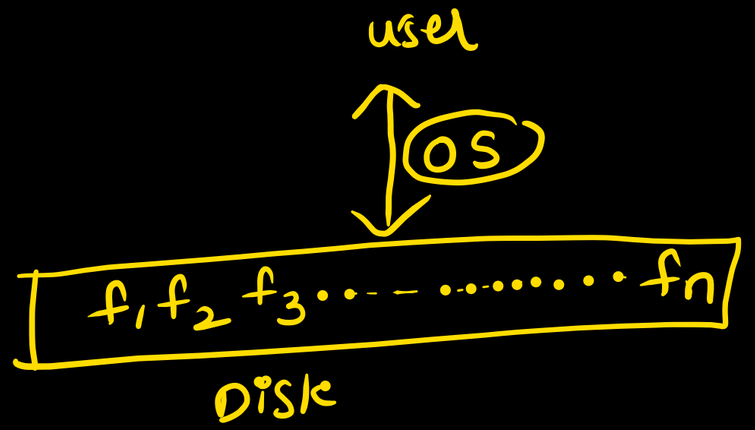
DBMS: S/w used to manage and access database files in mde efficient way.



Before DBMS → there were flat files on OS.

Flat file system (OS files) :

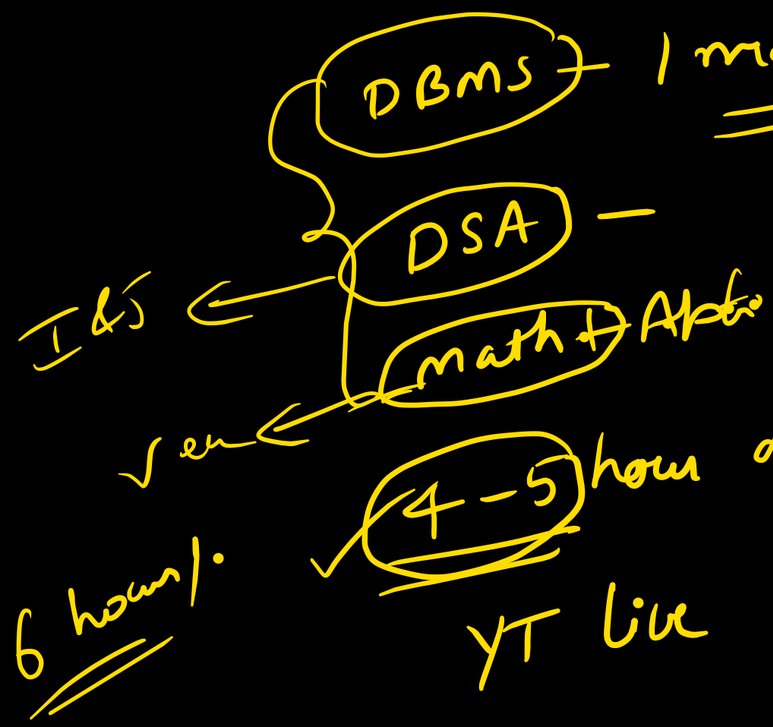
DB files are managed by users without DBMS S/W.



NO DBMS is used

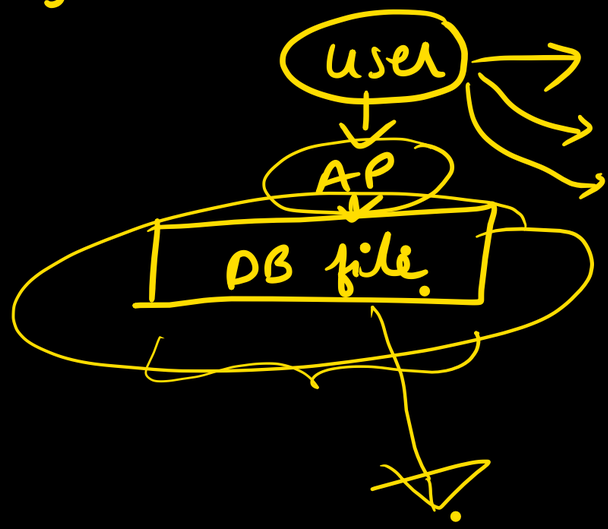
→ flat file system is good for small DB, KB's of MB's

→ If DB is huge, then flat file system fails.

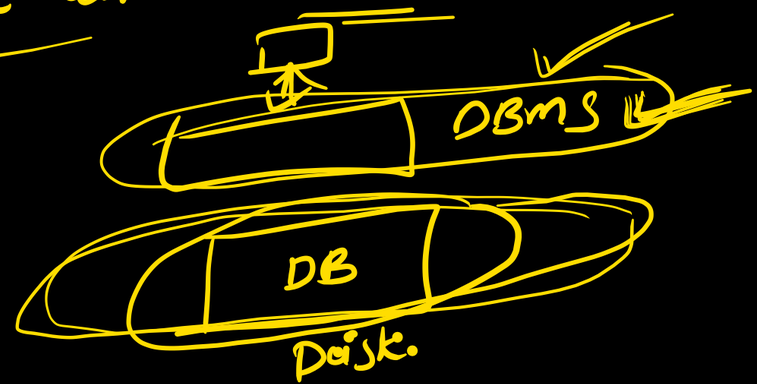


Limitations of flat system VS adv of DBMS

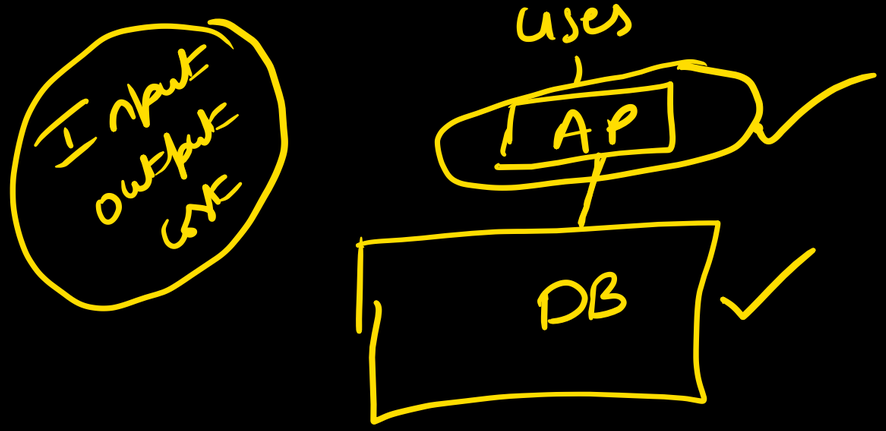
1) Too Complex to manage application programs to access data.



1) DBMS supports data independency [user can access data using SQL interface without knowing storage information of DB files]. So user need not know how data is physically stored in the DB.



2) more I/O cost to access data from DBMS files.

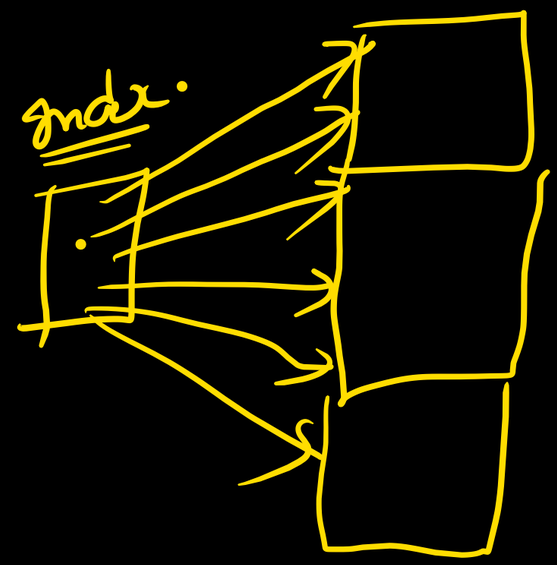


DBMS → S/W ✓ ↓ manages DB.

DB ✓ — Database ↓ Collection of related data.

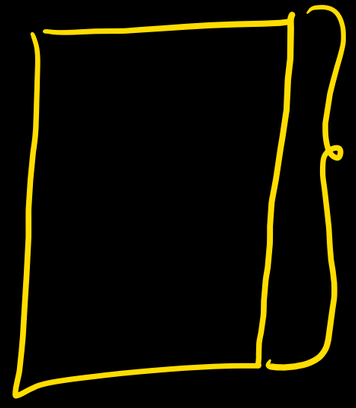
2) Because of indexing → end I/O cost will be less to access data

I/O fast.



3) Degree of Concurrency is very less

(1) → Parallel users using a file.



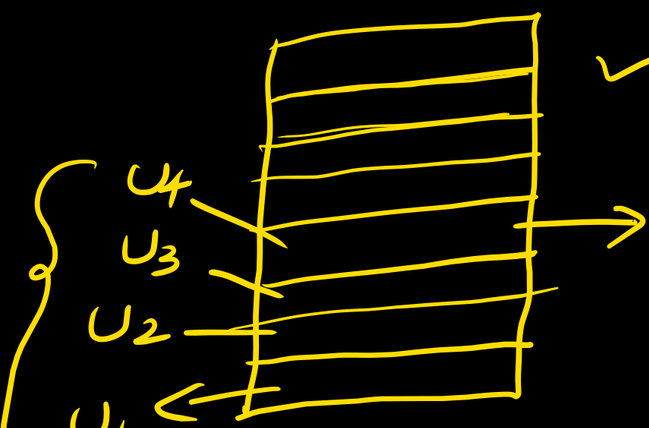
file level locking is done by OS → To give to one user

file

∴ only one user can access the file at a time.

Flat files.

3) Degree of Concurrency is very high

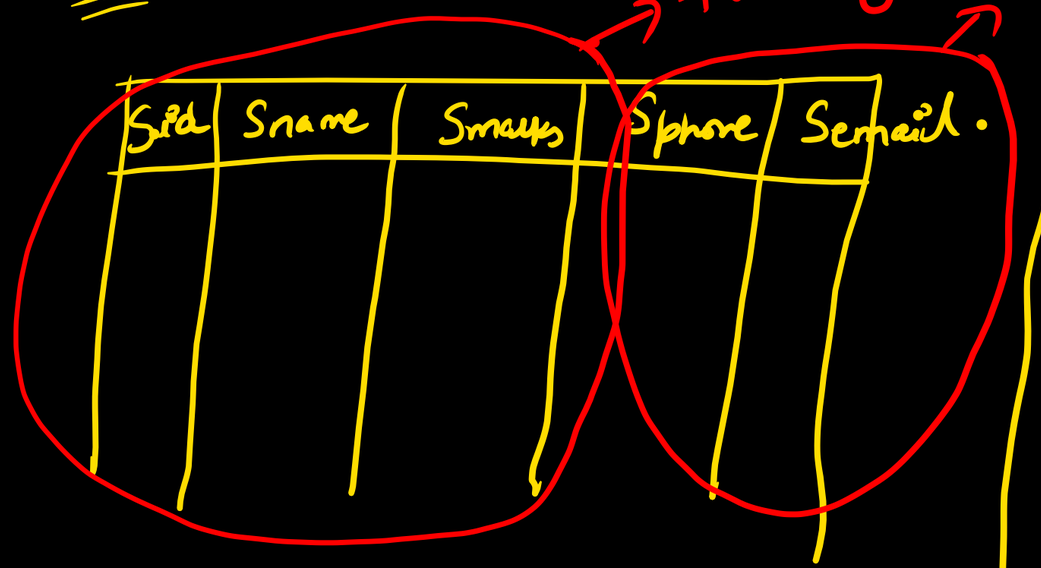


→ each row can be locked.

Parallel ∴ Degree of Concurrency is high in DBMS.

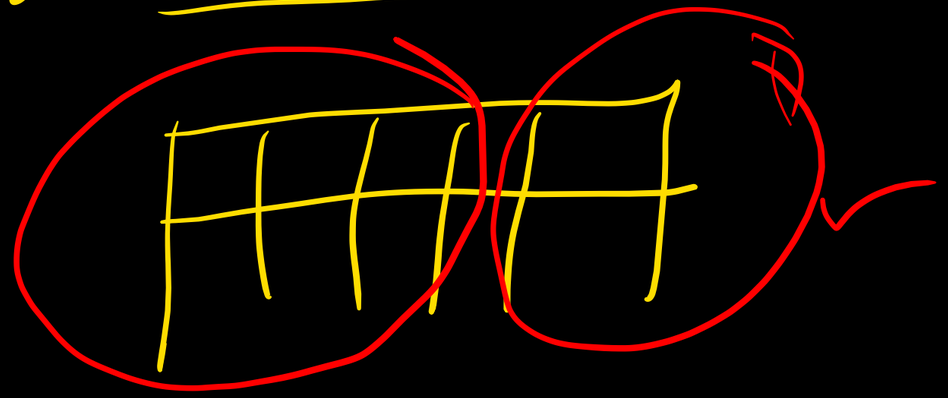
4) Too complex to maintain different levels of access control:

Ex:



faculty should see Sid, Sname, Smarks, but should not see Sphone, Semail. This is complex in flat files

4) In DBMS because of Views (virtual table), easy to manage access control.



5) Too complex to remove redundant data.

too complex to remove in flat files.

Sid	Sname	Cid	Cname
1	A	2	C
1	A	3	D
1	A	4	E

Redundant.

5) Because of Normalization of data, it is easy to maintain later not redundant data.

easy to remove redundancy

0% redundancy

Normalization

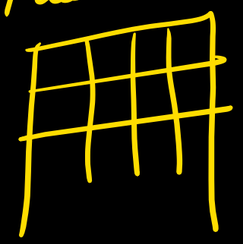
Integrity Constraints:

→ Correctness.

we are discussing integrity constraints based on RDBMS (Relational DBMS)

→ It is widely used data model, proposed by 'Codd' (name of the person who proposed RDBMS)

Relational → Tables ✓



→ Codd proposed 12 rules for design of DBMS S/w. These rules are called RDBMS guidelines.

→ I will explain each rule as the time comes.

Codd Rule I:

Data in DB file must be in tabular format (Collection of rows and columns)

NO two rows of DB file must be same.

Each row of the file is called record or tuple.

Set of all rows of a file is called relational instance or snapshot of record set


Sid
Sname
fee
DoB
S1
A
100
1990
S2
A
100
1990
S3
B
100
1991
attribute ∧ field
record ∧ tuple.
Record set
∧
relational instance
∧
snapshot

Relational schema: Definition/ Structure of the DB table

Informally: Heading of the table.

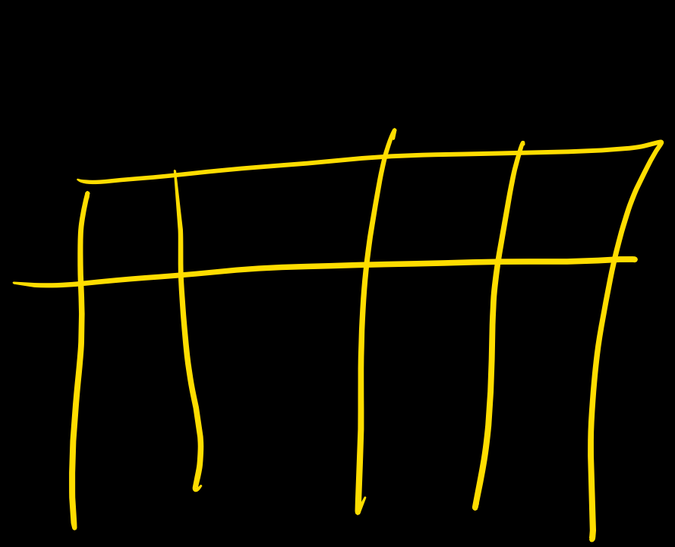
Student

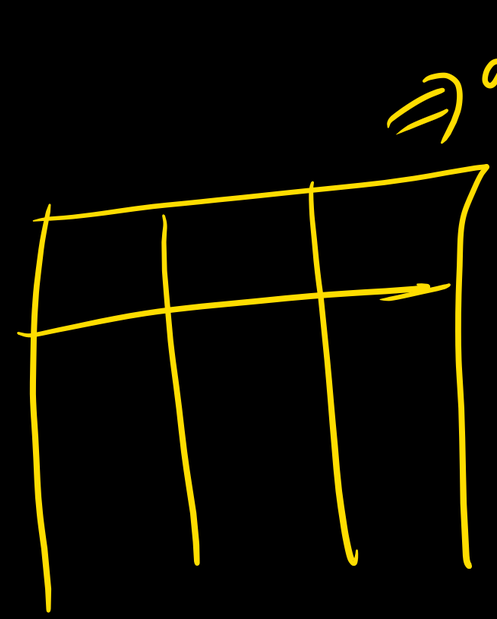
Sid	Sname	mark

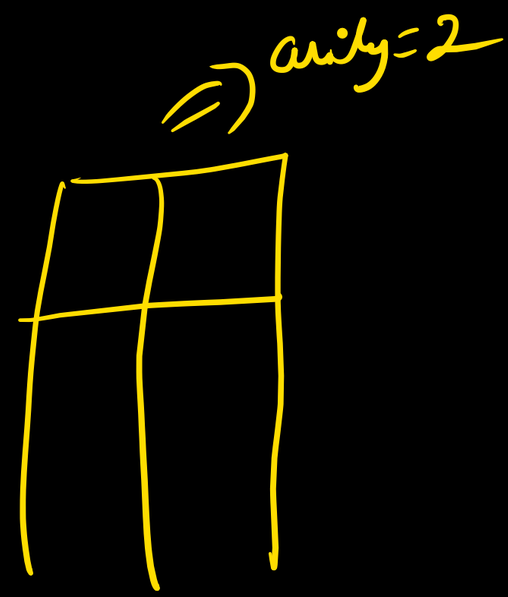
heading

Schemas: Student (Sid, Sname, marks).

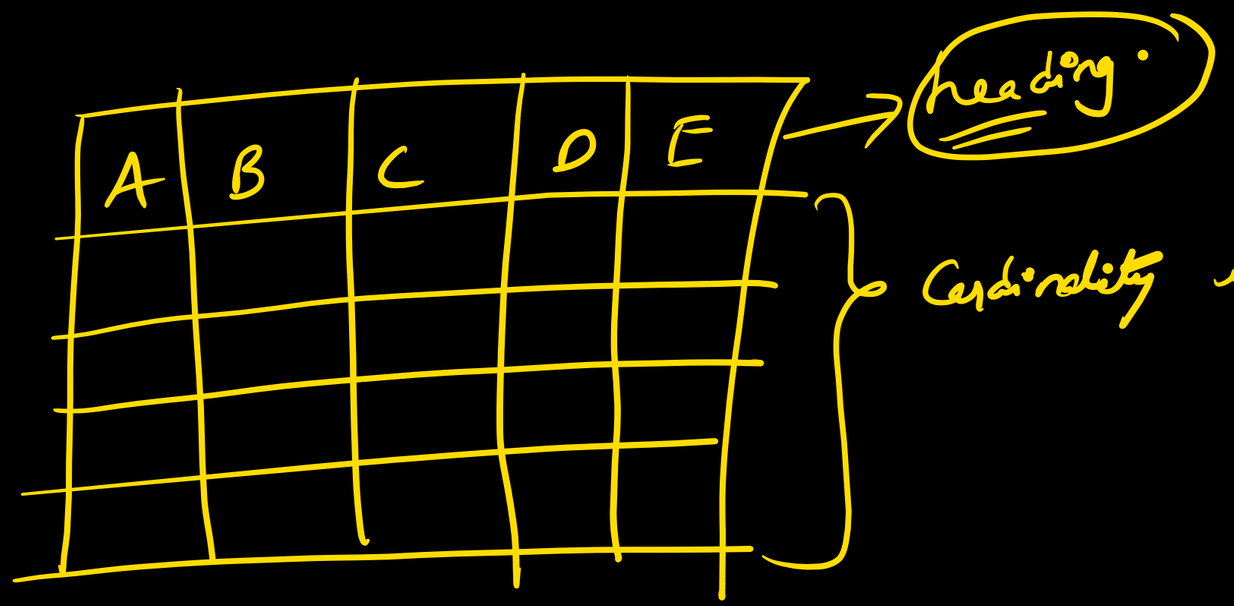
Arity: # fields of DB table







Cardinality: no of records of the DB table.

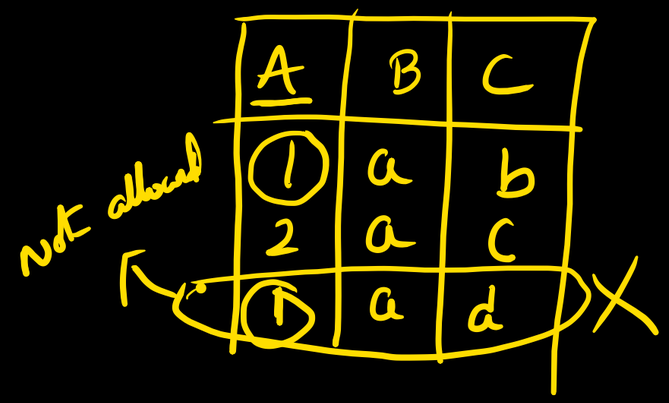


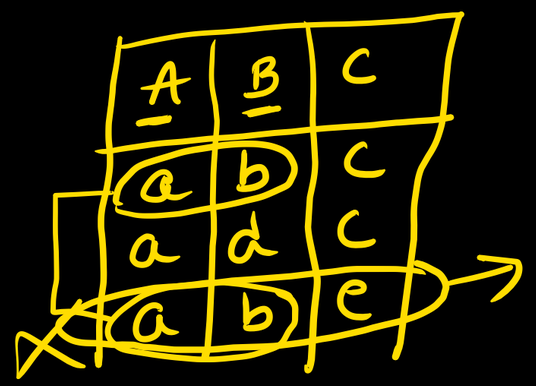
Candidate Key: *** very important question guaranteed.

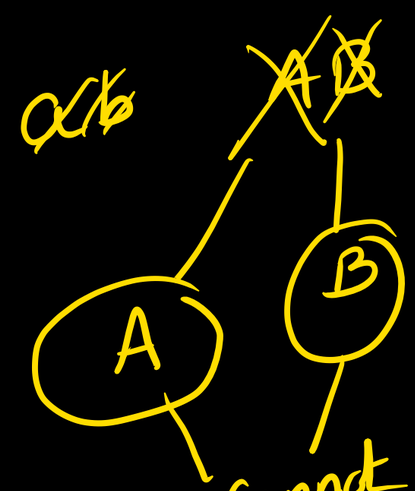
No two records of RDBMS table should be same. So we rely on Candidate Key.

Candidate Key is minimal set of attributes which can differentiate the records uniquely.

CK → 1 attribute
CK → 2 add
3
4
:
n







Can not be in Can key ↓ minimal.

Cannot be CK independently.

AB is CK.

ABC is CK.

A x CK

B x CK.

ABX

ACX

BCX

AX

BX

CX

A B C

No two rows should have same Sid.

Sid	Sname	fee
1		
2		
3		
1		

Enroll

Sid	Cid	fee
a	b	
a	c	
c	b	
a	b	

CK = {Sid, Cid}

X

{Sid, Cid} → CK → minimal → we cannot remove any attribute and still make it CK.

Sid X CK

Cid X CK.

ABC → CK.

AB × CK

BC × CK

AC × CK

A × CK

B × CK

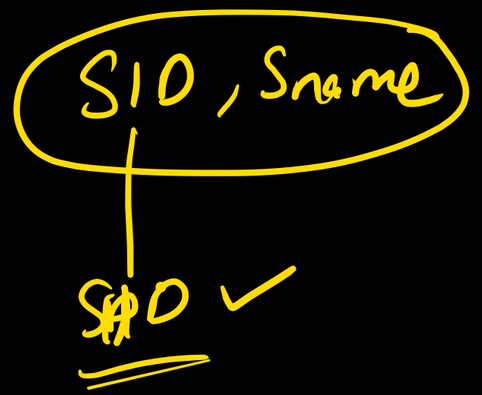
C × CK²

Stud (Sid , Sname , DOB)

Candidate key: SID

attribute that is underlined is Candidate key. If we make SID as candidate key, then no two records can have same SID.

If SID is CK, then anything add to SID cannot be CK.



Enroll (Sid , Cid, fee)

Sid cannot be CK because a Student can enroll in multiple Courses.

Cid cannot be CK because a Course can be enrolled by many Student.

∴ (Sid Cid) → Candidate Key.

Sid	Cid	fee
1	1	
1	2	
2	2	

Ex:

A	B	C
5	4	8
5	4	9
5	6	8
5	6	9
6	4	8

Ck = ?

A X AB X ABC.

B X BC X

C X AC X

Emp (Eid, ename, DOB, passpdt, Accno, ifsc, pan)

CK: { eid, passport, ACCNO ifsc, Pan }

Primary Key:

→ Any one Candidate Key whose value is not null

A, B, C, D.

↓

PK

↓

not null

Null → means unknown value or non existent value.

PK is one of the CKS ✓

PK does not allow null values ✓

Atmost one primary key should be present for a RDBMS Table. ✓

Ex: If Eid is primary key for employee table, then no two records can have same Eid & null values.

Alternative keys:

All candidate key of the relational schema except primary key are called alternative keys.

$$CK = \{A, (B, C, D)\}$$

(B, C, D) → alternative keys.

A ↓ Primary Key

alternative key are CK also.

AK fields allow null value

Only one AK's can be possible.

✓ CREATE TABLE EMP

{

Eid Varchar(10) PRIMARY KEY,

Ename Varchar(30),

DOB date,

PPno Varchar(15) UNIQUE, ... → allow null

Ano integer(10)

ifsc varchar(6)

Pan varchar(6) UNIQUE NOT NULL, → null is not allowed

UNIQUE (Ano, ifsc) ...

);

SQL

Simple Candidate Key:

A Candidate key with only one attribute is called simple CK.

Composite Candidate Key:

CK with atleast 2 attributes.

Eg: (Ano, Ifsc) ✓

(S10, C1D) ✓

Ano → CK.

(SBI 100) ✓

(CAN 100) ✓

2, 3, 4, 5 . . .

Prime attribute: ***

Attribute that belongs to some CK of selected schema.

Ex: CK = {ABCD}

A → Prime attribute

B → Pri " "

C → " "

D → " "

Prime attribute has nothing to do with primary key.

CKS = {AB, ED}

A, B, E, D.

non prime attribute:

Attributes which doesn't belong to any CK of relational schema.

R(ABCDE)

CK: (AB, CD)

Prime: {A, B, C, D}

non Prime: {E}

8:30

J

YT live.