# Database Systems

08-03-2023

Concise Guide to Databases by Peter Lake, Paul Crowther

- Data is raw facts and figures
- Info is data with meaning
- Knowledge is aggregate info with some use cases
- Intelligence is decision making analysis.

- Data stored in databases in a systematic way.
- Data ware house
- Data lake.

- Databases
  - ↳ dedicated machines → servers
  - ↳ cloud (for billions) but security issue.

CSV → comma separated values csv files.

→ File Systems (one file can have one type of data in no links).

→ Hierarchical Databases (Trees) relationships restricted.

→ Network Databases (Graphs) — —

→ Relational Database Management Systems. (Table form)

Entity 1: Customer  Entity 2: Transcations.

columns hold attributes of data.

SQL or sequel → Structured Query Language.

→ No SQL for unstructed data storing.

22-3-2023

Database: Shared data, structured and stored.

entity → attributes / tuple → record
table  columns  row

DBMS:

Query Parser, Indexer.

DBMS allows simultaneous access (for this DBMS has locks)

Data Integrity

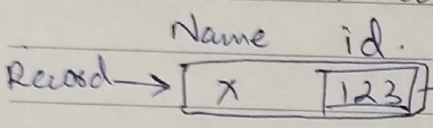
- Data Integrity (correctness تناقضات)
- Data Redundancy (repititions)
- Data Inconsistency (different or conflicting versions).

DBMS is centralized.
distributed DBMS.

Structured
RDBMS
Relation is a table

29-03-2023

Field is an attribute value.




File → collection of related records.

records have unique identifier.
- Ad hoc queries → temporary queries.

Database Management Systems.

- Data anomaly → abnormal (changes in redundant dat not made correctly)
  - Update anomaly.
  - Insertion anomaly.
  - Deletion anomaly.

① DDL (Data Definition Language).
create
alter
drop
truncate
modify
add.

② DML (Data Manupulation Language)
insert
upate
delete

③ DQL (Data Query Language)
select.

→ Metadata / Data dictionary.

# Chapter No. 02 Data Models

30-03-23

1. File based
2. Hierarchical based Model
3. Network based Model.
4. Relational Mode → Relations
   Entity
   Attributes
   Tuples.

ER - Model
Entity - Relationship

- Data Model is an abstraction of data flow, a prototype

## Data Model Basic

- Entity: anything about which data are to be collected or stored. It's a table name.
- Attribute: property / characteristic of an entity.
  Columns
- Relationship: an association between two entities.
  - one to many (1:M)
  - many to many (M:M or M:N)
  - one to one (1:1)

RELATIONAL → Table

instance → record.

Entity  attribute

05-04-23

- Cardinality:
  - 1:1 ex (CNIC for a person, degree of a student)
  - 1:M ex (car manufacturer makes many cars).
  - M:M ex (one student can enroll into multiple courses, also one course has multiple students).

- Constraints: DateEntered (not null) default now() ① Primary key ② Foreign key
  - Unique
  - Not Null
  - Default.
  - Check

→ Business rules are identifying cardinality of entities

adhoc queries

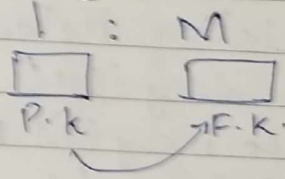
06/04/23

The Relational Model:

RDBMS

Primary key
A column which uniquely identifies each record.

Foreign key Referencial integrity key.
1 : M → Primary & Foreign keys.
Cardinality.




Select * from ~~AGENT~~ Customers.
where AGENT_CODE = "502";

Primary key - Foreign key
parent - child relationship.
constraint -- if foreign key exists, primary keys related to, it cannot be deleted.
through cascade, it is possible, then

if one table has more than one primary key then they're called composite keys (but it is not necessary).

Surrogate key.

```
create table Tbl (
    id int primary key autoIncrement,
    regNo varchar(20) not null
)
```

12-04-23

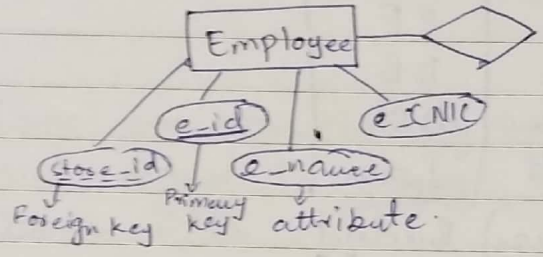
# The Entity Relationship Model.

- by Chen in 1976.
- Graphical representation of entities & their relationship.

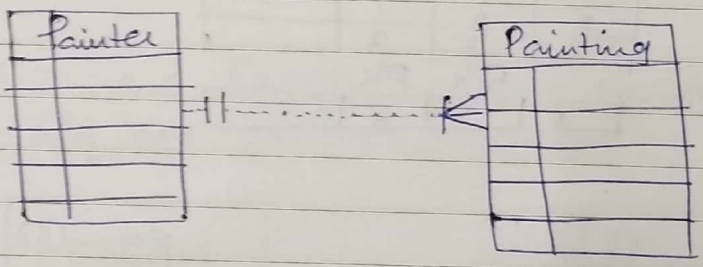
## Entity Relationship Diagram (ERD).

- Entity instance (or occurrence) is row in a table.
- Chen Notation.
  - ↳ By diamond. (relationship b/w tables)

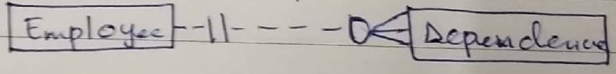
- Crows Foot Notation.



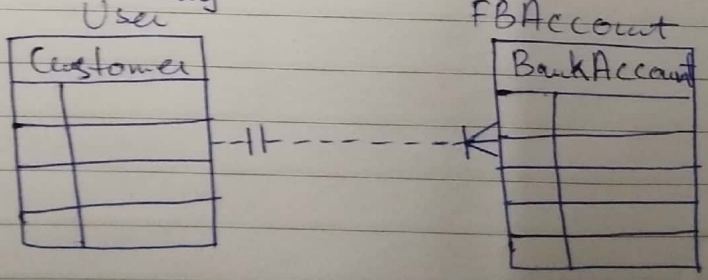

- Crows Foot Notation.




|| → one & only one
⫷ → one or many
⟨ → zero or many
..... → dependent entities.
——— → independent entities




one to many.



13-04-23

# How to implement M:N relationship.

1 : 1

Person — CNIC

or

Person + CNIC

1 : M

P.K : F.k

View

↓

query can be locked.

M : N

Students

| Sid | SName |
|---|---|
| 1 | Ali |
| 2 | Salman |
| 3 | Yasir |
| 4 | Danial |

Courses

| Cid | CName |
|---|---|
| 1 | PF |
| 2 | DB |
| 3 | DSA |
| 4 | Java |

Student-Course Bridge

| P.k id | sid | cid | grade |
|---|---|---|---|
| 1 | 1 | 2 | A |
| 2 | 1 | 4 | A- |
| 3 | 2 | 1 | |
| 4 | 2 | 2 | |

| P.k | Fk | |
|---|---|---|
| 5 | 2 | 3 |

pg-112.

Task: Make 1 eg. for each

1 : 1 1 : M M : M.

19-04-23

Keys

Key is one or more attributes that determines other. Key's role is based on determination.

→ Functional Dependency. (P.K).
→ Entity Integrity. (P.K)
→ Referential Integrity (F.K) FK Integrity
1 : M M : M.

Ch: 3 Slide. 18 Table 3.4 : Integrity Rules.
Read!

26-04-23.

THE ENTITY RELATIONSHIP (ER) MODEL.

ER model define conceptual structure of DB.
- Entity (an, active noun) table
- Attribute (properties of the entities).
- Relationships.

candidate key.

Business Rule,

1. One student can be enrolled in many courses and one course can be assigned to many students.

This is M : M relationship.

Implementation:

Create a bridge table (e.g. StudCourBridge) to implement this M : M relationship.

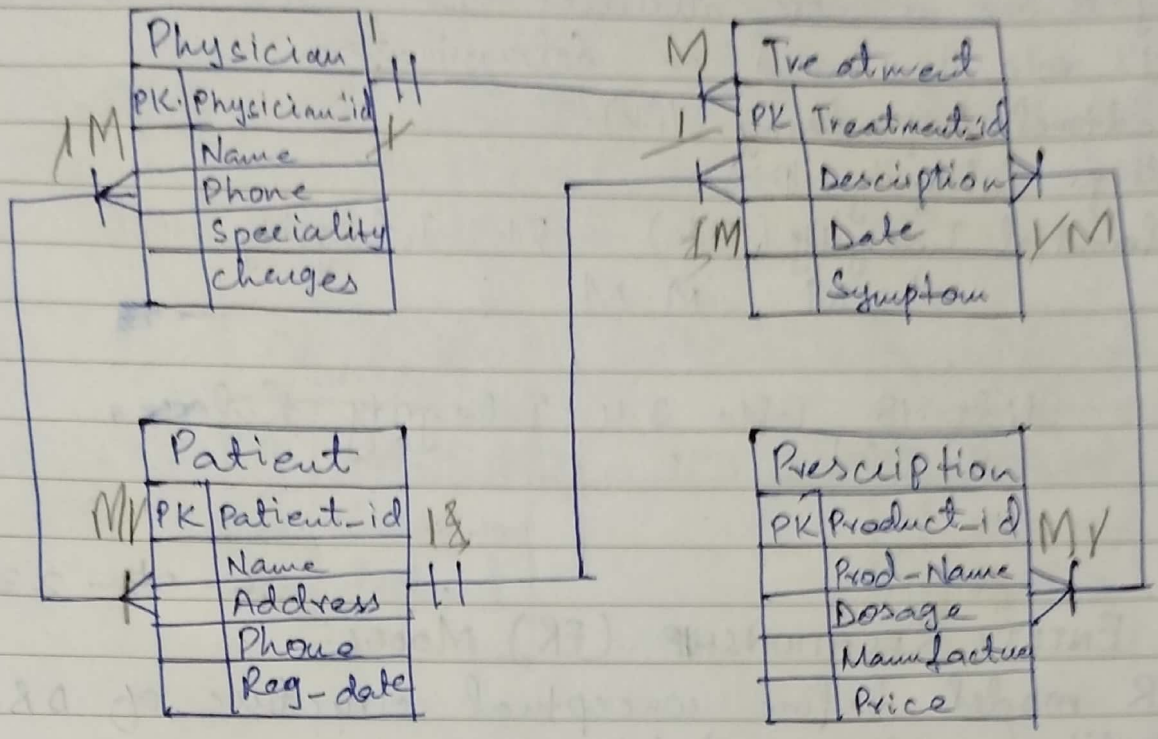
1 : M ??

Primary key – Foreign key.

Entity Integrity ?? In an ~~integrity~~ entity a PK can't be repeated.

Referential Integrity ?? We can't make a F.K if it is not a P.K in a parent table.

# Hospital Management System:




27-04-23.

Pg-18 Slides

3-2-1

Joins will be after mids

M:N many to many, implemented through bridge table, making it two 1:M relationships.

- - - - => dependent entity

_____ => independent entity.

1st part & 2nd part (Q).
Short Q/Ans.

3rd: ERD scenario

4th: Relational model Diagram. -- Pg: 53 of slides.

Slides, pg: 53.

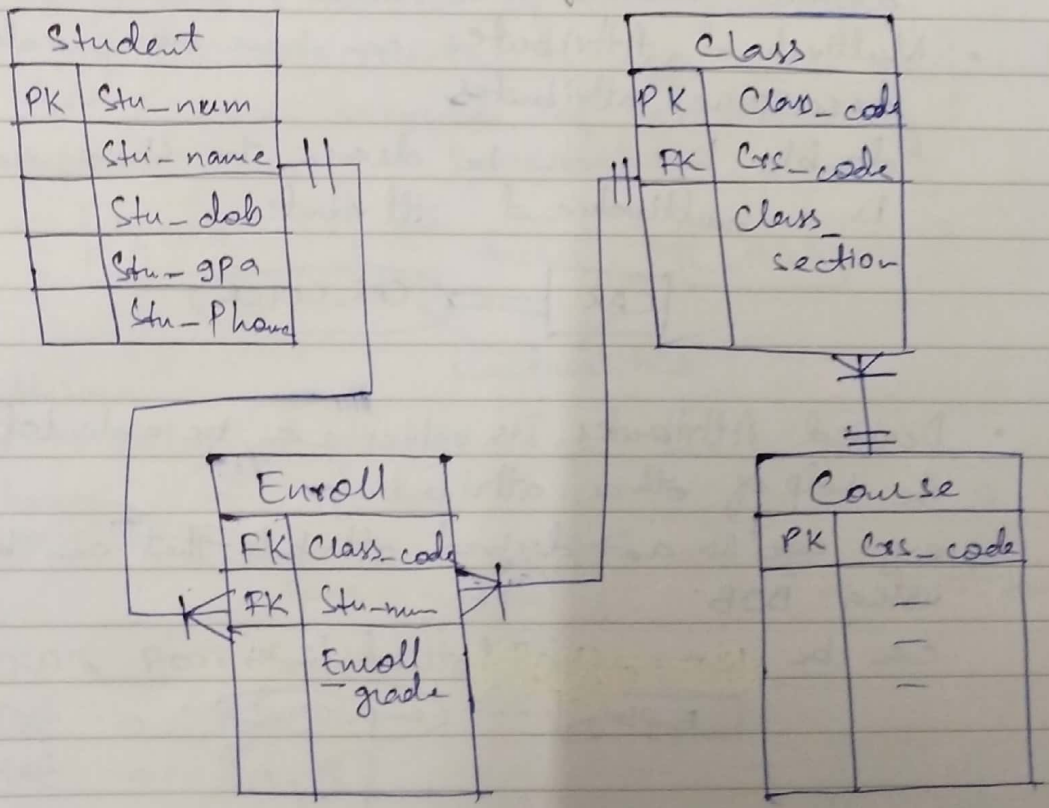
Relational Model.

Business Rules:

1. The system has four entities. Student, Enroll, Class and Course.
2. Student and Enroll has 1:M relationship. hence Student (Stu_id) would be foreign key in Enroll(stu_id) table.
3. Class and Enroll has 1:M relationship as well hence Class(class_code) would be foreign key in Enroll(class_code) table.
4. Course and Class has 1:M relationship. # hence Course (crs_code) would be a foreign key in Class (crs_code) table.

ER Diagram.

Entity Relationship Diagram




for UML Class.

|| 1 ... 1

|< 1 ... *

o< 0 ... *.

H.W: Ch: 3, pg: 55 slides

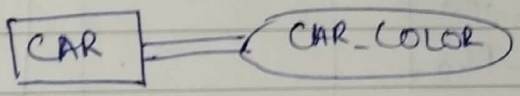
FINALS 10-05-23.

# Ch:4: Entity Relationship (ER) Modeling.

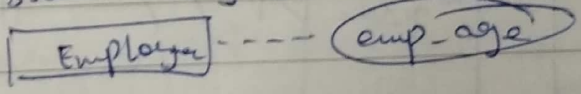
## Attributes

Properties of an entity.

- Required Attribute: can't be null, achieved by applying not null constraint
- Optional Attribute: can be null, allow null by default
- Identifiers: key attributes, it uniquely identifies each entity instance (a row or a record). One or more attributes which can be used to uniquely identify each row.
- Composite Identifiers: primary key composed of one or more attributes.
- Composite attribute: can be subdivided, e.g. name → first & last name
- Single-value Attribute: can't be subdivided e.g. date of birth, designation etc, as it contains atomic values (one value).
- Multivalued Attribute: can have many values descriptive attributes. double line can be drawn to show an attribute is a multivalued attribute.




- Derived Attribute: Its values can be calculated with the help of other attributes.
  e.g. age is a derived attribute that can be calculated using DOB.
  can be shown using dotted line.



## Relationships

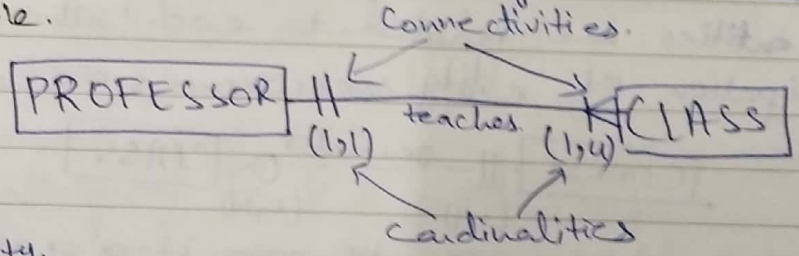
- Association b/w two or more entities
- Participants are the entities in a relationship
- Relationships b/w entities operates bi-directional eg. 1:M and M:1 are same because
- Relationships classification is difficult to establish & understand if only one side is known eg. student can have many courses → 1:M but we have not inferred that a course can have many students enrolled in it making this a M:M relationship.

Hence, relationships is a bi-directional approach.

**Connectivity:**
tells the relationship b/w entities

**Cardinality:**
→ expresses details of the connectivity
→ min & max number of occurances associated with one occurence of related entity.

Example.




connectivity:
one & only one professor can teach one or many classes

Cardinalities || => (1,1) (one & only one). |< => (1,4) one or atmost four

CROW'S FOOT Symbols: N => many.

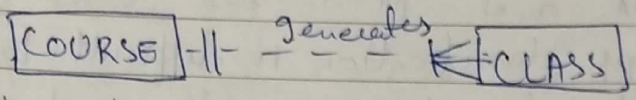
| | | |
|---|---|---|
| O< | => | Cardinality (0,N) |
| |< | => | (1,N) |
| \|\| | => | (1,1) |
| O\| | => | (0,1) |

Existence

- Existence Dependence:.

If some entity's existence is dependent on another entity, so this phenomenon is called as existence dependence.

Ex:




due to course ~~is there so~~ a class is there, so class depends on course.

weak relationship.

- Existence Independence:.

Entity can exist apart from one or more related entities.

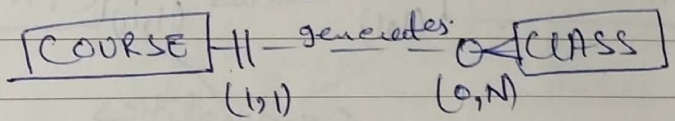
can be existed stand alone.

Relationship Participation:.

- Optional Participation.

One entity occurence (row) does not require corresponding entity occurence in a relationship.

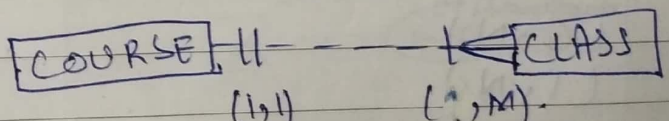
Two entities are related to each other but the participation of entity occurences are not mandatory




one course can be ~~exist~~ in 0 or many classes.

- Mandatory Participations.

One entity occurence from one entity could be related to atleast one or many all entity occurences in another entity.



## Relationship Degree:

- number of entities associated with a relationship.
- Unary relationship:
  association maintain within a single entity.
  1:1, unary relationship is a recursive relationship.
- Binary relationship:
  two entities are associated.
  1:M.
- Ternary relationship:
  three entities are associated.
  M:N.

## Associative (composite) Entities:

- Bridge entities
- used to implement M:N relationships

## Developing an ER Diagram:

→ Database design is an iterative process.

- detailed description
- business rules. ←
- entities & relationships
- Develop initial ERD.
- attributes & P.K. or other key.
- The final version of ERD should be implementable.

conceptual diagram means if there's M:N it is not shown by bridge table.
implementation-ready diagrame means if there's M:N it is shown by bridge.

## Task to submit

Connecting DB with Website using php

arifanedian@hotmail.com 17/05/23.

1. Xampp (Install) . ⇒ xamp\htdocs\...
2. Xampp Control Panel (start services)
3. Create database with a simple table.
4. create a connection/config file of php.
5.

24/05/23

Quiz
ERD (Scenario based)
Normalization

## NORMAIZATION:

- It is a process to remove data anomalies.
- Process for evaluating & correcting table structures to minimize data redundancies.

Database Anomalies. ⇒ Problem
- update
- Delete. → can create data inconsistency & incorrectness
- Insertion → creates unwanted nulls.

Conversion to 1st Normal form:.
1NF
2NF
3NF.

1. Eliminate the repeating groups (eliminate nulls)
2. Identify the primary keys
3. Create a dependency diagram.

Dependency Diagram.
1. Partial Dependency.
2. Transition dependency.

P.K

| A | B | C |
|---|---|---|
| | | |

functional dependency is achieved using P.K
(A) → Determinant
(B) → dependent attribute.

- When there are multiple PK in a table and some attributes are dependent on one PK, while some are on other PK. Then it is Partial Dependency. It should not be.

- when an attribute is dependent on a non-key attribute then it is transitive dependency.

Get an example for Partial & Transitive dependency.

25-05-23

empId empName DName HOD.

Conversion to 2ND NF:.

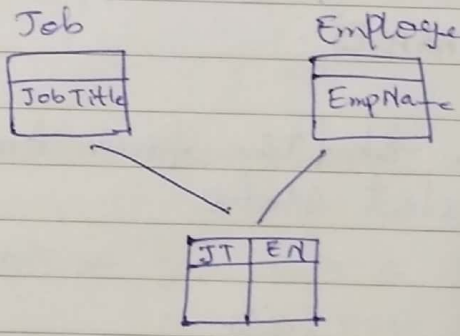
- Write separate tables for all primary keys & their related attributes.
- In this way, all partial dependencies will be removed.

Conversion to 3NF

- Make a separate table to remove transitive dependency and get related attributes
- Also, identify the relationship after breaking the table & make tables accordingly.

Joins

31-05-23.




When you need to get data from two or multiple table, we, use joins.

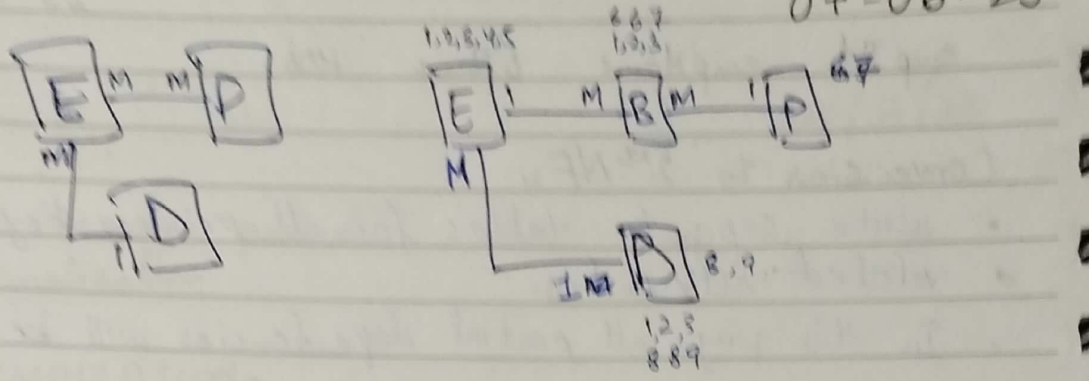
Joins —— equi-joins ——→ ① inner join
② outer join
left right
non-equi-joins. (theta)

- Inner join shows data on equality basis and does not show nulls.

Inner join do linear search

Select product.productName from products
left join order on product_id = products.product_id
where products.order_id is NULL;

07-06-23



E joins inner B joins inner P inner D

Select emp_name, project_name, dept_name from Employee as E join Bridge B on E.emp_Id = B.empID
join Project P on P.proj_id = B.proj_id
join Department D on D.dept_id = E.dept_id;

08-06-23

# TRANSACTION MANAGEMENT IN RELATIONAL DATABASES.

Database transaction shifts from one consistent state to another consistent state.

TCL
- commit → commits your data in H.D. (non-volatile memory) → store
- rollback.

- commited transaction can't be rolled back.
- ongoing transaction → DB log.

# ACID Properties

A → Atomicity :

- One transaction will either be completely executed or rolled back no partial, it is atomicity.
- Stops partial updates hence data integrity is maintained.

C - Consistency :

a transaction brings db from one consistent state to another.

I - Isolation :

Isolation ensures concurrent execution multiple transactions at one point of time, & one will not disturb the other.

D - Durability :

- Once data is committed, it can't be changed.
- it stops lost update problems.

14-06-23

## Transaction Management with SQL.

Commit → your data is saved in non-volatile memory, once commit is reached.

Transaction log.

→ Type of operations
→ Names of objects (tables, views, s.p, triggers.
→ Before & after values (updates) fields
→ pointers to prev. transactions & next, transaction in the serial list.

## Concurrency Control

→ on a field, one updates & other read, there'll be conflicts.

→ concurrent means, at same point of time, two same transactions hit the database.

→ lost updates problem; same transaction at same times, inconsistency of data.

→ Need to serialize the transactions.

## Uncommitted Data.

Phenomenon.

## Inconsistent Retrievals

① transaction access the data.

② transaction alters the data

③ transaction accesses the data again.

- This yields inconsistent results.
  - ↳ it'll be inconsistent, when it is not expected to be changed otherwise ~~&~~ expected retrievals.

## Serializability:

→ one by one execution of transaction

### Important factors

- two transactions, only reading same attribut (select statement), no conflict
- two transaction, reading different, no conflict.
- two transaction, reading & writing same attribute, conflict.

Scheduler: It is an algo or software which schedules the operations

Serializibility => locking, time stamping, optimistic techniques

15-06-23

# Concurrency Control Methods.

① Lock:

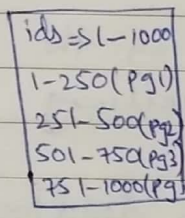
Lock Granularity.
- → database (no other transactions for that locked db)
- → table (no other transactions for that table)
- → page
- → row-level (that row is locked)
- → field-level (specific column.

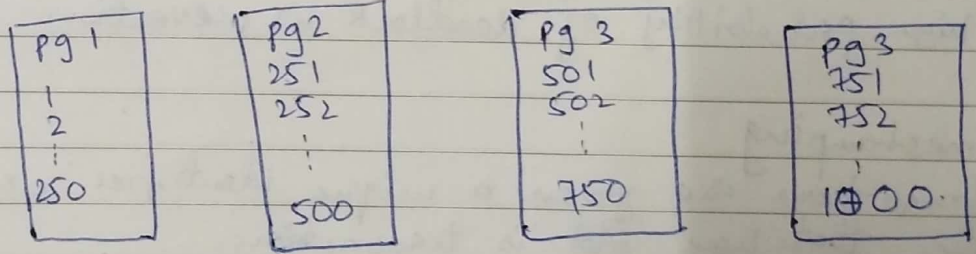
⇒ Indexes

if we have many no. of records, then searching will be costly

For catering that, database does indexing, divides records on basis of primary key, and that division of records are pages (stored in pages at backend).

Database page implements Binary Tree.




By default, db = indexes with PK which is clustered index

select * from tb where id is b/w _ and _
↓ page level lock.
or update query.

Lock types

① Binary lock :- 1 => lock , 0 => unlocked

② Exclusive lock :- doesn't allow read access in concurrent transactions.

③ Shared lock :- allows read access in concurrent transactions.

Deadlocks

One transaction is not leaving the processor & other transactions are waiting to process

⇒ Techniques to control deadlocks

① Detection :- If DB detects a transaction which is causing deadlock, dB rolls it back & that transaction is victim transaction.
Then next transactions continue executing.

② Prevention :- if a transaction can lead to deadlock then it is immediately roll back.

③ Avoidance :- Granting the locks to particular transaction, not rolling back.

- Low probability of deadlock => detection
- High probability of deadlock => prevention.

② Timestamping

Transactions are given a unique identifier or time that how old is transaction.

③ Optimistic Techniques.

- conflicts are rear
- if conflicts occurs, then problamatic trans rolled back (reen).
- Read Phase: transactions execute krega or local variables (log maintain) me store krega.
- Validation Phase: logs will be checked and if transactions are conflicting then they'll be rolled back & restarted.

WAL
wait ahead log.