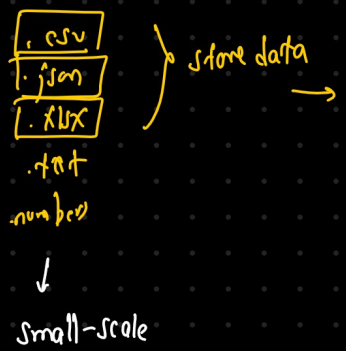
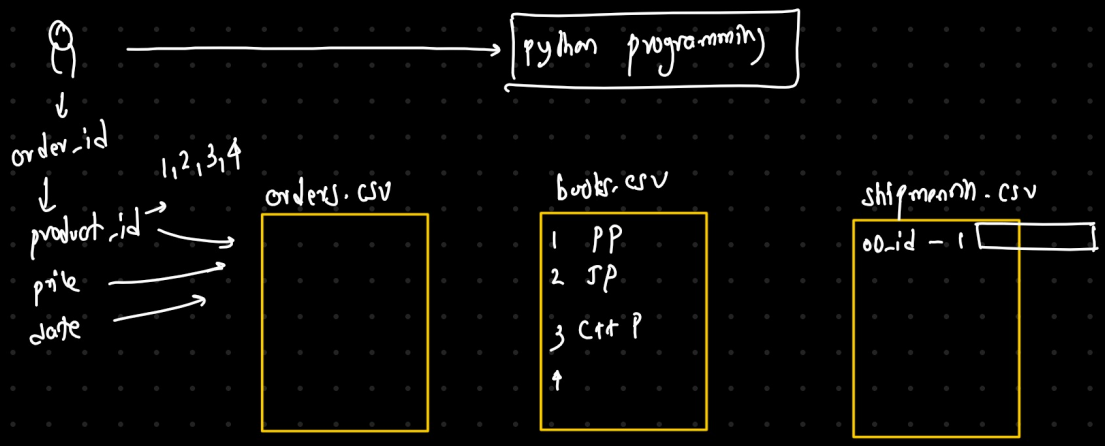
28-06-2025

Database - I : is an organized system to store, manage & retrieve data efficiently.



e.g. Big scale system (Amazon)



issue ! —

(1) Two employees open orders.csv (update) (concurrent)

(2) Employee, updates a price -100 → Corrupts.

Data integrity.

(3) million of rows → searching will be slow. → 40 sec

(4) .csv → Everyone can access it.

(5) orders.csv shipment.csv

or_id, date, productid or_id, track, status.

P.1 → get all the orders which are shipped this month.

Relationship: manually - error-prone and slow

Database

Relational Database

column (attribute) property

Name	Age	M.N	Email
Monal			

row - data point

Fixed schema

Table & row

Relation DB / SQL

Non-relational Database

Image_001.jpg 

a.mp3 → 

a.pdf → 

Link-1 → url

Flexible →

Name:
Age:

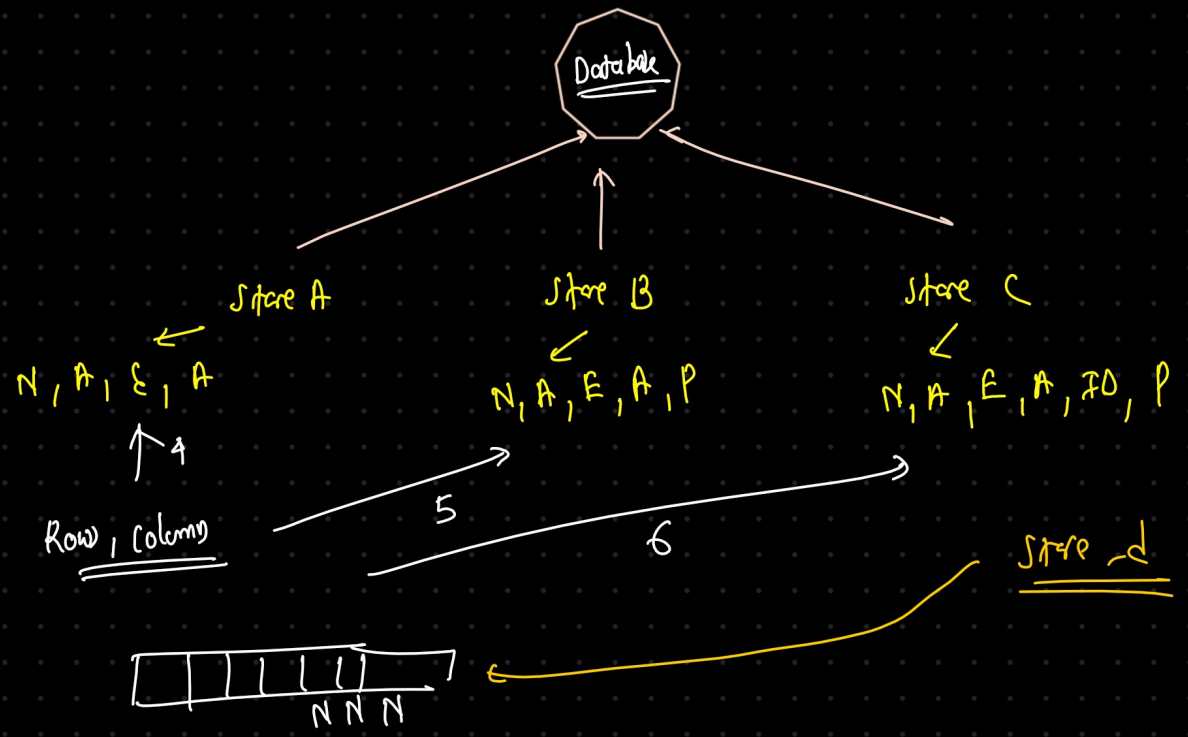
→

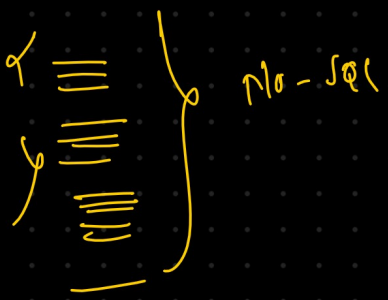
Name:
Age:
Email

→

Document, key-value, graph

Non-relation DB / NOSQL





Schema :— Blueprint of your table / Database.

P.S. School Admission.

Student No. / Roll No. → string
Class → string
Name → string
Age → int
Parent Name → string
gender → string
Admission Date → string → date time
DOB → date time

DD/MM/YYYY
MM/DD/YYYY

SQL → Structured Query language → interact with Database (Relational)

↓ ↘ sequel

S Q L

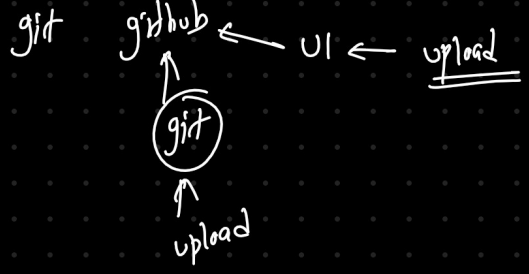
SQL ≠ Database

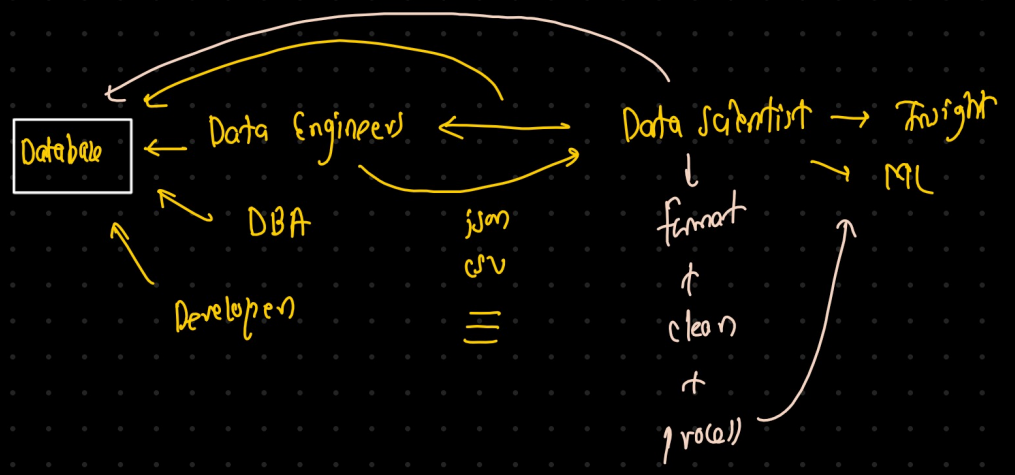
↓

- → PostgreSQL
- → MySQL
- → SQL Server
- → Oracle SQL

} Relational DB

→ PostgreSQL → free





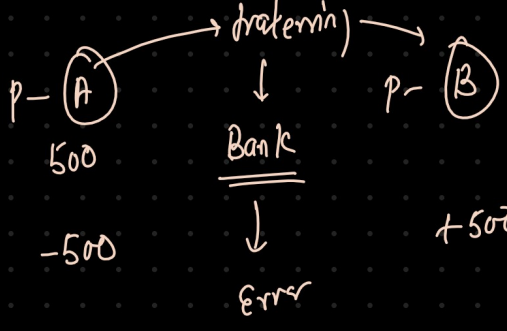
ACID

A → Atomicity →

C → Consistency → state

I → Isolation → lost update

D → Durability → commit, permanent



succedd or fail

Atomicity → w/o A, money could be debited but no product shipped.

Consistency → w/o C, you might buy products with invalid reference.

Isolation → w/o I, concurrent purchases could overwrite stock quantities.

Durability → w/o D, confirmed orders could vanish after crashes.

A – Atomicity – either the whole thing happens, or nothing happens.
C – Consistency — Always follow the rules, stays fair and balanced.
I – Isolation — one at a time
D – Durability — once it's done, it's saved forever.

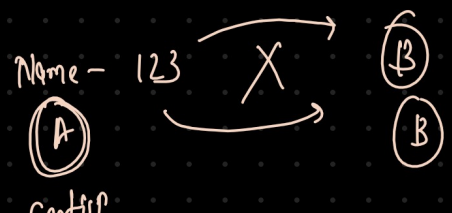
Name → 123 → X → Consistent

↓

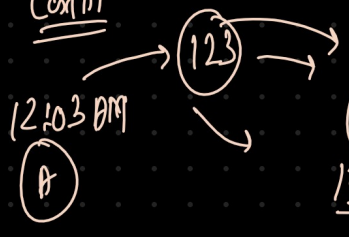
success → A → B

All defined rules

all defined rules are repeated.



Confirm



Name → Jenny

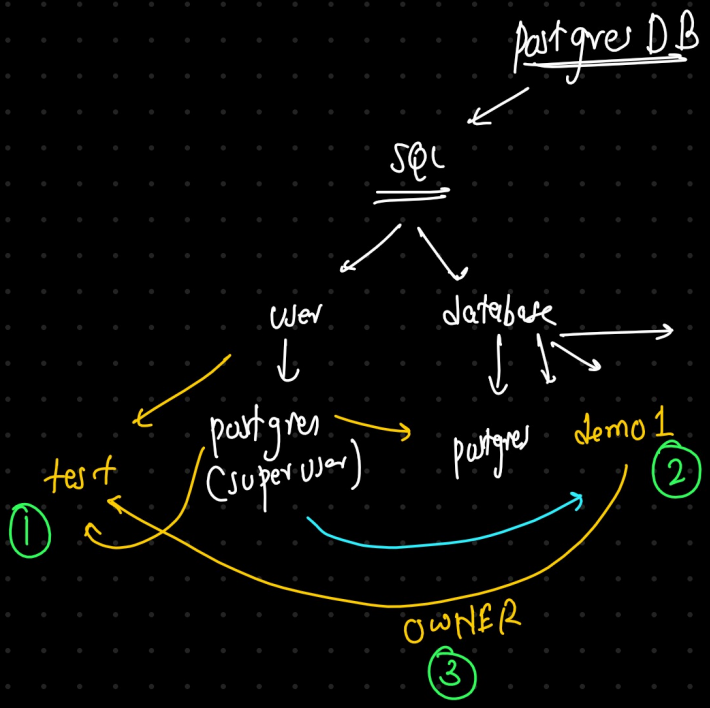
Name → SSN ↑ Number

A → 12:00 AM
B → 12:00 AM
C → 12:00 AM
} started at the same time

A
↓
B
↓
C

half update

Postgres DB



psql -U postgres -d postgres

psql -U postgres -d demo1

1. .sql → instruction
   - create db → test
   - create table
   - Insert data

2. open vs →

3. login superuser
   \i path/to/.sql

   successfull

4. logout - \q

5. psql -U test -d chinook

6. login to superuser
   - run 3 permission command

7. login to test with chinook
   chinook=# SELECT * FROM ARTIST;