Normalization

- Make database simple to keep data
- Make it optimize for insert/update
- Make it splitted as possible

- **Benifits**
- → Managing data easier / simpler/ Optimized
- → Data storage less used
- → Less redundant data

- **Drawbacks**
- Analytics is not recommended
- Projection/Selection less optimized

ACID

- **Isolation** (Levels)

- Read uncommitted
- Read Committed
- Repeatable
- Serializable

ACID

- **Isolation** (Levels)

- Read uncommitted: Allows reading uncommitted changes
- Concurrency problem:
  - **Dirty Read**: Reading uncommitted changes
- Table: products(apple, 300)
- Trx A: update> 300+50 = 350
- Trx B: Read > 350
- Trx A: Rollback (Uncommit)

Highest concurrency but concern with data accuracy (probably)

ACID

- **Isolation** (Levels)

- Read committed: Does not allow reading uncommitted changes by other trx
- Concurrency problem:
  - **Non repeatable read**: Gets different values from same data vin multiple reads
  - Table: products(apple, 300)
- Trx B: Read > 300
- Trx A: update> 300+50 = 350
- Trx A: Commit

In repeated read data will be different.

ACID

- **Isolation** (Levels)

- Repeatable Read: A trx can read a value , other trx cannot update until that trx commits, however other trx can insert new values.
- Concurrency problem:
    - Phantom read: When A trx Reading values , sees new rows
    - Table: products(apple, 300)
- Trx B: Read > ALL FRUITES…
- Trx A: update> 300+50 = 350 (LOCKED)
- Trx C: INSERT > BANANA
- Trx B: 2 rows
- ---------------------
  NAME | Price

  Apple | 120
  banana| 150

  Trx A read: Show me all fruites those are less than 160 tk
  Trx B update: --------------------------(LOCKED)
- Trx C Insert:
  Orange | 145
  Pineapple | 220

  Trx A: Sees 3 fruites [Apple, banana, orange] (1 fruit phantom)

- Solution: Serializable

ACID

- **Isolation** (Levels)

- Serializable: Highest level of isolation
- Concurrency problem:
  - No modification or insertion during other trx until committed

  - Trx A: read all fruits where price > 160
  - Trx B: update (X)
  - Trx C: Insert (X)
  - Trx D: Delete (X)

Solves all issues of concurrency.

PROBLEM: Performance issue, Query cost

Normalization

- Why?
- **1**. **Remove data redundancy**
- **2. Data Anomaly**
- **3. Data integrity**
- 4. Query easiness
- 5. Performance (Update/Delete)
- 6. Maintenance
- 7. Scalability
- 8. Storage

Needs

- Anomaly
  - Insert Anomaly
  - Update anomaly
  - Delete anomaly

Students

Roll	Class	Section	Subj	Teacher
121	10	C	**Math**	**Mr Abul**
545	10	C	**Math**	**Mr Abul**
646	8	B	Bangla	Ms Nargis
545	10	A	Eng	Mr XYZ
646	8	B	Bangla	Ms Nargis

Anomalies (Insert)

Students

Roll	Class	Section	Subj	Teacher
121	10	C	**Math**	**Mr Abul**
545	10	C	**Math**	**Mr Abul**
646	8	B	Bangla	Ms Nargis
545	10	A	Eng	Mr XYZ
646	8	B	Bangla	Ms Nargis
987	10	A	**Math**	**Mr Abul**

Anomalies (Updation Anomaly)

Students

Roll	Class	Section	Subj	Teacher
121	10	C	**Math**	**Mr Abul**
545	10	C	**Math**	**Mr Abul**
646	8	B	Bangla	Ms Nargis
545	10	A	Eng	Mr XYZ
646	8	B	Bangla	Ms Nargis
987	10	A	**Math**	**Mr Abul**

Roll	Class	Section	Subj	Teacher
121	10	C	**Math**	**Shahadat**
545	10	C	**Math**	**Shahadat**
646	8	B	Bangla	Ms Nargis
545	10	A	Eng	Mr XYZ
646	8	B	Bangla	Ms Nargis
987	10	A	**Math**	**Shahadat**

Anomalies (DeletionAnomaly)

Students

Roll	Class	Section	Subj	Teacher
121	10	C	**Math**	**Mr Abul**
545	10	C	**Math**	**Mr Abul**
646	8	B	Bangla	Ms Nargis
545	10	A	Eng	Mr XYZ
646	8	B	Bangla	Ms Nargis
987	10	A	**Math**	**Mr Abul**

Roll	Class	Section	Subj	Teacher

Normalization

- 1 NF
    - Has a PK
    - Each column should have unq values
    - Duplicate rows not allowed
- 2NF
    - Must be in 1nF
    - No partial dependency/ No Non prime attrbute
- 3NF
    - Must be in 2NF
    - No transtitive dependency
- …
- …
- …
- …

Students

1NF

Roll	Class	Section	Subj	Teacher
121	10	C	**Math, Eng, Bang**	**Mr Abul, Nargis, Mr X**
545	8	A	**Eng, Bang**	**Mr Abul, Nargis**

Roll (PK)	Class	Section	Subj	Teacher
545	8	A	**Eng**	**Mr Abul**
545	8	A	**Bang**	**Nargis**

Q&A