Unit-1: Fundamental Concepts

Definitions

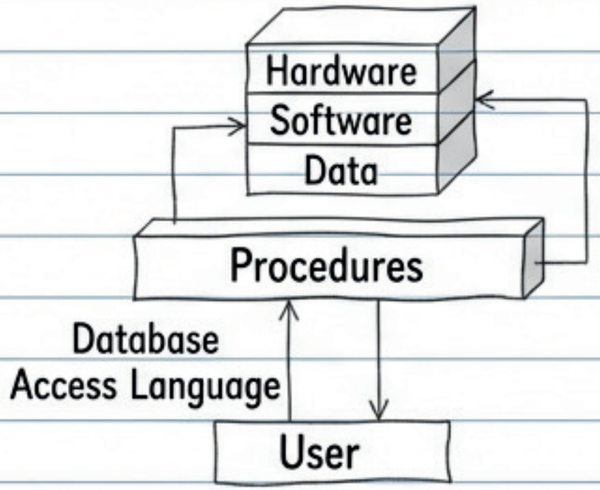
- **Data:** Known facts that can be recorded and that have implicit meaning.
- **Information:** Processed data; transform the data into a meaningful form.
- **Database:** A collection of related data.
- **DBMS (Database Management System):** Collection of programs that enables user to create and maintain a database. General purpose software that facilitates defining, constructing, manipulating, and sharing databases.
  - Defining: Datatype, structure, constraints.
  - Constructing: Storing on media.
  - Manipulating: Querying/retrieving.
  - Sharing: Concurrent access.

Characteristics of DBMS

1. Data stored in tables.
2. Reduced Redundancy.
3. Data consistency.
4. Support multiple users & concurrent access.
5. Query language.
6. Security.
7. Support transactions.

Components of DBMS

1) Hardware,
2) Software,
3) Data,
4) Procedures,
5) DB Access Language.



End Users

1. Casual: Occasionally access.
2. Naive/Parametric: Constantly query/update (e.g., Bank clerks).
3. Sophisticated: Scientists, engineers, analysts.
4. Standalone: Maintain personal database (e.g., tax package).

DBMS Architecture & Actors

Schema (Intension): Overall description/structure of DB. Rarely changes. (Types: Logical, Physical, View).

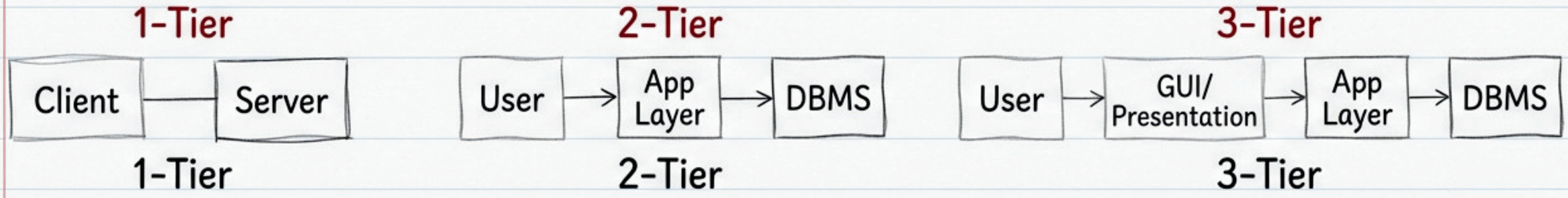
Instance (Extension): Collection of info at a particular moment. Changes frequently via updates/deletions.

	Schema	Instance
1	Overall description.	Collection of info at moment.
2	Same for whole database.	Data can be changed.
3	Does not change frequently.	Changes frequently.
4	Defines how data is stored.	Set of information at particular time.

Actors: Database Administrator (DBA), Database Designers.

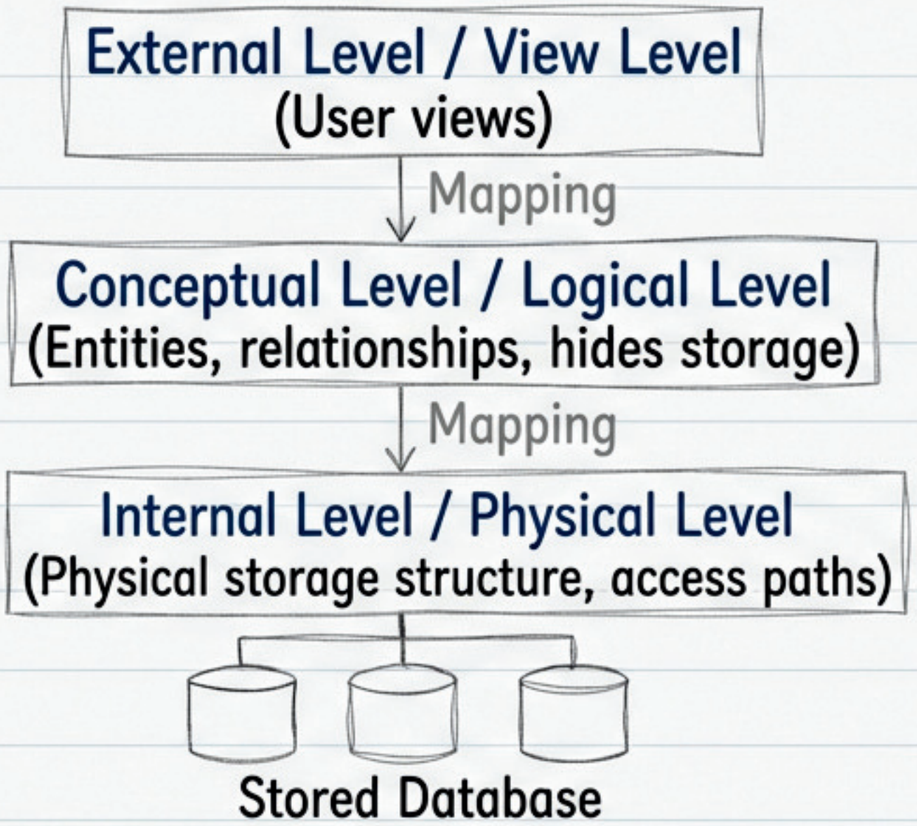
Advantages: Solves redundancy, Security, Data Integrity, Multi-user support, Backup.

Disadvantages: Costly, Complex, Large size.



Three Schema Architecture & Languages

The Architecture



Data Independence:

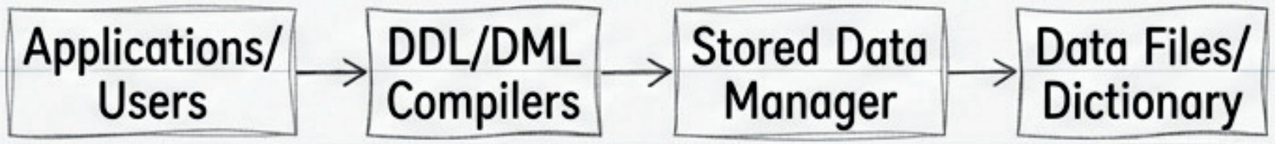
Logical DI: Change conceptual without changing external.

Physical DI: Change internal without changing conceptual.

Database Languages

- DDL (Data Definition): Create, Alter, Drop, Rename, Truncate. (Defines structure).
- DML (Data Manipulation): Insert, Update, Delete. (Handles data).
- DCL (Data Control): Grant, Revoke. (Access rights).
- TCL (Transaction Control): Commit, Rollback.
- DQL (Data Query): Select.

Interfaces: Menu-based, Forms, GUI, Natural Language, Speech I/O, DBA Interface.

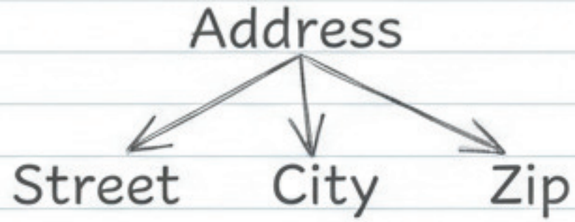


ER Model Concepts & Notation

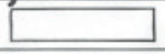
- **Entity:** Real-world thing with independent existence (Physical or Logical).
- **Entity Type:** Collection of entities with same attributes.
- **Entity Set:** Collection of all entities in DB at a point in time.
- **Attribute:** Property describing an entity (e.g., Name, Salary).
- **Key Attribute:** Unique value for each entity.

Attribute Types (Draw small examples):

1. Composite: Sub-parts (Address -> Street, City, Zip).
2. Multi-valued: Multiple values (Phone, Degree).
3. Single valued: (Age).
4. Derived: Calculated from stored (Age from DOB).
5. Null: Unknown/Not applicable.
6. Complex: Composite + Multi-valued.



Notation Table (Draw a grid):

Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Identifying Relationship
	Attribute
	Key Attribute
	Multi-valued Attribute
	Composite Attribute

Relationships & Cardinality

Definitions:

- **Relationship:** Association among n entities.
- **Degree:** Number of participating entity types (Binary=2, Ternary=3).
- **Recursive Relationship:** Entity type participates with itself (e.g., Employee supervises Employee).

Cardinality Ratios (Draw these 4 distinct diagrams):

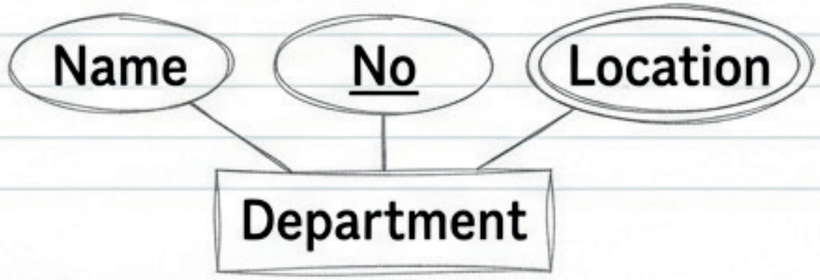
One to One (1:1):

One to Many (1:N):

Many to One (M:1):

Many to Many (M:N):

**Example Sketch:



Basic SQL Commands (Interlude)

1. CREATE Table:

```
CREATE TABLE Student (RollNo int, Name varchar(20), class varchar(20));
```

2. INSERT Data:

```
INSERT INTO Student (RollNo, Name, class) VALUES (10, 'ABC', 'MCA');
```

3. SELECT (Retrieve):

```
SELECT * FROM Student;
SELECT RollNo FROM Student;
SELECT * FROM Student WHERE class='MCA';
```

4. DELETE:

```
DELETE FROM Student WHERE Name='ABC';
```

5. UPDATE:

```
UPDATE Student SET Name='Gaurav' WHERE RollNo=10;
UPDATE Student SET class='BTECH' WHERE class='MCA';
```

Exercise: Create EMPLOYEE table (id, name, dept, salary), insert 5 records, display Name & Salary for 'MCA' dept, then update dept to 'MBA' where id=10.

Participation Constraints & Complex Diagrams

Concepts

Participation Constraints:

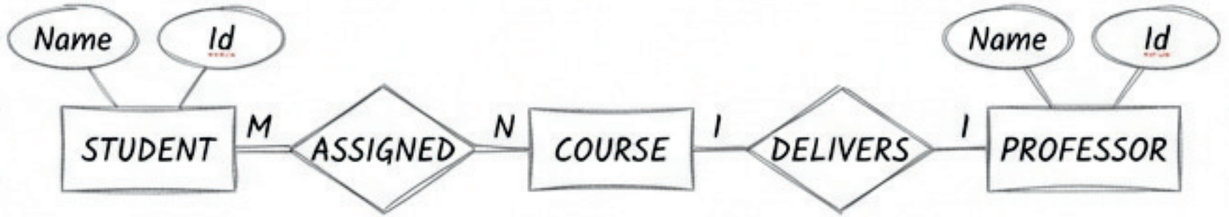
- **Total (Double Line):** Entity *must* participate. (e.g., Emp works for Dept).
- **Partial (Single Line):** Not all entities participate.

Weak Entity:

- No key attribute. Identified by Owner.
- **Symbol:** Double Rectangle.
- **Relationship:** Double Diamond (Identifying).

Full Diagrams

University System:



Railway System:

Concept of Keys

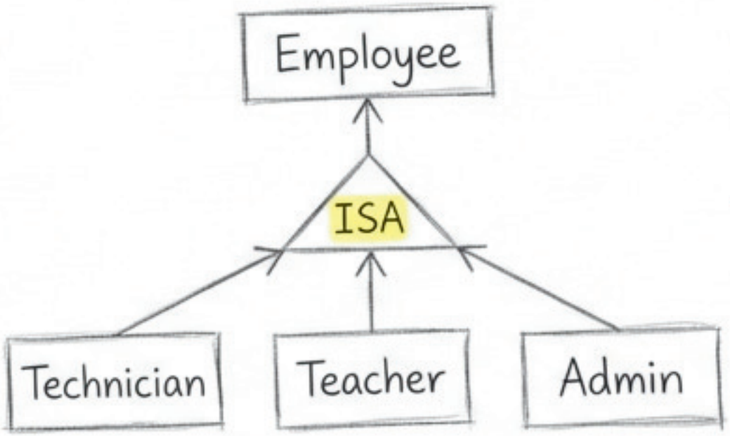
Attribute(s) which help to identify a row in a table.

Emp-id	Emp-name	Emp-adhar	Emp-sal	Email
1	ABC	1111	40k	a@gmail.com
2	XYZ	2222	50k	NULL
3	EFG	3333	60k	c@gmail.com

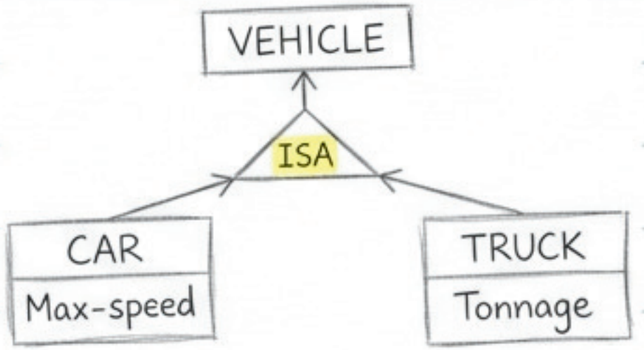
1. **Super Key:** Any combination identifying a row uniquely. (e.g., {Emp-id, {Emp-id, Name}, {Adhaar}).
2. **Candidate Key:** Minimal super key. (e.g., {Emp-id}, {Adhaar}, {Email}).
3. **Primary Key:** Selected candidate key. Not Null. (e.g., Emp-id).
4. **Unique Key:** Unique but can be Null. (e.g., Email).
5. **Alternate Key:** Candidate keys not chosen as PK. (e.g., Adhaar).
6. **Foreign Key:** Refers to PK of another table. Maintains integrity.
7. **Composite Key:** PK having >1 attribute.

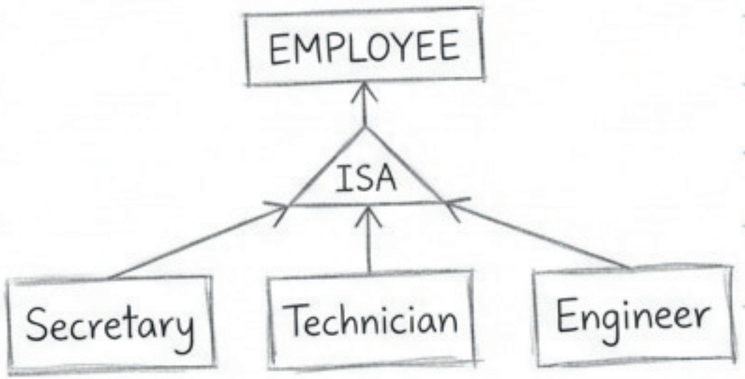
Extended/Enhanced E-R Model (EER)

1. **Generalization** (Bottom-Up):
   Combining **lower level entities** into a **higher level entity**.



2. **Specialization** (Top-Down):
   Dividing **higher level entity** into **specialized lower level entities**.





3. **Aggregation**:
   Treating a **relationship** as an **entity**.
   (Detailed in later slide)

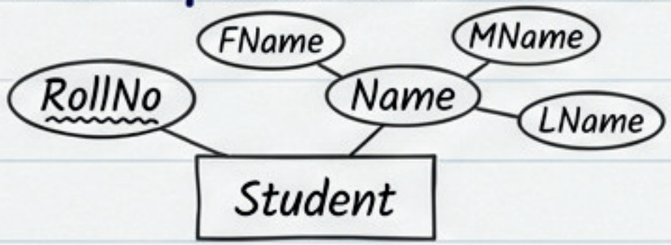
Converting ER Diagrams to Relational Model (Part I)

Step 1: Strong Entity: Entity becomes Table. Attributes become Columns.

Student

RollNo	Name	Gender

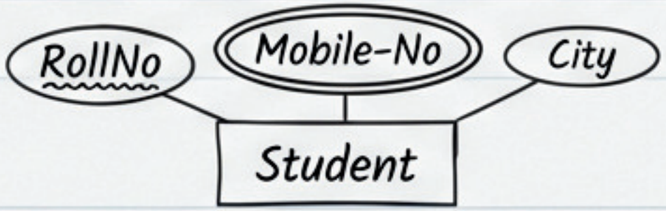
Step 2: Composite Attribute: Flatten into individual columns.



Student

RollNo	FName	MName	LName

Step 3: Multi-valued Attribute: Requires NEW separate table and Composite PK.



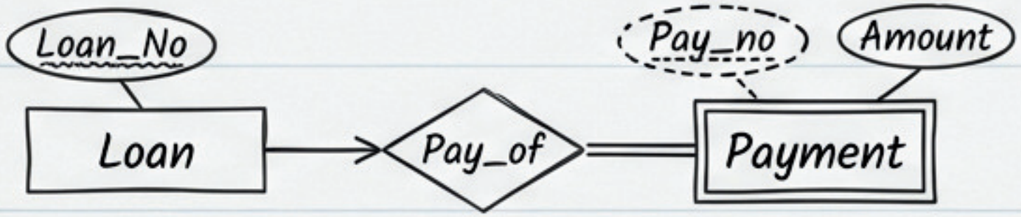
Student

RollNo	City

Mobile

RollNo	Mobile-No

Step 4: Weak Entity: Table includes Owner Key (FK), and Foreign Key.



Payment

Loan_No	Pay_no	Amount

Converting ER Diagrams to Relational Model (Part II)

Step 5: Binary Relationships:

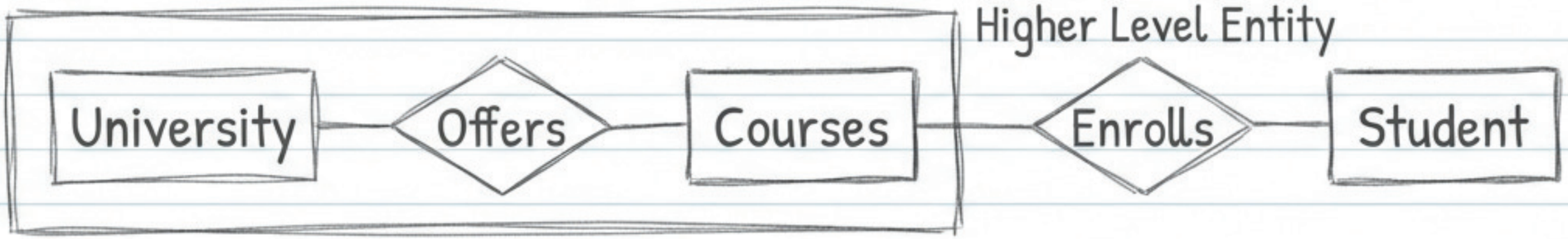
- M:N (Many-to-Many): Create 3rd Table containing PKs of both.
- 1:N (One-to-Many): Put PK of '1' side into 'N' side as FK.
- 1:1 (One-to-One): Add PK of one to other (preferably to Total Participation side).

Step 6: Generalization Mapping:

Create table for Superclass. Create separate tables for Subclasses with PK referencing Superclass.

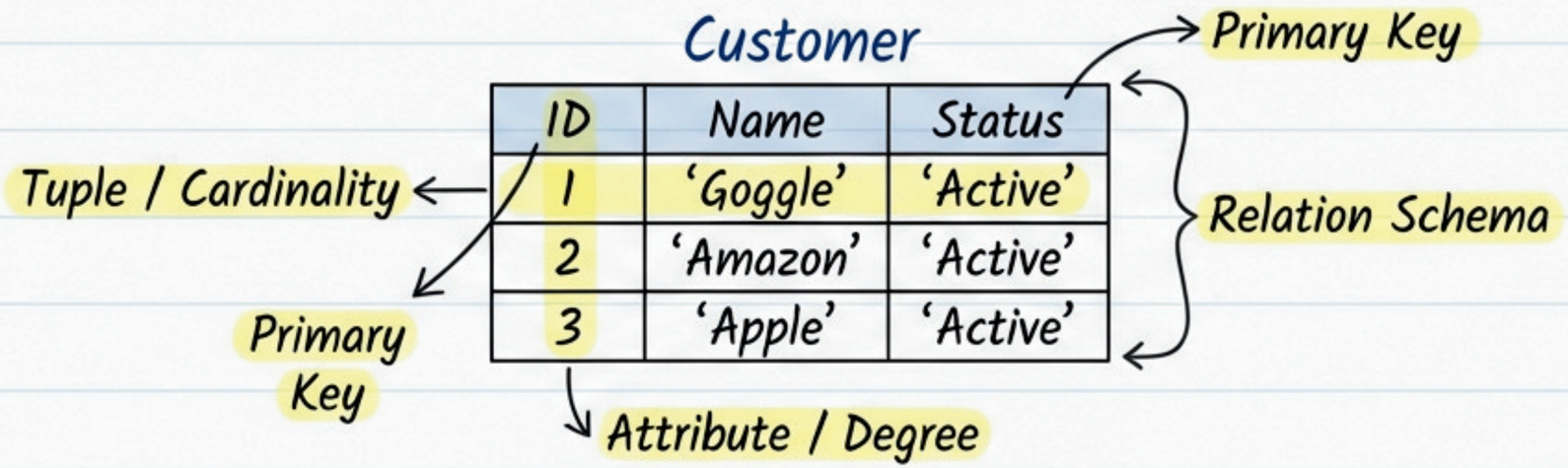
Step 7: Aggregation Mapping:

Design strategy: Relationship modeled as an entity.



Mapping: Create table for the relationship 'Offers' and use its ID to relate to Student.

The Relational Model & Integrity



Integrity Constraints

Domain Constraints:
Valid values (e.g., Age is int, not char).

ID	Name	Age
102	'ABC'	'A'

Age is integer value. ✗

Entity Integrity:
PK cannot be NULL.

ID	Name	Age
NULL	'John'	20

Not allowed as primary key can't contain NULL value.

Referential Integrity:
FK must match PK or be Null.

Table 1

Emp_No	Name	D-No
3	'EFG'	18

Not allowed - F.K ✗

Table 2

D-No	D-Location
11	Noida
24	Noida
35	Delhi

Key Constraints:
Keys must be unique.

ID	Name
11	'ABC'
22	'ABC'

Relational Algebra: Set Operations

Procedural query language. Input: Relation → Output: Relation.

Unary Operations:

- **Select (σ):** Selects horizontal subset (rows). Condition predicate.

$$\sigma_{\_topic='DB'}(Books)$$

- **Project (π):** Selects vertical subset (columns).

$$\pi_{\_Name,City}(Customer)$$

Set Theory Operations (Binary):

Table A

Col1	Col2
1	1
1	2

Table B

Col1	Col2
1	1
1	3

- **Union (A ∪ B):** {1, 2, 3} (All rows).
- **Intersection (A ∩ B):** {1} (Common rows).
- **Difference (A - B):** {2} (In A not B).
- **Difference (B - A):** {3} (In B not A).

Relational Algebra: Advanced Operations

1. **Cartesian Product (X)**: Combines every row of A with every row of B.

R_1

Rows
Vijay
Gopal

R_2

Rows
English
Maths

$R_1 \times R_2$

(Vijay	Eng)
(Vijay	Maths)
(Gopal	Eng)
(Gopal	Maths)

4 rows

2. **Rename (ρ)**: Unary operation to rename relation or attributes.

$\rho_S(R) \longrightarrow$ Renames R to S.

3. **Division Operator ($\div$)**: Used for "Select for all" queries.

A

Student	Project
S1	P1
S1	P2
S2	P1

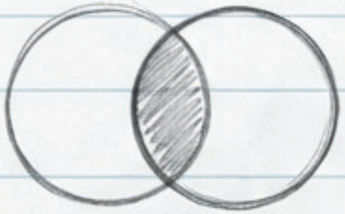
B

Project
P1
P2

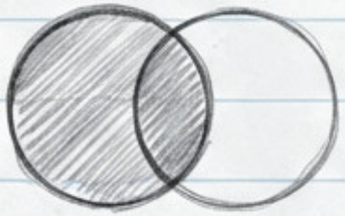
$A \div B = \{S1\}$ (Students who have done ALL projects in B)

PL/SQL Overview

Joins (Venn Diagrams)

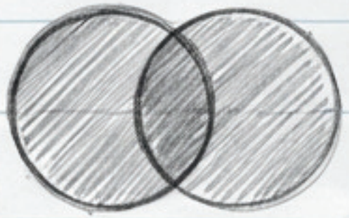


Inner Join
Matching rows only.



Left Outer Join
All left + matching right.

Right Outer Join
All right + matching left.



Full Outer Join
All rows.

Natural Join:
On common attribute.

PL/SQL

PL/SQL: Procedural Extension of SQL.
Features: Variables, Loops, Conditions, Exception Handling.
Structures: IF-ELSE, LOOP, WHILE, FOR.
Concepts:
- Cursors: Process multiple rows.
- Triggers: Auto-execution.
- Transactions: Commit, Rollback.
- Locking: Concurrency control.