# Chapter 12
# Relational Database and SQL

**Database**

Logically organized collection of data, can be considered a repository of stored data about an entity.

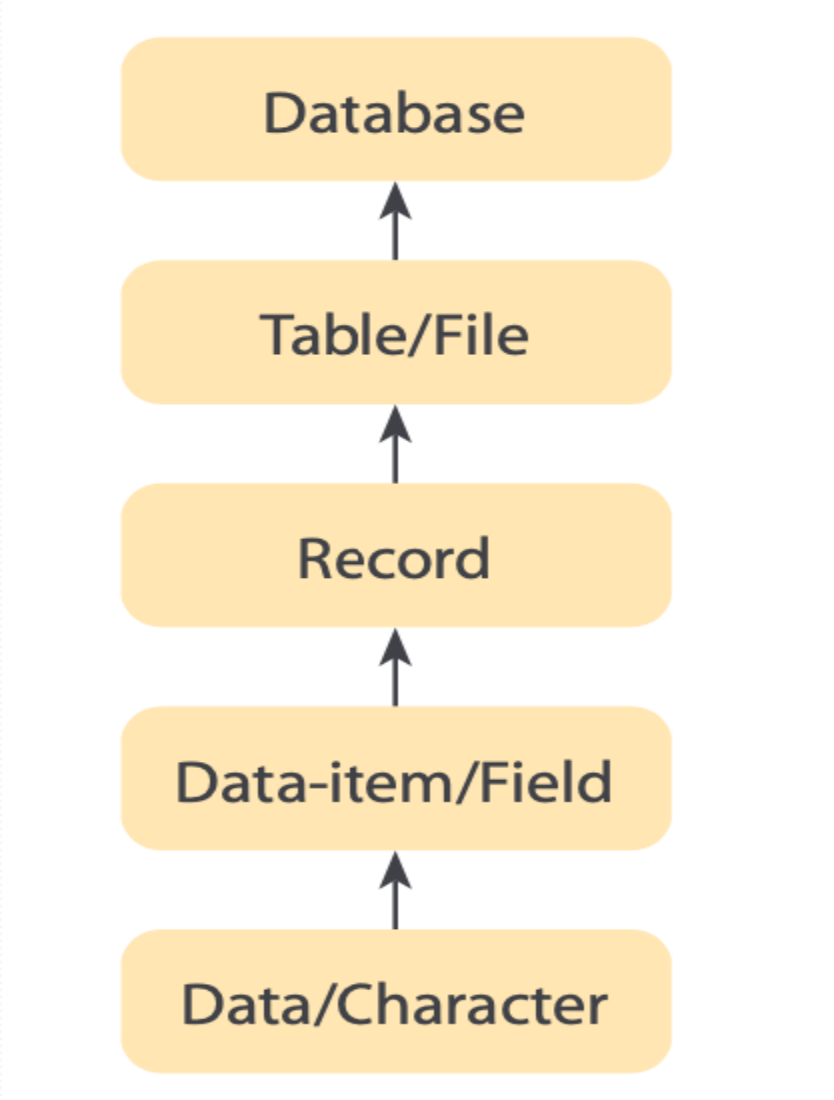
It is a shared collection of related data/information used to support the activities and decision-making of any organization.

Database - it is easy to enter, access, analyse data quickly and easily.

Database is integrated and well-shared.

Database is organised collection of interrelated data that serves many applications.

# FILE ORGANIZATION



## DBMS

**Database Management System is a collection of programs that enables users to create and maintain a database.**

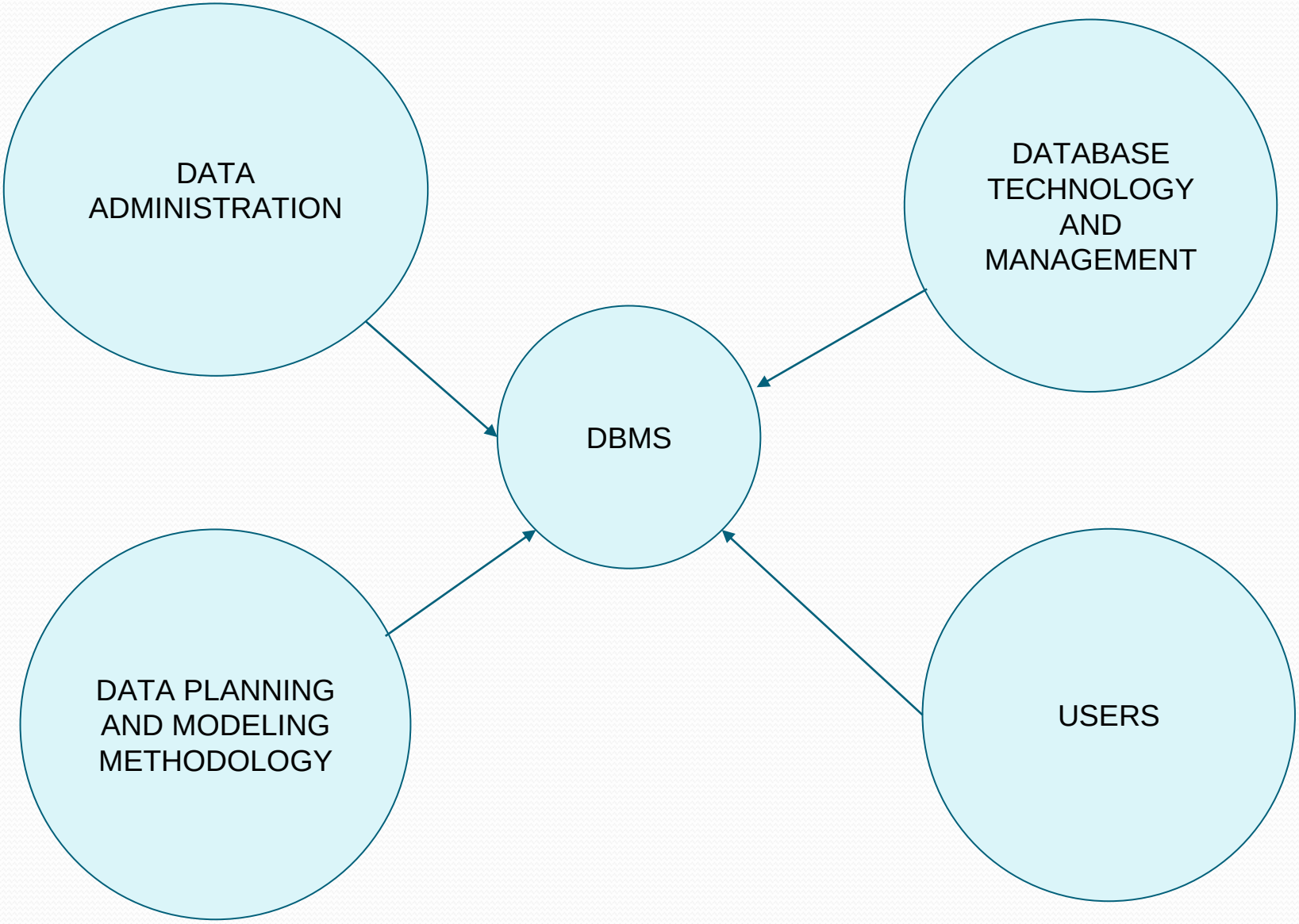
**It is a general purpose software that provides the users with the process of defining, creating, retrieving and manipulating the information when needed.**
**Eg. : Oracle, Sybase, PLSQL ..**

- **Database System, why is it needed?**

**The database and DBMS software together is called a database system.**

**It is basically a computer based record keeping system.**

**The intention of a database is that the same collection of data should serve as many applications as possible.**


DATA ADMINISTRATION
DATABASE TECHNOLOGY AND MANAGEMENT
DBMS
DATA PLANNING AND MODELING METHODOLOGY
USERS

## Advantages of Database system

Reduce data redundancy
Control data inconsistency
Data can be shared
Reduced programming effort
Standardized
Improved data integrity
Privacy and Secured
Economical
Integrate data
Improved Backup and Retrieval
Meeting enterprise requirements than individual requirements

## Disadvantages of Database System

Additional hardware is required, System is complex while developing, size …

## Types of users in database system

- End user
- Application system analyst
- Physical storage system analyst

## Data abstraction

**Provide the users only that much information that is required to them, this means that the system does not disclose all the details of data, rather it hides certain details of how the data is stored and maintained.**

# Levels of data abstraction

**Internal (physical)**

**Conceptual(logical)**

**External(view)**

## Data Independence

**Allows modification of a scheme definition of one level without affecting a scheme definition in the next higher level.**

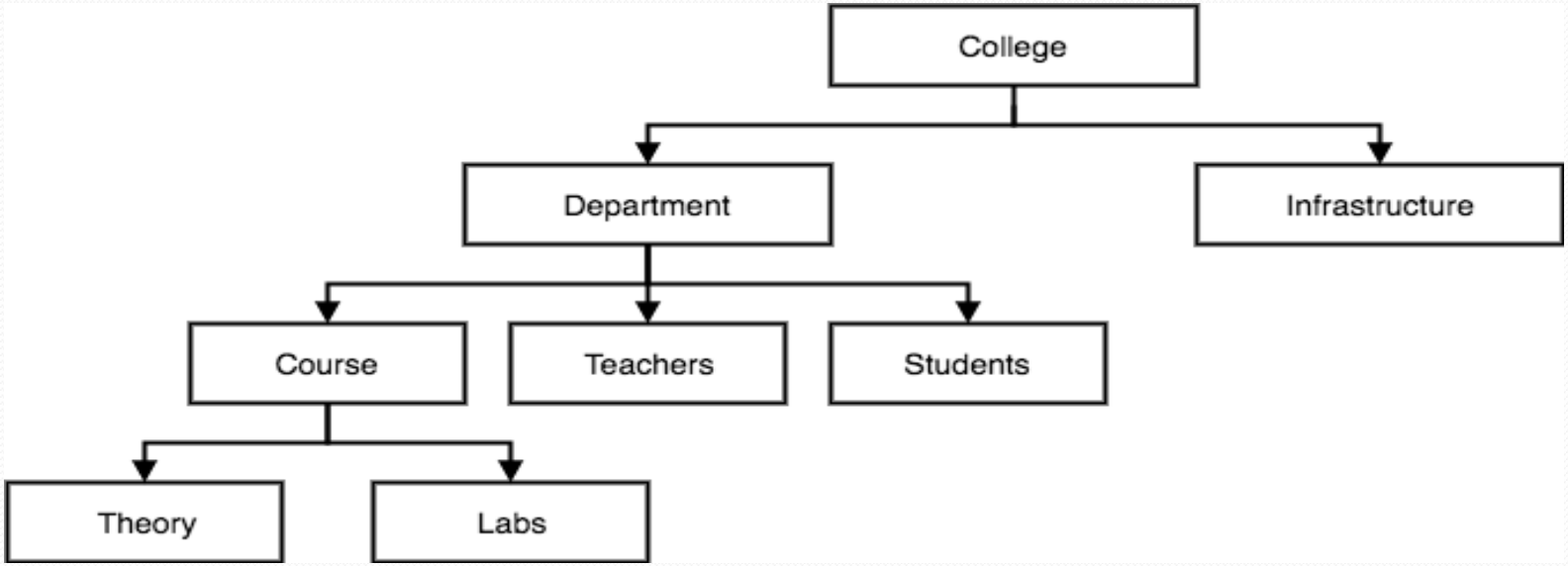
## Types of Data Models

1) **Object Based data model**
2) **Record Based data model**
   - **Hierarchical (Tree Structure)**
   - **Network (Directed graph)**
   - **Relational DBMS(Files/Tables/Relations)**

FEATURES OF DATA

A) Data models should be presented graphically using diagrams and symbols.
B) Data representation in a data model should have no data redundancy.
C) A data model should be made available and shared by various applications
D) Data represented should be consistent, stable and valid in all aspects.

# Hierarchical (Parent child – one to many)




Advantages :

1. Simple to represent
2. Easy to maintain data integrity
3. Promotes data sharing

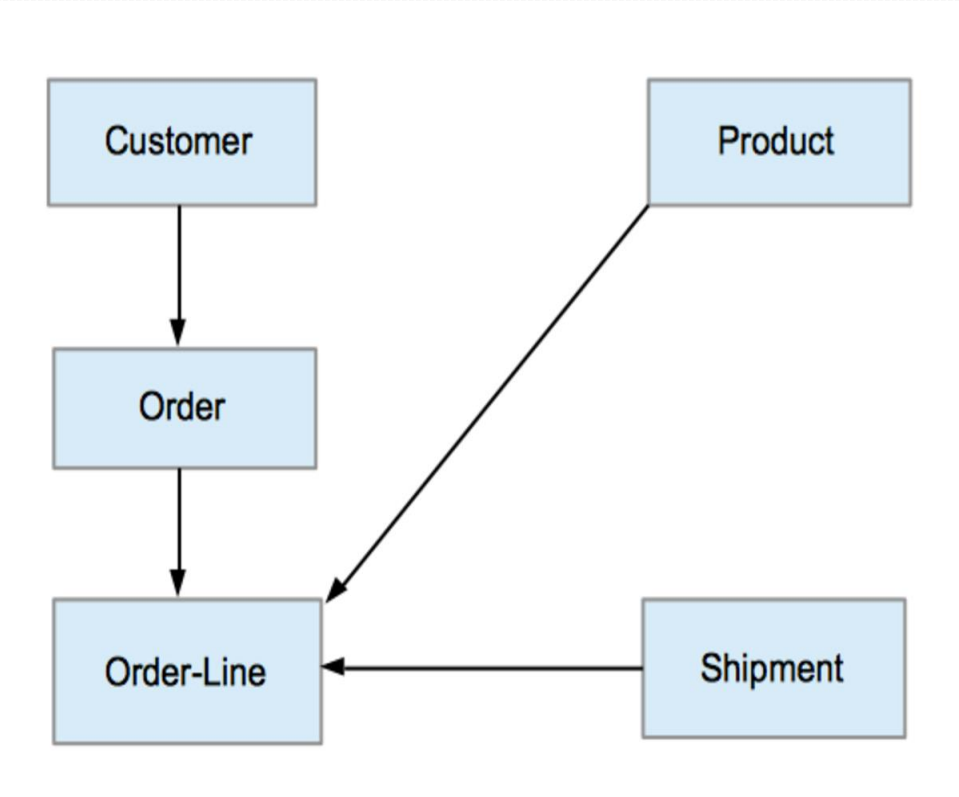
Disadvantages :

1. Implementation is possible only for hierarchical type and requires pointer concept to maintain the relationship and hence occupies more space.
2. Changes in structure (may be deletion of a node) affects/needs to be implemented in all application(time consuming)

Example:

IBM's Information Management System.

# Network (Many to Many)




Advantages :

1) Better data integrity than Hierarchical
2) Data access is easy
3) Supports both DDL and DML comamnds

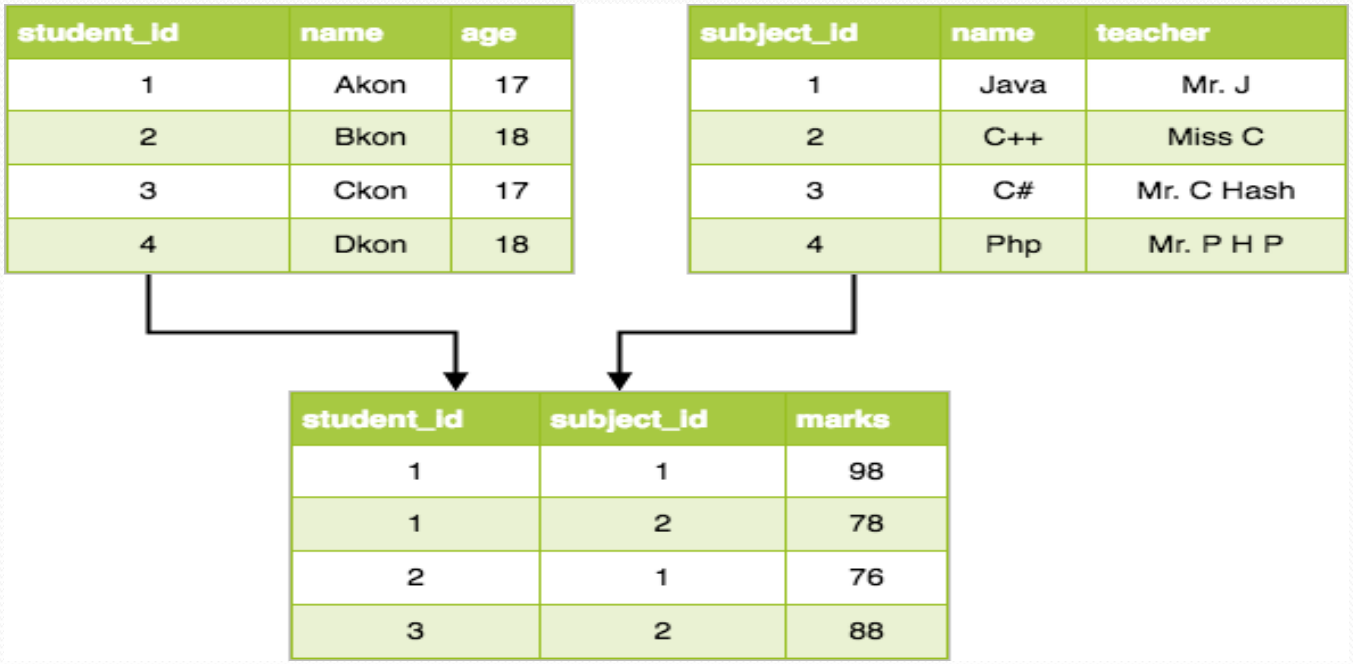
Disadvantages :

1) Structure development is more complex
2) Need more links to created using pointer concepts - more memory space
3) Change of structure need to be made effective in all applications

Example :

Univac DMS 1100

# Relational DBMS



| student_id | name | age |
|---|---|---|
| 1 | Akon | 17 |
| 2 | Bkon | 18 |
| 3 | Ckon | 17 |
| 4 | Dkon | 18 |

| subject_id | name | teacher |
|---|---|---|
| 1 | Java | Mr. J |
| 2 | C++ | Miss C |
| 3 | C# | Mr. C Hash |
| 4 | Php | Mr. P H P |

| student_id | subject_id | marks |
|---|---|---|
| 1 | 1 | 98 |
| 1 | 2 | 78 |
| 2 | 1 | 76 |
| 3 | 2 | 88 |

A RDBMS - is a collection of several relations or tables with unique name.

Data is represented in an orderly arrangement of data in rows and columns which makes a relation.

Columns represent the attributes that belong to an entity which describes the table.

Every row in a table is a record or a single entity.

## Advantages of using Relational DBMS

**Simplicity:**
A relational data model is simpler than the hierarchical and network model.

**Structural Independence:**
The relational database is only concerned with data and not with a structure. This can improve the performance of the model.

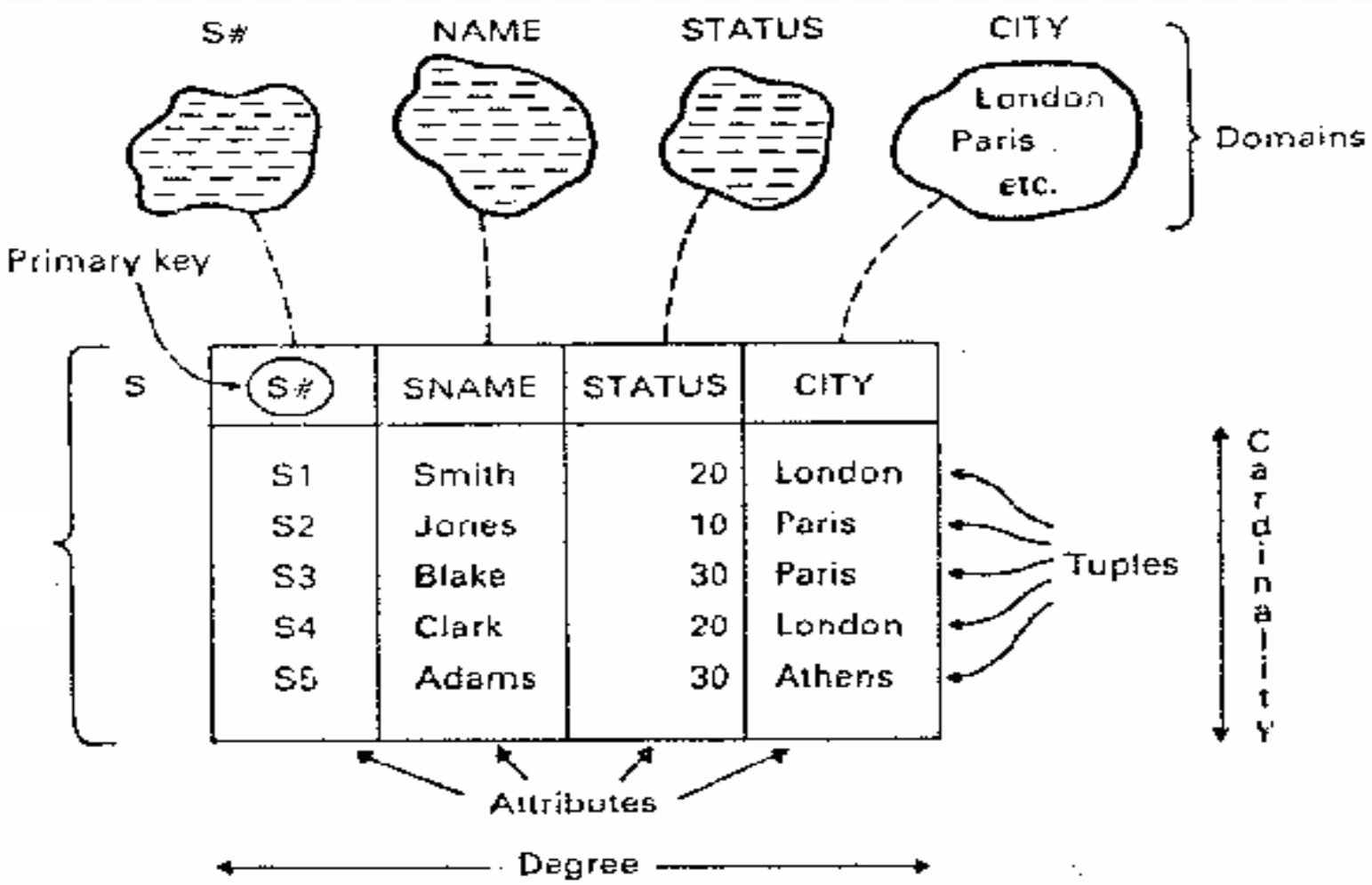
**Easy to use:**
The relational model is easy as tables consisting of rows and columns is quite natural and simple to understand

**Query capability:**
It makes possible for a high-level query language like SQL to avoid complex database navigation.

**Data independence:**
The structure of a database can be changed without having to change any application.

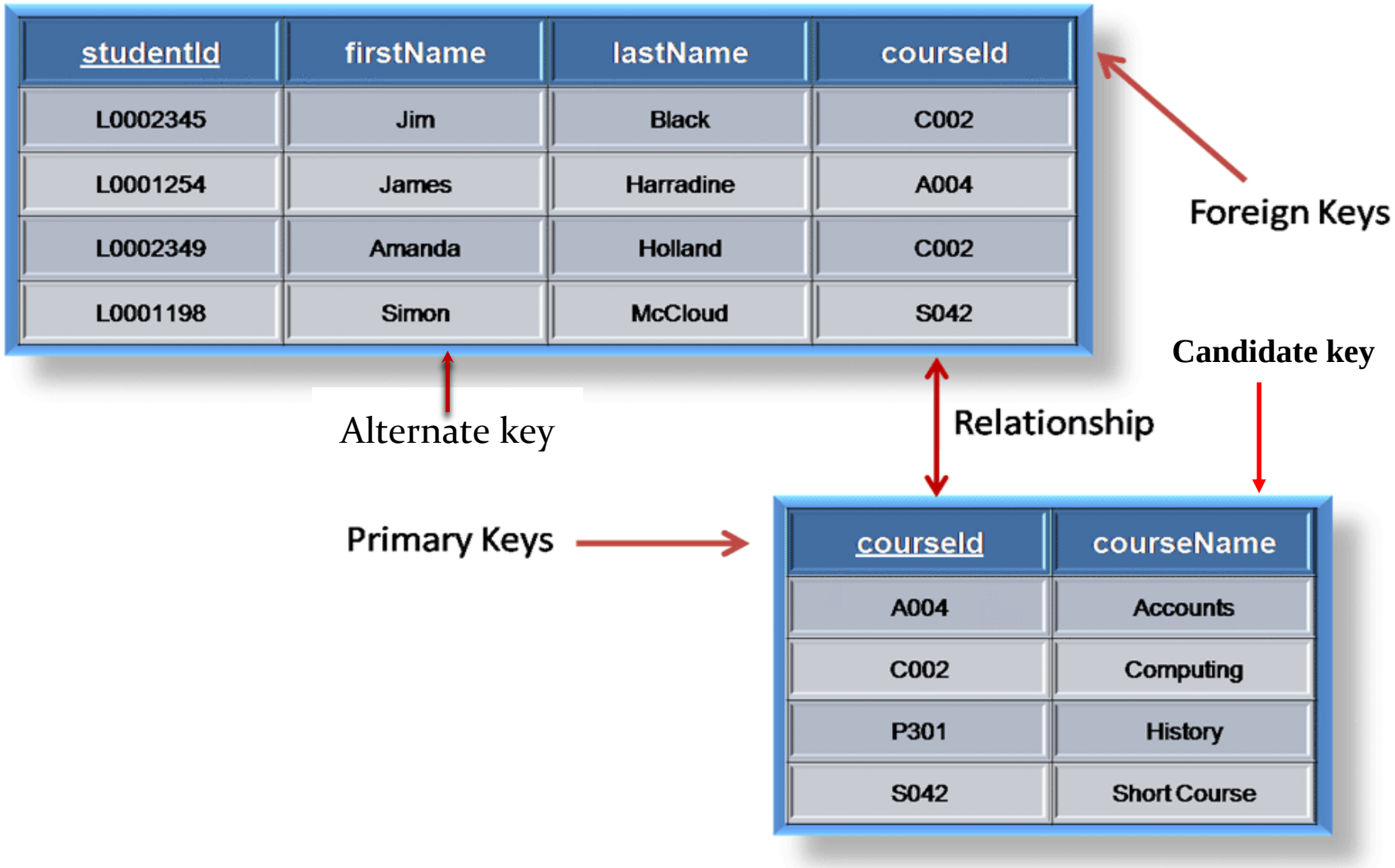
**Scalable:**
Regarding a number of records, or rows, and the number of fields, a database should be enlarged to enhance its usability.

Relation


S#
NAME
STATUS
CITY
London
Paris
etc.
Domains
Primary key
S
S#
SNAME
STATUS
CITY
S1
Smith
20
London
S2
Jones
10
Paris
S3
Blake
30
Paris
S4
Clark
20
London
S5
Adams
30
Athens
Tuples
Cardinality
Attributes
Degree

# Terms used in RDBMS

- Relation <-> Table
- Tuple <-> Row or record
- Attribute <-> Column or field
- Cardinality <-> Number of tuples(rows)
- Degree <-> Number of attributes(columns)
- Primary key <-> Unique identifier
- Domain <-> Pool of legal values


studentId
firstName
lastName
courseId
L0002345
Jim
Black
C002
L0001254
James
Harradine
A004
L0002349
Amanda
Holland
C002
L0001198
Simon
McCloud
S042
Foreign Keys
Candidate key
Alternate key
Relationship
Primary Keys
courseId
courseName
A004
Accounts
C002
Computing
P301
History
S042
Short Course

Primary Key - A Primary key is an attribute or attributes that uniquely identify tuples(records) in a relation(table).
Features of primary key:

1) It cannot be left null or cannot be redeclared.
2) One table can have only one primary key
3) Helps to arrange the table in specific order
4) Easy to query the table

Candidate key: A candidate key is an attribute in a relation which can be made primary key.

Alternate key: A candidate key that in not a primary key is called an alternate key.

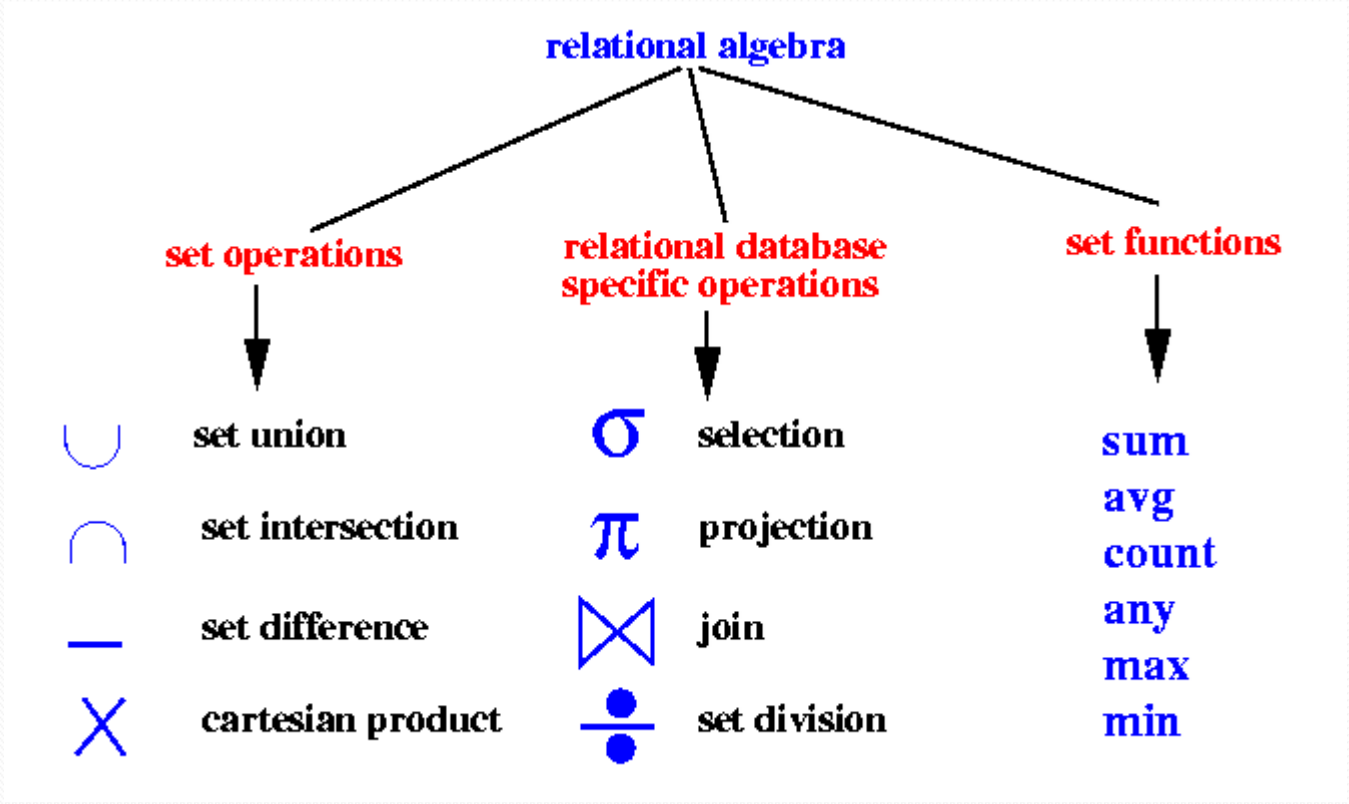
Foreign key : A non-key attribute, whose value is derived from the primary key of some other table.

# RELATIONAL ALGEBRA

**Relational Algebra** is a procedural query language used to query the database tables to access data in different ways.

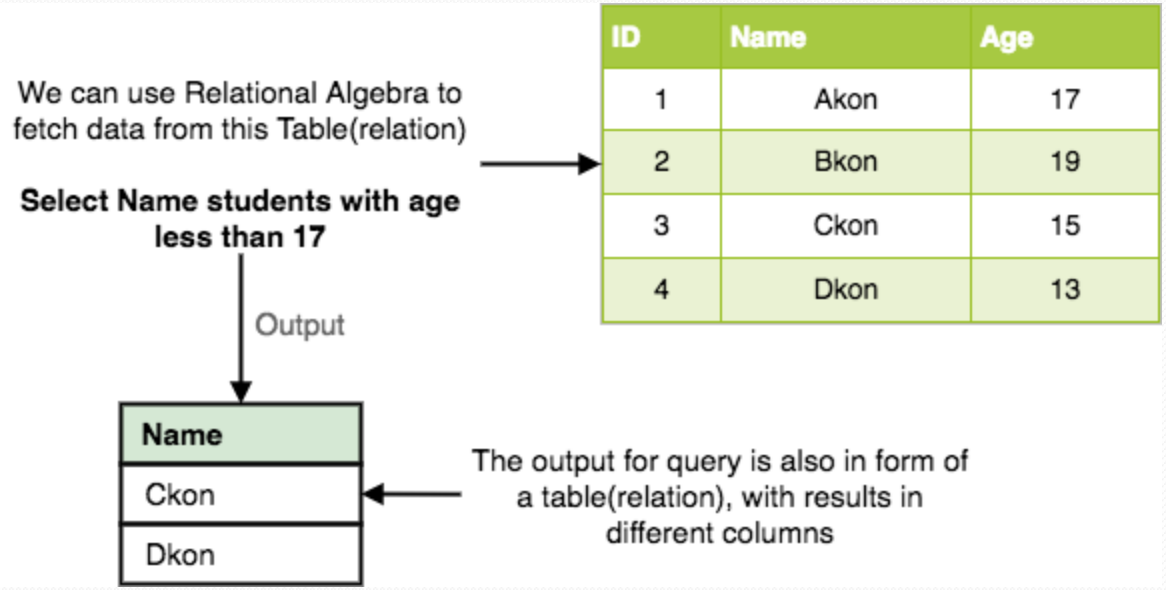
VARIOUS OPERATIONS IN RELATIONAL ALGEBRA

**SELECT (UNARY)**
**PROJECT (UNARY)**
**CARTESIAN PRODUCT (BINARY)**
**UNION (BINARY)**
**SET DIFFERENCE (BINARY)**
**SET INTERSECTION (BINARY)**


relational algebra
set operations
relational database specific operations
set functions
set union
set intersection
set difference
cartesian product
selection
projection
join
set division
sum
avg
count
any
max
min

# SELECT OPERATION (σ)

This is used to fetch rows(tuples) from table(relation) which satisfies a given condition.



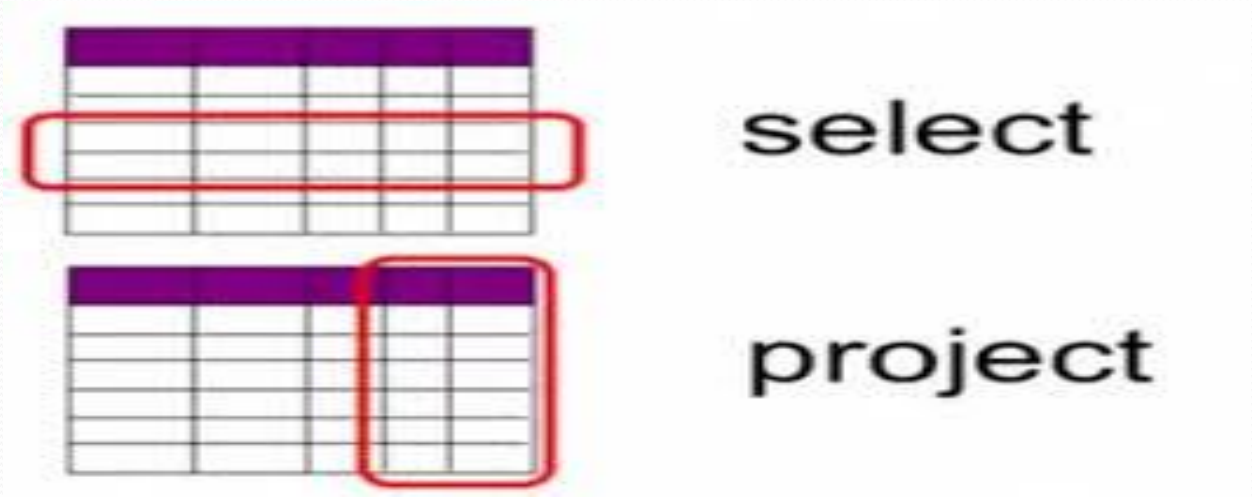
Result for Query $\pi_{Name, Age}(Student)$-

# Project Operation (∏)

Project operation is used to project only a certain set of attributes of a relation.

| ID | Name | Subject | Age |
|---|---|---|---|
| 100 | Ashish | Maths | 19 |
| 200 | Rahul | Science | 20 |
| 300 | Naina | Physics | 20 |
| 400 | Sameer | Chemistry | 21 |

| Name | Age |
|---|---|
| Ashish | 19 |
| Rahul | 20 |
| Naina | 20 |
| Sameer | 21 |


select
project

Union – returns all tuples from interconnected tables by eliminating duplicate tuples

| EMP_TEST | | | |
|---|---|---|---|
| EMP_ID | EMP_NAME | EMP_ADDRESS | EMP_SSN |
| 100 | James | Troy | 232434 |
| 104 | Kathy | Holland | 324343 |

Union

| EMP_DESIGN | | | |
|---|---|---|---|
| EMP_ID | ENAME | EMP_ADDRESS | SSN |
| 103 | Rose | Freser Town | 6744545 |
| 102 | Marry | Novi | 343613 |
| 105 | Laurry | Rochester Hills | 97676 |
| 104 | Kathy | Holland | 324343 |

| UNION | | | |
|---|---|---|---|
| EMP_ID | EMP_NAME | EMP_ADDRESS | EMP_SSN |
| 100 | James | Troy | 232434 |
| 102 | Marry | Novi | 343613 |
| 103 | Rose | Freser Town | 6744545 |
| 104 | Kathy | Holland | 324343 |
| 105 | Laurry | Rochester Hills | 97676 |

# Union Compatible

- Same number of fields.
- `Corresponding' fields have the same type.

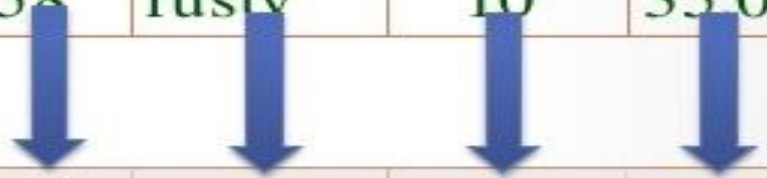
Schema of S1 = Schema of S1

S1(sid,sname,rating,age)

S2(sid,sname,rating,age)

*S1*

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |



*S2*

| sid | sname | rating | age |
|---|---|---|---|
| 28 | yuppy | 9 | 35.0 |
| 31 | lubber | 8 | 55.5 |
| 44 | guppy | 5 | 35.0 |
| 58 | rusty | 10 | 35.0 |

# Union, Intersection, Set-Difference

*S1*

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

*S2*

| sid | sname | rating | age |
|---|---|---|---|
| 28 | yuppy | 9 | 35.0 |
| 31 | lubber | 8 | 55.5 |
| 44 | guppy | 5 | 35.0 |
| 58 | rusty | 10 | 35.0 |

- All have the same schema

$S1 \cup S2$

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |
| 44 | guppy | 5 | 35.0 |
| 28 | yuppy | 9 | 35.0 |

$S1 \cap S2$

| sid | sname | rating | age |
|---|---|---|---|
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

$S1 - S2$

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |

Cartesian Product – returns all combinations of tuples from two relations.

R

| | |
|---|---|
| A | 1 |
| B | 2 |
| D | 3 |
| F | 4 |
| E | 5 |

S

| | |
|---|---|
| A | 1 |
| C | 2 |
| D | 3 |
| E | 4 |

R CROSS S

| | | | |
|---|---|---|---|
| A | 1 | A | 1 |
| A | 1 | C | 2 |
| A | 1 | D | 3 |
| A | 1 | E | 4 |
| B | 2 | A | 1 |
| B | 2 | C | 2 |
| B | 2 | D | 3 |
| B | 2 | E | 4 |
| D | 3 | A | 1 |
| D | 3 | C | 2 |
| D | 3 | D | 3 |
| D | 3 | E | 4 |

| | | | |
|---|---|---|---|
| F | 4 | A | 1 |
| F | 4 | C | 2 |
| F | 4 | D | 3 |
| F | 4 | E | 4 |
| E | 5 | A | 1 |
| E | 5 | C | 2 |
| E | 5 | D | 3 |
| E | 5 | E | 4 |

Cartesian Product of two relations – R x S
Degree = sum of the degrees of R and S
Cardinality = Product of cardinality of R and S