

# Normalization

# Normalization

- Normalization is a process of designing a consistent database by minimizing redundancy and ensuring data integrity through decomposition which is lossless.
- Normalization is a set of rules/guidelines/technique that is used while designing a database.

# Normalization

- Normalization is used for mainly following purpose,
  - *Avoid Data Redundancy.*
  - *Removing Anomalies.*
  - *Ensuring Functional Dependency.*
  - *Ensuring Functional Decomposition.*

# Data Redundancy

- **Data redundancy** is a condition created within a **database** or **data tables** in which the same piece of **data** is held in two separate places (i.e. duplication of data).
- Disadvantages of data redundancy:
  - Increases the size of the database unnecessarily.
  - Causes data inconsistency.
  - Decreases efficiency of database.
  - Leads to data corruption.

# Anomalies

- Anomaly is a deviation or condition occurred when any correct operation on data may lead to inconsistent data or violation of database constraints is called as anomaly in database.
- Types of anomaly
  - Insertion
  - Updating
  - deletion

# Functional Dependency

- Functional dependency is a relationship that exists when one attribute uniquely determines another attribute.
- If  $R$  is a relation with attributes  $X$  and  $Y$ , a functional dependency between the attributes is represented as  $X \rightarrow Y$ , which specifies  $Y$  is functionally dependent on  $X$ .
- Here  $X$  is a determinant set and  $Y$  is a dependent attribute.
- If column  $A$  of a table uniquely identifies the column  $B$  of same table then it can be represented as  $A \rightarrow B$  (Attribute  $B$  is functionally dependent on attribute  $A$ ).

# Functional Dependency

- Types of functional dependency
  - *Trivial functional dependency*
  - *Non-trivial functional dependency*
  - *Multi-valued dependency*
  - *Transitive dependency*

# Functional Decomposition

- Decomposition is the process of breaking down in parts or elements. It replaces a relation with a collection of smaller relations. It breaks the table into multiple tables in a database.
- Properties
  - Lossless Decomposition
  - Dependency Preservation
  - Lack of Data Redundancy



# Normalization

- Types of Normalization:
  - *First Normal Form (1NF)*
  - *Second Normal Form (2NF)*
  - *Third Normal Form (3NF)*
  - *Boyce-Codd Normal Form (3.5NF)*
  - *Forth Normal Form (4NF)*
  - *Fifth Normal Form (5NF)*
  - *Domain Key Normal Form (DKNF)*

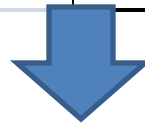
Form	Description
<u>1NF</u>	A relation is in 1NF if it contains an atomic value.
<u>2NF</u>	A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functionally dependent on the prime attribute.
<u>3NF</u>	A relation will be in 3NF if it is in 2NF and no transitive dependency exists.
<u>BCNF</u>	A relation is in BCNF if every functional dependency $X \rightarrow Y$ , where X is the super key of the table.
<u>4NF</u>	A relation will be in 4NF if it is in Boyce Codd normal form and has no multi-valued dependency.
<u>5NF</u>	A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless.

# First Normal Form i.e. 1NF

- "A relation schema R is in 1NF,
  - if it does not have any **composite** attributes,
  - **OR multi-valued** attribute or their combination".
- For a table to be in the First Normal Form.
  - All attributes (column) in the entity (table) must be single valued (atomic).
  - Each record needs to be unique.

# 1NF

ROLLNO	NAME	COURSES		ROLLNO	NAME	SUB1	SUB2	SUB3	SUB4
5101	XYZ	DST,DBMS,CN,JAVA		5101	XYZ	DST	DBMS	CN	JAVA
5102	ABC	DBMS,CN		5102	ABC	DBMS	CN	-	-
5103	LMN	DST,OM,CN		5103	LMN	DST	OM	CN	-
5104	PQR	JAVA		5104	PQR	JAVA	-	-	-



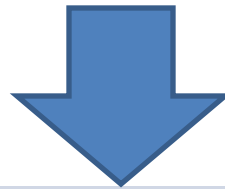
ROLLNO	NAME	COURSES
5101	XYZ	DST
5101	XYZ	DBMS
5101	XYZ	CN
5101	XYZ	JAVA
5102	ABC	DBMS
5102	ABC	CN
5103	LMN	DST
5103	LMN	OM
5103	LMN	CN
5104	PQR	JAVA

# Second Normal Form i.e. 2NF

- **Prime attribute** – An attribute, which is a part of the candidate-key, is known as a prime attribute.
- **Non-prime attribute** – An attribute, which is not a part of the candidate-key, is said to be a non-prime.
- For a table to be in the Second Normal Form,
  - It should be in the First Normal form.
  - And, it should not have Partial Dependency.
- **Partial Dependency:** If the proper subset of candidate key determines non-prime attribute, it is called partial dependency.

# 2NF

ROLLNO	COURSECODE	MARKS	TEACHER
1	6235	80	DST TEACHER
1	6236	85	DBM TEACHER
1	6238	75	CPP TEACHER
2	6235	80	DST TEACHER
2	6236	90	DBM TEACHER
3	6238	70	CPP TEACHER



ROLLNO	COURSECODE	MARKS		COURSECODE	TEACHER
1	6235	80		6235	DST TEACHER
1	6236	85		6236	DBM TEACHER
1	6238	75		6238	CPP TEACHER
2	6235	80		6235	DST TEACHER
2	6236	90		6236	DBM TEACHER
3	6238	70		6238	CPP TEACHER

# Third Normal Form i.e. 3NF

- A table is said to be in the Third Normal Form when,
  - It is in the Second Normal form.
  - And, it doesn't have Transitive Dependency.
- **Transitive Dependency:** if any non-prime attribute is functionally dependant on other non-prime attribute which is functionally dependant on prime attribute.
- **R(ABC) :  $A \rightarrow B \rightarrow C$**  (A is prime, B is non-prime, C is non-prime )

# 3NF

ROLLNO	NAME	PINCODE	CITY	STATE
1	XYZ	422101	NASHIK	MH
2	ABC	423105	THANE	MH
3	ABC	414103	PUNE	MH
4	PQR	422101	NASHIK	MH
5	XCZ	450114	SANGALI	MH



ROLLNO	NAME	PINCODE	PINCODE	CITY	STATE
1	XYZ	422101	422101	NASHIK	MH
2	ABC	423105	423105	THANE	MH
3	ABC	414103	414103	PUNE	MH
4	PQR	422101	422101	NASHIK	MH
5	XCZ	450114	450114	SANGALI	MH

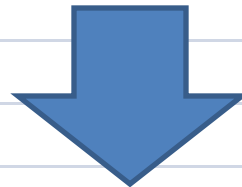


# Boyce-Codd Normal Form (BCNF)

- Boyce-Codd Normal Form or BCNF is an extension to the [third normal form](#), and is also known as 3.5 Normal Form.
- A table is said to be in the Boyce-Codd Normal Form when,
  - Relation must be in 3rd Normal Form
  - and, for each functional dependency (  $X \rightarrow Y$  ),  
X should be a super Key. (In simple words, it means, that for a dependency  $X \rightarrow Y$ , then X cannot be a **non-prime attribute**, if Y is a **prime attribute**. )

# BCNF

ROLLNO	SUBJECT	TEACHER
1	DST	XYZ
1	DBM	ABC
1	CPP	PQR
2	DST	LMN
2	CN	TUV
3	CPP	PQR



ROLLNO	SUBJECT	ROLLNO	TEACHER
1	DST	1	XYZ
1	DBM	1	ABC
1	CPP	1	PQR
2	DST	2	LMN
2	CN	2	TUV
3	CPP	3	PQR

# 4NF

- A table is said to be in the Fourth Normal Form when,
  - It is in the Boyce-Codd Normal Form.
  - And, it doesn't have Multi-Valued Dependency.
- **Multi-Valued Dependency:** For a dependency  $X \twoheadrightarrow Y$ , if for a single value of  $X$ , multiple value of  $Y$  exists, then the table may have multi-valued dependency.

# 4NF

		<b>ROLLNO</b>	<b>SUBJECT</b>	<b>HOBBY</b>	
		1	DST	CRICKET	
		1	DST	HOCKEY	
		1	DBM	CRICKET	
		1	DBM	HOCKEY	
		2	CN	FOOTBALL	
		2	CN	VOLLEYBALL	
	<b>ROLLNO</b>	<b>SUBJECT</b>		<b>ROLLNO</b>	<b>HOBBY</b>
	1	DST		1	CRICKET
	1	DBM		1	HOCKEY
	2	CN		2	FOOTBALL
				2	VOLLEYBALL

# De-normalization

- De-normalization is a strategy that database managers use to increase the performance of a database infrastructure.
- It involves adding redundant data to a normalized database to reduce certain types of problems with database queries
- De-normalization is the process of attempting to optimize the performance of a database by adding redundant data or by grouping data

# De-normalization

- **Methods of De-normalization**

- *Adding Redundant columns*
- *Adding derived columns*
- *Collapsing the tables*
- *Snapshots*
- *VARRAYS*

# File Organization

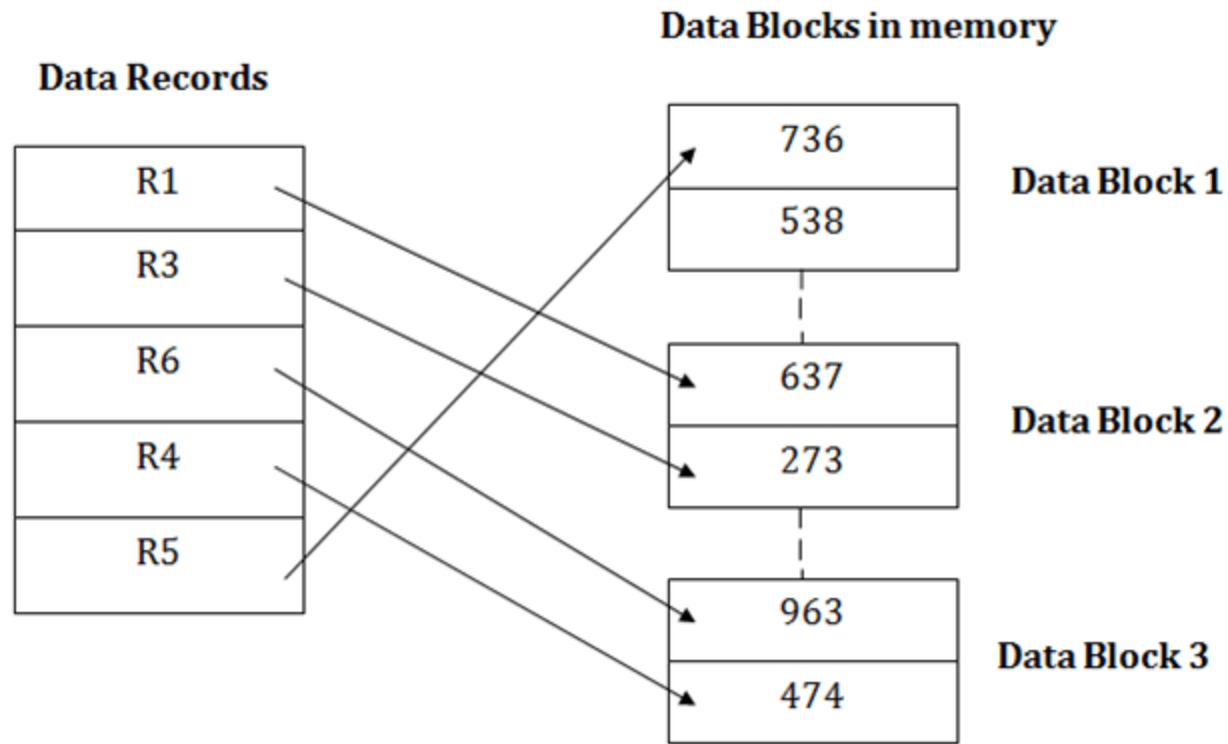
- The process of storing data items in a files in certain order on physical memory so retrieval of data becomes simple, easier, and faster is called as *File Organization*.
- *The main objectives of file organization are:*
  - Optimal selection of records
  - Any DML Operation should be easy and quick.
  - No duplicate records should be induced.
  - Records should be stored efficiently so that cost of storage is minimal.

# **File Organization**

- **Types of File Organization**
  - Heap File Organization
  - Sequential File Organization
  - Hash/Direct File Organization
  - Cluster File Organization



# Heap File Organization



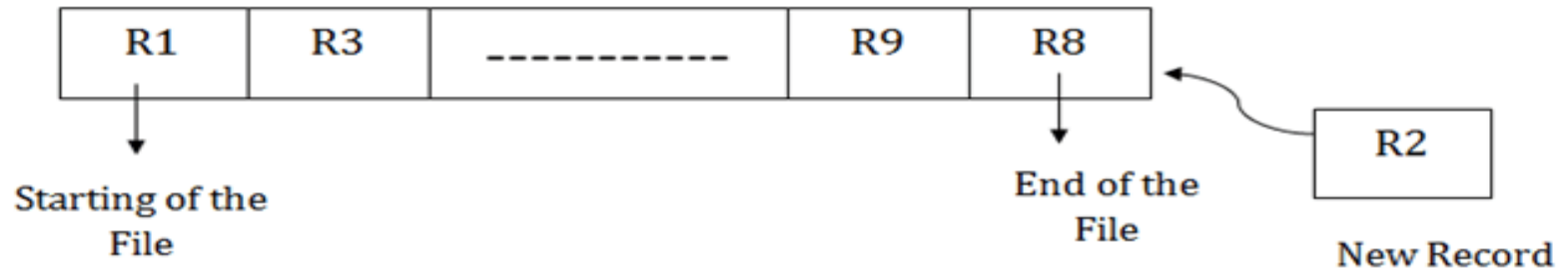
- Here records are inserted at the end of the file in the data block called as heap
- Once the data block is full, the next record is stored in the new block.

# Sequential File Organization

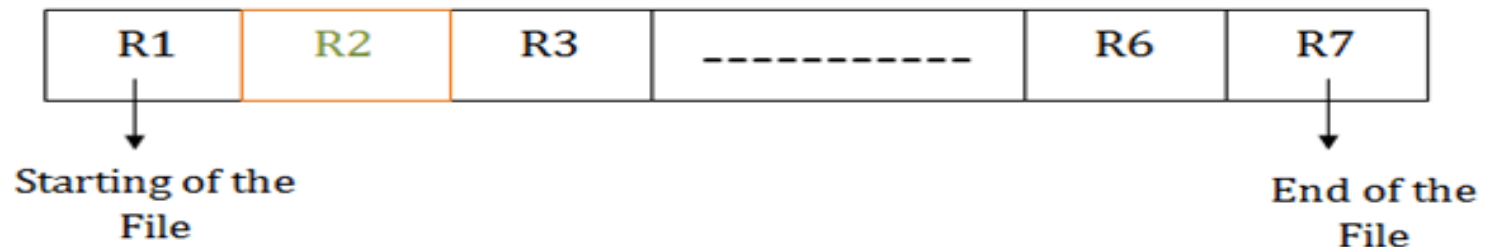
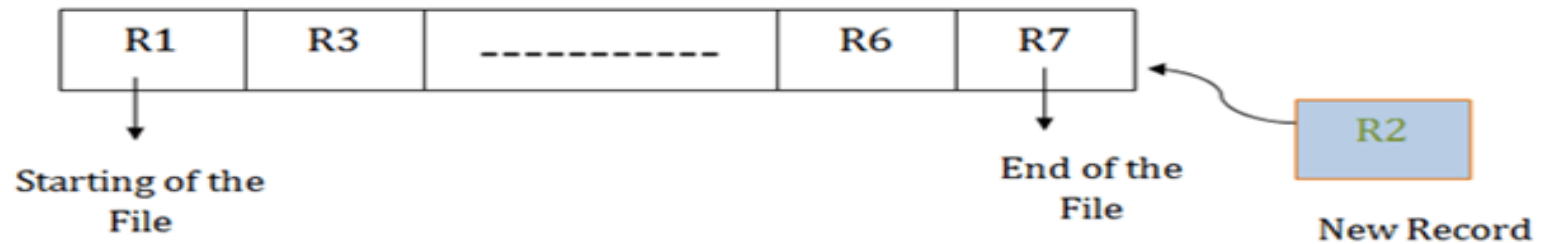
- Records are stored one after the other as they are inserted into the tables. This method is called **Sequential file method**.
- Types:
  - **Pile File Method**
  - **Sorted File Method**

# Sequential File Organization

- Pile File Method

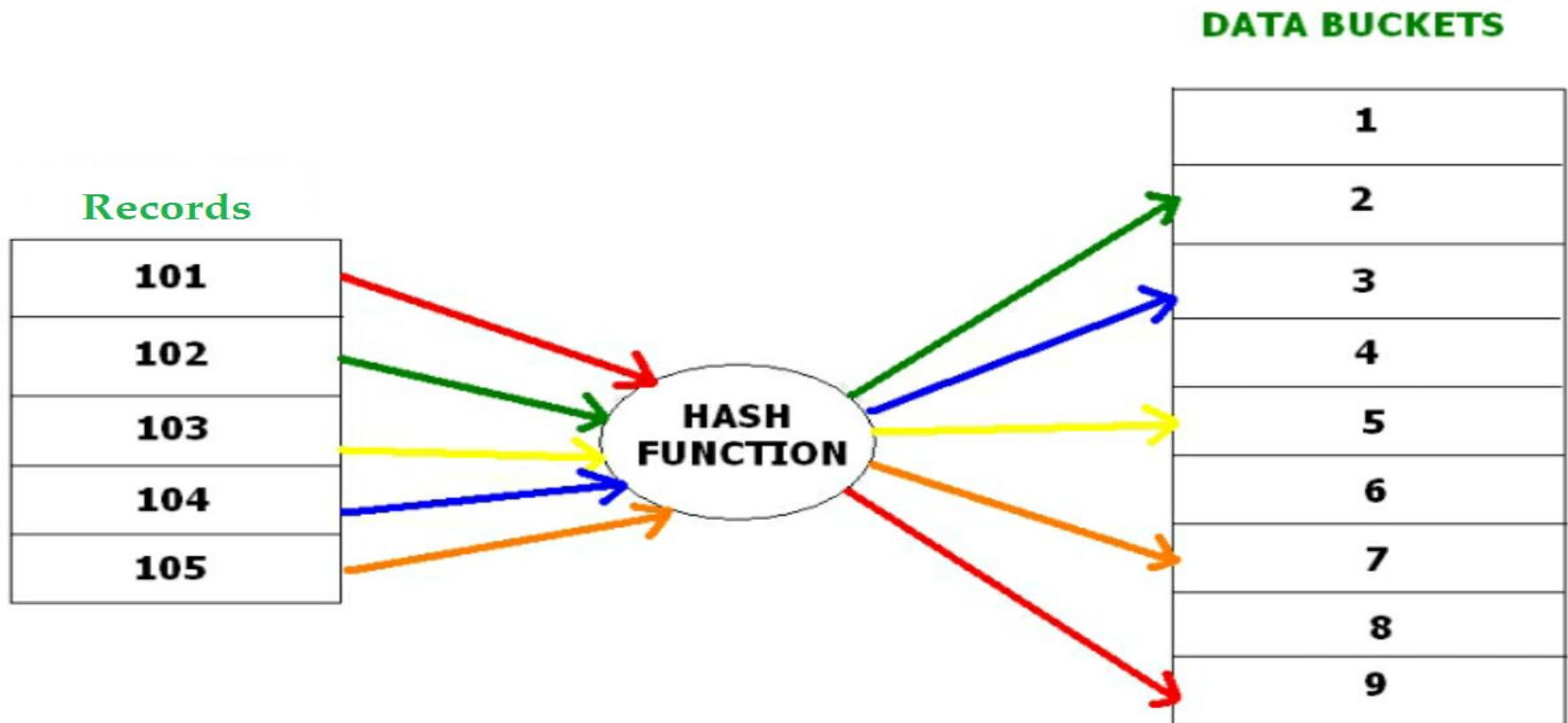


- Sorted File Method



# Hash/Direct File Organization

- hash function is used to calculate the address of the block to store the records.
- The hash function can be any simple or complex mathematical function.



# Clustered File Organization


**EMPLOYEE**

EMP_ID	EMP_NAME	ADDRESS	DEP_ID
1	John	Delhi	14
2	Robert	Gujarat	12
3	David	Mumbai	15
4	Amelia	Meerut	11
5	Kristen	Noida	14
6	Jackson	Delhi	13
7	Amy	Bihar	10
8	Sonoo	UP	12

**DEPARTMENT**

DEP_ID	DEP_NAME
10	Math
11	English
12	Java
13	Physics
14	Civil
15	Chemistry

Cluster Key



DEP_ID	DEP_NAME	EMP_ID	EMP_NAME	ADDRESS
10	Math	7	Amy	Bihar
11	English	4	Amelia	Meerut
12	Java	2	Robert	Gujarat
12		8	Sonoo	UP
13	Physics	6	Jackson	Delhi
14	Civil	1	John	Delhi
14		5	Kristen	Noida
15	Chemistry	3	David	Mumbai

# Object Oriented Database (OODB)

- Encapsulation
- Abstraction
- Inheritance
- Polymorphism
- Classes
- objects

# Indexing & Hashing

# Indexing

- “Indexing is a data structure technique to efficiently retrieve records from the database files based on some attributes on which the indexing has been done”.
- Indexing in database systems is similar to what we see in books.
- Types of Indices:
  - Ordered Index,
  - Primary Index,
  - Secondary Index,
  - Clustered Index etc.



# Types of indices

- Ordered indices:
  - If the indexes are sorted, then it is called as **ordered indices**.
- Primary indices
  - If the index is created on the primary key of the table then it is called as Primary Indexing. Types of primary index: **Dense index** and **sparse index**
- Secondary indices
  - Created on secondary storage to increase efficiency of primary indices.
- Clustered indices
  - records with similar characteristics are grouped together and indexes are created for these groups.

# Hashing

- Hash File organization method is the one where data is stored at the data blocks whose address is generated by using hash function. The hash function can use any of the column value to generate the address.
- hash function uses primary key to generate the hash index – address of the data block.
- Hash function can be simple mathematical function to any complex mathematical function.

# Types of Hashing

- Static Hashing
  - resultant data bucket address will be always same
  - Types
    - Closed
    - Open
    - Quadratic
    - Double
- Dynamic Hashing
  - This hashing method is used to overcome the problems of static hashing – bucket overflow
  - data buckets grows or shrinks as the records increases or decreases

**Thank you**