Database Normalization & Indexing: Student Handout

Overview

This handout provides a concise summary of the key concepts of **Database Normalization** and **Indexing**. By understanding these concepts, you will be able to structure and optimize databases effectively.

---

1. Database Normalization

Definition: Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity.

Normal Forms

First Normal Form (1NF)

- **Rule**: Each column should contain atomic values, and each record should be unique.

Examples:

1. A table with separate columns for "First Name" and "Last Name" instead of a single "Name" column.
2. A table where each row is unique, with no duplicate entries.
3. A table with a single value in each cell, such as a single phone number per contact.

Second Normal Form (2NF)

- **Rule**: Remove partial dependencies; every non-key column should depend on the entire primary key.

Examples:

1. Splitting a table with a composite key into two tables to ensure non-key columns depend on the full key.

2. Separating customer details from order details when the primary key is a combination of "Order ID" and "Product ID."
3. Creating a separate table for product details when they only depend on "Product ID."

Third Normal Form (3NF)

- **Rule**: Remove transitive dependencies; non-key columns should not depend on other non-key columns.

Examples:

1. Creating a separate table for customer addresses that depend on "Customer ID" rather than "Customer Name."
2. Ensuring that "Employee Department" depends directly on "Employee ID" and not on "Employee Name."
3. Separating supplier contact information into its own table, linked by "Supplier ID."

Advantages of Normalization

1. **Reduces Data Redundancy**: Avoids repeating information.
2. **Improves Data Integrity**: Maintains accuracy and consistency.
3. **Easier to Update**: Changes are made in one place.

---

2. Denormalization

Definition: Denormalization is the process of combining tables to reduce the number of joins needed in queries, improving performance at the cost of increased redundancy.

Examples:

1. Combining customer and order tables to speed up query performance.
2. Storing calculated fields to avoid complex calculations during queries.
3. Merging product and supplier tables to reduce join operations.

---

3. Indexing in SQL Databases

Definition: An index is a database structure that improves the speed of data retrieval operations.

Types of Indexes

Clustered Index

- **Definition**: Determines the physical order of data in a table. Only one per table.

Examples:

1. A clustered index on "Order ID" to sort orders by their entry sequence.
2. Using a clustered index on "Employee ID" for quick access to employee records.
3. Implementing a clustered index on "Invoice Number" for efficient invoice retrieval.

Non-Clustered Index

- **Definition**: A separate lookup table that points to the actual data. Multiple per table.

Examples:

1. A non-clustered index on "Product Name" for faster product searches.
2. Creating a non-clustered index on "Customer Email" for quick email lookups.
3. Using a non-clustered index on "Transaction Date" to speed up date-based queries.

Unique Index

- **Definition**: Ensures that the values in a column are unique.

Examples:

1. A unique index on "Customer Email" to prevent duplicate emails.
2. Implementing a unique index on "Username" to ensure each user has a distinct username.
3. Creating a unique index on "Social Security Number" to maintain unique identifiers.

---

4. Performance Optimization Techniques

1. **Use Indexes**: Speed up queries, especially on large tables.

2. **Avoid Over-Normalization**: Balance normalization with performance needs.
3. **Query Optimization**: Write efficient SQL queries, avoiding unnecessary data retrieval.

Activity: Normalize and Index a Sample Database

Sample Table:

Order ID	Customer Name	Product Name	Customer Address
1	Raj Sharma	Rice	Delhi
2	Priya Singh	Wheat	Mumbai

Step 1: Normalize the Table

- Split into "Customers" and "Products" tables.

Customers Table

Customer ID	Customer Name	Customer Address
1	Raj Sharma	Delhi
2	Priya Singh	Mumbai

Products Table

Product ID	Product Name
101	Rice
102	Wheat

Step 2: Create Indexes

- Create a clustered index on "Customer ID" in the Customers table.
- Create a non-clustered index on "Product Name" in the Products table.

Conclusion

This handout covered the basics of **Database Normalization** and **Indexing**, highlighting their importance in reducing redundancy, improving data integrity, and optimizing performance. Use these principles to design efficient and effective databases.