# DATABASE DESIGN OF AN ONLINE E-COMMERCE STORE

**Members** – Kemal Batu Turgut, Krish Sanjaybhai Patel, Mehakpreet Kaur

**Section** – DBS211 ZII

# NORMALIZATION PROCESS

1. **Identifying the Entities**: First, we identified all the entities in our database. For example, the database that we decided to design for an online e-commerce website could include entities like (customers, orders, suppliers, categories, products, and so on).

2. **Defining Attributes:** Next, we determined the attributes (properties or characteristics) of each entity. For instance, for the "customers" entity, attributes might include 'customer id, customer name, customer email, phone number, and address

3. **Eliminating Repeating Groups:** We made sure that each attribute contains only atomic values and doesn't contain repeating groups. For example, if a customer can have multiple phone numbers, we didn't include them directly in the customer table. Instead, we would create a separate table for phone numbers and link it to the customer table using a foreign key.

4**. Identifying Primary Keys:** We identified a primary key for each entity, which uniquely identifies each record in the table. For example, in the customer table, the customer ID might serve as the primary key.

5. **Applying First Normal Form (1NF):** We ensured that each table satisfies the first normal form by making sure that each column contains atomic values and that there are no repeating groups.

6. **Applying Second Normal Form (2NF):** We checked if the table satisfies the second normal form by ensuring that it is in 1NF and that all non-key attributes are fully dependent on the primary key. If any non-key attribute is dependent on only a portion of the primary key, we move it to a separate table.

7. **Applying Third Normal Form (3NF):** We confirmed that the table satisfies the third normal form by ensuring that it is in 2NF and all transitive dependencies are removed. This means that no non-key attribute should depend on another non-key attribute.

By following these steps, we ensured that our database schema was well-structured, normalized, and free from anomalies, leading to better data integrity and efficiency in querying and updating data.

# EXPLANATION OF THE SQL COMMANDS AND THE TRANSACTION IMPLEMENTATION

1**. Creation of Tables:** The `CREATE TABLE` statements define the structure of your database by creating tables for different entities like employees, suppliers, customers, orders, products, and their attributes.

2. **Insertion of Sample Data:** The `INSERT INTO` statements populate the tables with sample data, simulating a realistic dataset for testing and development purposes.

3. **SQL Queries:**

**a. Selection and Projection Queries:** These queries retrieve specific columns from one or more tables using `SELECT` statements, often joined with other tables to gather related information. For example, querying orders along with customer details and employee responsible for the order.

**b. Aggregate Queries**: These queries utilize aggregate functions like `SUM` and `AVG` along with `GROUP BY` clauses to summarize data, such as calculating total sales per customer or average product prices per category.

**c. Subqueries and Nested Queries**: These queries use subqueries to answer complex questions about the data. For instance, selecting orders where the customer's email domain is 'example.com'.

4. **Transaction:**

Insertion of a New Customer: The `INSERT INTO` statement adds a new customer record to the `Customers` table.

Update of an Existing Product Price: The `UPDATE` statement modifies the price of a product with `ProductID = 1` to $24.99.

Deletion of an Employee: The `DELETE FROM` statement removes an employee record from the `Employees` table based on their `EmployeeID`.

These transactional operations demonstrate typical data manipulation actions that might occur in a real-world scenario, such as adding new records, updating existing data, or removing outdated information.

## **SIGNIFICANCE**

By executing these SQL queries and transactions, we could effectively demonstrate the functionality and versatility of your database design, showcasing how it can handle data retrieval, analysis, and modification tasks. Additionally, these operations are crucial for maintaining data accuracy, integrity, and consistency within our database system.