# INVENTORY SYSTEM

# TABLE OF CONTENTS

# INTRODUCTION

## 1.1 PURPOSE:

An inventory system is a process that tracks stock, supplier and sales through an entire supply chain. Companies use inventory systems to ensure they know exactly what items they have available and the location in which they reside.

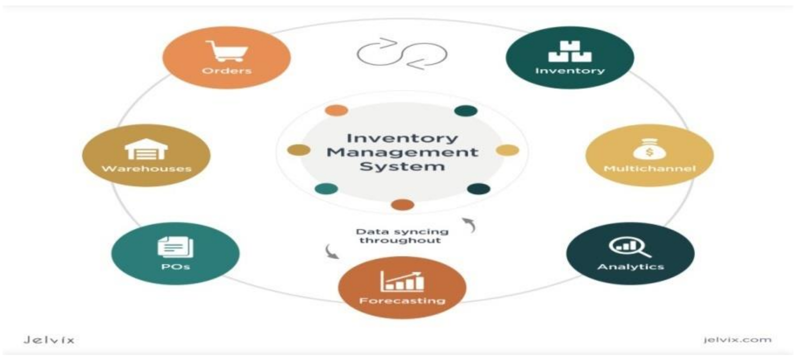
## 1.2. SCOPE:

Inventory systems provide detailed records of new and returned products as they're entering or leaving the warehouse to help companies organize and account for their stock. These systems can also track data such as the number of units, cost per unit, serial number, lot numbers, purchase dates and production dates.

- **Inventory control :** keeping track of stock levels, managing re-order Points and ensuring optimal stock level.
- **Supply Chain management :** coordinating with suppliers and managing lead times to ensure timely replenishment of stack.
- **Demand forecasting:** Analysing Sales data to Predict future inventory needs and adjusting Purchasing accordingly.
- **Cost management :** monitoring the cost of goods sold (COGS) and overall inventors carrying costs to maximize Profitability.
- **Automation:** utilizing technology to automate tasks such as re-ordering, stock counting, and reporting to improve efficiency.

## 1.3 NEEDS OF THE SYSTEM

A good inventory system needs to accurately track stock levels, facilitate efficient ordering and replenishment, and provide real-time visibility to ensure optimal inventory management.



- Stream lines operations
- Reduces costs
- Improves customer satisfaction
- Allows for better financial management

## 1.3.1 EXISTING SYSTEM AND ITS DRAWBACKS:

Existing inventory systems, whether manual or computerized, often struggle with inaccuracies, inefficiencies, and high costs, leading to problems like stock-outs, overstocking, and wasted resources.

**DRAWBACKS**

**Inaccurate Data and Analysis Manual Systems:**

- Relying on spreadsheets or paper records can lead to human errors, incomplete data, and difficulty in tracking inventory across different locations or departments.

**Computerized Systems:**

- Even with software, data inaccuracies can arise from poor integration with other systems, outdated data, or errors in data entry.

**Inefficiencies and High Costs :**

- Inaccurate demand forecasts, which leads to wasteful spending ,leaving your business bloated with costs.

**Security Risks :**

- Overstocking , leading to storage costs and obsolescence, or understanding ,causing stock-outs and missed sales.

**Lack of Real-Time Visibility :**

- Limited Information: Manual systems often provide limited real-time visibility into inventory levels and locations. .

## 1.3.2 PROPOSED SYSTEM AND ITS ADVANTAGES

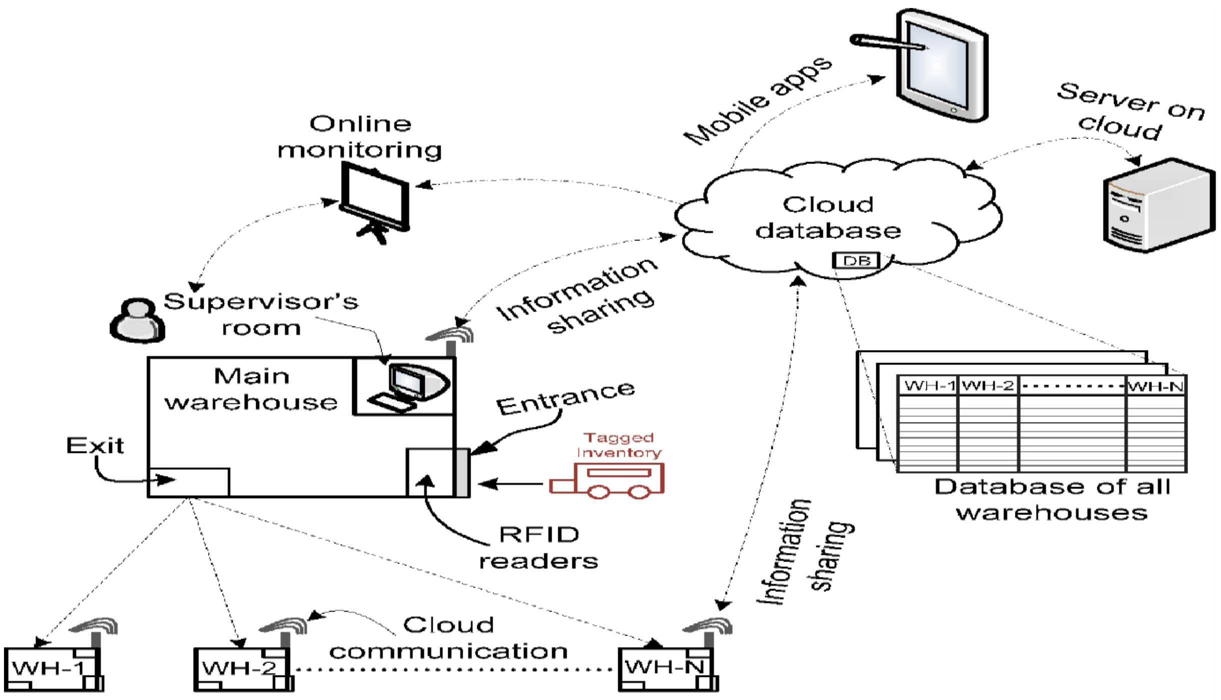
A well-designed inventory management system, whether perpetual or periodic, offers numerous advantages, including improved forecasting, reduced operational costs, and better visibility and transparency, ultimately leading to streamlined processes and increased efficiency.

Perpetual Inventory System: This system tracks inventory in real-time, updating stock levels with each sale or purchase.

**Advantages**:

- **Real-time visibility:** Businesses have constant access to accurate inventory levels.
- **Improved forecasting**: Data from real-time tracking helps in better demand prediction.
- **Reduced stock-outs:** Knowing exact stock levels minimizes the risk of running out of products.
- **Better inventory control:** Real-time data allows for more efficient management of inventory.

## 1.4 Architecture



# SOFTWARE REQUIREMENTS SPECIFICATIONS & ANALYSIS

## 2.1 PRODUCT PERSPECTIVE :

An inventory management system is how businesses track and control stock before it is sold. Whether automated or manual, inventory systems seek to bring your inventory carrying costs down while ensuring sufficient stock is available to meet customer demand.

An inventory management system (or inventory system) is the process by which you track your goods throughout your entire supply chain, from purchasing to production to end sales. It governs how you approach inventory management for your business

## 2.2PRODUCT FUNCTIONS :

- **Inventory Tracking:** IMS software helps monitor the movement and location of inventory, ensuring real-time visibility of stock levels.
- **Inventory Control**: This involves managing and regulating the flow of goods, tracking stock levels, and minimizing stock loss.
- **Order Management**: IMS facilitates the processing of orders, including picking, packing, and shipping, ensuring efficient fulfilment.
- **Forecasting and Planning:** IMS helps predict future demand and plan inventory levels accordingly, minimizing the risk of stock outs or overstocking.
- **Storage and Warehousing**: IMS manages the storage and handling of inventory, including warehouse layout, space allocation, and inventory movement within the warehouse.

## 2.3 USER CHARACTERISTICS:

- **Software Literacy:** Users should be comfortable using computer software and databases, as inventory management systems often involve data entry, analysis, and reporting.
- **Barcode Scanning/Tagging:** with barcode scanning and tagging systems can significantly improve efficiency in tracking and managing inventory.

- **Data Entry Accuracy:** Users need to be able to accurately input and update inventory data to maintain the integrity of the system.

## 2.4 MODULES

**Inventory Management:**

- **Item Management:** Allows users to add, edit, and manage product information (SKU, description, price, etc.).
- **Stock Tracking:** Keeps track of current inventory levels, including real time updates on stock changes.
- **Inventory Valuation:** Calculates the value of the inventory based on different costing methods (e.g., FIFO, LIFO, weighted average).
- **Inventory Adjustments:** Facilitates adjustments to inventory levels due to write-offs, returns, or other reasons.
- **Batch/Serial Number Tracking:** Allows tracking of specific batches or serial numbers for better control and traceability.
- **Barcode/QR Code Scanning:** Enables fast and accurate inventory updates using barcode or QR code technology.

**Sales Order Management:**

- **Order Entry:** Allows users to create and manage sales orders.
- **Order Fulfillment:** Tracks the status of orders from placement to shipment.
- **Invoicing:** Generates invoices for sales orders.

**Reporting and Analytics:**

- **Inventory Reports:** Provides reports on inventory levels, stock turnover, and other key metrics.
- **Sales Reports:** Provides reports on sales performance, customer orders, and other sales-related data.
- **Purchase Reports:** Provides reports on purchasing activity, supplier performance, and other purchasing-related data.
- **Reduced stock-outs:** Knowing exact stock levels minimizes the risk of running out of products.

- **Better inventory control:** Real-time data allows for more efficient management of inventory.

## 2.5 Functional and Non-Functional Requirements

### 2.5.1 Functional Requirements

Functional requirements define the functions that are requested by our stakeholders Different Functions are needed by different. Of the system. Different administrators have authority to maintain the system.

- **Part Modeling:**

Creating and modifying 3D part geometry using various modeling techniques (e.g., extrusions, revolves, sweeps).

- **Assembly Modeling:**

Assembling multiple parts into a larger product, defining constraints and relationships between components.

- **Drawing Generation:**

Creating 2D engineering drawings from 3D models, including views, dimensions, and annotations.

- **Parametric Design:**

Defining relationships between features and dimensions so that changes in one part or assembly propagate to others.

- **Data Management:**

Organizing, storing, and retrieving design data, including version control and revision management.

- **Simulation and Analysis:**

Performing stress analysis, dynamic simulations, and other analyses to evaluate product performance.

- **Customization and Automation:**

Providing tools for users to customize the interface and automate repetitive tasks through Logic.

### 2.5.2 Non functional Requirements

Non-functional Requirements of a system specify the performance, reliability, availability, security, maintainability, portability of a system. So below there are a short description each of those non-functional requirements.

**Static Requirements**: The requirements do not impose any constriants on the execution characteristics of the system .they are:

1) **Number of Terminals:** while the front end will be available to the administrative computer
2) **Number of Users:** The number users can be administrator only.

**Dynamic Requirements**: They typically include response time and throughout the system. Since these factors are not applicable to proposed software.

- **Reliabilit**y: The software will not be able to connect to the database.
- **Availabilit**y: only to administrator of the organization .
- **Security:** The security requirements deal with the primary security.
- **Maintainability:** Backups for database are available.
- **Portability:** The software is a web-based application and built in JAVA and SQL.

## 2.6 SYSTEM SPECIFICATIONS:

### 2.6.1 HARDWARE REQUIREMENTS:

Languages and tools, checking the specifications are addressed during this activity

| | |
|---|---|
| ➢ Processor | : i3 /i5 /i7 Intel processor |
| ➢ RAM | : 4GB(min) |
| ➢ Hard disk | : 256 GB |

External Interface Requirement

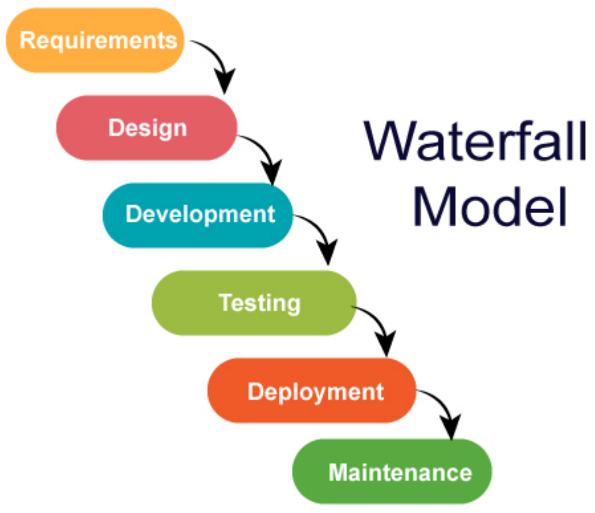
### 2.6.2 SOFTWARE REQUIREMENTS:

| | |
|---|---|
| ➢ Operating System | : Window7/8/9/10/11 |
| ➢ Application server | :Tomcat 5.0/6 |
| ➢ Front End | : HTML, JSP |
| ➢ Scripts | : java |
| ➢ Server side Script | : java Server pages |

## 2.7 Software Development Life Cycle

Software Development Life Cycle Models and Methodologies The Software Development Life Cycle (SDLC) is a series of stages that provide a structured approach to the software development process. It encompasses understanding the business requirements, eliciting needs, converting concepts into functionalities and features, and ultimately delivering a product that meets business needs. A proficient software developer should possess adequate knowledge to select the appropriate SDLC model based on project context and business requirements. Therefore, it is essential to select the right SDLC model tailored to the specific concerns and requirements of the project to ensure its success. To explore more about choosing the right SDLC model, you can follow this link for additional information. Furthermore, to delve deeper into software lifecycle testing and SDLC stages, follow the highlighted links here. The exploration will cover various types of SDLC models, their benefits, disadvantages, and when to use them. SDLC models can be viewed as tools to enhance product delivery.

**Waterfall model**



The Waterfall Model follows a linear, sequential flow, where progress moves steadily downwards (like a waterfall) through the phases of software development. Each stage in the development cycle begins only after the previous stage is completed. The waterfall approach does not accommodate going back to a previous stage to address changes in requirements. It is the oldest and most well-known method used for software development. The five-stage waterfall model, based on Winston W. Royce's requirements,

Divides development processes into the following stages

1. Analysis
2. Design
3. Implementation
4. Testing and
5. Operation

**Advantages**

1. Simple to clarify for the clients.
2. Structures approach.
3. Stages and exercises are distinct.
4. Assists with arranging and timetable the task.

## 2.8 system study

Feasibility Analysis Feasibility study is an important phase in the software development process. It enables the developer to have an assessment of the product being developed. It refers to the feasibility study of the product in terms of outcomes of the product, operational use and technical support required for implementing it. Feasibility study should be performed on the basis of various criteria and parameters. The various feasibility studies are:

- Technical Feasibility
- Economic Feasibility
- Organizational Feasibility
- Operational Feasibility

**Technical Feasibility:**

The system is self-explanatory and does not need any extra sophisticated training. As the system has been built by concentrating on the Graphical User Interface Concepts, the application can also be handled very easily with a novice User. The overall time that is required to train the users upon the system is less than half an hour. The System has been added with features of menu-driven and button interaction methods, which makes the user the master as he starts working through the environment. The net time the customer should concentrate is on the installation time.

**Financial Feasibility:**

i) **Time Based:** Contrast to the manual system management can generate any report just by single click. In manual system it is too difficult to maintain historical data which become easier in this system. Time consumed to add new records or to view the reports is very less compared to manual system. So this project is feasible in this point of view.

ii) **Cost Based:** No special investment need to manage the tool. No specific training is required for employees to use the tool. Investment requires only once at the time of installation. The software used in this project is freeware so the cost of developing the tool is minimal and hence the overall cost.

iii) **Implementation Plan:** The main plan for the system developed is to mimic the existing system asit is in the proposed system.

**Operational Feasibility**

Operation feasibility deals with risk factors. It checks the impact of the propose software on required project or module. This system is being developed for analyzing the packets and black listed IP addresses. As this system is generalize one so anybody can use that. Here we are implementing this system module as one of industry application, which is innovative and new advance concept, which helps the user to use it more effectively and easily and instead always go for system work.

**Organizational Feasibility:**

From an organizational perspective, the system can be considered as low risk project. The management of Transmission Specialist INC. has a big interest in the researcher's project.

The Users and the management of the Inventory Management System of Transmission Specialist INC. are expected to appreciate and find the new system offered easy to use.

The Costs and Benefits is based from Philippine Currency (PHP). Current conversion of USD/PHP is 53.00PHP-1 USD.

## 2.9 Methodology And Algorithms

- **Requirements Gathering:** Identify functional and non-functional requirements.
- **System Design:** Design the system architecture, database schema, and user interface.
- **Implementation:** Develop the system using a programming language (e.g., Java, Python).
- **Testing**: Perform unit testing, integration testing, and system testing.

**Algorithms:**

**Inventory Management:**

- Stock level updates based on orders, deliveries, and stock movements.
- Low-stock alerts and notifications.

**Order Management:**

- Order processing and fulfillment logic.
- Order tracking and status updates.

**Reporting:**

- Generating reports on inventory levels, order history, and sales trends.

**Data Structures:**

**1. Arrays or Lists:** Store inventory items, orders, and customers.

**2. Hash Tables or Dictionaries:** Store and retrieve data efficiently (e.g., inventory item details).

## 2.10 Techniques Used

**Programming Languages:**

**1. Java**: Known for its platform independence and robust security features.

**2. Python**: Easy to learn and versatile, with extensive libraries for data analysis and visualization.

**3. C++:** A popular choice for Windows-based applications, with strong integration with Microsoft technologies.

**Database Management Systems:**

**1. MySQL:** A popular open-source relational database management system.

**2. PostgreSQL:** A powerful, open-source relational database with advanced features.

**Front-end Development:**

**1. HTML/CSS**: For building user interfaces and web applications.

**2. JavaScript:** For adding interactivity and dynamic effects to web pages.

**3. React or Angular:** Popular front-end frameworks for building complex web applications.

**Back-end Development:**

**1. Spring Boot (Java):** A popular framework for building robust and scalable back-end services.

**2. Django (Python):** A high-level framework for building web applications quickly and efficiently.

.

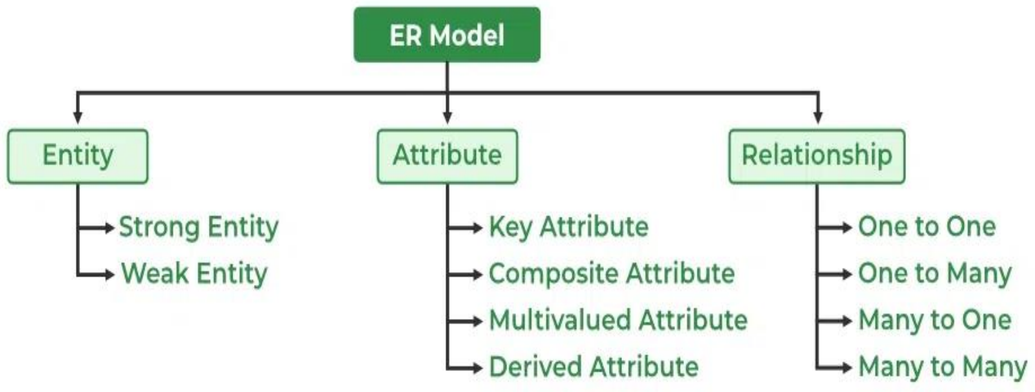
# SYSTEM DESIGN

# 3 SYSTEM DESIGN

## 3.1 E-R MODEL

The Entity-Relationship Model (ER Model) is a conceptual model for designing a databases. This model represents the logical structure of a database, including entities, their attributes and relationships between them.

- **Entity:** An objects that is stored as data such as  strong entity or weak entity.
- **Attribute:** Properties that describes an entity such as key, composite, multivalued, derived attribute.
- **Relationship:** A connection between entities such as "one to one ,one to many ,many to one, many to many ".

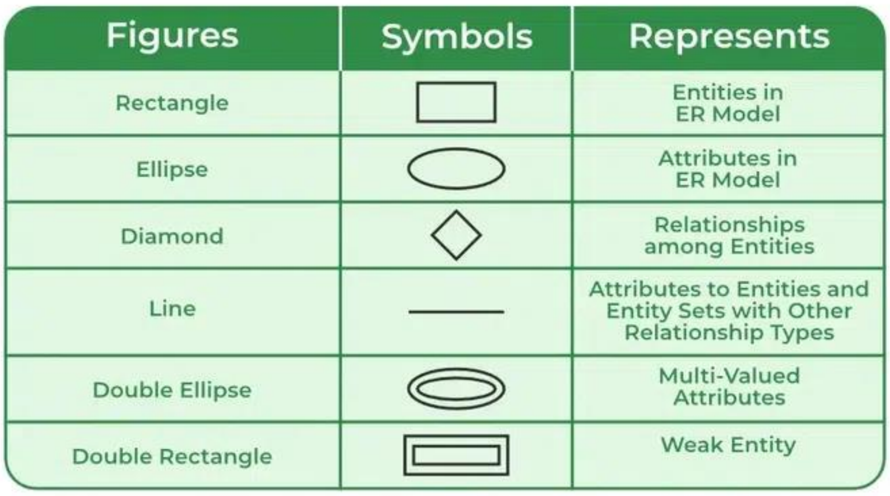


### Symbols Used in ER Model

ER Model is used to model the logical view of the system from a data perspective which consists of these symbols:

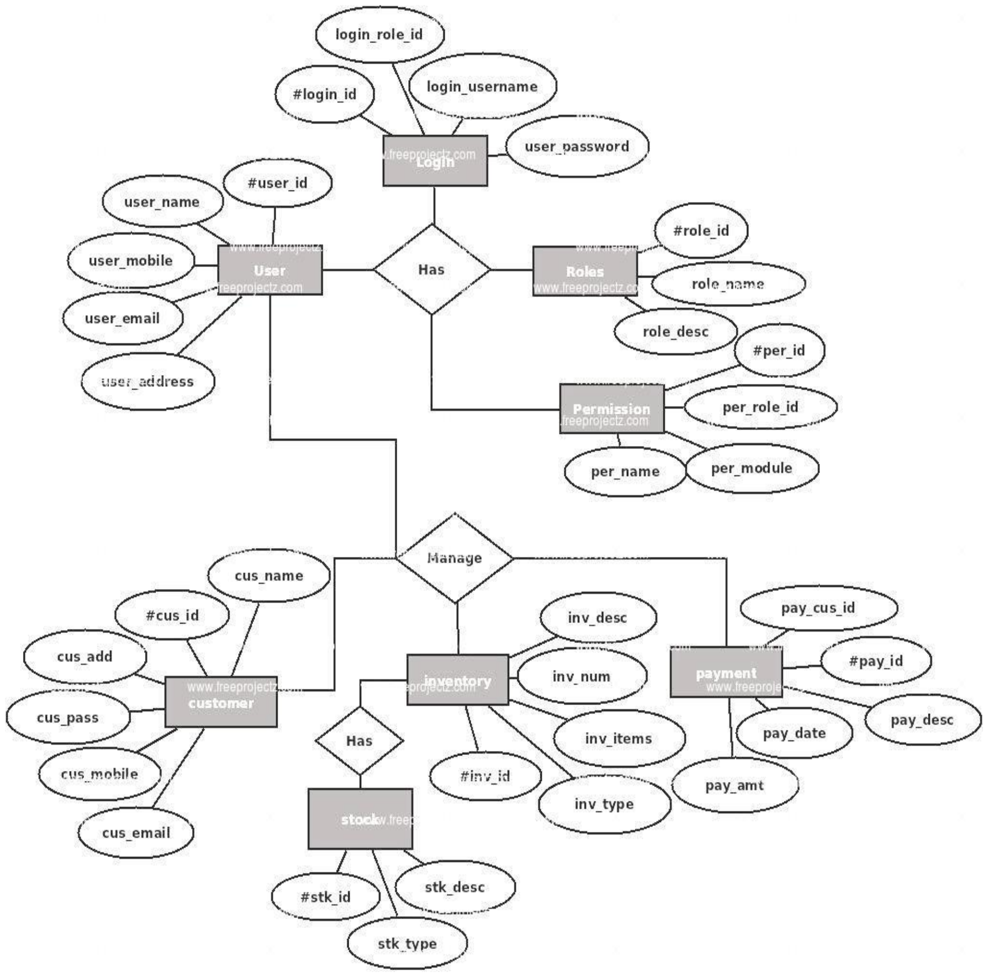
- **Rectangles:** Rectangles represent entities in the ER Model.
- **Ellipses:** Ellipses represent attributes in the ER Model.
- **Diamond:** Diamonds represent relationships among Entities.
- **Lines:** Lines represent attributes to entities and entity sets with other relationship types.
- **Double Ellipse:** Double ellipses represent multi-valued Attributes, such as a student's multiple phone numbers

- **Double Rectangle:** Represents weak entities, which depend on other entities for identification.



| Figures | Symbols | Represents |
|---|---|---|
| Rectangle | ▭ | Entities in ER Model |
| Ellipse | ⬭ | Attributes in ER Model |
| Diamond | ◇ | Relationships among Entities |
| Line | — | Attributes to Entities and Entity Sets with Other Relationship Types |
| Double Ellipse | ◎ | Multi-Valued Attributes |
| Double Rectangle | ⧈ | Weak Entity |

## E-R DIAGRAM



## 3.2 Data Dictionary

### Log in

| Name | Description | Type | Size |
|---|---|---|---|
| Login_id | Login id | Int | 20 |
| Login_id role_id | Log id number | Int | 20 |
| User_name | Name of the user | Varchar | 50 |
| User password | Password of the user | Varchar | 50 |

### User

| Name | Description | Type | Size |
|---|---|---|---|
| User id | User id number | Int | 25 |
| user_name | Name of the user | varchar | 25 |
| User_mobile | Number of the user | Int | 10 |
| User _address | Address of the user | varchar | 100 |
| User_email | Email of the address | varchar | 20 |

### Roles

| Name | Description | Type | Size |
|---|---|---|---|
| Role _id | Id number | int | 10 |
| Role_name | Name of the role | varchar | 25 |
| Role _desc | Work an employee is expected to perform | varchar | 30 |

### Customer

| Name | Description | Type | Size |
|---|---|---|---|
| Cust-id | Id number | int | 15 |
| Cust-name | Name | varchar | 25 |
| Cust-mobile | Mobile | Int | 10 |
| Cust- address | Current address | varchar | 100 |
| Cust- email | email | varchar | 30 |

### Stock

| Name | Description | Type | Size |
|---|---|---|---|
| Stock -id | Id number | Int | 15 |
| Stock-name | Name | varchar | 50 |
| Stock-type | Types of stock | varchar | 80 |
| Stock-desc | Desc of stock | varchar | 100 |

**Inventory**

| Name | Description | Type | Size |
|---|---|---|---|
| Inventory name | Name | Varchar | 25 |
| Inventory id | Id | Int | 15 |
| Inventory desc | Description of the items | Varchar | 50 |
| Inventory type | Items of models | Varchar | 30 |

**Payment**

| Name | Description | Type | Size |
|---|---|---|---|
| Payment-cust-id | Cust-id | Int | 15 |
| Payment-id | Id number | int | 20 |
| Payment-type | Types of upi | varchar | 10 |
| Payment-desc | Payment of desc UTR ID | int | 12 |
| Payment - amount | Amount | int | 6 |
| Return payment | Return amount | int | 6 |

## 3.3 Normalization

Normal forms in DBMS are rules that guide database design, aiming to minimize data redundancy and dependency issues, leading to a more efficient and maintainable database. These forms, often referred to as 1NF, 2NF, 3NF, BCNF, 4NF, and 5NF, provide a structured approach to organizing data within tables to ensure data integrity and consistency.

**1. First Normal Form (1NF):**

**Goal**: Ensures that each column contains only atomic values (indivisible values) and eliminates repeating groups.

Example: A table with a column "Phone Numbers" storing multiple phone numbers in a single cell violates 1NF. It needs to be restructured to have separate columns for each phone number, or separate rows for each number.

**Example:**

| S.No | Phone Number |
|---|---|
| 1 | 9876543210 |
| 2 | 6987543210,8569741230 |
| 3 | 6322114488 |
| 4 | 9668855443 |

| S.No | Phone Number |
|---|---|
| 1 | 9876543210 |
| 2 | 6987543210 |
| 2 | 8569741230 |
| 3 | 6322114488 |
| 4 | 9668855443 |

**2. Second Normal Form (2NF):**

**Goal**: Builds upon 1NF by eliminating partial dependencies.

**Explanation:** A table with a composite key (e.g., Student ID, Course ID) is in 2NF if every non-key attribute depends on the entire primary key, not just a part of it.

**Example**: If Student Name depends only on Student-ID and not on the entire key, it violates 2NF. Student Name should be moved to a separate table with Student ID as the primary key.

| S.no | Student Name | Phone Number |
|---|---|---|
| 1 | Ram | 9876543210 |
| 2 | Vishnu | 6987543210 |
| 3 | Venky | 6322114488 |
| 4 | Vara | 9668855443 |

**3. Third Normal Form (3NF):**

**Goal**: Eliminates transitive dependencies while maintaining 1NF and 2NF.

**Explanation**: A table is in 3NF if there are no transitive dependencies, meaning no non-key attribute depends on another non-key attribute.

**Example**: If a table has columns for Student ID, Course ID, and Course Name, and Course Name depends on Course ID, which in turn depends on Student ID, it has a transitive dependency. Course Name should be moved to a separate table with Course ID as the primary key.

**Student Courses**

| **Student id** | **Couse id** | **Course name** |
|---|---|---|
| 01 | 101 | Mathematical |
| 02 | 102 | Science |
| 03 | 103 | English |
| 04 | 104 | Hindi |

**Courses Table**

| Couse id | Course name |
|---|---|
| 101 | Mathematical |
| 102 | Science |
| 103 | English |
| 104 | Hindi |

**Student Courses Table**

| **Student id** | **Course id** |
|---|---|
| 01 | 101 |
| 02 | 102 |
| 03 | 103 |
| 04 | 104 |

**4. Boyce-Codd Normal Form (BCNF):**

**Goal**: An extension of 3NF that removes dependencies where the determinant (the column on which another column depends) is not a candidate key.

**Explanation**: In BCNF, for every functional dependency X -> Y, X must be a superkey (a key that can uniquely identify all rows in the table).

**Example:** Consider a relation R with attributes (student, teacher, subject).

**R table**

| Student | Teacher | Subject |
|---|---|---|
| Raghu | R.Das | Data Base |
| Mohan | K.Raman | C |
| Vara | R.Das | Data Base |
| Venky | N.Gupta | C |

**R1 table**

| Teacher | Subject |
|---|---|
| R.Das | Database |
| K.Raman | C |
| N.Gupta | C |

**R2 Table**

| Student | Teacher |
|---|---|
| Raghu | R.Das |
| Mohan | K.Raman |
| Vara | R.Das |
| Venky | N.Gupta |

**5. Fourth Normal Form (4NF):**

**Goal**: Addresses multi-valued dependencies.

**Explanation**: A table is in 4NF if it's in 3NF and there are no multi-valued dependencies.

**Example**: A table that stores skills for employees might have a multi-valued dependency where an employee can have multiple skills. This can be resolved by creating a separate table for skills and a relationship between the employee and skill ta Employee Skills Table

| Employee id | Name | skills |
|---|---|---|
| 1 | John | Programming |
| 1 | John | Design |
| 2 | Mohan | Marketing |
| 2 | Mohan | sales |

**Employees  Table**

| Employee ID | Name |
|---|---|
| 1 | John |
| 2 | Mohan |

**Skills Table**

| Skill ID | Skill Name |
|---|---|
| 1 | Programming |
| 2 | Design |
| 3 | Marketing |
| 4 | Sales |

**Employee skills table(bridge table)**

| Employee  ID | Skill  ID |
|---|---|
| 1 | 1 |
| 1 | 2 |
| 2 | 3 |
| 2 | 4 |

**6. Fifth Normal Form (5NF):**

**Goal**: Addresses lossless join dependencies.

**Explanation**: A table is in 5NF if it's in 4NF and all join dependencies are implied by the candidate keys.

**Example**: This form is less frequently encountered in practice and is often used to deal with complex relational schemas.

**Sales table**

| Salesperson | Product | Customer |
|---|---|---|
| Mohan | Phone | Alice |
| Vishnu | Laptop | Bob |
| Vishnu | Phone | Alice |
| Venky | laptop | Charlie |

**Salesperson Product Table**

| Salesperson | Product |
|---|---|
| Mohan | Phone |
| Vishnu | Laptop |
| Vishnu | Phone |
| Venky | laptop |

**Salesperson  Customer Table**

| Salesperson | Customer |
|---|---|
| Mohan | Alice |
| Vishnu | Bob |
| Vishnu | Alice |
| Venky | Charlie |

**Product Customer Table**

| Product | Customer |
|---|---|
| Phone | Alice |
| Laptop | Bob |
| Phone | Alice |
| Laptop | Charlie |

# CONCLUSION

This project on "The study on Inventory System" gave me an opportunity to understand the level of inventory management in the organization. This research will help the organization to make necessary measure to the inventories. This will certainly bring down the causes of inventory problems and help the management of inventories. The high turnover ratio indicates efficient management of inventory because more frequently the stock sold. So the organization should try to improve the inventory turnover ratio.

The current study helped me to understand the current inventory control measures practiced in any organization. The cordial and corporate relationship between management and employees is the secret behind the success every organization.