General store Database management

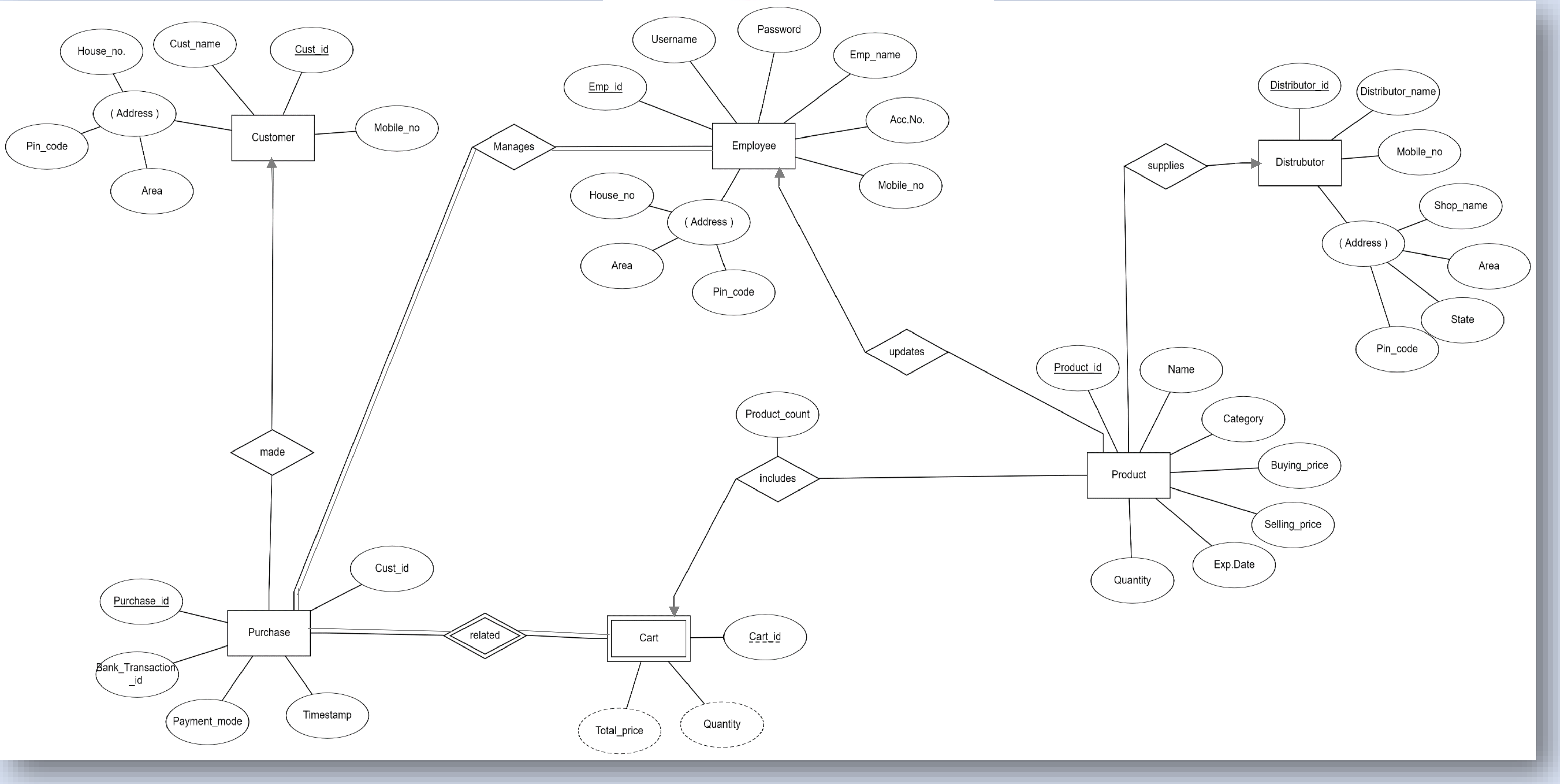
# DBMS Project

Sai Krishna store

***No. of students** : 04*

| | |
|---|---|
| U19CS070 | Shubham kumar jayswal |
| U19CS017 | Chakka Vasudev |
| U19CS069 | Akkaladevi Shreyas |
| U19CS065 | Sai Satya |

Sai Krishna store



# Schema

## Entity Table attributes-

1. **Customer**
    - a. Cust_id // primary key, auto_increment
    - b. Cust_name // not null
    - c. Address
        - i. H_No. // not null
        - ii. Area // not null
        - iii. Pin_code // not null
    - d. Mobile no. // single valued, not null

2. **Employee**
    - a. Emp_id // primary key, auto_increment
    - b. Username // not null, Unique
    - c. Password // not null
    - d. Emp_name // not null
    - e. Mobile_no // not null, single valued,
    - f. Acc_no // null
    - g. Address
        - i. H_No. // not null
        - ii. Area // not null
        - iii. Pin_code // not null

3. **Product**
    - a. Pdt_id // primary key, auto_increment
    - b. Pdt_name // not null
    - c. Category // not null
    - d. Buying_price // not null
    - e. Selling_price // not null
    - f. Exp_date // null
    - g. Stock // not null
    - h. Distributor_id // not null, Foreign key of Distributor(Distributor_id)

4. **Distributor**

    a. Distributor_id // primary key, auto_increment

    b. Distributor_name // not null

    c. Mobile_no // not null

    d. Address

        i. Shop_name // not null

        ii. Area // not null

        iii. State // not null

        iv. Pin code // not null

5. **Purchase**

    a. Purchase_id // primary key, auto_increment

    b. Cust_id // not null, Foreign key of Customer(Cust_id)

    c. Timestamp // default System current timestamp

    d. Payment_Mode // not null

    e. Bank_Transaction_id // null

6. **Cart** //Weak entity

    **Primary key**:-( Purchase_id + Pdt_id)

    a. Purchase_id // not null, Foreign key of Purchase(Purchase_id),

    b. Pdt_id // not null, Foreign key of Product(Pdt_id),

    c. Quantity // not null

    d. Total_Price // not null, derived attribute

## Relationship sets-

1. **Manages** : Employee/Manager manages the Purchase
    - Fully participation
    - One to many

2. **Updates** : Employee/Manager updates the information of products
    - One to many

3. **Supplies** : Distributor supplies Product
    - One to many

4. **Made** : Customer made Purchase
    - One to many

5. **Related** : Purchase related Cart information
    - Fully participation
    - One to many

6. **Includes** : Cart includes product_count number of Product
    - Product_count = Quantity in cart
    - One to many

# Relational Schema

## For Entity -

- **Customer** *( Cust_id , Cust_name , H_no , Area , Pin_code , Mobile_no )*
- **Employee** *( Emp_id , Username , Password , Emp_name , Acc_no , Mobile_no , H_no , Area , Pin_code )*
- **Product** *( Pdt_id , Pdt_name , Category , Buying_price , Selling_price , Exp_date , Quantity )*
- **Distributor** *( Distributor_id , Distributor_name , Mobile_no , Shop_name , Area , State , Pin_code )*
- **Purchase** *( Purchase_id , Cust_id , Bank_transaction_id , Payment_mode , Timestamp )*
- **Cart** *( Purchase_id , Pdt_id , Total_price , Quantity )*

## For relationship sets -

- **Manages** *(Emp_id , Purchase_id)*
- **Updates** *(Emp_id , Pdt_id)*
- **Supplies** *(Distributor_id , Pdt_id)*
- **Made** *(Cust_id , Purchase_id)*
- **Related** *(p.Purchase_id , c.Purchase_id , c.Pdt_id)*
- **Includes** *( c.Purchase_id , c.Pdt_id , product_count , i.Pdt_id)*

# Normalization

- **1NF :-** A relation is in first normal form if and only if the domain of each attribute contains only atomic (indivisible) values, and the value of each attribute contains only a single value from that domain.

  - All tables in our database are in 1NF, because all attributes are indivisible (i.e.;) in our table we don't have any multivalued attributes or complex attributes.

- **2NF :-** A relation is in the second normal form if it fulfills the following two requirements:
  1. It is in first normal form.
  2. It does not have any non-prime attribute that is functionally dependent on any proper subset of any candidate key of the relation.

  - All tables in our database are in 2NF, because we don't have any partial dependency.

- **3NF :-** A table is in 3NF if and only if both of the following conditions hold:
  1. The relation R (table) is in second normal form (2NF).
  2. Every non-prime attribute of R is non-transitively dependent on every key of R.

  - All tables in our database are in 3NF, because we don't have any transitive functional dependency.

- **BCNF :-** BCNF is the advanced version of 3NF. It is stricter than 3NF. A table is in BCNF if every functional dependency X → Y, X is the super key of the table. For BCNF, the table should be in 3NF, and for every FD, LHS is super key.

  - All tables in our database are in BCNF, because all functional dependencies are in the form of X → Y, where X is the super key of the table.

## Features-

1. Customer entry
2. Employee information
3. Product details (entry, deletion)
4. Distributor entry/Removal
5. Transaction information
6. Product Stock detail
7. Sales Statistics
8. Order Products
9. Return order within one day
10. Alert for Out of stock

## Important notes-

- This project is related to a small general store management system.
- In the store, there is only one employee (owner itself).
- All the basic features we have included in this project.
- Payment mode is either cash or UPI
- For UPI payment, we will store the UPI related bank transaction id
- We have added the total amount in the cart of every group of items, because if an employee changes the selling price in future, then it will not reflect in calculation of profit.
- Employee will collect the cash or UPI payment
- Customers can also return the items within a day.
- We have added a one-day limit because product prices may change next day and expiry date issues will also be problematic if we cancel the order next day.
- For Customers, we have not added any delete functionality because we don't want to lose any customer.

- Deletion of any Product means we will make the stock of that product zero.
- If expiry date is not applicable to any product, then by default it will be null or 0000-00-00.
- Deletion of distributors is also not allowed because Products are related to Distributors.
- Employee can change his profile information including password, in the profile section.
- We can check Products related to a Distributor by Pressing the eye button in the Distributor page.
- We can update the Distributor information, Customer information, Product related information on their detail page.
- The Employee can check all the items related to a particular transaction by pressing the eye button in the Transaction detail page.
- Cancel order option is also there on this page.
- Cancel order option will appear only if a transaction has been made on the same date/current date.
- We have used triggers for deleting the related row in cart and updating the product stock, when the employee is deleting any transaction.
- We have also added some basic but important features on all the details page like search by name and search by date.
- On the Product page, we have added sort features. By using this the employee can sort the product by their different attributes' values.
- On the main Page, Employee can see the sales statistics in the form of profit.
- Monthly and daily statistics are available on the main page. The employee can also check the profit on a particular date and a particular month by giving inputs on the main page.
- A list of out-of-stock items is also there on the main page.
- All the Pages will be accessible only if the user is logged in. If a user is not logged in and he tries to access any of the page then he will redirect to the main page.
- In the Payment Page, we have implemented views so that the employee can make transactions for those products only, which are in stock and not expired.