DATA AND APPLICATIONS
PROJECT PHASE 3
RELATIONAL MODEL

Team 54
Maanasa Kovuru - 2021101101
Anusha Nemani - 2021101082
Muskan Raina - 2021101066

ASSUMPTIONS:

1. Each canteen has a Mess Manager who manages it. This mess manager has a Mess_Manager_ID.
2. Staff has 2 key attributes. We chose S_ID to be the primary key of the relation and set up foreign key relationships in other related tables with the S_ID attribute.
3. The cost of an order is not only determined by the cost of the item. It includes additional costs like transportation fee, taxes, etc.

ER DIAGRAM TO RELATIONAL MODEL:

Step 1: Mapping of Regular Entity Types

- For each strong entity in the ER schema we created a relation R that includes all the simple attributes of the strong entity.
- The entity types CANTEEN, CUSTOMER, STAFF, MESS COMMITTEE, FOOD, BEVERAGES, COMPANY, and ORDER were made into relations that contain all the simple attributes of the entity types.
- We separated composite attributes into their simple component attributes. This helps remove the 1-NF violation of composite attributes from the database by taking all the attributes of the composite attribute and adding them in the relation R.

Step 2: Mapping of Weak Entity Types

- For each weak entity type (W) in the ER schema with the owner entity type (E), we created a relation R and included all simple attributes of W as attributes of R.
- We also included the primary key attributes of the owner entity type as foreign key attributes of R. These attributes along with its own partial key forms the primary key of the relation. For example, Canteen_name is the foreign key attribute of INVENTORY entity type, and S_ID is the foreign key attribute of DEPENDENT entity type (primary key attributes of CANTEENS and STAFF entity types respectively).

Step 3: Mapping of Binary 1:1 Relationship Types

- As we do not have any binary 1:1 Relationship types in our ER model, there were no changes made to the relational model during this step.

Step 4: Mapping of Binary 1:N Relationship Types

- For every binary 1:N relationship type R, we identify the relation S that represents the participating entity type at the N-side of the 1:N relationship type. We include the primary key of T (the relation that represents the participating entity type at the 1-side of the 1:N relationship type) as a foreign key in the relation S.
- For example, we included the Mess_Manager_Id (the primary key from the MESS_COMMITTEE relation) as a foreign key in the CANTEEN table as a Mess Committee member can manage N canteens.

Step 5: Mapping of Binary M:N Relationship Types

- For every binary M:N relationship type R, we created a new relation S to represent R. We include the primary keys of relations that represent the participating entity types as the foreign key attributes in S; the combination of all these attributes will be the primary key of the relation S.
- In our ER diagram, the relationship type PROCURED_FROM is a binary M:N relationship. We created a PROCURED_FROM relation that includes all the primary keys (Company_Name, Order_No, Canteen_Name) as foreign keys in it.
- We also created a PRESENT_IN relation with the Canteen_Name and Item_Name as its attributes. However, this created a redundancy as these attributes are already present together in the INVENTORY relation. Thus the newly added PRESENT_IN relation was omitted.

Step 6: Mapping of Multivalued Attributes

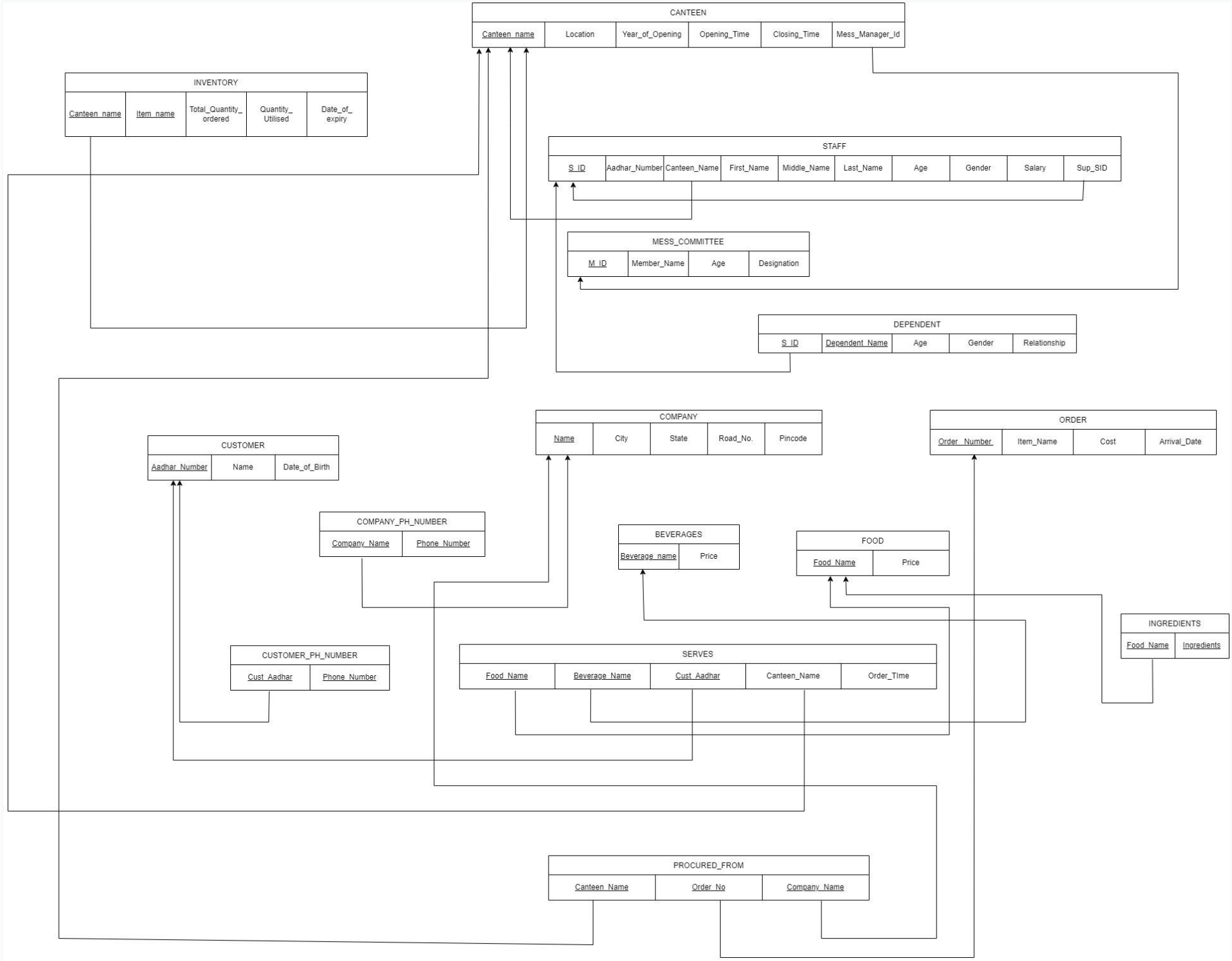
- For every multivalued attribute A, we created a new relation R. This relation will include an attribute corresponding to A, as well as the primary key K (as the foreign key attribute of R) of the relation that represents the entity type that has A as the multivalued attribute. In this relation, the combination of A and K will be the primary key of the relation.
- Here we had three multivalued attributes: Ingredients, Phone Number (from CUSTOMER), Phone Number (from COMPANY). We created three new relations: INGREDIENTS, CUSTOMER_PH_NUMBER, COMPANY_PH_NUMBER. We added the primary keys of the FOOD, CUSTOMER, COMPANY relations respectively as foreign keys in the newly added relations.
- This removed the violation of 1-NF (multivalued attributes) from our database.

Step 7: Mapping of N-ary Relationship Types

- For each degree > 2 relationship type, we create a new relationship S to represent the relation R. We include the primary keys of the relations that represent the participating entity types as foreign key attributes in S.
- For example, in the SERVES quaternary relationship type, we included the primary keys Food_Name, Beverage_Name, Aadhar_Number from the FOOD, BEVERAGES, CUSTOMER respectively, as foreign keys in the SERVES relation.

1-NF:

Our database does not need to be explicitly changed for it to be in 1 NF as there are no multivalued, composite or nested attributes remaining in the database after we map the ER diagram to a relational model.
We reduced composite attributes to their lowest atomic attributes and multivalued attributes were converted to relations and kept in separate tables.

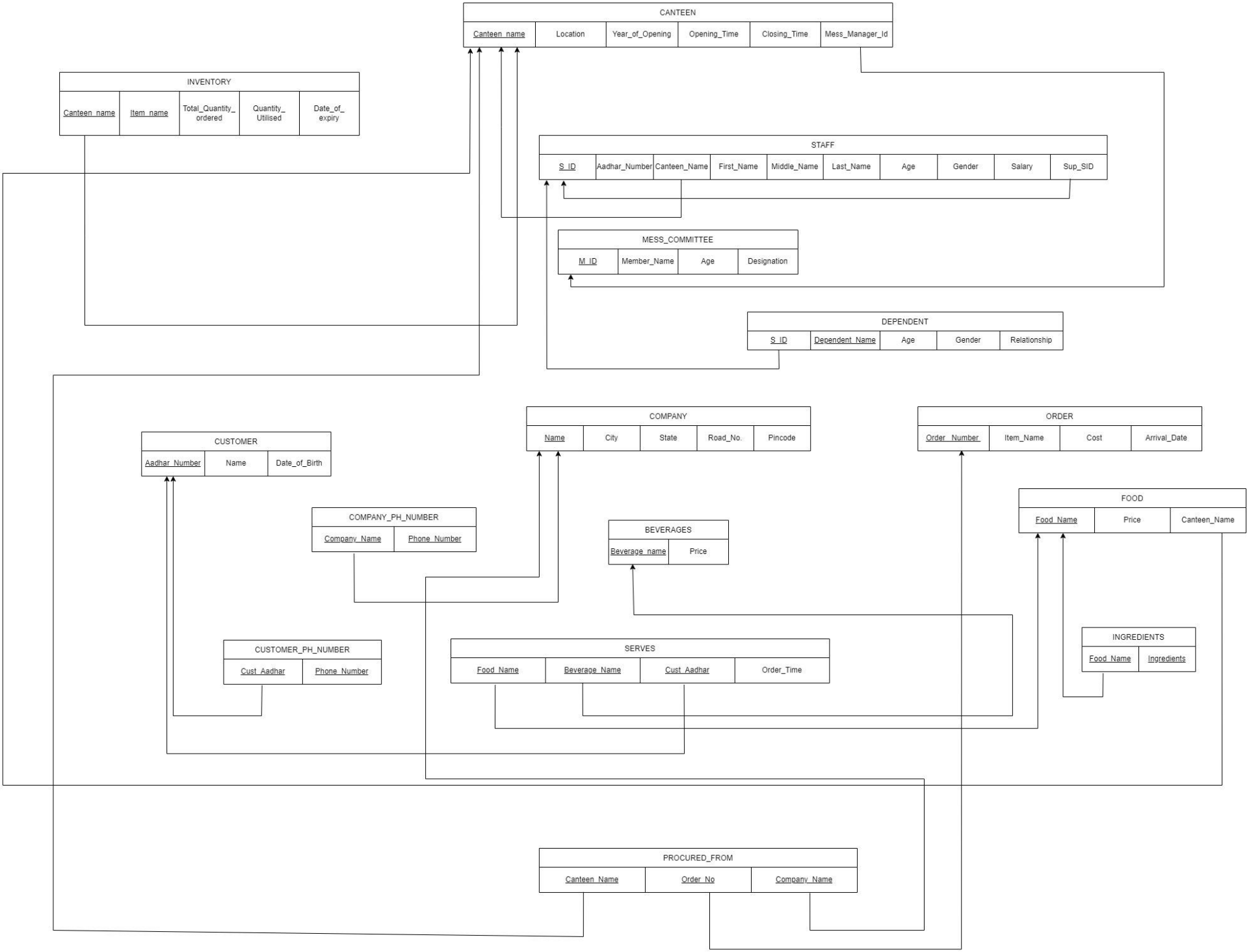

CANTEEN
Canteen_name
Location
Year_of_Opening
Opening_Time
Closing_Time
Mess_Manager_Id
INVENTORY
Canteen_name
Item_name
Total_Quantity_ordered
Quantity_Utilised
Date_of_expiry
STAFF
S_ID
Aadhar_Number
Canteen_Name
First_Name
Middle_Name
Last_Name
Age
Gender
Salary
Sup_SID
MESS_COMMITTEE
M_ID
Member_Name
Age
Designation
DEPENDENT
S_ID
Dependent_Name
Age
Gender
Relationship
COMPANY
Name
City
State
Road_No.
Pincode
ORDER
Order_Number
Item_Name
Cost
Arrival_Date
CUSTOMER
Aadhar_Number
Name
Date_of_Birth
COMPANY_PH_NUMBER
Company_Name
Phone_Number
BEVERAGES
Beverage_name
Price
FOOD
Food_Name
Price
INGREDIENTS
Food_Name
Ingredients
CUSTOMER_PH_NUMBER
Cust_Aadhar
Phone_Number
SERVES
Food_Name
Beverage_Name
Cust_Aadhar
Canteen_Name
Order_TIme
PROCURED_FROM
Canteen_Name
Order_No
Company_Name

2NF:

The FD {Food_Name} ->{Canteen_Name} in SERVES relation violates 2NF and thus needs to be normalized.

This can be done by taking out Canteen_Name and Food_Name from the table and forming a separate table. In this new table, {Food_Name} would form a primary key and the other non-prime attribute would be Canteen_Name.

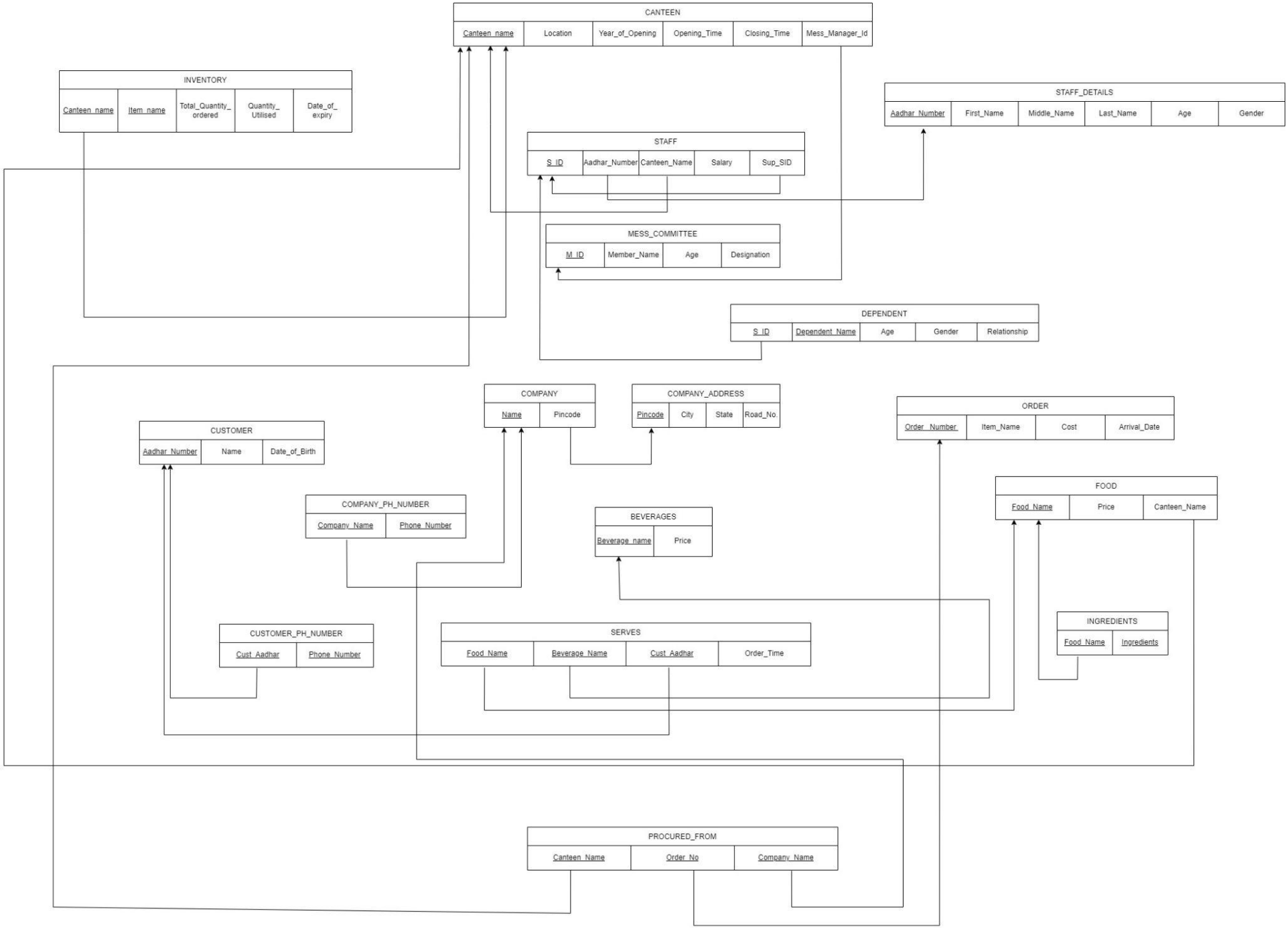
However, since the FOOD table already exists with Food_Name as a key attribute, creating another such table would be redundant. Thus, Canteen_Name is added as an attribute to the FOOD table.


CANTEEN
Canteen_name
Location
Year_of_Opening
Opening_Time
Closing_Time
Mess_Manager_Id
INVENTORY
Canteen_name
Item_name
Total_Quantity_ ordered
Quantity_ Utilised
Date_of_ expiry
STAFF
S_ID
Aadhar_Number
Canteen_Name
First_Name
Middle_Name
Last_Name
Age
Gender
Salary
Sup_SID
MESS_COMMITTEE
M_ID
Member_Name
Age
Designation
DEPENDENT
S_ID
Dependent_Name
Age
Gender
Relationship
COMPANY
Name
City
State
Road_No.
Pincode
ORDER
Order_Number
Item_Name
Cost
Arrival_Date
CUSTOMER
Aadhar_Number
Name
Date_of_Birth
FOOD
Food_Name
Price
Canteen_Name
COMPANY_PH_NUMBER
Company_Name
Phone_Number
BEVERAGES
Beverage_name
Price
INGREDIENTS
Food_Name
Ingredients
CUSTOMER_PH_NUMBER
Cust_Aadhar
Phone_Number
SERVES
Food_Name
Beverage_Name
Cust_Aadhar
Order_Time
PROCURED_FROM
Canteen_Name
Order_No
Company_Name

3NF:

In the relation COMPANY, the colattributes), City, State, Road_No. are transitively dependent on the primary key, Company_Name.
{Pincode} -> {City, State, Road_No.} forms an FD and thus violates 3NF. It has to thus be normalized.
We do this by creating a new table COMPANY_ADDRESS which has Pincode as the primary key and {City, State, Road_No.} as the other non prime attributes. The original COMPANY table now has Name as primary key and Pincode as a foreign key which references Pincode from the COMPANY_ADDRESS table. Pincode is now COMPANY's only non prime attribute.

In the relation STAFF, the attributes First_Name, Middle_Name, Last_Name, Age, Gender are transitively dependent on the primary key, S_ID.
{Aadhar_Number} -> {First_Name, Middle_Name, Last_Name, Age, Gender} forms an FD and thus violates 3NF. It has to thus be normalized.
We do this by creating a new table STAFF_DETAILS which has Aadhar_Number as the primary key and {First_Name, Middle_Name, Last_Name, Age, Gender} as the other non prime attributes. The original STAFF table now has S_ID as primary key and Aadhar_Number as a foreign key which references Aadhar_Number from STAFF_DETAILS table. Now Aadhar_Number, Canteen_Name, Salary, Sup_SID are the non-prime attributes in the STAFF table.


CANTEEN
Canteen_name
Location
Year_of_Opening
Opening_Time
Closing_Time
Mess_Manager_Id
INVENTORY
Canteen_name
Item_name
Total_Quantity_ ordered
Quantity_ Utilised
Date_of_ expiry
STAFF_DETAILS
Aadhar_Number
First_Name
Middle_Name
Last_Name
Age
Gender
STAFF
S_ID
Aadhar_Number
Canteen_Name
Salary
Sup_SID
MESS_COMMITTEE
M_ID
Member_Name
Age
Designation
DEPENDENT
S_ID
Dependent_Name
Age
Gender
Relationship
COMPANY
Name
Pincode
COMPANY_ADDRESS
Pincode
City
State
Road_No.
ORDER
Order_Number
Item_Name
Cost
Arrival_Date
CUSTOMER
Aadhar_Number
Name
Date_of_Birth
FOOD
Food_Name
Price
Canteen_Name
COMPANY_PH_NUMBER
Company_Name
Phone_Number
BEVERAGES
Beverage_name
Price
INGREDIENTS
Food_Name
Ingredients
CUSTOMER_PH_NUMBER
Cust_Aadhar
Phone_Number
SERVES
Food_Name
Beverage_Name
Cust_Aadhar
Order_Time
PROCURED_FROM
Canteen_Name
Order_No
Company_Name