Database for The Andhra Pradesh State Handloom Weavers Cooperative Society (APCO)

Group Members:

Rohan Patnaik (U101115FCS135)
Santosh Kumar Boyina (U101115FCS140)
Shailesh Mohta (U101115FCS305)

INTRODUCTION:

The Andhra Pradesh State Handloom Weavers Cooperative Society popularly known as APCO, is a cooperative of traditional handloom weavers of the Indian state of Andhra Pradesh. This is under the control of Department of Handlooms and Textiles of Government of Andhra Pradesh. The organisation owns a number of shopping outlets in Andhra Pradesh. The Society was registered in the year 1976 with registered No.T.P.W.44 under the Andhra Pradesh Cooperative Societies Act.

The objectives of APCO are to serve and function as a State Level Apex Society for cotton, silk and wool handloom weavers and their primary societies, to organise the handloom industry on commercial lines in Andhra Pradesh and few other parts of the country and to expand and develop the market for handloom fabrics.

APCO makes arrangements for the training of weavers in the latest methods of weaving and also for training of its employees in sales promotion techniques, business administration, bookkeeping and account maintenance of stocks, orders,etc.
APCO has been rendering service by purchasing handloom products from the affiliated Primary Handloom Weavers' Co-Operative societies and selling them in the state and all over the country through its wide network of handloom fabrics showrooms.

Thus, it is very important for them to keep an efficient record of data of :-
→ supply of inputs like yarn to the affiliated Handloom Weaver's societies.
→ procurement of cloth from the primaries
→ availability of stock of handloom clothes like dhotis , sarees , towels, etc. at the showrooms, shops, etc.
→ sale of the products to the customers
→ various showrooms ,shops ,sale outlets associated with APCO
→ employees , vendors - as it is very critical for the successful running of the industry.
→ customers :-
Each customer will have basic information stored and this will be linked to their purchases. Managers can run reports viewing which store they shopped at, what they bought, how many total transactions and how much money they spent.

The main problem with this society is that most of the work is done manually and a lot paperwork is done. Our effort is to reduce this paperwork and manual job.We intend to manage not only the data of products and orders, but also of the customers, employees, weavers, vendors, and different production societies involved by creating a database and storing the data online and managing it in a better way, thus offering better efficiency and helping them improving their business**.**

PROBLEM STATEMENT:

Firstly, every Weaver has a wavers_Name and weaver_ID and they make the Product and give it to the Production Society. Every Production Society has a ps_Name, a ps_ID and an address_ID. The Production Society marks the p_name, p_ID and the manufactured_date to the Product. They also mention the Specification of each Product which are length, cost, and thread count. This stock is then transferred to the Vendors, who have a vendor_name and vendor_ID. The Vendor then distribute the stock to the Showroom. The Showroom has a showroom_ID and showroom_Name and an address_ID. The Showroom has Employees, who have name, emp_ID, contact number and take fixed salary and of a particular age. Various Customers come to Showroom and buy the Products. From every Customer, his/her customer_name, phone, customer_address is taken and is given a unique customer_ID. As the Customer may pay in various forms, the Payment Method is noted with a unique payment_ID and payment_type. The purchases are collectively grouped under Order of the Customer which is given a order_ID and calculates the total_price. In this entire system, the Addresses contain address_ID and zipcode of the place.

ENTITIES AND ATTRIBUTES:

PAYMENT METHOD (payment_ID , payment_type , total_price)

CUSTOMER (customer_name , phone , customer_address , payment method_ID , customer_ID)

ADDRESS (address_ID , zipcode , place)

EMPLOYEE (name , contact number , salary , age , emp_ID)

PRODUCTS (p_name , p_ID , manufactured_date)

SPECIFICATION (length , cost , thread count , quantity , product_Name)

WEAVERS (weavers_Name , weaver_ID)

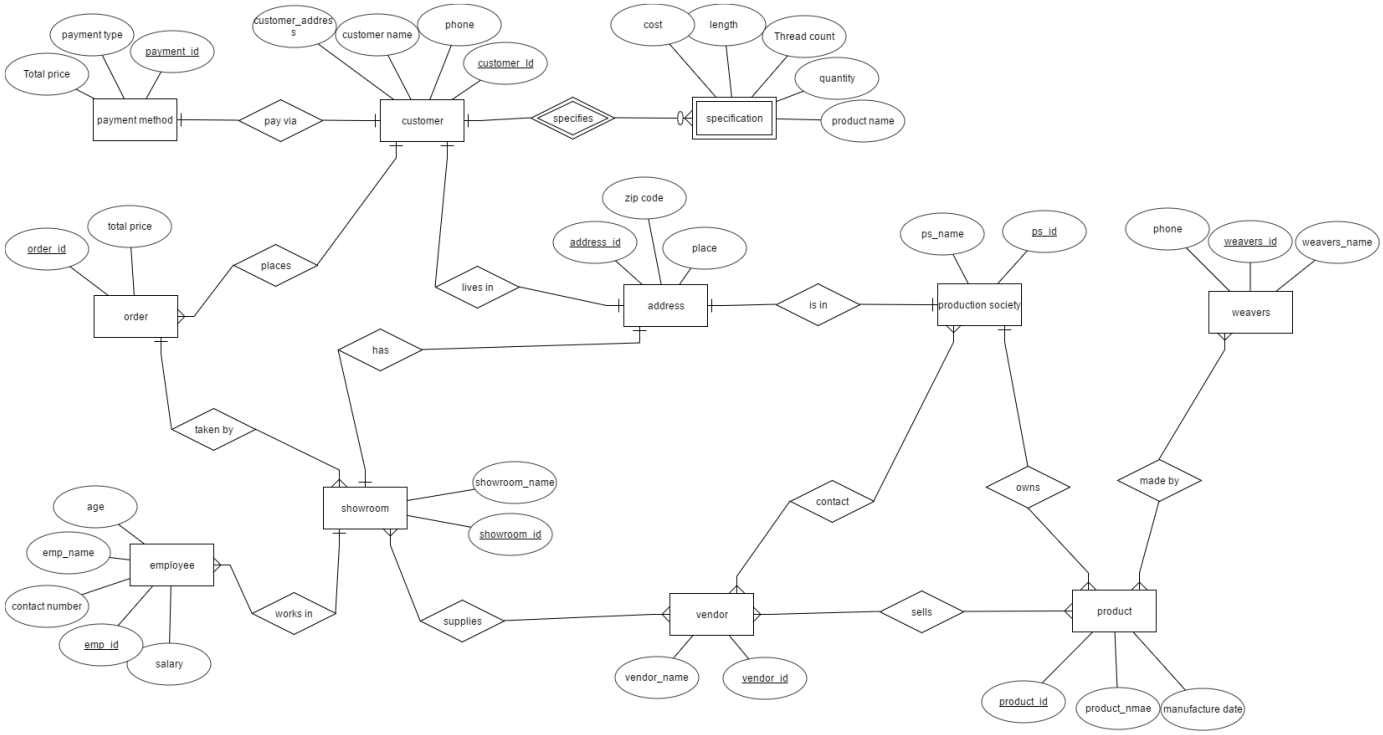
PRODUCTION SOCIETY (ps_Name, ps_ID)

VENDOR (vendor_name , vendor_ID)

ORDERS (total_price , order_ID)

SHOWROOM (showroom_Name , showroom_ID)

ER DIAGRAM



MYSQL TABLES

Payment method

Create table payment method(payment_type varchar(30),payment_Id int(10),total price int(10),primary key(payment_id));

Attributes	Type	Key
payment_type	varchar(30)	
total_price	int(10)	
payment_Id	int(10)	PRI

Customer

Create table customer(phone int(10),customer_name varchar(30),customer_id int(10),payment method_id int(10),customer_address varchar(30),primary key(customer_id));

Attributes	Type	Key
phone	int(10)	
Customer name	varchar(30)	
customer_id	int(10)	PRI
payment method_id	int(10)	
customer_address	varchar(30)	

Address

Create table address(zipcode int(10),place varchar(30),address_id int(10),primary key(address_id));

Attributes	Type	Key
zipcode	int(10)	
place	varchar(30)	
address_id	int(10)	PRI

Employee

Create table employee(name varchar(30),age int(10),salary int(10),showroom_id int(10),contact number int(10),emp_id int(10),primary key(emp_id));

Attributes	Type	Key
name	varchar(30)	

salary	int(10)	
age	int(10)	
showroom_id	int(10)	
Contact number	int(10)	
emp_id	int(10)	PRI

Products

Create table products(p_name varchar(30),p_id int(10),manufacture_date date,primary key(p_id));

Attributes	Type	Key
p_name	varchar(30)	
p_id	int(10)	PRI
manufacture_date	date	

Specification

Create table specification(length int(10),thread count int(10),cost int(10),quantity int(10),product name varchar(30),customer_id int(10));
Alter table specification add foreign key(customer_id);

Attributes	Type	Key
length	int(10)	
Thread count	int(10)	
cost	int(10)	
quantity	int(10)	
product_name	varchar(30)	
customer_id	int(10)	

Weaver

Create table weaver(weaver_name varchar(30),weaver_id int(10),production society_id int(10),product_id int(10),primary key (weaver_id));

Attributes	Type	Key
weaver _name	varchar(30)	
weaver_id	int(10)	PRI
production society_id	int(10)	
product_id	int(10)	

Production Society

Create table production society(ps_name varchar(30),ps_id int(10),address_idnt(10),primary key (ps_id));

Attributes	Type	Key
ps_name	varchar(30)	
ps_id	int(10)	PRI
address_id	int(10)	

Vendor

Create table vendor(vendor_name varchar(30),vendor_id int(10),primary key(vendor_id));

Attributes	Type	Key
vendor_name	varchar(30)	
vendor_id	int(10)	PRI

Orders

Create table orders(total_price int(10),order_id int(10),showroom_id int(10),customer_id int(10),primary key(order_id));

Attributes	Type	Key
total_price	int(10)	
order_id	int(10)	PRI
showroom_id	int(10)	
customer_id	int(10)	

Showroom

Create table showroom(showroom_id int(10),showroom_name varchar(30),address_id int(10),primary key(showroom_id));

Attributes	Type	Key
showroom_id	int(10)	PRI
showroom_name	varchar(30)	
address_id	int(10)	

TRANSFORM ER DIAGRAM INTO TABLES

- Convert all the Entities in the diagram to tables
- All single valued attributes of an entity is converted to a column of the table
- Key attribute in the ER diagram becomes the Primary key of the table.
- Declare the foreign key column, if applicable
- One can ignore derived attribute, since it can be calculated at any time

- Any multi-valued attributes are converted into new table.
- Any composite attributes are merged into same table as different columns.

· **Representing 1:1 relationship**

Imagine ADDRESS is an entity and PRODUCTION SOCIETY is another entity and we have address is in production society relation.It is one-one relation. i.e.; ; it implies one production society has one address We can represent this case in two ways

1. Create table for both ADDRESS and PRODUCTION SOCIETY. Add the primary key of ADDRESS in PRODUCTION SOCIETY table as foreign key. It implies the address for that particular production society
2. Create table for both ADDRESS and PRODUCTION SOCIETY. Add the primary key of PRODUCTION SOCIETY in address table as foreign key. It implies that production society is in address

Representing 1:N relationship

Consider ORDER and SHOWROOM relation, where each showroom can take multiple orders but particular order taken by one showroom. This is a 1: N relation. In this case, primary key of order table is added to the showroom table. i.e.; the primary key at 1 cardinality entity is added as foreign key to N cardinality entity

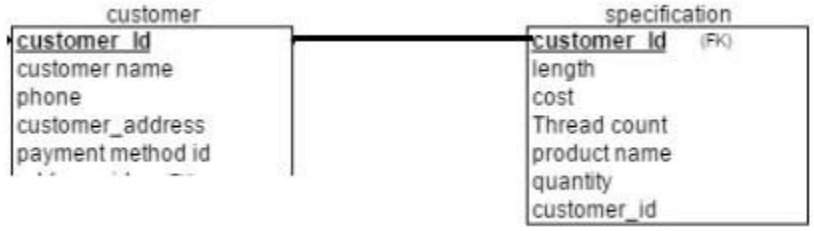
Representing M:N relationship

Consider the example vendor can supply products to multiple showrooms and showroom can receive products from multiple vendors which is M:N relation. In this case, we create SHOWROOM and VENDOR tables for the entities. Create one more table for the relation SUPPLY. Add the primary keys of showroom and vendor into it, which forms the composite primary key of the new table

Converting Weak Entity

Weak entity is also represented as table. All the attributes of the weak entity forms the column of the table. But the key attribute represented in the diagram cannot form the primary key of this table. We have to add a foreign key column, which would be the primary key column of its strong entity. This foreign key column along with its key attribute column forms the primary key of the table.

In our example SPECIFICATION is the weak entity. Hence, we create a table for it. Its attributes forms the column of this table. we have to find the foreign key first. CUSTOMER is the strong entity related to SPECIFICATION. Hence the primary key CUSTOMER_ID of CUSTOMER is added to SPECIFICATION table as foreign key.



RELATIONAL TABLE

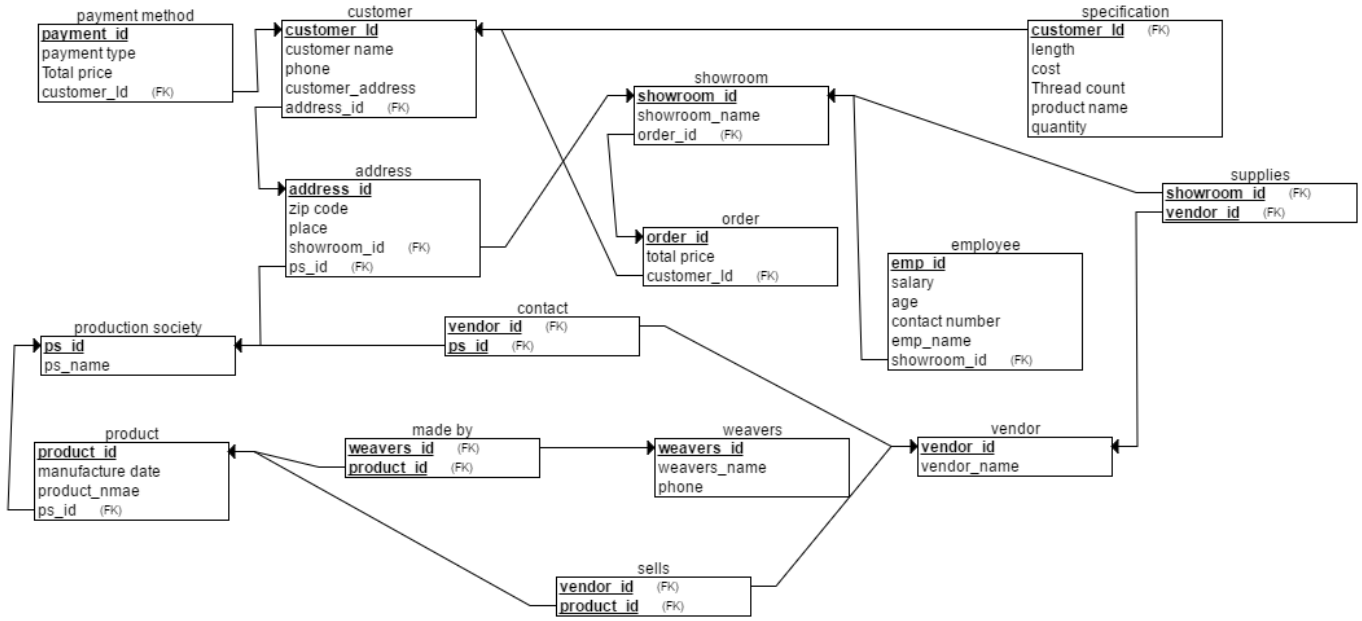


TABLE SCHEMA:

CREATE TABLE vendor

(

vendor_name INT NOT NULL,

vendor_id INT NOT NULL,

PRIMARY KEY (vendor_id)

);

CREATE TABLE production_society

(

```
 ps_name INT NOT NULL,
  ps_id INT NOT NULL,
  PRIMARY KEY (ps_id)
);
CREATE TABLE product
(
  product_id INT NOT NULL,
  manufacture_date INT NOT NULL,
  product_nmae INT NOT NULL,
  ps_id INT NOT NULL,
  PRIMARY KEY (product_id),
  FOREIGN KEY (ps_id) REFERENCES production_society(ps_id)
);
CREATE TABLE weavers
(
  weavers_id INT NOT NULL,
  weavers_name INT NOT NULL,
  phone INT NOT NULL,
  PRIMARY KEY (weavers_id)
);
CREATE TABLE made_by
```

```
(
  weavers_id INT NOT NULL,
  product_id INT NOT NULL,
  PRIMARY KEY (weavers_id, product_id),
  FOREIGN KEY (weavers_id) REFERENCES weavers(weavers_id),
  FOREIGN KEY (product_id) REFERENCES product(product_id)
);
CREATE TABLE sells
(
  vendor_id INT NOT NULL,
  product_id INT NOT NULL,
  PRIMARY KEY (vendor_id, product_id),
  FOREIGN KEY (vendor_id) REFERENCES vendor(vendor_id),
  FOREIGN KEY (product_id) REFERENCES product(product_id)
);
CREATE TABLE contact
(
  vendor_id INT NOT NULL,
  ps_id INT NOT NULL,
  PRIMARY KEY (vendor_id, ps_id),
  FOREIGN KEY (vendor_id) REFERENCES vendor(vendor_id),
```

```
  FOREIGN KEY (ps_id) REFERENCES production_society(ps_id)
);
CREATE TABLE payment_method
(
  payment_id INT NOT NULL,
  payment_type INT NOT NULL,
  Total_price INT NOT NULL,
  customer_Id INT NOT NULL,
  PRIMARY KEY (payment_id),
  FOREIGN KEY (customer_Id) REFERENCES customer(customer_Id)
);
CREATE TABLE customer
(
  customer_name INT NOT NULL,
  phone INT NOT NULL,
  customer_address INT NOT NULL,
  customer_Id INT NOT NULL,
  address_id INT NOT NULL,
  PRIMARY KEY (customer_Id),
  FOREIGN KEY (address_id) REFERENCES address(address_id)
);
```

```
CREATE TABLE address
(
  zip_code INT NOT NULL,
  address_id INT NOT NULL,
  place INT NOT NULL,
  showroom_id INT NOT NULL,
  ps_id INT NOT NULL,
  PRIMARY KEY (address_id),
  FOREIGN KEY (showroom_id) REFERENCES showroom(showroom_id),
  FOREIGN KEY (ps_id) REFERENCES production_society(ps_id)
);
CREATE TABLE specification
(
  length INT NOT NULL,
  cost INT NOT NULL,
  Thread_count INT NOT NULL,
  product_name INT NOT NULL,
  quantity INT NOT NULL,
  customer_Id INT NOT NULL,
  PRIMARY KEY (customer_Id),
  FOREIGN KEY (customer_Id) REFERENCES customer(customer_Id)
```

```
);
CREATE TABLE order_
(
  order_id INT NOT NULL,
  total_price INT NOT NULL,
  customer_Id INT NOT NULL,
  PRIMARY KEY (order_id),
  FOREIGN KEY (customer_Id) REFERENCES customer(customer_Id)
);
CREATE TABLE showroom
(
  showroom_name INT NOT NULL,
  showroom_id INT NOT NULL,
  order_id INT NOT NULL,
  PRIMARY KEY (showroom_id),
  FOREIGN KEY (order_id) REFERENCES order_(order_id)
);
CREATE TABLE employee
(
 salary INT NOT NULL,
  age INT NOT NULL,
```

```
  emp_id INT NOT NULL,
  contact_number INT NOT NULL,
  emp_name INT NOT NULL,
  showroom_id INT NOT NULL,
  PRIMARY KEY (emp_id),
  FOREIGN KEY (showroom_id) REFERENCES showroom(showroom_id)
);
CREATE TABLE supplies
(
  showroom_id INT NOT NULL,
  vendor_id INT NOT NULL,
  PRIMARY KEY (showroom_id, vendor_id),
  FOREIGN KEY (showroom_id) REFERENCES showroom(showroom_id),
  FOREIGN KEY (vendor_id) REFERENCES vendor(vendor_id)
);
```

FUNCTIONAL DEPENDENCIES:

PAYMENT METHOD

- payment_id → { payment_type },
- payment_id → { total_price }

- payment_id → { customer_id }
- payment_id → { customer_id , payment_type }
- payment_id → { customer_id , total_price }
- payment_id → { total_price , payment_type }
- payment_id → { payment_type , total_price , customer_id }

CUSTOMERS

- customer_id → { customer_name , address_id , customer_address , phone }
- customer_id→ { customer_name }
- customer_id → { address_id }
- customer_id → { customer_address }
- customer_id→ { phone }
- customer_id → { customer_name , address_id }
- customer_id → { customer_name , customer_address }
- customer_id → { customer_name , phone }
- customer_id → { address_id , customer_address }
- customer_id → { address_id , phone }
- customer_id → { customer_address , phone }

- customer_id → { customer_name , address_id ,customer_address }
- customer_id → { customer_name , address_id , phone }
- customer_id → { address_id , customer_address , phone }
- customer_name , phone → { customer_address , address_id , customer_id }

ADDRESS

- address_id → {zipcode , place, showroom_id , ps_id }
- address_id → zipcode
- address_id → place
- address_id → showroom_id
- address_id → ps_id
- address_id → showroom_id , ps_id
- address_id → showroom_id , place
- address_id → showroom_id , zipcode
- address_id → ps_id,zip code
- address_id → ps_id , place
- address_id → zipcode , place

EMPLOYEE

- emp_id → showroom_id , name , age , contact_number , salary
- name , contact_number → showroom_id , emp_id , age , salary

PRODUCTS

- produuct_id → product_name , manufacture_date , ps_id

SPECIFICATIONS

- customer_id , product_name → length, thread_count, quantity, cost

SHOWROOM

- Showroom_id → showroom_name , order_id
- Showroom_id → showroom_name
- Showroom_id → order_id

PRODUCTION SOCIETY

- Ps_id → ps_name

WEAVERS

- Weavers_id → weavers_name , phone

- Weavers_id → weavers_name
- Weavers_id → phone

Vendor

- Vendor_id → vendor_name

Order

- Order_id → total_price , customer_id
- Order_id → total_price
- Order_id → customer_id

NORMALISATION

PAYMENT_METHOD

2NF:

The candiates keys are { payment_id}, The set of key attributes are: { payment_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: payment_id → payment_type,total_price,customer_id

3NF:

The candiates keys are { payment_id}, The set of key attributes are: { payment_id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency payment_id → payment_type,total_price,customer_id

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

CUSTOMERS (It satisfies 1NF, 2NF, 3NF, and BCNF)

2NF:

The candidates keys are { customer_id}, { customer_name,phone}, The set of key attributes are: { customer_id,customer_name,phone }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: customer_id → customer_name,address_id,phone,customer_address
checking FD: customer_name,phone → address_id,customer_id,customer_address

3NF:

The candidates keys are { customer_id}, { customer_name,phone}, The set of key attributes are: { customer_id,customer_name,phone }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency customer_id → customer_name,address_id,phone,customer_address
checking functional dependency customer_name,phone → address_id,customer_id,customer_address

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey

ADDRESS (It satisfies 1NF, 2NF, 3NF, and BCNF)

2NF:

The candidates keys are { address_id}, The set of key attributes are: { address_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: address_id → zipcode,place,showroom_id,ps_id

3NF:

The candidates keys are { address_id}, The set of key attributes are: { address_id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency address_id → zipcode,place,showroom_id,ps_id

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

EMPLOYEE (It satisfies 1NF, 2NF, 3NF, and BCNF)

2NF:

The candidates keys are { emp_id}, { contact_number,name}, The set of key attributes are: { emp_id,contact_number,name }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: emp_id → showroom_id,name,age,contact_number,salary
checking FD: name,contact_number → showroom_id,age,salary,emp_id

3NF:

The candidates keys are { emp_id}, { contact_number,name}, The set of key attributes are: { emp_id,contact_number,name }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency emp_id → showroom_id,name,age,contact_number,salary
checking functional dependency name,contact_number → showroom_id,age,salary,emp_id

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

PRODUCTS (It satisfies 1NF, 2NF, 3NF, and BCNF)

2NF:

The candidates keys are { produuct_id}, The set of key attributes are: { produuct_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: produuct_id → product_name,manufacture_date,ps_id

3NF:

The candidates keys are { produuct_id}, The set of key attributes are: { produuct_id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency produuct_id → product_name,manufacture_date,ps_id

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey

SPECIFICATIONS (It satisfies 1NF, 2NF, 3NF, and BCNF)

2NF:

The candidates keys are { customer_id,product_name}, The set of key attributes are: { customer_id,product_name }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: customer_id,product_name → length,thread_count,quantity,cost

3NF:

The candidates keys are { customer_id,product_name}, The set of key attributes are: { customer_id,product_name }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency customer_id,product_name → length,thread_count,quantity,cost

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

SHOWROOM (It satisfies 1NF, 2NF, 3NF, and BCNF)

2NF:

The candidates keys are { Showroom_id}, The set of key attributes are: { Showroom_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: Showroom_id → showroom_name,order_id

3NF:

The candidates keys are { Showroom_id}, The set of key attributes are: { Showroom_id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency Showroom_id → showroom_name,order_id

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

PRODUCTION SOCIETY (It satisfies 1NF, 2NF, 3NF, and BCNF)

2NF:

The candidates keys are { Ps_id}, The set of key attributes are: { Ps_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes

checking FD: Ps_id → ps_name

3NF:

The candidates keys are { Ps_id}, The set of key attributes are: { Ps_id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency Ps_id → ps_name

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

WEAVERS (It satisfies 1NF, 2NF, 3NF, and BCNF)

2NF:

The candidates keys are { Weavers_id}, The set of key attributes are: { Weavers_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: Weavers_id → weavers_name,phone

3NF:

The candidates keys are { Weavers_id}, The set of key attributes are: { Weavers_id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency Weavers_id → weavers_name,phone

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey

VENDOR (It satisfies 1NF, 2NF, 3NF, and BCNF)

2NF:

The candidates keys are { vendor_id}, The set of key attributes are: { vendor_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: vendor_id → vendor_name

3NF:

The candidates keys are { vendor_id}, The set of key attributes are: { vendor_id }

for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency vendor_id → vendor_name

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

ORDER (It satisfies 1NF, 2NF, 3NF, and BCNF)

2NF:

The candiates keys are { Order_id}, The set of key attributes are: { Order_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: Order_id → total_price,customer_id

3NF:

The candiates keys are { Order_id}, The set of key attributes are: { Order_id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency Order_id → total_price,customer_id

BCNF:

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey

RELATIONAL DATABASE TABLE SCHEMA WITH SQL CODE

Entity	Description	SQL Code
Vendor	-	CREATE TABLE vendor (vendor_name INT NOT NULL,

		vendor_id INT NOT NULL, PRIMARY KEY (vendor_id));
Production Society	-	CREATE TABLE production_society (ps_name INT NOT NULL, ps_id INT NOT NULL, PRIMARY KEY (ps_id));
Product	-	CREATE TABLE product (product_id INT NOT NULL, manufacture_date INT NOT NULL, product_nmae INT NOT NULL, ps_id INT NOT NULL, PRIMARY KEY (product_id), FOREIGN KEY (ps_id) REFERENCES

		production_society(ps_id));
Weavers		CREATE TABLE weavers (weavers_id INT NOT NULL, weavers_name INT NOT NULL, phone INT NOT NULL, PRIMARY KEY (weavers_id));
Payment Method		CREATE TABLE payment_method (payment_id INT NOT NULL, payment_type INT NOT NULL, Total_price INT NOT NULL, customer_Id INT NOT NULL, PRIMARY KEY (payment_id), FOREIGN KEY (customer_Id) REFERENCES

		customer(customer_Id));
Customer		CREATE TABLE customer (customer_name INT NOT NULL, phone INT NOT NULL, customer_address INT NOT NULL, customer_Id INT NOT NULL, address_id INT NOT NULL, PRIMARY KEY (customer_Id), FOREIGN KEY (address_id) REFERENCES address(address_id));
Address		CREATE TABLE address (zip_code INT NOT NULL, address_id INT NOT NULL,

		place INT NOT NULL, showroom_id INT NOT NULL, ps_id INT NOT NULL, PRIMARY KEY (address_id), FOREIGN KEY (showroom_id) REFERENCES showroom(showroom_id), FOREIGN KEY (ps_id) REFERENCES production_society(ps_id));
Specifications		CREATE TABLE specification (length INT NOT NULL, cost INT NOT NULL, Thread_count INT NOT NULL, product_name INT NOT NULL, quantity INT NOT NULL, customer_Id INT NOT NULL, PRIMARY KEY (customer_Id), FOREIGN KEY (customer_Id) REFERENCES

		customer(customer_Id));
Order		CREATE TABLE order_ (order_id INT NOT NULL, total_price INT NOT NULL, customer_Id INT NOT NULL, PRIMARY KEY (order_id), FOREIGN KEY (customer_Id) REFERENCES customer(customer_Id));
Showroom		CREATE TABLE showroom (showroom_name INT NOT NULL, showroom_id INT NOT NULL, order_id INT NOT NULL, PRIMARY KEY (showroom_id), FOREIGN KEY (order_id)

		REFERENCES order(order_id));
Employee		CREATE TABLE employee (salary INT NOT NULL, age INT NOT NULL, emp_id INT NOT NULL, contact_number INT NOT NULL, emp_name INT NOT NULL, showroom_id INT NOT NULL, PRIMARY KEY (emp_id), FOREIGN KEY (showroom_id) REFERENCES showroom(showroom_id));

Relation	Description	SQL Code
made_by	Participating Entities: product and weavers A weaver makes many products. A product is made by many weavers	CREATE TABLE made_by (weavers_id INT NOT NULL, product_id INT NOT NULL, PRIMARY KEY (weavers_id, product_id), FOREIGN KEY (weavers_id) REFERENCES weavers(weavers_id), FOREIGN KEY (product_id) REFERENCES product(product_id));
sells	Participating Entities: vendor and product A vendor sells many products. A product is sold by many vendors	CREATE TABLE sells (vendor_id INT NOT NULL, product_id INT NOT NULL, PRIMARY KEY (vendor_id, product_id), FOREIGN KEY (vendor_id) REFERENCES vendor(vendor_id), FOREIGN KEY (product_id)

		REFERENCES product(product_id));
contact	Participating Entities: vendor and production society A vendor contacts many production societies. A production society contacts many vendors	CREATE TABLE contact (vendor_id INT NOT NULL, ps_id INT NOT NULL, PRIMARY KEY (vendor_id, ps_id), FOREIGN KEY (vendor_id) REFERENCES vendor(vendor_id), FOREIGN KEY (ps_id) REFERENCES production_society(ps_id));
supplies	Participating Entities: showroom and vendor A vendor supplies to many showrooms. A showroom is supplied by many vendors.	CREATE TABLE supplies (showroom_id INT NOT NULL, vendor_id INT NOT NULL, PRIMARY KEY (showroom_id, vendor_id), FOREIGN KEY (showroom_id)

		REFERENCES showroom(showroom_id), FOREIGN KEY (vendor_id) REFERENCES vendor(vendor_id));
pay via	Participating Entities: payment method and customer. A customer can pay via one payment method at a time. One payment method is used by one customer to pay at a time.	-
specifies	Participating Entities: customer and specification. This is a weak relation and customer_id of customer table is the primary key of specification table. A customer may specify many specifications. A specification may be specified by one customer.	-
places	Participating Entities:order and customer. A customer can place many orders. An order can be placed by one customer	-
taken by	Participating Entities: order and showroom An order is taken by one showroom. A showroom takes many	-

	orders.	
works in	Participating Entities: employee and showroom. An employee works in one showroom. Many employees work in one showroom.	-
has	Participating Entities: showroom and address A showroom has one address. In one address there is one showroom.	-
lives in	Participating Entities: customer and address. A customer lives in one address. In an address there is one customer.	-
is in	Participating Entities: address and production society A production society is in one address. In an address there is one production society	-
owns	Participating Entities: production society and product A production society owns many products. A product is owned by one production society	-

Sample table outputs

Vendor

vendor_name	vendor_id
nagaraju	1
subbaraju	2
appalaaraju	3
kondarao	4
kamaraju	5
ramaraju	6
ramarao	7
venkataramana	8
venkatarao	9
venkatesh	10
NULL	NULL

Production Society

ps_name	ps_id
aravind	11
warangal	12
ompole	13
gadwal	14
pochampalli	15
venkatagiri	16
mangalagiri	17
chirala	18
uppada	19
narayanapet	20
NULL	NULL

Product

product_id	manufacture_date	product_name	ps_id
21	1996-07-30	saree	11
22	2000-08-01	lungi	12
23	2010-06-05	towel	13
24	2004-05-15	bedsheet	14
25	2011-11-23	handkerchief	15
26	1998-10-25	shirt	16
27	2000-01-14	trouser	17
28	2003-02-16	pillowcovers	18
29	2009-03-26	musquitonets	19
30	2002-04-28	curtains	20
NULL	NULL	NULL	NULL

Weavers

weavers_id	weavers_name	phone
31	hari	9441569338
32	suri	9442589338
33	giri	9442569338
34	ravi	9441569808
35	nari	9441569698
36	naresh	9441569333
37	nagesh	9441561038
38	ramesh	9441567738
39	suresh	9441564438
40	devi	9441569368
NULL	NULL	NULL

made_by

weavers_id	product_id
31	21
32	22
33	23
34	24
35	25
36	26
37	27
38	28
39	29
40	30
NULL	NULL

sells

vendor_id	product_id
1	21
2	22
3	23
4	24
5	25
6	26
7	27
8	28
9	29
10	30
NULL	NULL

contact

vendor_id	ps_id
1	11
2	12
3	13
4	14
5	15
6	16
7	17
8	18
9	19
10	20
NULL	NULL

address

zip_code	address_id	place	showroom_id	ps_id
530007	61	duvvada	71	11
530002	62	madhurwava	72	12
530004	63	pandurangapuram	73	13
530011	64	palasa	74	14
530008	65	srikakulam	75	15
530025	66	vijayanagaram	76	16
530013	67	vijayawada	77	17
530018	68	rajamundry	78	18
530017	69	godavari	79	19
530003	70	krishna	80	20
NULL	NULL	NULL	NULL	NULL

payment_method

payment_id	payment_type	Total_price	customer_Id
41	online	2999	51
42	online	3458	52
43	offline	3089	53
44	offline	2590	54
45	online	3089	55
46	online	3079	56
47	offline	989	57
48	online	9989	58
49	online	4589	59
50	offline	6566	60
NULL	NULL	NULL	NULL

customer

customer_name	phone	customer_address	customer_Id	address_id
rahul	9441278678	madhavadhara	51	61
raghav	9441247089	seethamadhara	52	62
roshan	9441242233	seethamapeta	53	63
ranjan	9441245649	dwarakanagar	54	64
anjali	9441737378	ramnagar	55	65
likhitha	9446247378	nehrunagar	56	66
vashwitha	9441280378	madhurwada	57	67
rakshitha	9441256378	akkavapalem	58	68
subhanvitha	9441247380	mvpcolony	59	69
sahithi	9441247379	gajuwaka	60	70
NULL	NULL	NULL	NULL	NULL

specification

length	cost	Thread_count	product_name	quantity	customer_Id
5	303	80	saree	1	51
7	256	100	lungi	1	52
5	258	100	towel	1	53
6	313	80	bedsheet	1	54
7	111	80	handkerchief	1	55
6	58	80	shirt	1	56
5	88	100	trouser	1	57
4	101	80	pillowcovers	1	58
5	109	80	musquitonets	1	59
6	666	100	curtains	1	60
NULL	NULL	NULL	NULL	NULL	NULL

order_

order_id	total_price	customer_Id
81	2999	51
82	3458	52
83	3089	53
84	2590	54
85	3089	55
86	3079	56
87	989	57
88	9989	58
89	4589	59
90	6566	60
NULL	NULL	NULL

showroom

showroom_name	showroom_id	order_id
balaji	71	81
venkateswara	72	82
sana	73	83
rana	74	84
vaibhav	75	85
kalaniketan	76	86
kankatala	77	87
cmr	78	88
rs	79	89
chermas	80	90
NULL	NULL	NULL

employee

salary	age	emp_id	contact_number	emp_name	showroom_id
30000	27	91	8897150223	lohiva	71
25000	26	92	8897859022	santosh	72
15000	24	93	8897189022	kushal	73
30000	28	94	8897178223	rohan	74
25000	33	95	8897155223	naidu	75
30000	26	96	8897152322	revanth	76
30000	28	97	8897155823	harihanth	77
30000	27	98	8897155622	manish	78
25000	30	99	8897154523	vamsheeth	79
30000	27	100	8897155223	shailesh	80
NULL	NULL	NULL	NULL	NULL	NULL

Supplies

showroom_id	vendor_id
71	1
72	2
73	3
74	4
75	5
76	6
77	7
78	8
79	9
80	10
NULL	NULL