**Database design & Modeling Assignment 1**

**Jushira Thelakkat**

**jt39634**

# PART A: Coffee database

1. **Identify various functional dependencies. Using functional dependencies, evaluate the normal form of the table (Excel spreadsheet)? Justify**.

**Solution:**

This example has 21 variables , namely,

|  |  |
| --- | --- |
| ProductId |  |
| Product |  |
| ProductLine |  |
| Type |  |
| ProductType |  |
| Market |  |
| State |  |
| Area Code |  |
| Market Size |  |
| Date |  |
| Profit |  |
| Margin |  |
| Sales |  |
| COGS |  |
| Total Expenses |  |
| Marketing |  |
| Inventory |  |
| Budget Profit |  |
| Budget Margin |  |
| Budget Sales |  |
| Budget COGS |  |

* There are a large number of functional dependencies in this Coffee example.

ProductId🡪Product

ProductId🡪Product Line

ProductId🡪Type

ProductId🡪Product Type

* Similarly, variable Product individually has a functional dependency with each of Product Line, Type and Product Type

State🡪Market

State🡪Market Size

Area Code🡪Market

Area Code🡪State

Area Code🡪Market Size

* We also have a combination of columns that give us an idea about a new column.

Product, Area Code, Date🡪Profit

Product, Area Code, Date🡪Margin

Product, Area Code, Date🡪Sales

Product, Area Code, Date🡪COGS

Product, Area Code, Date🡪Total Expenses

Product, Area Code, Date🡪Marketing

Product, Area Code, Date🡪Inventory

Product, Area Code, Date🡪Budget Margin

Product, Area Code, Date🡪Budget Profit

Product, Area Code, Date🡪Budget Sales

Product, Area Code, Date🡪Budget COGS

Since we see from the functional dependencies that there still exists partial dependency and transitive dependencies, we can say that this dataset is in its FIRST functional form.

1. **Normalize the database into 3rd normal form using functional dependencies identified above. Identify the various tables including primary keys and foreign keys. (Feel free to create unique keys when you have composite primary keys)**

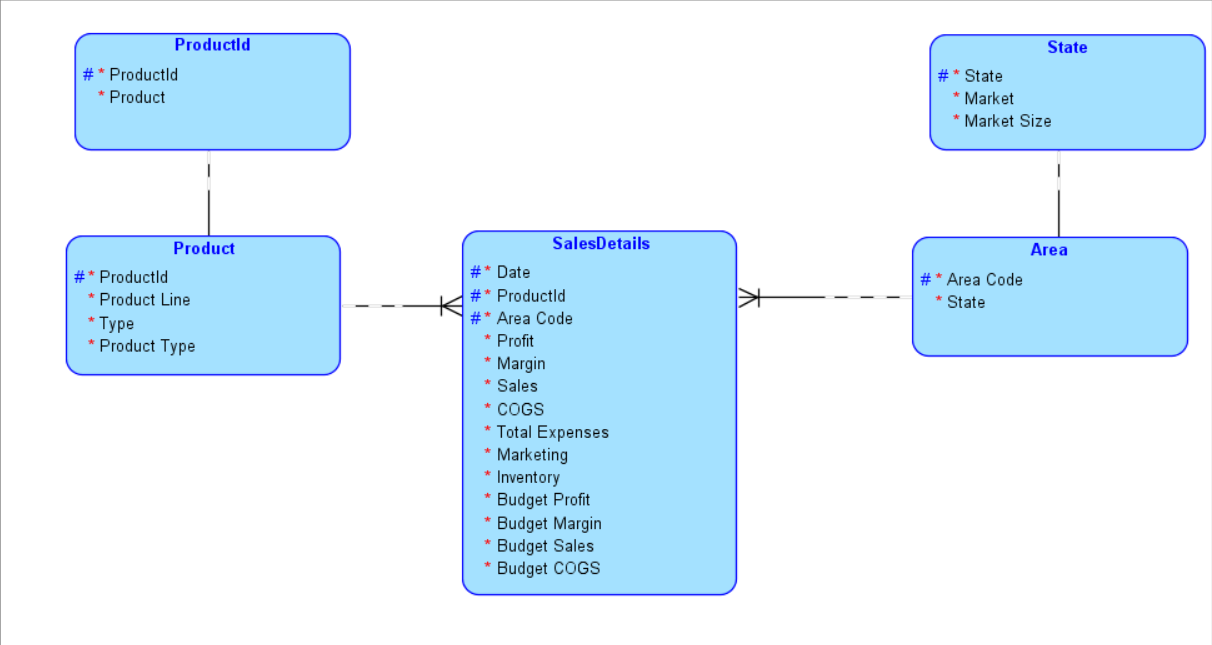
**Solution:**

Generally, the third normal form is considered to be a good standard for a relational database.

To convert database into 3rd normal form, we have to remove the PARTIAL dependencies and the TRANSITIVE dependencies.

Since ProductID and Product have partial dependencies, we split them up into different tables.

* An entity named ProductId with ProductId as the primary key
* An entity named product with ProductId as the primary key and Line and Type related attributes. The foreign key here is the ProductId from the ProductId entity.
* An entity named State with State as primary key as it uniquely describes Market and MarketSize
* An entity named Area with Area Code as the primary key as it uniquely describes State. The foreign key here is the State from the State entity/table.
* An entity named SalesDetails with composite primary key (Date,Area Code,ProductId) and attributes such as Profit, Margin, Sales etc. This table will have two foreign keys namely Area Code and ProductId.

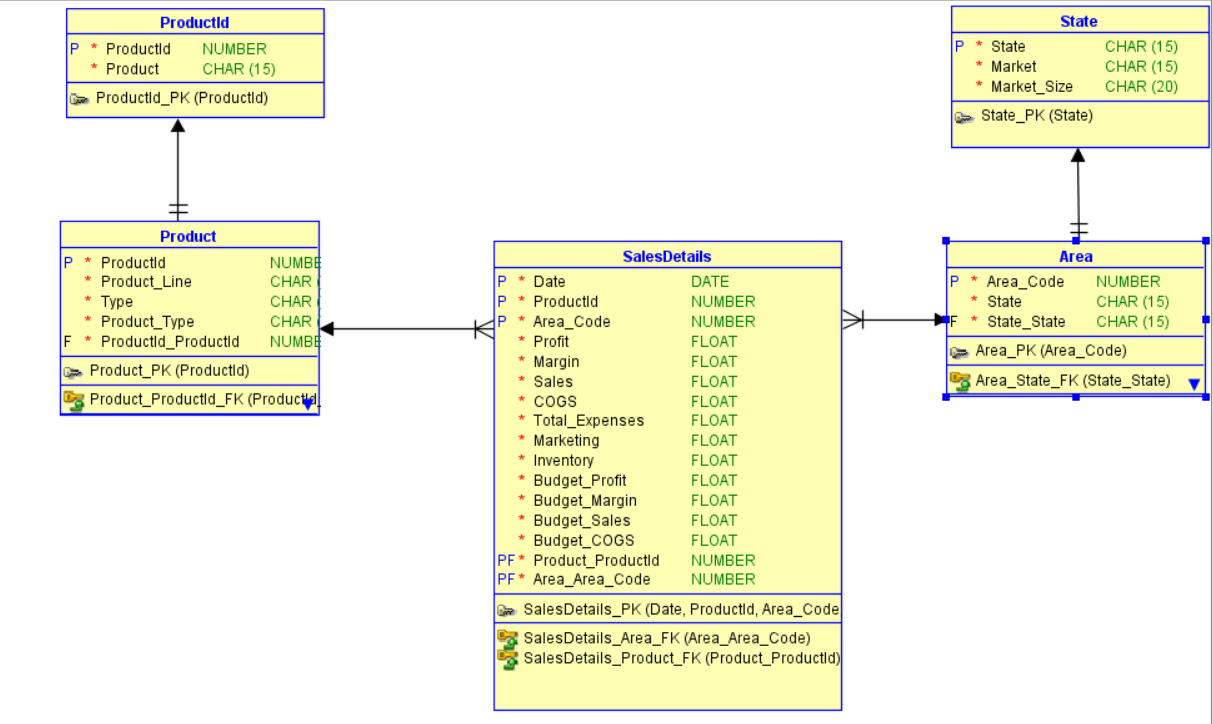


Now this above Logical diagram is in the 3rd Functional form as I have tried to remove all partial and transitive dependencies.

1. Now, identify various entities and relationship types (1-1; 1-M; M-N), draw the E-R diagram using Oracle Data Modeler (or other tools). Convert the logical schema into actual table structure. Clearly identify PKs and FKs in the table structure.

**Solution:** As mentioned in the above solution, we have five entities namely ProductId, Product, SalesDetails, State and Area.

* There is 1-1 relationship between ProductId and Product
* Similarly there is a 1-1 relationship between State and Area as one attribute in State matches to one in Area
* There is a 1-N relationship between Product and SalesDetails because each product has more than one row for the attributes mentioned in the SalesDetails entity.
* Similarly, there is a 1-N relationship between Area and SalesDetails.



The above is the ER diagram on the Data Modeler. In each entity, the primary keys (PK) and foreign keys (FK) are clearly visible.

The CREATE statements in DDL File Editor are as follows.

-- Generated by Oracle SQL Developer Data Modeler 17.2.0.188.1059

-- at: 2017-09-09 13:30:14 CDT

-- site: Oracle Database 11g

-- type: Oracle Database 11g

CREATE TABLE area (

area\_code NUMBER NOT NULL,

state CHAR(15) NOT NULL,

state\_state CHAR(15) NOT NULL

);

CREATE UNIQUE INDEX area\_\_idx ON

area ( state\_state ASC );

ALTER TABLE area ADD CONSTRAINT area\_pk PRIMARY KEY ( area\_code );

CREATE TABLE product (

productid NUMBER NOT NULL,

product\_line CHAR(15) NOT NULL,

type CHAR(15) NOT NULL,

product\_type CHAR(15) NOT NULL,

productid\_productid NUMBER NOT NULL

);

CREATE UNIQUE INDEX product\_\_idx ON

product ( productid\_productid ASC );

ALTER TABLE product ADD CONSTRAINT product\_pk PRIMARY KEY ( productid );

CREATE TABLE productid (

productid NUMBER NOT NULL,

product CHAR(15) NOT NULL

);

ALTER TABLE productid ADD CONSTRAINT productid\_pk PRIMARY KEY ( productid );

CREATE TABLE salesdetails (

"Date" DATE NOT NULL,

productid NUMBER NOT NULL,

area\_code NUMBER NOT NULL,

profit FLOAT NOT NULL,

margin FLOAT NOT NULL,

sales FLOAT NOT NULL,

cogs FLOAT NOT NULL,

total\_expenses FLOAT NOT NULL,

marketing FLOAT NOT NULL,

inventory FLOAT NOT NULL,

budget\_profit FLOAT NOT NULL,

budget\_margin FLOAT NOT NULL,

budget\_sales FLOAT NOT NULL,

budget\_cogs FLOAT NOT NULL,

product\_productid NUMBER NOT NULL,

area\_area\_code NUMBER NOT NULL

);

ALTER TABLE salesdetails

ADD CONSTRAINT salesdetails\_pk PRIMARY KEY ( "Date",productid,area\_code,product\_productid,area\_area\_code );

CREATE TABLE state (

state CHAR(15) NOT NULL,

market CHAR(15) NOT NULL,

market\_size CHAR(20) NOT NULL

);

ALTER TABLE state ADD CONSTRAINT state\_pk PRIMARY KEY ( state );

ALTER TABLE area

ADD CONSTRAINT area\_state\_fk FOREIGN KEY ( state\_state )

REFERENCES state ( state );

ALTER TABLE product

ADD CONSTRAINT product\_productid\_fk FOREIGN KEY ( productid\_productid )

REFERENCES productid ( productid );

ALTER TABLE salesdetails

ADD CONSTRAINT salesdetails\_area\_fk FOREIGN KEY ( area\_area\_code )

REFERENCES area ( area\_code );

ALTER TABLE salesdetails

ADD CONSTRAINT salesdetails\_product\_fk FOREIGN KEY ( product\_productid )

REFERENCES product ( productid );

-- Oracle SQL Developer Data Modeler Summary Report:

--

-- CREATE TABLE 5

-- CREATE INDEX 2

-- ALTER TABLE 9

-- CREATE VIEW 0

-- ALTER VIEW 0

-- CREATE PACKAGE 0

-- CREATE PACKAGE BODY 0

-- CREATE PROCEDURE 0

-- CREATE FUNCTION 0

-- CREATE TRIGGER 0

-- ALTER TRIGGER 0

-- CREATE COLLECTION TYPE 0

-- CREATE STRUCTURED TYPE 0

-- CREATE STRUCTURED TYPE BODY 0

-- CREATE CLUSTER 0

-- CREATE CONTEXT 0

-- CREATE DATABASE 0

-- CREATE DIMENSION 0

-- CREATE DIRECTORY 0

-- CREATE DISK GROUP 0

-- CREATE ROLE 0

-- CREATE ROLLBACK SEGMENT 0

-- CREATE SEQUENCE 0

-- CREATE MATERIALIZED VIEW 0

-- CREATE SYNONYM 0

-- CREATE TABLESPACE 0

-- CREATE USER 0

--

-- DROP TABLESPACE 0

-- DROP DATABASE 0

--

-- REDACTION POLICY 0

--

-- ORDS DROP SCHEMA 0

-- ORDS ENABLE SCHEMA 0

-- ORDS ENABLE OBJECT 0

--

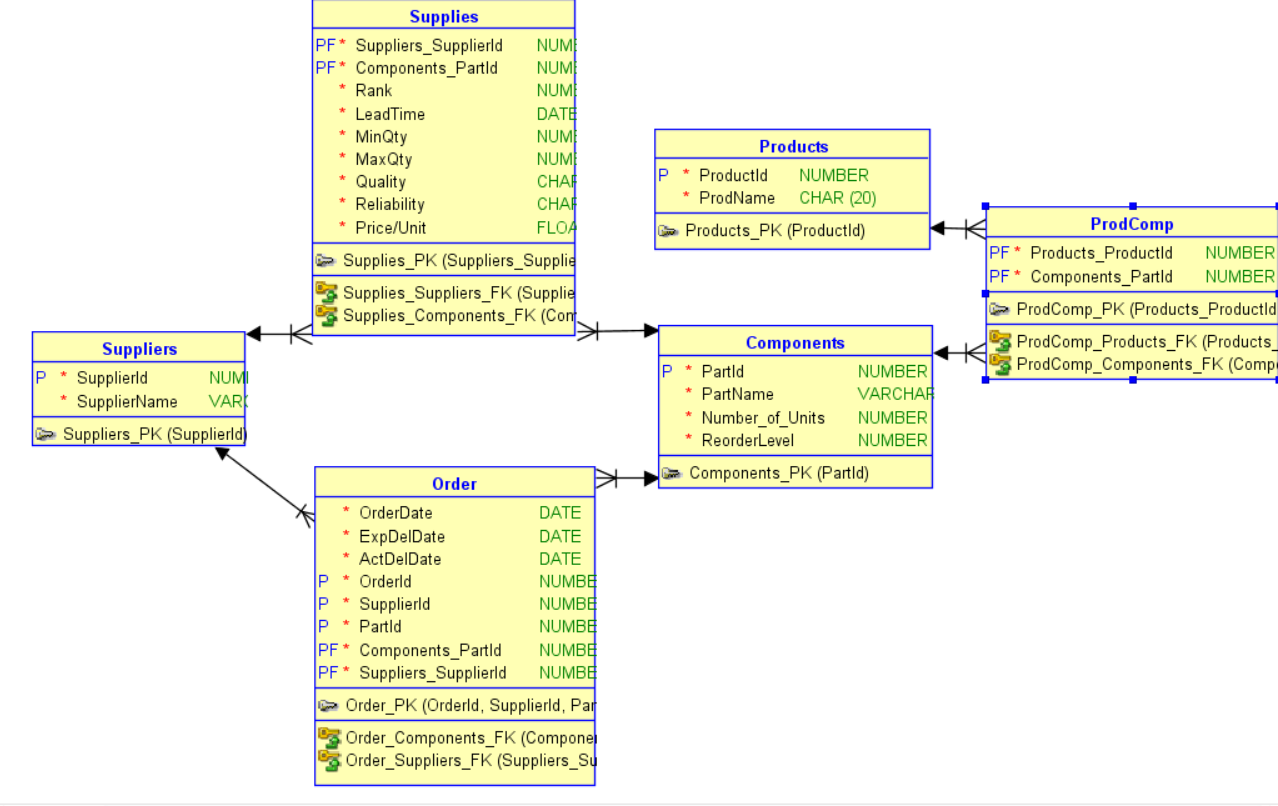
-- ERRORS 0

-- WARNINGS 0

# PART B:

1. **Identify various entities and the relationships and draw the E-R diagram. Convert the E-R diagram into tables (relational model).**
2. **What if an order can be supplied in small quantities at various times (that is, a large order can be split into many small orders). How does your E-R diagram and table structure change?**

**Solution:**



We have four entities here – Suppliers, Products, Components and Order

The primary key for Products will be ProductId and that for the Components table will be PartId.

We have each product having a lot of components and each component can be a part of different products so a many🡪many relationship between entities Products and Components, which gives us a third bridge table ProdComp.

The relationship between suppliers and components is SUPPLIES which has Rank, LeadTime, Min and Max Quantities, Quality, Reliability and Price/unit has its attributes.

In the Components table, if the number of units falls below the reorder value, then a purchase order is placed (the Order table) where the attributes such as Expected Delivery Date, Actual Delivery Date and Date of the order are captured.

There is a 1🡪N relationship between Suppliers and Order because this re purchase order is placed only when units fall below reorder and then the rank is considered and the order is released only to ONE supplier.

If one order is now supplied in many small quantities, we will now have more than one OrderId for the same order. Say one large order is now supplied in five different quantities, we will have five different orderIds for the same order on the same date.

By manipulating the primary key in the order table a little, this problem can be solved.

For order A, instead of one unique OrderId, we will now have five unique OrderIds. And we can now incorporate the OrderDate as a part of the primary key too.

So now we will have a primary key which is a combination of OrderId (different from OrderId of Part (a)), OrderDate, SuuplierId and PartId.