

Survey Application Database for a Product Manufacturing Company

Eshwar N Kumar

Team Name: Targaryen

Team Members(Total:1): Eshwar N Kumar

University at Buffalo, State University of New York

UB ID: eshwarna

eshwarna@buffalo.edu

ABSTRACT

Data is nothing but facts that can be recorded. A Database is a collection of related data that represents some real-world entities with which the user directly interacts. A Database Management System manages the database by ensuring that there is no redundancy, inconsistency in the data and ensures optimal storage of data. In this paper, we will design an optimal database for a survey application of a product manufacturing company and ensure that each relation is in accordance with Boyce-Codd Normal Form.

1. INTRODUCTION

A survey application database is mainly used by the product manufacturing company to analyze the customers opinion about the products sold by the company and about the company itself. On the other hand, the values to the database will be input by the customers who provide feedback by taking the survey. The company using the survey application and the people who want to see the customer's feedback(like engineers, HRs, etc) will administer the database.

Let us understand a real life scenario to understand the usage of the survey application database. Let's consider a Company by name Rubix utilize the survey application that uses our survey application database for storing the data. A purchaser buys a new product named X manufactured by Rubix at Buff Branch. Post his/her purchase, he/she will complete the survey about his purchase of product X and the company Rubix. The company after few days makes use of the database that stores the customers feedback and makes necessary improvements with respect to both product and the company to reach a higher position.

2. PROBLEM STATEMENT

Develop a Survey Application Database for the products of a company with the below requirements:

1. A user will have to complete the survey after he/she completes purchase of a product from the company.
2. The survey can consist of 3 different types of questions. It can be objective with one answer choice selection, objective with multiple answer choice selection, or it can be subjective where the user types textual answer to the question.
3. The survey should collect company feedback, product/s feedback, branch feedback, and the location where the purchaser resides.

4. The survey should be such that it can be used for multiple products that are manufactured by the company.

Below are the reasons why a database is needed to solve this problem instead of an excel file/spread sheet:

1. Every cell in a excel file has no constraints on the type of data that can be stored in it. Due to this, a string like the product name can be inserted in the place where the product id which is an integer value have to be inserted. This results in lots of erroneous information in the excel file. However, every index in a database has constraints on the type of data which can be stored in it to improve data integrity and avoid erroneous information storage in the database.
2. In a excel file, we can create multiple worksheets(one for each table) and relate one worksheet with the other but such relations will be very limited in spreadsheets. However, there is no limit for the number of relations in a database and a database can store more than a million of records. Hence, using a database over excel file will increase the performance of operations and can store millions of records of data which results in higher speed of manipulation of data in the dataset.
3. In the case of excel file, it is very hard to find who changed what data. Also, we cannot impose user restrictions for an excel file. But with a database, the tables will have a highly stable structure and we can implement access permissions and user restrictions. This drastically improves the security and data consistency of the data stored. In addition, retrieval and updating of data is more easier, safer using databases than with an excel file.

3. IMPELEMENTED SOLUTION

3.1 Survey Application Overview

Initially based on the database schema described in section 3.2, the tables are created with necessary constraints like primary key, foreign key, etc. using Data Definition Language(DDL) commands.

According to the real-life scenario described in the section 1, the survey data from the customer feedback is inserted into the respective database relations using SQL INSERT commands. In this way, we update data into the tables.

In order to view/insert/modify the data stored in the database tables, we can use SQL Data Manipulation Language(DML) commands like INSERT/DELETE/JOIN to perform data manipulation according to our needs.

Since we are using Databases, there is typically no limit for the number of tuples that can be stored for each relation. In addition, the databases support update of data and the queries can be performed without any limitation.

3.2 Database Schema for our Survey Application Database

Below are the list of relations along with the constraints applicable for each attribute of every relation in the schema
Product(id, name, category, companyid)

Constraints:

Id: Primary Key, companyid: Foreign Key, name: not null, category: not null

Product table contains details of all the products manufactured by company Rubix.

id is the Primary Key because it is unique for every Product.

companyid is the Foreign key that references to the id in the Company table. It indicates the company that has manufactured the product. If the company is deleted in the company table, then all the products associated with the company must also be deleted. Hence, companyid is associated with 'on delete cascade'.

name is the name of the product and it should not be a null value.

category is the category to which the product belongs to and it also must be not null.

Company(id, name)

Constraints:

id: Primary Key, name: not null

Company relation contains the company name and id. Please note that for this project, we are considering only one company and surveying all the products manufactured by that company. However, the same database schema can be extended to more than one company with some additional modifications.

id is the primary key.

name is the name of the company and it should not be null.

Customer(id, name, age, gender, branch, country)

Constraints:

id: Primary Key, name: not null, age: not null, gender: not null, branch: not null, country: not null

Customer relation stores all the details of the customer who has purchased one or more products manufactured by Rubix.

id is the Primary Key.

name is the name of the customer and it must not be null.

age is the age of the customer which is int value and should not be null

gender indicates the gender identity of the customer and should not be null

branch is the branch name from which the customer has purchased the product and should not be null

country is the country from which the customer has purchased the product and should not be null.

Survey(id, title)

Constraints:

Id: Primary Key, title: not null

Survey relation stores the names of all the surveys that is provided to different customers based on different requirements.

id is the Primary key.

title is the title of the survey. For example, Product Survey, Company Survey, Branch Survey. This attribute must not be null.

In addition, there can be more than one survey with the same name. Since the questions asked in the survey changes by time, we can add/delete the questions in each survey and create a new survey tuple in this table.

Questions(id, type, question, surveyid)

Constraints:

id: Primary Key, surveyid: Foreign key, type, question: not null

Questions table stores the list of all questions that are associated with any of the survey present in the survey table.

id is the primary key.

type is the kind of question and it should not be a null value. In this relation, we use only three types of questions which are objective_single_choice which means the customer has to choose only one option among the list of options, objective_multiple_choice which means customer can choose one or more option from the list of options, and subjective which means customer has to manually type the answer for the question. surveyid is the foreign key that references to a particular survey present in survey table. If a survey is deleted from the survey table, still the same question can be utilized for other surveys. Hence, surveyid is associated with 'on delete set null'.

question is the actual question that is asked to the customer in the survey. It should not be a null value.

Answer(id, text, questionid)

Constraints:

id: Primary Key, questionid: foreign key

Answer relation stores the options for every objective question present in the questions relation.

id is the Primary Key

text is the answer options for every objective question present in the questions relation. If the question in the questions relation is subjective, then corresponding tuple in the Answer relation will be filled by null. So, the text field can be null.

questionid is the foreign key attribute that references to the primary key in the questions relation. If there is no questionid, then the corresponding tuple in the answer relation must be deleted. Hence this attribute is set to 'on delete cascade'.

Responder(id, customerid, productid, surveyid)

Constraints:

id: Primary Key, customerid, productid, surveyid: Foreign Key

Responder relation stores the data about the customer who is answering various questions in different surveys. It indicates which user is responding for a survey due to the purchase of a particular product.

id is the primary key for the responder relation.

customerid is the foreign key that references to the primary key in the Customer relation.

productid is the foreign key that references to the primary key in the Product relation.

surveyid is the foreign key that references to the primary key in the Survey relation.

If a customerid or productid or customerid is deleted, still the data provided by the customer in a survey about a product the customer purchased before getting deleted can be used by Rubix for analysis. Hence, customerid, productid, and surveyid are set to 'on delete set null'.

Response(id, questionid, answerid, responsetext, responderid)

Constraints:

Id: Primary key, questionid, answerid, responderid: Foreign Key
Response is the relation that stores the data about the responses provided by customers in the survey.

id is the primary key.

questionid is the foreign key that references to the primary key in the Questions table.

answerid is the foreign key that references to the primary key in the Answer table.

responsetext is the attribute that stores the option/options chosen by the customer for an objective question and the answer typed by the customer for a subjective question in the survey. In this survey application, it is mandatory for every customer who takes the survey to provide answers to all the questions in the survey i.e; choose option/s for objective questions and input answer/feedback for subjective questions. Hence, responsetext field must not be null value.

responderid is the foreign key that references to the primary key in the Responder table which provides details about the customers who have taken the survey for a product. If the id in the responder table is deleted, still the data record can be used by the company for data analysis. Hence, this attribute is set to 'on delete set null'.

3.3 SQL Data Definition Language(DDL) Commands for creating the database according to the schema specified in 3.2

```
Create Table Company(
    id int,
    name varchar(20) not null,
    Primary key (id)
);
Create Table Product (
    id int,
    name varchar(20) not null,
    category varchar(20) not null,
    companyid int,
    primary key (id),
    foreign key (companyid) references company(id) on
    delete cascade
);
Create Table Customer (
    id int,
    name varchar(20) not null,
    age int not null,
    gender varchar(20) not null,
    branch varchar(20) not null,
    country varchar(20) not null,
    primary key (id)
);
Create Table Survey (
    id int,
    title varchar(100) not null,
    primary key (id)
);
Create Table Questions(
    id int,
    type varchar(50) not null,
    question varchar(500) not null,
    surveyid int,
    primary key(id),
```

```
    foreign key(surveyid) references Survey(id) on delete
    set null
);
Create Table Answer(
    id int,
    text varchar(1000),
    questionid int,
    primary key(id),
    foreign key(questionid) references Questions(id) on
    delete cascade
);
Create Table Responder(
    id int,
    customerid int,
    productid int,
    surveyid int,
    primary key(id),
    foreign key (customerid) references Customer(id) on
    delete set null,
    foreign key (productid) references Product(id) on delete
    set null,
    foreign key (surveyid) references Survey(id) on delete
    set null
);
Create Table Response (
    id int,
    questionid int,
    answerid int,
    responsetext varchar(1000) not null,
    responderid int,
    primary key (id),
    foreign key (questionid) references Questions(id) on
    delete set null,
    foreign key (answerid) references Answer(id) on delete
    set null,
    foreign key (responderid) references Responder(id) on
    delete set null
);
```

3.4 Data Insertion Commands with necessary data

3.4.1 Data Manipulation Language(DML) Commands with necessary data

Company Table

```
INSERT INTO Company (id, name)
VALUES (1, 'Rubix');
```

Product Table

```
INSERT INTO Product (id, name, category, companyid)
VALUES (1, 'Coconut oil', 'Grocery', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (2, 'Tomato Ketchup', 'Sauces', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (3, 'Maggi', 'Instant Food', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (4, 'Poha', 'Grocery', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (5, 'Corn Flakes', 'Instant Food', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (6, 'Bread-sticks', 'Milk Products', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (7, 'Paneer', 'Milk Products', 1);
INSERT INTO Product (id, name, category, companyid)
```

```
VALUES (8, 'pasta', 'Instant Food', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (9, 'Basmati Rice', 'Grocery', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (10, 'Choco Flakes', 'Instant Food', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (11, 'Chana Dal', 'Grocery', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (12, 'Wheat Atta', 'Food', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (13, 'Rava Idly Mix', 'Instant Food', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (14, 'Rubix Biscuits', 'Instant Food', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (15, 'Shampoo', 'Personal Care', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (16, 'Detergent Liquid', 'Cleaning', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (17, 'Sensitive toothpaste', 'Personal Care', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (18, 'Honey', 'Food', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (19, 'Peanuts', 'Grocery', 1);
INSERT INTO Product (id, name, category, companyid)
VALUES (20, 'Cheese', 'Milk Products', 1);
```

Customer Table

```
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (1, 'Alex', 25, 'Male', 'Texas branch', 'United States');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (2, 'Raju', 32, 'Male', 'London branch', 'United Kingdom');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (3, 'Mary', 48, 'Female', 'Buffalo branch', 'United States');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (4, 'Romeo', 28, 'Male', 'Delhi branch', 'India');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (5, 'Sara', 29, 'Female', 'Mumbai branch', 'India');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (6, 'Alen', 35, 'Male', 'Texas branch', 'United States');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (7, 'Sandhu', 42, 'Male', 'London branch', 'United Kingdom');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (8, 'Mira', 58, 'Female', 'Buffalo branch', 'United States');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (9, 'Rambo', 38, 'Male', 'Delhi branch', 'India');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (10, 'Sirah', 39, 'Female', 'Mumbai branch', 'India');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (11, 'Alice', 27, 'Male', 'Texas branch', 'United States');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (12, 'Saro', 35, 'Male', 'London branch', 'United Kingdom');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (13, 'Cary', 49, 'Female', 'Buffalo branch', 'United States');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (14, 'Roni', 24, 'Male', 'Delhi branch', 'India');
```

```
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (15, 'Seema', 28, 'Female', 'Mumbai branch', 'India');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (16, 'James', 45, 'Male', 'Texas branch', 'United States');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (17, 'Sanga', 37, 'Male', 'London branch', 'United Kingdom');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (18, 'Judy', 32, 'Female', 'Buffalo branch', 'United States');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (19, 'Henry', 26, 'Male', 'Delhi branch', 'India');
INSERT INTO Customer (id, name, age, gender, branch, country)
VALUES (20, 'Sama', 39, 'Female', 'Mumbai branch', 'India');
```

Survey Table

```
INSERT INTO Survey (id, title)
VALUES (1, 'Product Survey');
INSERT INTO Survey (id, title)
VALUES (2, 'Company Survey');
INSERT INTO Survey (id, title)
VALUES (3, 'Branch Survey');
INSERT INTO Survey (id, title)
VALUES (4, 'Product Survey');
INSERT INTO Survey (id, title)
VALUES (5, 'Company Survey');
INSERT INTO Survey (id, title)
VALUES (6, 'Branch Survey');
INSERT INTO Survey (id, title)
VALUES (7, 'Product Survey');
INSERT INTO Survey (id, title)
VALUES (8, 'Company Survey');
INSERT INTO Survey (id, title)
VALUES (9, 'Branch Survey');
```

Questions Table

```
INSERT INTO Questions (id, type, question, surveyid)
VALUES (1, 'objective_single_choice', 'How do you think is the price of the Product?', 1);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (2, 'objective_single_choice', 'How is the quality of the Product?', 1);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (3, 'objective_single_choice', 'How is the packaging of the Product?', 1);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (4, 'objective_single_choice', 'How often do you use the product?', 1);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (5, 'objective_single_choice', 'How do you rate the value for money of the product?', 1);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (6, 'objective_single_choice', 'How long have you been using this product?', 1);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (7, 'objective_single_choice', 'Will you purchase the product again?', 1);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (8, 'objective_multiple_choice', 'Features of the product that needs improvment?', 1);
INSERT INTO Questions (id, type, question, surveyid)
```

```
VALUES (9, 'subjective', 'What is the main benefit you receive
from this product?', 1);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (10, 'subjective', 'What are the top 3 benefits you
receive from this product?', 1);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (11, 'objective_single_choice', 'How many products of
Rubix do you use?', 2);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (12, 'objective_single_choice', 'Since how long have
you been using Rubix products?', 2);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (13, 'subjective', 'What are the main qualities of Rubix
products that you like?', 2);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (14, 'subjective', 'What are the main qualities of Rubix
products that you dont like?', 2);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (15, 'objective_single_choice', 'Do you recommend
Rubix Products to others?', 2);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (16, 'objective_multiple_choice', 'What alternative
brands would you choose when Rubix products are not
available?', 2);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (17, 'objective_single_choice', 'Was the branch you
visited nearby to your home?', 3);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (18, 'objective_single_choice', 'What is the distance of
the branch you visited from your home?', 3);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (19, 'objective_single_choice', 'How was the customer
service at the branch you visited?', 3);
INSERT INTO Questions (id, type, question, surveyid)
VALUES (20, 'subjective', 'What recommendations do you offer
to improve our products or service', 3);
```

Answer Table

```
INSERT INTO Answer (id, text, questionid)
VALUES (1, 'Low, Medium, High', 1);
INSERT INTO Answer (id, text, questionid)
VALUES (2, 'Low, Medium, High', 2);
INSERT INTO Answer (id, text, questionid)
VALUES (3, 'Low, Medium, High', 3);
INSERT INTO Answer (id, text, questionid)
VALUES (4, '1 month, 3 months, 6 months, I do not use it', 4);
INSERT INTO Answer (id, text, questionid)
VALUES (5, '1 star, 2 stars, 3 stars, 4 stars, 5 stars', 5);
INSERT INTO Answer (id, text, questionid)
VALUES (6, '1 month, 3 months, 6 months, 12 months, More
than 12 months', 6);
INSERT INTO Answer (id, text, questionid)
VALUES (7, 'Never, Probably, Sure, Not Sure', 7);
INSERT INTO Answer (id, text, questionid)
VALUES (8, 'Packaging, Taste, Freebies, Ingredients,
Availability', 8);
INSERT INTO Answer (id, text, questionid)
VALUES (9, "", 9);
INSERT INTO Answer (id, text, questionid)
VALUES (10, "", 10);
INSERT INTO Answer (id, text, questionid)
```

```
VALUES (11, 'One, One to Five, More than Five', 11);
INSERT INTO Answer (id, text, questionid)
VALUES (12, '1 month, 3 months, 6 months, 1 year, More than 1
year', 12);
INSERT INTO Answer (id, text, questionid)
VALUES (13, "", 13);
INSERT INTO Answer (id, text, questionid)
VALUES (14, "", 14);
INSERT INTO Answer (id, text, questionid)
VALUES (15, 'Yes, No', 15);
INSERT INTO Answer (id, text, questionid)
VALUES (16, 'Chubix, Zento, Zest', 16);
INSERT INTO Answer (id, text, questionid)
VALUES (17, 'Yes, No', 17);
INSERT INTO Answer (id, text, questionid)
VALUES (18, 'less than 1 mile, between 1 to 5 miles, more than 5
miles', 18);
INSERT INTO Answer (id, text, questionid)
VALUES (19, 'Not good, Good, Excellent', 19);
INSERT INTO Answer (id, text, questionid)
VALUES (20, "", 20);
```

Responder Table

```
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (1, 1, 1, 1);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (2, 1, 1, 2);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (3, 1, 1, 3);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (4, 2, 1, 1);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (5, 2, 2, 1);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (6, 2, 4, 1);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (7, 2, 11, 1);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (8, 2, 11, 2);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (9, 2, 11, 3);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (10, 5, 3, 1);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (11, 5, 4, 1);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (12, 5, 4, 2);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (13, 5, 4, 3);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (14, 7, 12, 1);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (15, 7, 13, 1);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (16, 7, 13, 2);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (17, 7, 13, 3);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (18, 20, 13, 1);
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (19, 20, 13, 2);
```

```
INSERT INTO Responder (id, customerid, productid, surveyid)
VALUES (20, 20, 13, 3);
```

Response Table

```
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (1, 1, 1, 'Medium', 1);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (2, 2, 2, 'High', 1);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (3, 3, 3, 'High', 1);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (4, 4, 4, '6 months', 1);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (5, 5, 5, '5 stars', 1);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (6, 6, 6, 'More than 12 months', 1);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (7, 7, 7, 'Sure', 1);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (8, 8, 8, 'Availability', 1);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (9, 9, 9, 'Oil is good for health', 1);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (10, 10, 10, 'Oil mixes with rice properly, good health,
low cholestrol', 1);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (11, 11, 11, 'One to Five', 2);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (12, 12, 12, '1 year', 2);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (13, 14, 14, 'Unavailability of products at all times', 2);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (14, 13, 13, 'Good packaging and quality of products',
2);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (15, 15, 15, 'Yes', 2);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (16, 16, 16, 'Zento', 2);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (17, 18, 18, 'less than 1 mile', 3);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (18, 17, 17, 'Yes', 3);
```

```
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (19, 19, 19, 'Excellent', 3);
INSERT INTO Response (id, questionid, answerid, responsetext,
responderid)
VALUES (20, 20, 20, 'Increase the number of branches and
availability of products in highly populated locations', 3);
```

3.4.2 Screenshots of the created database relations using PostgreSQL(PgAdmin) indicating the above data

Fig 1, Fig 2, Fig 3, Fig 4, Fig 5, Fig 6, Fig 7, Fig 8 depict all the relations that have been created using PgAdmin Application

Company

	id [PK] integer	name character varying (20)
1	1	Rubix

Fig 1

Product





Data Output	Explain	Messages	Notifications	
	 id [PK] integer	 name character varying (20)	 category character varying (20)	 companyid integer
1	1	Coconut oil	Grocery	1
2	2	Tomato Ketchup	Sauces	1
3	3	Maggi	Instant Food	1
4	4	Poha	Grocery	1
5	5	Corn Flakes	Instant Food	1
6	6	Bread-sticks	Milk Products	1
7	7	Paneer	Milk Products	1
8	8	pasta	Instant Food	1
9	9	Basmati Rice	Grocery	1
10	10	Choco Flakes	Instant Food	1
11	11	Chana Dal	Grocery	1
12	12	Wheat Atta	Food	1
13	13	Rava Idly Mix	Instant Food	1
14	14	Rubix Biscuits	Instant Food	1
15	15	Shampoo	Personal Care	1
16	16	Detergent Liquid	Cleaning	1
17	17	Sensitive toothpaste	Personal Care	1
18	18	Honey	Food	1
19	19	Peanuts	Grocery	1
20	20	Cheese	Milk Products	1

Fig 2

Customer

Data Output		Explain	Messages	Notifications		
	id [PK] integer	name character varying (20)	age integer	gender character varying (20)	branch character varying (20)	country character varying (20)
1	1	Alex	25	Male	Texas branch	United States
2	2	Raju	32	Male	London branch	United Kingdom
3	3	Mary	48	Female	Buffalo branch	United States
4	4	Romeo	28	Male	Delhi branch	India
5	5	Sara	29	Female	Mumbai branch	India
6	6	Allen	35	Male	Texas branch	United States
7	7	Sandhu	42	Male	London branch	United Kingdom
8	8	Mira	58	Female	Buffalo branch	United States
9	9	Rambo	38	Male	Delhi branch	India
10	10	Sirah	39	Female	Mumbai branch	India
11	11	Alice	27	Male	Texas branch	United States
12	12	Saro	35	Male	London branch	United Kingdom
13	13	Cary	49	Female	Buffalo branch	United States
14	14	Roni	24	Male	Delhi branch	India
15	15	Seema	28	Female	Mumbai branch	India
16	16	James	45	Male	Texas branch	United States
17	17	Sanga	37	Male	London branch	United Kingdom
18	18	Judy	32	Female	Buffalo branch	United States
19	19	Henry	26	Male	Delhi branch	India
20	20	Sama	39	Female	Mumbai branch	India

Fig 3

Survey

Data Output	Explain	Messages	Notifications
id [PK] integer	title character varying (100)		
1	Product Survey		
2	Company Survey		
3	Branch Survey		
4	Product Survey		
5	Branch Survey		
6	Branch Survey		
7	Product Survey		
8	Product Survey		
9	Branch Survey		
10	Product Survey		
11	Product Survey		
12	Product Survey		
13	Branch Survey		
14	Company Survey		
15	Product Survey		
16	Product Survey		
17	Company Survey		
18	Branch Survey		
19	Product Survey		
20	Branch Survey		

Fig 4

Questions

Data Output	Explain	Messages	Notifications
id [PK] integer	type character varying (50)	question character varying (500)	surveyid integer
1	objective_single_choice	How do you think is the price ...	1
2	objective_single_choice	How is the quality of the Prod...	1
3	objective_single_choice	How is the packaging of the P...	1
4	objective_single_choice	How often do you use the pro...	1
5	objective_single_choice	How do you rate the value for ...	1
6	objective_single_choice	How long have you been usin...	1
7	objective_single_choice	Will you purchase the product ...	1
8	objective_multiple_choice	Features of the product that n...	1
9	subjective	What is the main benefit you r...	1
10	subjective	What are the top 3 benefits yo...	1
11	objective_single_choice	How many products of Rubix ...	2
12	objective_single_choice	Since how long have you been...	2
13	subjective	What are the main qualities of...	2
14	subjective	What are the main qualities of...	2
15	objective_single_choice	Do you recommend Rubix Pro...	2
16	objective_multiple_choice	What alternative brands woul...	2
17	objective_single_choice	Was the branch you visited ne...	3
18	objective_single_choice	What is the distance of the br...	3
19	objective_single_choice	How was the customer servic...	3
20	subjective	What recommendations do yo...	3

Fig 5

Answer

Data Output	Explain	Messages	Notifications
id [PK] integer	text character varying (1000)	questionid integer	
1	Low, Medium, High	1	
2	Low, Medium, High	2	
3	Low, Medium, High	3	
4	1 month, 3 months, 6 months, 1 ...	4	
5	1 star, 2 stars, 3 stars, 4 stars, 5...	5	
6	1 month, 3 months, 6 months, 1...	6	
7	Never, Probably, Sure, Not Sure	7	
8	Packaging, Taste, Freebies, Ingr...	8	
9		9	
10		10	
11	One, One to Five, More than Five	11	
12	1 month, 3 months, 6 months, 1...	12	
13		13	
14		14	
15	Yes, No	15	
16	Chubix, Zento, Zest	16	
17	Yes, No	17	
18	less than 1 mile, between 1 to 5...	18	
19	Not good, Good, Excellent	19	
20		20	

Fig 6

Responder

Data Output	Explain	Messages	Notifications
id [PK] integer	customerid integer	productid integer	surveyid integer
1	1	1	1
2	2	1	2
3	3	1	3
4	4	2	1
5	5	2	1
6	6	2	4
7	7	2	11
8	8	2	11
9	9	2	11
10	10	5	3
11	11	5	4
12	12	5	4
13	13	5	4
14	14	7	12
15	15	7	13
16	16	7	13
17	17	7	13
18	18	20	13
19	19	20	13
20	20	20	13

Fig 7

Response

	Data Output	Explain	Messages	Notifications	
	id [PK] integer	questionid integer	answerid integer	responsetext character varying (1000)	responderid integer
1	1	1	1	Medium	1
2	2	2	2	High	1
3	3	3	3	High	1
4	4	4	4	6 months	1
5	5	5	5	5 stars	1
6	6	6	6	More than 12 months	1
7	7	7	7	Sure	1
8	8	8	8	Availability	1
9	9	9	9	Oil is good for health	1
10	10	10	10	Oil mixes with rice properly, go...	1
11	11	11	11	One to Five	2
12	12	12	12	1 year	2
13	13	14	14	Unavailability of products at all ...	2
14	14	13	13	Good packaging and quality of ...	2
15	15	15	15	Yes	2
16	16	16	16	Zento	2
17	17	18	18	less than 1 mile	3
18	18	17	17	Yes	3
19	19	19	19	Excellent	3
20	20	20	20	Increase the number of branch...	3

Fig 8

3.5 List of all the Functional Dependencies(FDs) for each relation in our Survey Application Database, Transformation from Initial Schema to final Schema and ensuring that all the final set of relations for this project are in Boyce-Codd Normal Form(BCNF)

3.5.1 List of all FDs for every relation in our application

Relation 1:

Company(id, name)

FDs : {id -> name}

Relation 2:

Product(id, name, category, companyid)

FDs : {id -> name, category, companyid}

Relation 3:

Customer(id, name, age, gender, branch, country)

FDs : {id -> name, age, gender, branch, country}

Relation 4:

Survey(id, title)

FDs : {id -> title}

Relation 5:

Questions(id, type, question, surveyid)

FDs : {id -> type, question, surveyid}

Relation 6:

Answer(id, text, questionid)

FDs : {id -> text, questionid}

Relation 7:

Responder(id, customerid, productid, surveyid)

FDs: {id -> customerid, productid, surveyid}

Relation 8:

Response(id, questionid, answerid, responsetext, responderid)

FDs : {id -> questionid, answerid, responsetext, responderid}

3.5.2 Transformation from Initial Schema to Final Schema and ensuring that all the final set of relations for this project are in BCNF Form

We say that a Relation R is in BCNF if for every non-trivial Function Dependency (X->Y) of R, X must be a Superkey.

Let us now analyze if the Functional Dependencies for each relation listed above satisfy BCNF property:

Relation 1:

Company(id, name)

FDs : {id -> name}

Here, id->name is a non-trivial FD since name is not contained in id. Also, id is the superkey since we can find the name of the company relation if we have id. Hence, Company Relation is in BCNF.

Relation 2:

Product(id, name, category, companyid)

FDs : {id -> name, category, companyid}

Here, id->name is a non-trivial FD since name, category, companyid are not contained in id. Also, id is the superkey since we can find all the other attributes of the Product Relation if we have id. Hence, Product Relation is in BCNF.

Relation 3:

Customer(id, name, age, gender, branch, country)

FDs : {id -> name, age, gender, branch, country}

Here, id->name, age, gender, branch, country is a non-trivial FD since name, age, gender, branch, country are not contained in id. Also, id is the superkey since we can find all the other attributes of the Customer Relation if we have id. Hence, Customer Relation is in BCNF.

Relation 4:

Survey(id, title)

FDs : {id -> title}

Here, id->title is a non-trivial FD since title is not contained in id. Also, id is the superkey since we can find title of the Survey Relation if we have id. Hence, Survey Relation is in BCNF.

Relation 5:

Questions(id, type, question, surveyid)

FDs : {id -> type, question, surveyid}

Here, id->type, question, surveyid is a non-trivial FD since type, question, surveyid are not contained in id. Also, id is the superkey since we can find all the other attributes of the Questions Relation if we have id. Hence, Questions Relation is in BCNF.

Relation 6:

Answer(id, text, questionid)

FDs : {id -> text, questionid}

Here, id->text, questionid is a non-trivial FD since text, questionid are not contained in id. Also, id is the superkey since we can find all the other attributes of the Answer Relation if we have id. Hence, Answer Relation is in BCNF.

Relation 7:

Responder(id, customerid, productid, surveyid)

FDs: {id -> customerid, productid, surveyid}

Here, id->customerid, productid, surveyid is a non-trivial FD since customerid, productid, surveyid are not contained in id. Also, id is the superkey since we can find all the other attributes of the Responder Relation if we have id. Hence, Responder Relation is in BCNF.

Relation 8:

Response(id, questionid, answerid, responsetext, responderid)

FDs : {id -> questionid, answerid, responsetext, responderid}

Here, id->questionid, answerid, responsetext, responderid is a non-trivial FD since questionid, answerid, responsetext, responderid are not contained in id. Also, id is the superkey since we can find all the other attributes of the Response Relation if we have id. Hence, Response Relation is in BCNF.

3.5.3 Final Database schema

Based on the analysis made in section 3.5.1 and section 3.5.2, the final schema with each relation in BCNF format for our Survey Application Database is as follows:

```
Create Table Company(
    id int,
    name varchar(20) not null,
    Primary key (id)
);
Create Table Product (
    id int,
    name varchar(20) not null,
    category varchar(20) not null,
    companyid int,
    primary key (id),
    foreign key (companyid) references company(id) on
delete cascade
);
Create Table Customer (
    id int,
    name varchar(20) not null,
    age int not null,
    gender varchar(20) not null,
    branch varchar(20) not null,
    country varchar(20) not null,
    primary key (id)
);
Create Table Survey (
    id int,
    title varchar(100) not null,
    primary key (id)
);
Create Table Questions(
    id int,
    type varchar(50) not null,
    question varchar(500) not null,
    surveyid int,
    primary key(id),
    foreign key(surveyid) references Survey(id) on delete
set null
);
Create Table Answer(
    id int,
```

```
text varchar(1000),
questionid int,
primary key(id),
foreign key(questionid) references Questions(id) on
delete cascade
);
Create Table Responder(
    id int,
    customerid int,
    productid int,
    surveyid int,
    primary key(id),
    foreign key (customerid) references Customer(id) on
delete set null,
    foreign key (productid) references Product(id) on delete
set null,
    foreign key (surveyid) references Survey(id) on delete
set null
);
Create Table Response (
    id int,
    questionid int,
    answerid int,
    responsetext varchar(1000) not null,
    responderid int,
    primary key (id),
    foreign key (questionid) references Questions(id) on
delete set null,
    foreign key (answerid) references Answer(id) on delete
set null,
    foreign key (responderid) references Responder(id) on
delete set null
);
```

3.6 Application of Indexing

While handling large dataset, I did not run into any problem. SQL by default adds all the primary keys to Binary trees and performs index-based search for attributes that are primary keys of a relation. No explicit indexing strategy apart from this was used during the design of the database schema.

3.7 Smaller to Larger data sample transition to the schema

The schema has been initially tested with small dataset consisting of a handful of tuples in each relation. Then, the number of tuples was gradually increased for each reach relation and design corrections like adding/removing foreign key constraints according to the requirement, increasing the attribute length, etc. have been made.

3.8 Query Execution Analysis

Here, we use PostgreSQL's EXPLAIN/EXPLAIN ANALYZE command to identify the cost of SQL queries and to find out the best practices that should be used while writing SQL queries to get better performance in terms of query execution time.

QUERY 1:

```
explain analyze select id from customer where id=1;
"Index Only Scan using customer_pkey on customer
(cost=0.15..8.17 rows=1 width=4) (actual time=0.010..0.010
rows=1 loops=1)"
```

```
" Index Cond: (id = 1)"
" Heap Fetches: 1"
"Planning Time: 0.050 ms"
"Execution Time: 0.021 ms"
```

In this query we find the id of a customer. Since id is the primary key, it will be stored as an index. After execution of the query, a query scan happens using Indexing where only the Indices will be scanned using Binary tree data structure. This way the search takes very less amount of time when compared to the search using a non-indexed attribute.

QUERY 2:

```
explain analyze select name from customer where
branch='London Branch';
"Seq Scan on customer (cost=0.00..13.75 rows=2 width=58)
(actual time=0.039..0.042 rows=4 loops=1)"
" Filter: ((branch)::text = 'London branch'::text)"
" Rows Removed by Filter: 16"
"Planning Time: 0.222 ms"
"Execution Time: 0.083 ms"
```

In this query, we find the name of all the persons in the customer table having branch at London. Here, since we searching for a non-indexed attribute, we observe that Sequential scan happens due to which the time required for execution is much more for the case of non-indexed attribute search. In the case of sequential search, search happens tuple by tuple which is another reason for increase in the execution step.

Recommendation: It is always recommended to create index for all frequently searched attributes so that the time needed for fetching the results decreases.

QUERY 3:

```
explain analyze select name from customer where branch like
'%don%';
"Seq Scan on customer (cost=0.00..13.75 rows=2 width=58)
(actual time=0.086..0.089 rows=4 loops=1)"
" Filter: ((branch)::text ~~ '%Lond%'::text)"
" Rows Removed by Filter: 16"
"Planning Time: 0.047 ms"
"Execution Time: 0.097 ms"
```

In this query, we find the name of the customer whose branch contains the string “don”. We observe that in this case as well, sequential scan happens to fetch the requested results. We also observe that the execution time in this case is even more. This is because, “don” is not a value and for each tuple the value from branch attribute is to be compared with “don” before returning the result. When there are millions of records in a table, then such a difference of a millisecond leads to drastic decrease in the performance and execution time.

Recommendation: It is never recommended to use non valued inputs like “don” with like keyword. Instead it is always recommended to create a new index for branch attribute and always search using “=” operator with a value instead of using like with “don”.

Few efficient queries for our survey application database are as follows:

-- 1. To find all the questions, questiontypes in the Product survey with id=1

```
select q.question, q.type
from questions as q, survey as survey
```

where q.surveyid=survey.id and survey.title='Product Survey' and survey.id=1;

-- 2. To find list of all customers who have purchased products

```
from Mumbai branch
select c.branch, c.name
from customer c
where c.branch = 'Mumbai branch';
```

-- 3. To find the number of customers in each branch

```
select count(name), branch
from customer
group by branch;
```

-- 4. To find all the responders for the product with name equal to Coconut oil

```
select distinct product.name, customer.name
from responder re, product product, customer customer
where product.id=re.productid and customer.id=re.customerid and
product.name='Coconut oil';
```

-- 5. To find all the question-answer/response pair for the product with name Rava Idly Mix

```
select distinct q.question, res.responsetext
from response res, responder re, questions as q
where res.questionid=q.id and
re.productid=(select id
from product prod
where prod.name='Rava Idly Mix');
```

4. CONCLUSION

In this paper, I have come up with a systematic way of developing a real-time database with database schema design for a survey application which can be utilized by a product manufacturing company. We also learned about the few unique characteristics of our database. Finally, we discuss few query optimization techniques which will produce the required results in a very efficient manner. For any database design, it is very important for the database designer to identify quick efficient ways to obtain data from the database in addition to developing a good schema.

5. FUTURE ENHANCEMENTS

The database design of the survey application which is currently applicable for one single company can be extended for multiple companies by performing few additional modifications to the schema.

6. REFERENCES

- [1] Database Systems, The Complete Book. Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom
- [2] Database System Concepts, 7th Edition. Abraham Silberschatz, Henry F. Korth, S. Sudarshan