Database Design Project

Team 1:
Austin Ellsworth
aells2@Brockport.edu
Cameron Goldberg
cgold6@Brockport.edu
Gregory Evevskey
gevev1@Brockport.edu

CIS 422
Database Design/Implementation

Implementation Report

Table of Contents

Statement of Objectives

The purpose of this project is to create an easy to use database where technicians can log the chemical names and quantities used for each customer. The database will allow generation of reports which will show total monthly billing by each technician in each zip code area. This will also include functionality to produce individual bills for each customer that will itemize the chemicals used and their price.

Description of Figures throughout report:

Figure 1 on page 3 shows the original document describing the needs for this database.
Figure 2 on page 4 shows the requirements matrix for this database.
Figure 3 on page 6 shows the ERD for this database.
Figure 4 on page 12 shows the test data that will be implemented in the database.
Figure 5 on page 12 shows the expected resulting view tables from the test data.

)UND

One of the maintenance personnel at your university has sent your instructor an e-mail message, asking for help designing a database for his business. Here's what he wrote.

TO: professor@youruniversity.edu

FROM: joe.green@youruniversity.edu

I own a small lawn care business, and I need your help designing a database. Since the start of my company in 1995, business has grown rapidly due to a core of good customers who have recommended me to friends. I have been doing all my own bookkeeping by hand, but it's becoming a difficult task with the expansion of my business. I would like to implement a computerized billing system this winter, while business is a little slow.

Here's how my business works. I have four technicians who travel to customers' homes to apply chemicals to treat lawns. Each technician has an area defined by Zip Code. The technician travels to the customer's home, inspects the lawn, and then decides which chemicals to apply. The customer is only charged for the chemicals used that day, so the bills to individual customers vary month by month. The charge for each chemical applied depends on the quantity applied. All of my lawn chemicals are charged by the gallon.

A few of my customers require that I use only organic products on their lawn. This is noted on their customer records. When technicians go to the customer's home, they know from the list whether to apply only organic products. Since the products carried on the truck are normally synthetic chemicals, I need a list of those customers who prefer "organic only," so their lawns can be treated at a separate time.

I would like a system that is easy enough for my technicians to use, so when they return to the office after a day's work, they can log the chemical names and quantities used for each customer. Then I will take that information and generate a report which will show my total monthly billing by each technician in each Zip Code area. This will help me to identify which technicians are generating the most income. I also want to be able to produce individual bills for each customer that will itemize the chemicals used and their price.

I hope you can help me set up my system.

Thank you.

Joe Green

Figure 1. Original Document Describing Project

Requirements

1. This database system must hold information for Technicians, Customers, Treatments, and Chemicals.

2. The table for technicians shall hold the technician's first name, last name, and their associated zip code.

3. The table for customers shall hold their name, address, zip code, an id, and if they want organic chemicals.

4. The chemicals table shall hold the name of the chemical, its cost per gallon, and if it is organic.

5. The treatments table will contain the customer id, chemical name, chemical quantity, if the chemical is organic, the zip code, the price corresponding to the chemical and its amount, and the data and time of the treatment.

6. The treatment and chemical table must also produce individual bills for each customer that will itemize the chemicals used and their price.

7. The technician, treatment, and chemical table will all be used to create a view that shows total monthly billing by each tech in each zip code.

	1	2	3	4	5	6
Req 1	X	X	X	X		
Req 2	X					
Req 3		X				
Req 4			X			
Req 5				X		
Req 6					X	
Req 7						X

Figure 2. Requirements Matrix

Description of E-R Diagram

For the construction of the E-R Diagram, Treatment will be represented with a diamond because it is the relationship between Technician, Customer, and Chemical. The attributes of Treatment, represented with ovals, will be Date, Time, ChemQuant, Price, CustomerId, ChemName, and Organic, with Date, Time, CustomerID, ChemName, and Organic underlined because they are the Primary Key of Treatment.

The entities Technician, Customer, and Chemical will be represented with rectangles, and they will all be joined to Treatment with arrowed lines because there is only one Technician, Customer, and Chemical per treatment.

The attributes of Technician, represented with ovals, will be ZipCode, FirstName, and LastName, with ZipCode underlined because it is the Primary Key of Technician.

The attributes of Customer, represented with ovals, will be CustomerId, FirstName, LastName, Zipcode, PrefOrganic, and Address, with CustomerId underlined because it is the Primary Key of Customer.

The attributes of Chemical, represented with ovals, will be Organic, ChemName, and CostPerGallon, with Organic and ChemName underlined because they are the Primary Key of Chemical.

E-R Diagram

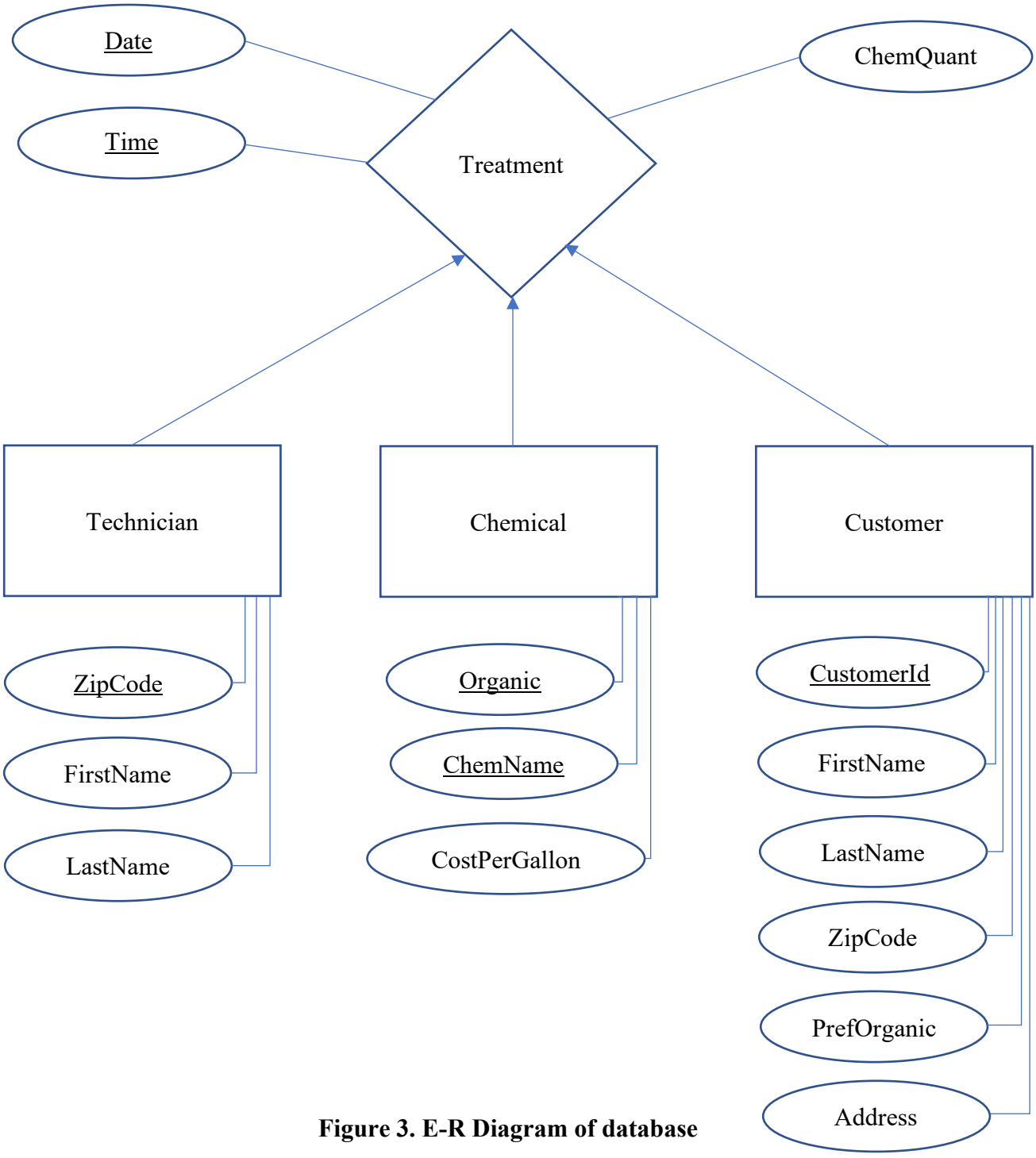

Figure 3. E-R Diagram of database

Description of Schema

The schema will consist of four parts, technician, chemical, customer and treatment.

Technician will consist of a zip code, first name and last name.

Chemical will include a true or false field describing whether it is organic or not. The chemical name and cost per gallon of a chemical is also included.

Customer will include fields for first name, last name, zip code, address, customer ID as well as a true or false field for the customers preference on organic chemicals.

The final table in the schema is Treatment, including fields such as customer ID, chemical name, quantity of the chemical, price, date of treatment, and time of treatment.

Treatment also references customer for customer ID, and it references chemical for values on chemical name and if the chemical is organic.

Together these four parts create the schema for our database.

Schema

Technician (ZipCode: STRING, FirstName: STRING, LastName: STRING)
Key: {ZipCode}

Chemical (Organic: BOOL, ChemName: STRING, CostPerGallon: DOUBLE)
Key: {Organic, ChemName}

Customer (FirstName: STRING, LastName: STRING, ZipCode: STRING, PrefOrganic: BOOL,
Address: STRING, CustomerID: STRING)
Key: {CustomerId}

Treatment (CustomerId: STRING, ChemName: STRING, ChemQuant: DOUBLE, Price DOUBLE,
Date: STRING, Time: STRING)
Key: {CustomerId, ChemName, Organic, Date, Time}
Treatment (CustomerId) references Customer
Treatment (ChemName) references Chemical
Treatment (Organic) references Chemical

Queries in Structured English:

1. Create a table for technicians with zip code, first name and last name attributes.

2. Create a table for customers with first name, last name, zip code, address, ID, and organic preference attributes.

3. Create a table for chemicals with organic, name, and cost attributes.

4. Create a table for treatments with customer id, chemical name, quantity, organic, price, date and time attributes.

5. Retrieve a table that shows customer billing; including ID, chemical name, quantity and price attributes.

6. Retrieve a table for monthly reports showing zip codes and prices.

Description of SQL

Query 1:

This query creates the Technician table, and it includes ZipCode as CHAR(5), FirstName as CHAR(20), and LastName as CHAR(20).

Technicians are sorted by zip code and only one technician operates per each zip code, making zip code a primary key

Query 2:

This query creates the Customer table, and it includes FirstName as CHAR(20), LastName as CHAR(20), ZipCode as CHAR(5), PrefOrganic as BOOLEAN, Address as CHAR(100), and CustomerId as CHAR(6).

Each customer is given a customer ID to simplify the SQL in the treatment table. Organic is true if and only if the customer prefers organic chemicals over non-organic chemicals added to their lawn. The customer ID will be the primary key for this table.

Query 3:

This query creates the Chemical table, and it includes Organic as a BOOLEAN, ChemName as CHAR(50), and CostPerGallon as DOUBLE(5,2).

Each chemical is either organic or not organic. Cost per gallon is stored as a double with 2 points after the decimal to represent a dollar amount. The primary key is organic and the chemical name because there may be a single chemical that can be made organic or not organic.

Query 4:

This query creates the Treatment table, and it includes CustomerId as CHAR(6), ChemName as CHAR(20), ChemQuant as DOUBLE(7,2), Organic as BOOLEAN, Date as CHAR(8), and Time as CHAR(6).

Each treatment will consist the chemical name and organic quality that will reference the Chemical table, chemical quantity, and a date and time of treatment. Two different chemicals being applied to the same lawn will be two separate treatments. Each treatment also needs a customer that is refenced in the customer table. Each treatment will be done by a technician from the Technician table. The primary key for this table is the customer ID, chemical name, organic quality, and the date and time.

Query 5:

This query creates a view to display how much each customer owes. This will display the price of each treatment to provide an itemized bill per each treatment for a customer.

Query 6:

This query creates a view that shows the sum of all treatments performed by a technician within a specified time period.

SQL

Query 1:

```
CREATE TABLE Technician (
        ZipCode         CHAR(5),
        FirstName       CHAR(20),
        LastName        CHAR(20),
        PRIMARY KEY(ZipCode)
)
```

Query 2:

```
CREATE TABLE Customer (
        FirstName       CHAR(20),
        LastName        CHAR(20),
        ZipCode         CHAR(5),
        PrefOrganic     BOOLEAN,
        Address         CHAR(100),
        CustomerId      CHAR(6),
        PRIMARY KEY     (CustomerId)
)
```

Query 3:

```
CREATE TABLE Chemical (
        Organic         BOOLEAN,
        ChemName        CHAR(50),
        CostPerGallon DOUBLE(5,2),
        PRIMARY KEY(Organic, ChemName)
)
```

Query 4:

```
CREATE TABLE Treatment (
        CustomerId      CHAR(6),
        ChemName        CHAR(20),
        ChemQuant       DOUBLE(7,2),
        Organic         BOOLEAN,
        Date            CHAR(8),
        Time            CHAR(5),
        PRIMARY KEY     (CustomerId, ChemName, Organic, Date, Time),
        FOREIGN KEY     (CustomerId) REFERENCES Customer,
        FOREIGN KEY     (ChemName) REFERENCES  Chemical,
        FOREIGN KEY     (Organic) REFERENCES Chemical
)
```

Query 5:

```
CREATE VIEW CustomerBilling(CustomerId, ChemName, ChemQuant, Price) AS
        SELECT      T.CustomerId, T.ChemName, T.ChemQuant,
                            T.ChemQuant * C.CostPerGallon AS Price
        FROM        Treatment T, Chemical C
        WHERE       C.ChemName = T. ChemName
```

Query 6:

```
CREATE VIEW MonthlyReport (ZipCode, Price) AS
        SELECT TE.ZipCode, SUM(T.ChemQuant * C.CostPerGallon) AS Price
        FROM Technician TE, Treatment T, Chemical C
        Where Date > USERDATA
                AND Date < USERDATA
                AND TE.ZipCode = (Select ZipCode FROM Customer CU Where
                                        CU.CustomerId = T.CustomerId)
                AND C.ChemName = T.ChemName
        GROUP BY ZipCode
```

Test Plan

Technician

ZipCode	First name	Last name
14580	Bob	Bobsmen
14607	Tim	Timson

Customer

FirstName	LastName	ZipCode	PrefOrganic	Address	CustomerId
Greg	Evevsky	14580	0	123 Fake St	111111
Cameron	Goldberg	14607	1	246 Real St	222222

Chemical

Organic	name	CostPerGallon
0	Chemical X	25.52
1	Chemical Y	11.11

Treatment

CustomerId	ChemName	ChemQuant	Organic	Date	Time
1111111	Chemical X	2.00	0	03/12/20	16:30
1111111	Chemical Y	5.00	1	03/12/20	16:50
2222222	Chemical X	2.00	0	03/13/20	12:00

Figure 4. Test Data to Be Implemented in The Database

CustomerBilling

CustomerId	ChemName	ChemQuant	Price
1111111	Chemical X	2.00	51.04
1111111	Chemical Y	5.00	55.55
2222222	Chemical X	2.00	51.04

MonthlyReport (Given the USERDATA is 03/01/20 to 03/31/20)

ZipCode	Price
14580	106.59
14607	51.04

Figure 5. Expected Resulting View Tables from Test Data

Implementation

Entering Test Values Into Technician:

INSERT INTO Technician(ZipCode,FirstName,LastName)
VALUES('14580','Bob','Bobsmen')

INSERT INTO Technician(ZipCode,FirstName,LastName)
VALUES('14607','Tim','Timson')

SELECT *
FROM Technician

SQL result

Host: localhost
Database: test
Generation Time: Apr 26, 2020 at 07:19 PM
Generated by: phpMyAdmin 4.4.15.10 / MySQL 5.5.64-MariaDB
SQL query: SELECT * FROM Technician LIMIT 0, 25 ;
Rows: 2

ZipCode	FirstName	LastName
14580	Bob	Bobsmen
14607	Tim	Timson

Entering Test Values Into Customer:

INSERT INTO Customer(FirstName,LastName,ZipCode,PrefOrganic,Address,CustomerId)
VALUES('Gregory','Evevsky','14580',FALSE,'123 Fake Street','111111')

INSERT INTO Customer(FirstName,LastName,ZipCode,PrefOrganic,Address,CustomerId)
VALUES('Cameron','Goldberg','14607',TRUE,'246 Real Street','222222')

SELECT *
FROM Customer

SQL result

Host: localhost
Database: test
Generation Time: Apr 26, 2020 at 07:28 PM
Generated by: phpMyAdmin 4.4.15.10 / MySQL 5.5.64-MariaDB
SQL query: SELECT * FROM `Customer` LIMIT 0, 25 ;
Rows: 2

FirstName	LastName	ZipCode	PrefOrganic	Address	CustomerId
Greg	Evevsky	14580	0	123 Fake Street	111111
Cameron	Goldberg	14607	1	246 Real Street	222222

Entering Test Values Into Chemical

```
INSERT INTO Chemical(Organic,ChemName,CostPerGallon)
VALUES(FALSE,'Chemical X', 25.52)

INSERT INTO Chemical(Organic,ChemName,CostPerGallon)
VALUES(TRUE,'Chemical Y', 11.11)

SELECT *
FROM Chemical
```

SQL result

Host: localhost
Database: test
Generation Time: Apr 26, 2020 at 07:33 PM
Generated by: phpMyAdmin 4.4.15.10 / MySQL 5.5.64-MariaDB
SQL query: SELECT * FROM Chemical LIMIT 0, 25 ;
Rows: 2

Organic	ChemName	CostPerGallon
0	Chemical X	25.52
1	Chemical Y	11.11

Entering Test Values Into Treatment:

```
INSERT INTO Treatment(CustomerId,ChemName,ChemQuant,Organic,Date,Time)
VALUES('111111','Chemical X',2,FALSE,'3/12/2020','16:30')

INSERT INTO Treatment(CustomerId,ChemName,ChemQuant,Organic,Date,Time)
VALUES('111111','Chemical Y',5,TRUE,'3/12/2020','16:50')

INSERT INTO Treatment(CustomerId,ChemName,ChemQuant,Organic,Date,Time)
VALUES('222222','Chemical X',2,FALSE,'3/13/2020','12:00')

SELECT *
FROM Treatment
```

SQL result

Host: localhost
Database: test
Generation Time: Apr 26, 2020 at 07:54 PM
Generated by: phpMyAdmin 4.4.15.10 / MySQL 5.5.64-MariaDB
SQL query: SELECT * FROM Treatment LIMIT 0, 25 ;
Rows: 3

CustomerId	ChemName	ChemQuant	Organic	Date	Time
111111	Chemical X	2.00	0	03/12/20	16:30
111111	Chemical Y	5.00	1	03/12/20	16:50
222222	Chemical X	2.00	0	03/13/20	12:00

Creating View For CustomerBilling:

SQL Result

Host: localhost
Database: test
Generation Time: Apr 27, 2020 at 09:24 PM
Generated by: phpMyAdmin 4.4.15.10 / MySQL 5.5.64-MariaDB
SQL query: SELECT * FROM `CustomerBilling` LIMIT 0, 25 ;
Rows: 3

CustomerId	ChemName	ChemQuant	Price
111111	Chemical X	2.00	51.04
111111	Chemical Y	5.00	55.55
222222	Chemical X	2.00	51.04

Creating View For MonthlyReport:

SQL Result

Host: localhost
Database: test
Generation Time: Apr 27, 2020 at 09:43 PM
Generated by: phpMyAdmin 4.4.15.10 / MySQL 5.5.64-MariaDB
SQL query: SELECT * FROM `MonthlyReport` LIMIT 0, 25 ;
Rows: 2

ZipCode	Price
14580	106.59
14607	51.04

Conclusion of Results

Following our design, we were able to successfully complete our goal of creating a database where technicians can log the chemical names and quantities used for each customer. We were also successful in creating an itemized customer billing report and a monthly report for technicians based on the data entered for lawn treatments. During our implementation, we did find some small parts of the design that needed to be altered or corrected in order to more closely fit the requirements of this project.

For instance, the attribute of Price was removed from the Treatment table. It was easier to calculate this within the VIEW tables for each report. We also found that we neglected to place the Organic attribute within our original SQL statements, and that the domain for CustomerId had to be changed to CHAR(6) to match the Customer table. Another change to the Treatment table was making Time CHAR(5) rather than CHAR(6) because there are only five characters needed to display the time. The final change for treatment is that we removed the attributes CustomerId, ChemName, and Organic from the ERD because they are attributes that come from the relationship, and they are not actual attributes of Treatment. The only other modification to our tables was changing ZipCode in Customer to CHAR(5) because only five characters are needed, and this is consistent with our Technician table.

Within our CustomerBilling VIEW we are calculating the Price attribute since it was removed from Treatment, and the WHERE statement has been changed to C.ChemName = T.ChemName in order to list things properly. The final changes were for the MonthlyReport VIEW. These changes were made to calculate the price within this view and to sum all prices within the time period that was specified. We also made several adjustments to the WHERE statement and added a GROUP BY statement in order to display things correctly. The alterations to the MonthlyReport are reflected in our test plan as well. All of the changes mentioned above are shown in the Change History section below.

All in all, this project came together nicely. By starting with a design using project requirements, an ERD, schema, and structured English we were able to develop SQL statements and a test plan to ensure the implementation of our database. During the implementation phase we did find some items to be corrected, but our initial design made this a smooth process. This project has been a great application of our newly found database design and querying skills, and, going forward, I believe we will find that this will have given us a solid foundation for future database applications.

Change History

Queries 2,4,5, and 6 have been modified. They were:

Query 2:

```
CREATE TABLE Customer (
        FirstName       CHAR(20),
        LastName        CHAR(20),
        ZipCode         CHAR(20),
        PrefOrganic     BOOLEAN,
        Address         CHAR(100),
        CustomerId      CHAR(6),
        PRIMARY KEY         (CustomerId)
)
```

Query 4:

```
CREATE TABLE Treatment (
        CustomerId   CHAR(20),
        ChemName     CHAR(20),
        ChemQuant    (DOUBLE(7,2)),
        Price        (DOUBLE(7,2))
                        DEFAULT = ChemQuant * SELECT CostPerGallon IN Chemical
                                WHERE ChemName = ChemName,
        Date         CHAR(8),
        Time         CHAR(6),
        PRIMARY KEY      (CustomerId, ChemName, Organic, Date, Time)
        FOREIGN KEY (CustomerId) REFERENCES Customer,
        FOREIGN KEY (ChemName) REFERENCES  Chemical
)
```

Query 5:

```
CREATE VIEW CustomerBilling(CustomerId, ChemName, ChemQuant, Price) AS
        SELECT      CustomerId, ChemName, ChemQuant, Price
        FROM        Treatment
        WHERE       EXIST
```

Query 6:

```
CREATE VIEW MonthlyReport (ZipCode, Price) AS
        SELECT TE.ZipCode, T.Price
        FROM Technician TE, Treatment T
        Where Date > USERDATA AND Date < USERDATA
```

The removal of the Price attribute is also reflected in the ERD.
CustomerId, ChemName, and Organic were removed from Treatment in the ERD as well.

The test plan for MonthlyReport was changed to reflect design requirements. It was:

ZipCode	Price
14580	51.04
14580	55.55
14607	51.04

All modifications can be seen in their respective location of the report.