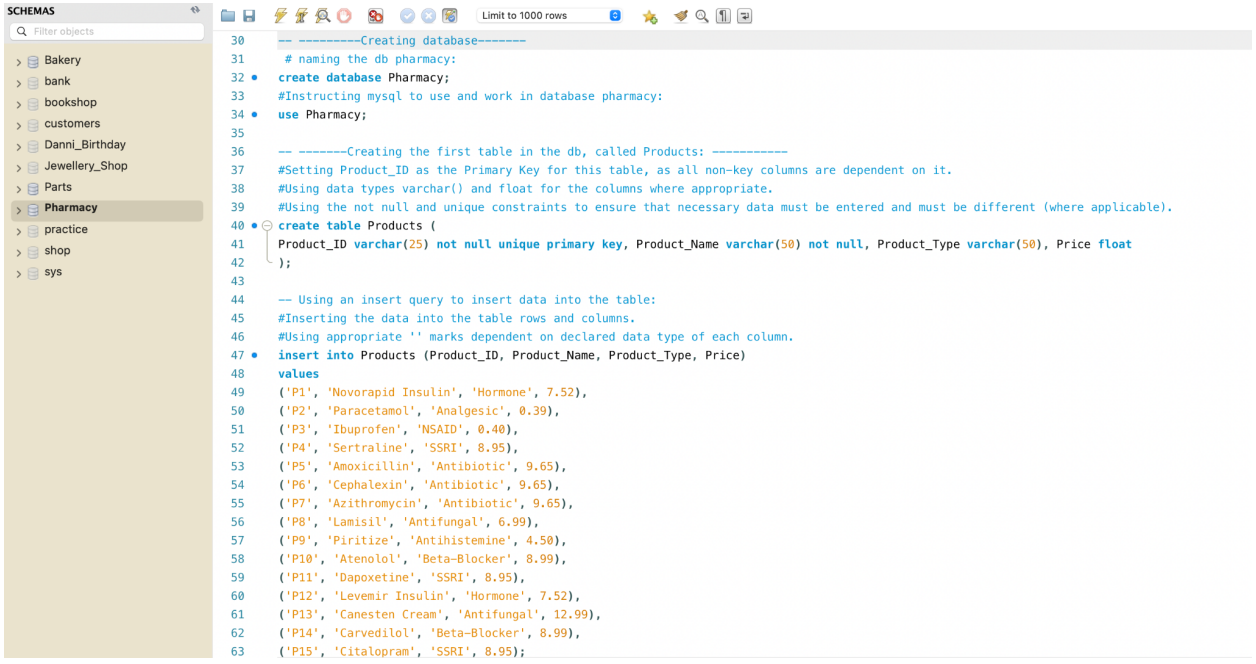
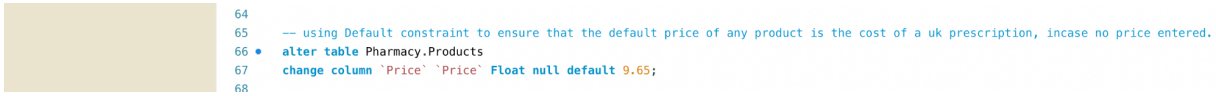
Scenario:

I am creating a database for a chemist / pharmacy. The database contains product, customer and doctors surgery information, with queries, in-built functions and stored procedures that enable pharmacy staff to easily access information.
The product information will populate 2 tables, one with basic product info called 'Products' and a second table with medicinal product info called 'Product_Specs'.
The 'Products' table has a default value for the 'Price' column. This is set to the cost of a UK prescription, so that if no price is entered this will autopopulate.
The 'Product_Specs' table has easily accessible saftey / medical information in case a customer has a query or a product is prescribed to someone of the wrong age, etc.
The database will also have a stock table. This will show the quantity of each unit of product in the pharmacy. The stock table will also contain delivery time in days for each product when the pharmacy requests it from the manufacturers.
The customer info is also split into 2 tables, one with relevant customer information called 'Customers' and another with the customer contact details called 'Customers_Contact'.
The 'Customer' table has the relevant information that the pharmacy needs for issuing prescriptions, such as customer age and any existing medical conditions that may effect the medication they are prescribed or buying.
The Doctor's Surgery tables are split into 2 tables, one providing a Surgery_ID to each practice to maintain db normalisation, and the second table contains the contact information for each surgery and is called 'Surgery_Info'.
There is a final table called 'Cust_Surgery', which contains a list of which customers are registered to which surgery. This way if there is ever an issue with a prescription for a patient, the pharmacy can easily query which Drs surgery to contact.

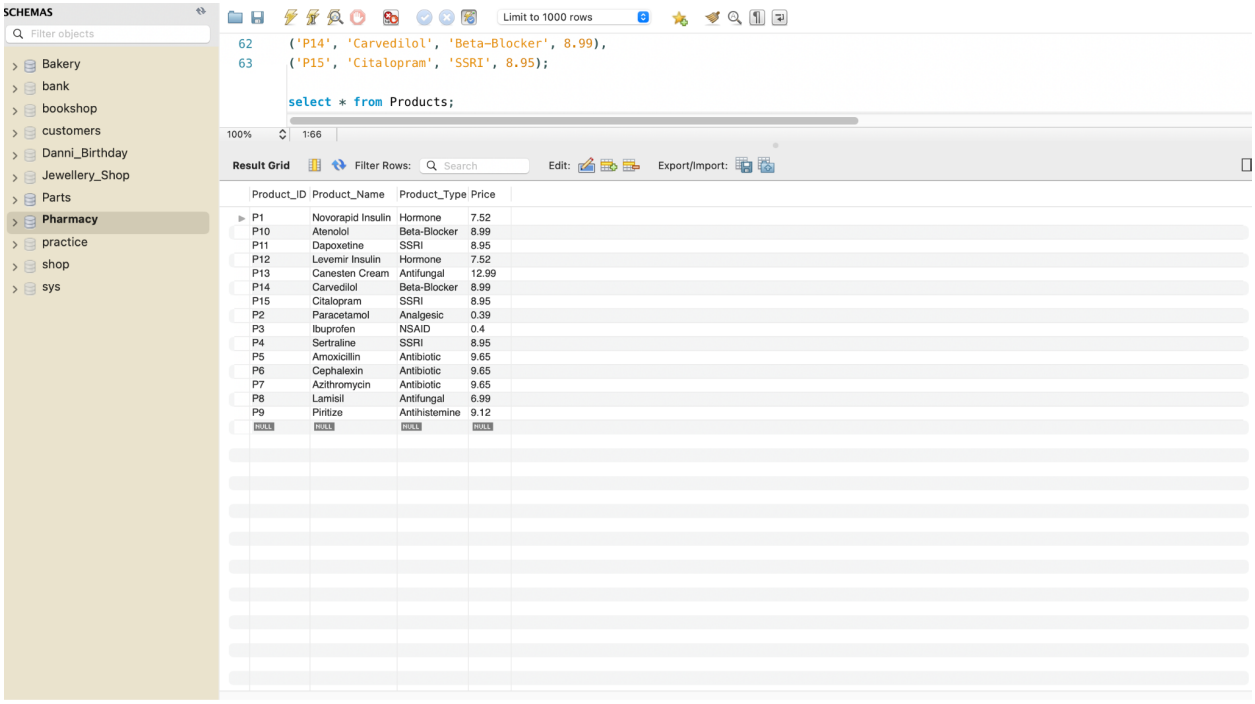
Creating Pharmacy Database and tables:

Creating first table:

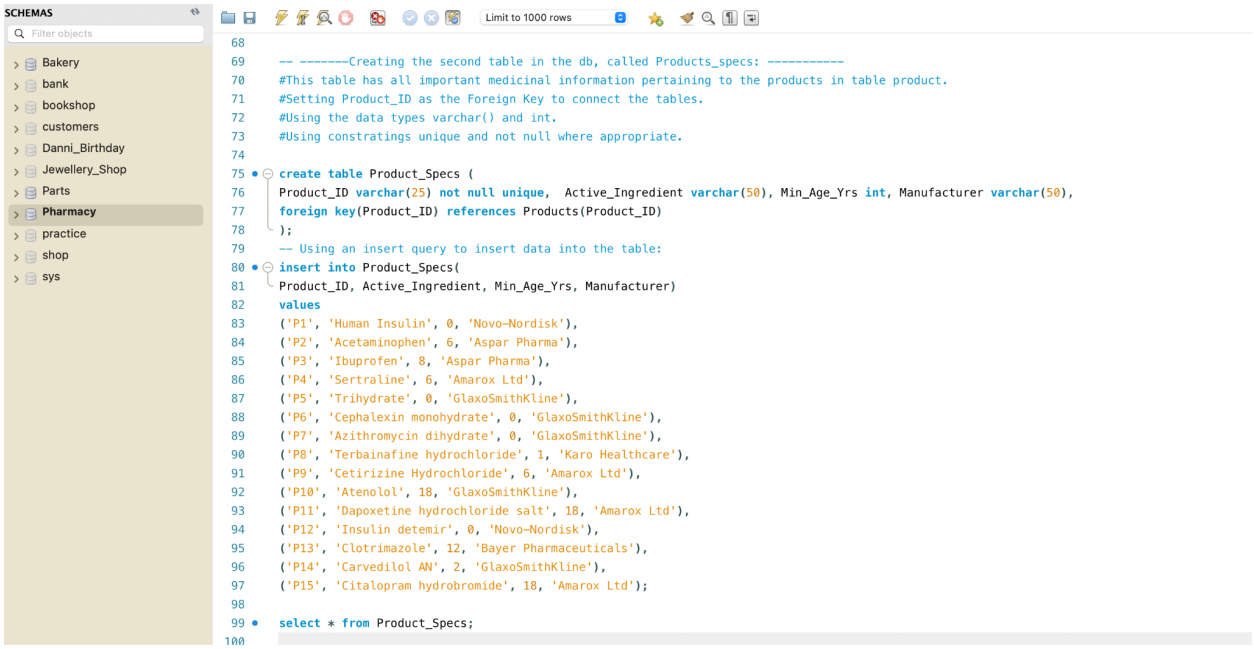




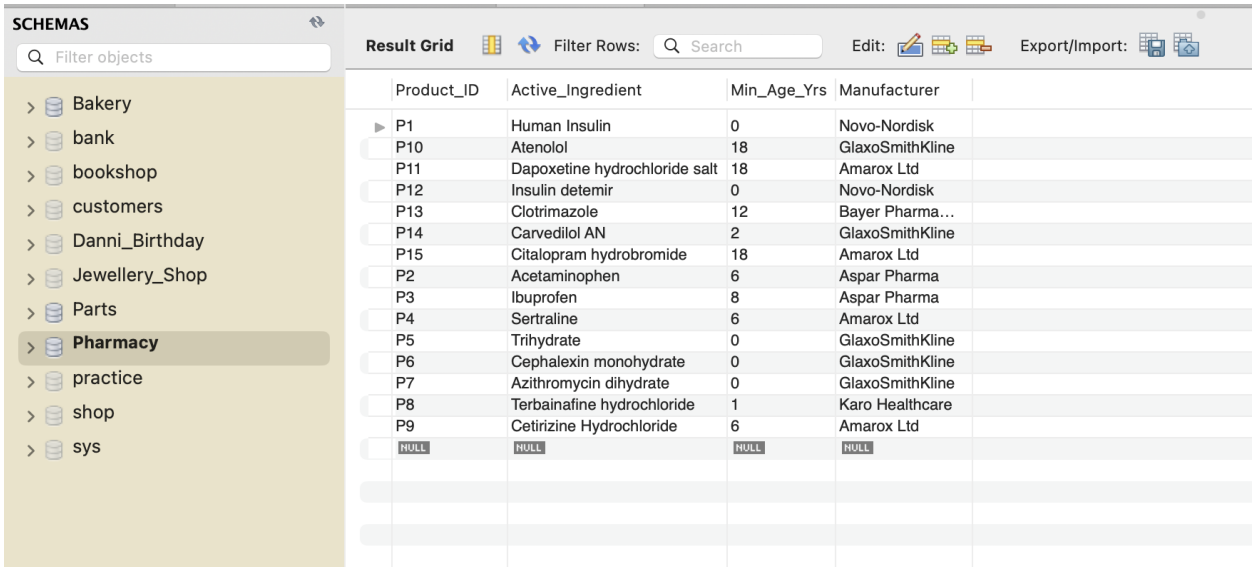
Output:



Creating second table, ‘Product_Specs’:

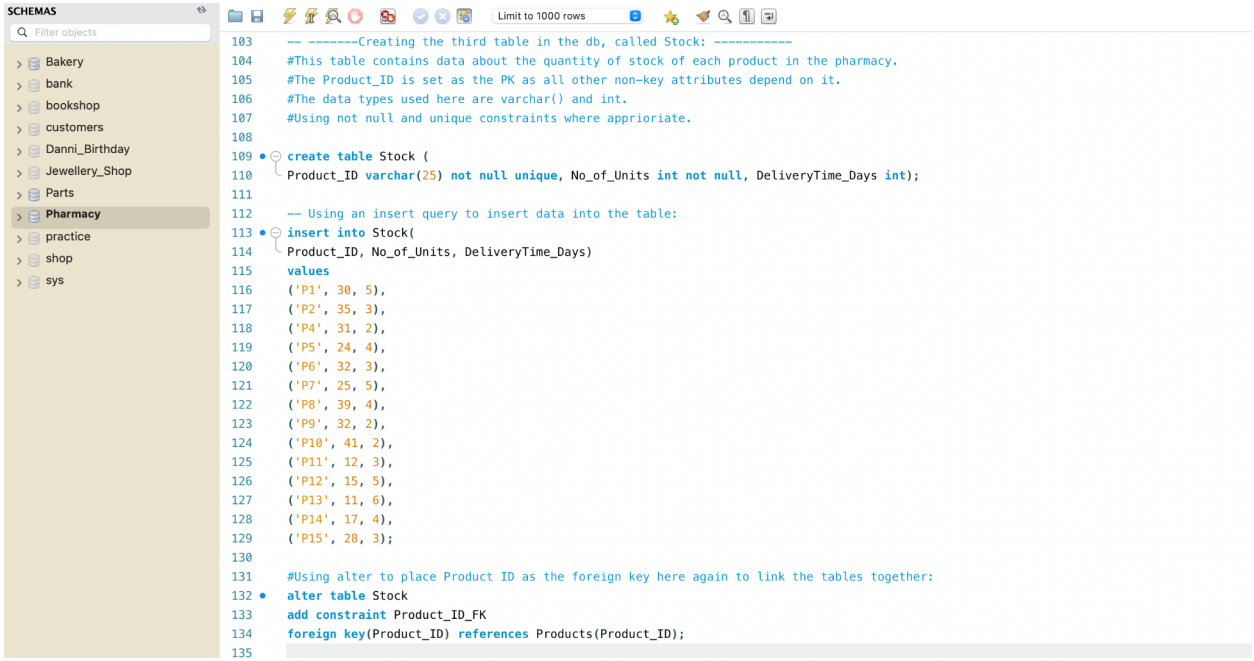


Output:



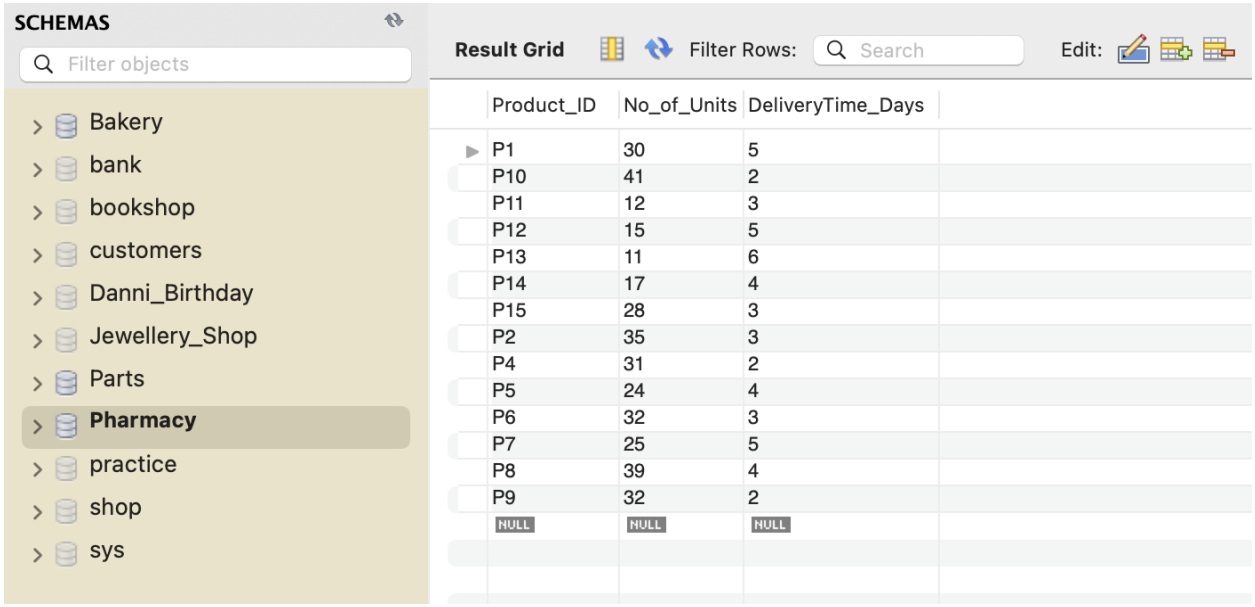
Product_ID	Active_Ingredient	Min_Age_Yrs	Manufacturer
P1	Human Insulin	0	Novo-Nordisk
P10	Atenolol	18	GlaxoSmithKline
P11	Dapoxetine hydrochloride salt	18	Amarox Ltd
P12	Insulin detemir	0	Novo-Nordisk
P13	Clotrimazole	12	Bayer Pharma…
P14	Carvedilol AN	2	GlaxoSmithKline
P15	Citalopram hydrobromide	18	Amarox Ltd
P2	Acetaminophen	6	Aspar Pharma
P3	Ibuprofen	8	Aspar Pharma
P4	Sertraline	6	Amarox Ltd
P5	Trihydrate	0	GlaxoSmithKline
P6	Cephalexin monohydrate	0	GlaxoSmithKline
P7	Azithromycin dihydrate	0	GlaxoSmithKline
P8	Terbainafine hydrochloride	1	Karo Healthcare
P9	Cetirizine Hydrochloride	6	Amarox Ltd
NULL	NULL	NULL	NULL

Creating third table, ‘Stock’:

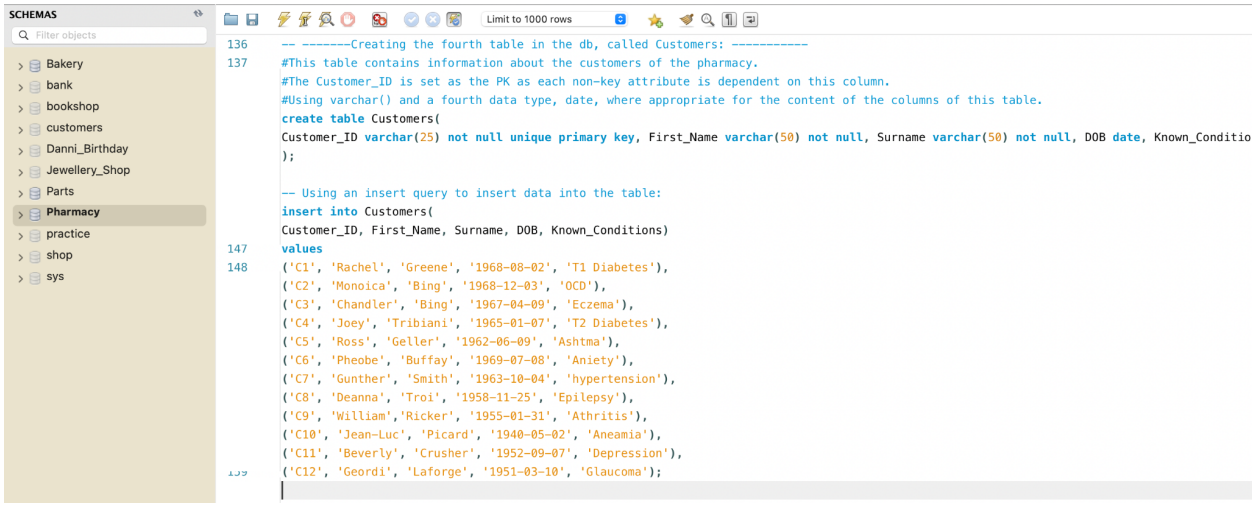



```
-- -------Creating the third table in the db, called Stock: ----------
#This table contains data about the quantity of stock of each product in the pharmacy.
#The Product_ID is set as the PK as all other non-key attributes depend on it.
#The data types used here are varchar() and int.
#Using not null and unique constraints where appriopriate.

create table Stock (
Product_ID varchar(25) not null unique, No_of_Units int not null, DeliveryTime_Days int);

-- Using an insert query to insert data into the table:
insert into Stock(
Product_ID, No_of_Units, DeliveryTime_Days)
values
('P1', 30, 5),
('P2', 35, 3),
('P4', 31, 2),
('P5', 24, 4),
('P6', 32, 3),
('P7', 25, 5),
('P8', 39, 4),
('P9', 32, 2),
('P10', 41, 2),
('P11', 12, 3),
('P12', 15, 5),
('P13', 11, 6),
('P14', 17, 4),
('P15', 28, 3);

#Using alter to place Product ID as the foreign key here again to link the tables together:
alter table Stock
add constraint Product_ID_FK
foreign key(Product_ID) references Products(Product_ID);
```

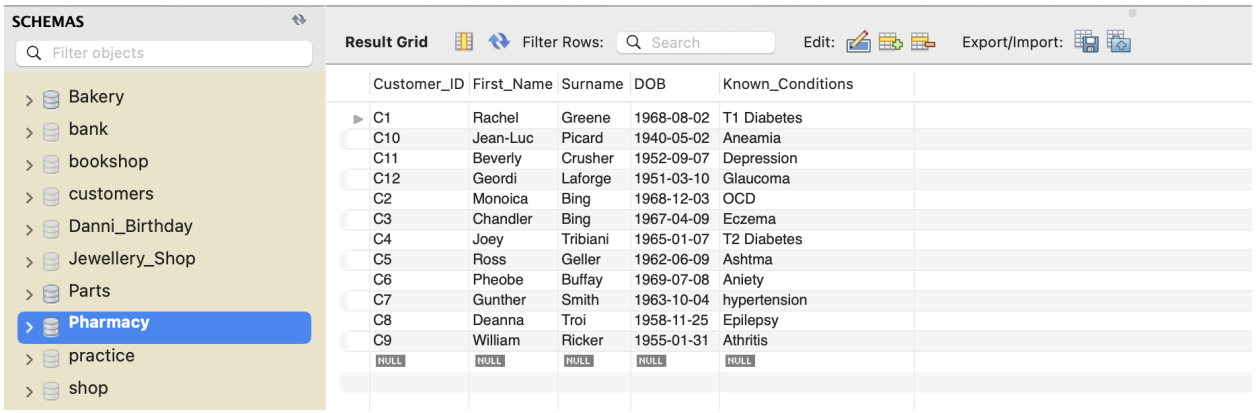
Output:



Creating fourth table, ‘Customers’:

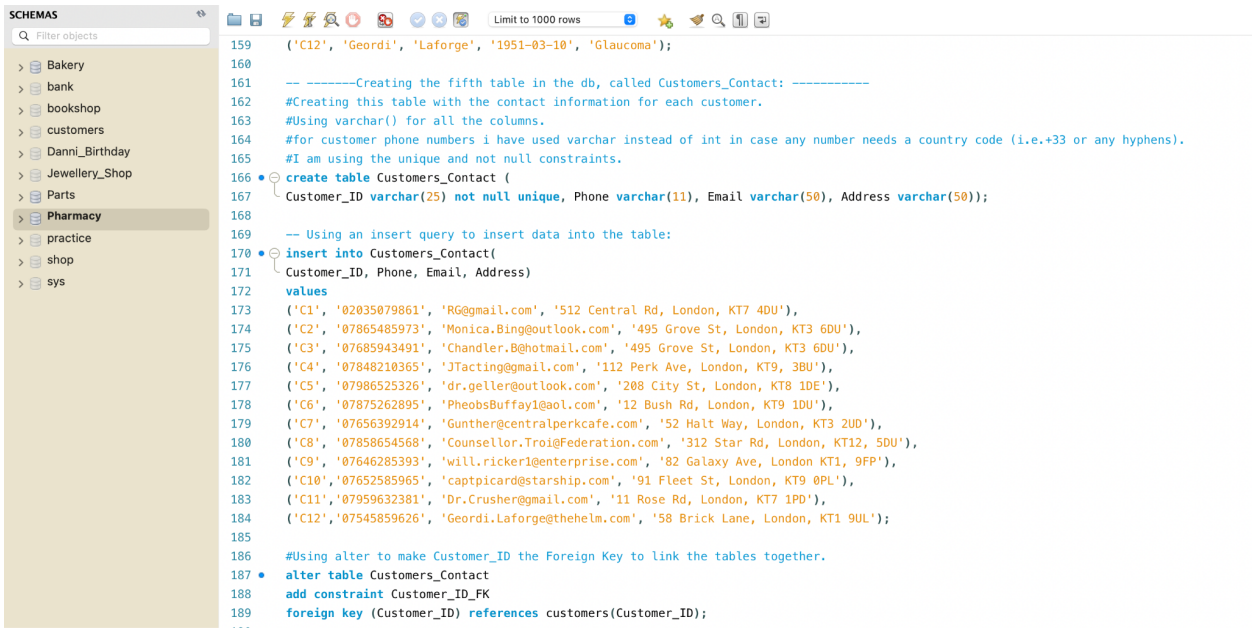


Output:

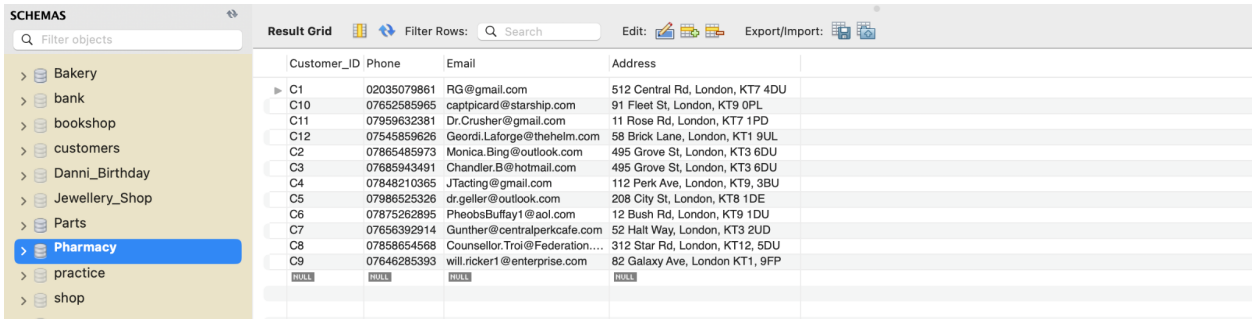


Customer_ID	First_Name	Surname	DOB	Known_Conditions
C1	Rachel	Greene	1968-08-02	T1 Diabetes
C10	Jean-Luc	Picard	1940-05-02	Aneamia
C11	Beverly	Crusher	1952-09-07	Depression
C12	Geordi	Laforge	1951-03-10	Glaucoma
C2	Monoica	Bing	1968-12-03	OCD
C3	Chandler	Bing	1967-04-09	Eczema
C4	Joey	Tribiani	1965-01-07	T2 Diabetes
C5	Ross	Geller	1962-06-09	Ashtma
C6	Pheobe	Buffay	1969-07-08	Aniety
C7	Gunther	Smith	1963-10-04	hypertension
C8	Deanna	Troi	1958-11-25	Epilepsy
C9	William	Ricker	1955-01-31	Athritis
NULL	NULL	NULL	NULL	NULL

Creating fifth table, ‘Customers_Contact’:

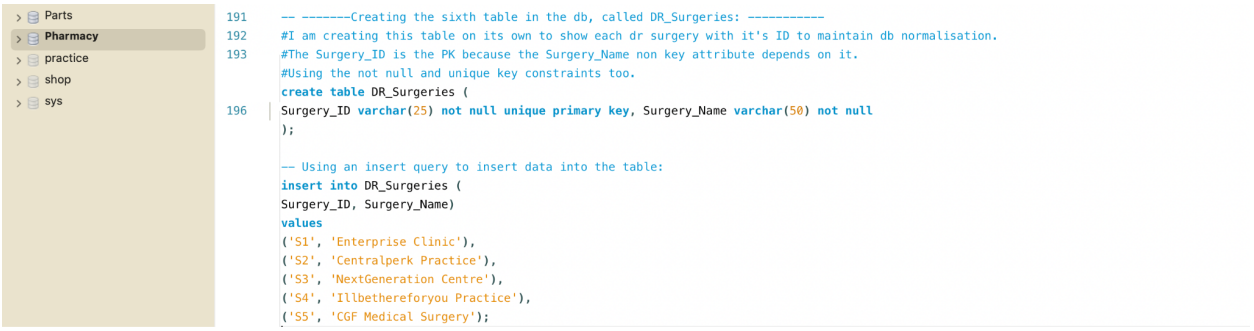


Output:



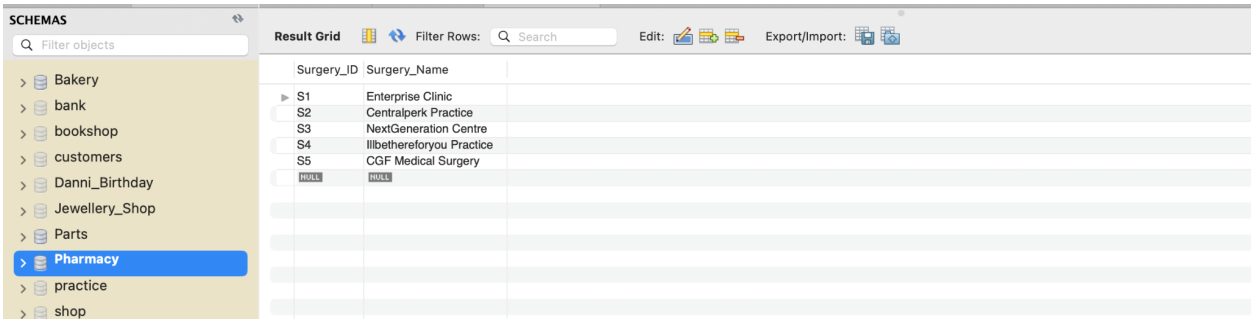
Customer_ID	Phone	Email	Address
C1	02035079861	RG@gmail.com	512 Central Rd, London, KT7 4DU
C10	07652585965	captpicard@starship.com	91 Fleet St, London, KT9 0PL
C11	07959632381	Dr.Crusher@gmail.com	11 Rose Rd, London, KT7 1PD
C12	07545859626	Geordi.Laforge@thehelm.com	58 Brick Lane, London, KT1 9UL
C2	07865485973	Monica.Bing@outlook.com	495 Grove St, London, KT3 6DU
C3	07685943491	Chandler.B@hotmail.com	495 Grove St, London, KT3 6DU
C4	07848210365	JTacting@gmail.com	112 Perk Ave, London, KT9, 3BU
C5	07986525326	dr.geller@outlook.com	208 City St, London, KT8 1DE
C6	07875262895	PheobsBuffay1@aol.com	12 Bush Rd, London, KT9 1DU
C7	07656392914	Gunther@centralperkcafe.com	52 Halt Way, London, KT3 2UD
C8	07858654568	Counsellor.Troi@Federation....	312 Star Rd, London, KT12, 5DU
C9	07646285393	will.ricker1@enterprise.com	82 Galaxy Ave, London KT1, 9FP
NULL	NULL	NULL	NULL

Creating sixth table, ‘DR_Surgeries’:



```
-- -------Creating the sixth table in the db, called DR_Surgeries: -----------
#I am creating this table on its own to show each dr surgery with it's ID to maintain db normalisation.
#The Surgery_ID is the PK because the Surgery_Name non key attribute depends on it.
#Using the not null and unique key constraints too.
create table DR_Surgeries (
Surgery_ID varchar(25) not null unique primary key, Surgery_Name varchar(50) not null
);

-- Using an insert query to insert data into the table:
insert into DR_Surgeries (
Surgery_ID, Surgery_Name)
values
('S1', 'Enterprise Clinic'),
('S2', 'Centralperk Practice'),
('S3', 'NextGeneration Centre'),
('S4', 'Illbethereforyou Practice'),
('S5', 'CGF Medical Surgery');
```

Output:



SCHEMAS

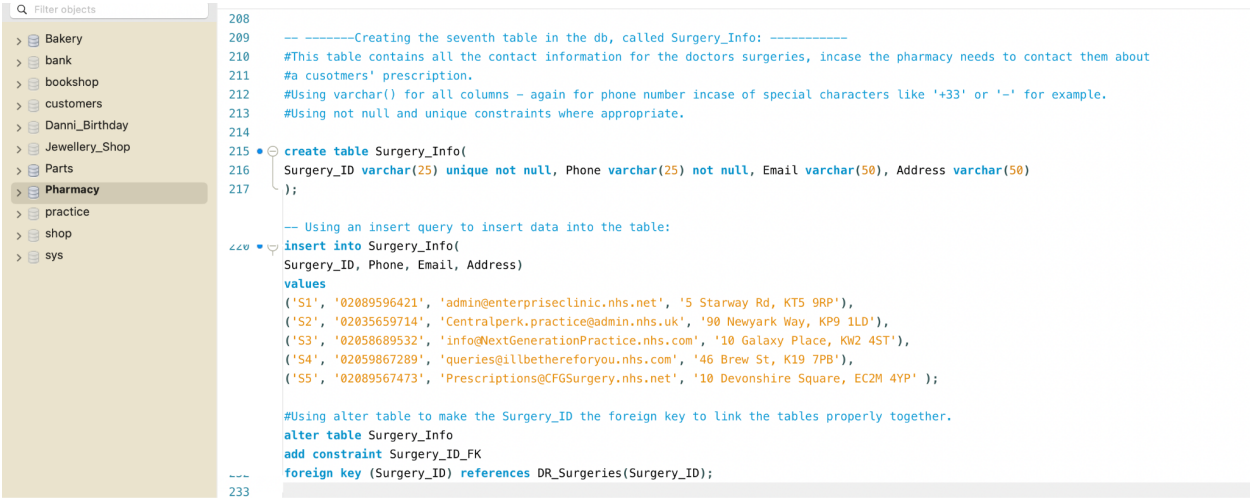
Filter objects

- Bakery
- bank
- bookshop
- customers
- Danni_Birthday
- Jewellery_Shop
- Parts
- Pharmacy
- practice
- shop

Result Grid | Filter Rows: Search | Edit: | Export/Import:

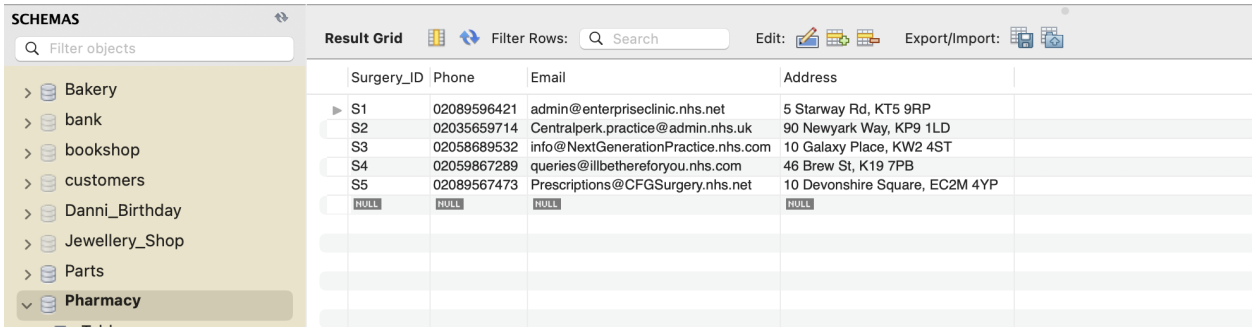
Surgery_ID	Surgery_Name
S1	Enterprise Clinic
S2	Centralperk Practice
S3	NextGeneration Centre
S4	Illbethereforyou Practice
S5	CGF Medical Surgery
NULL	NULL

Creating seventh table, ‘Surgery_Info’:

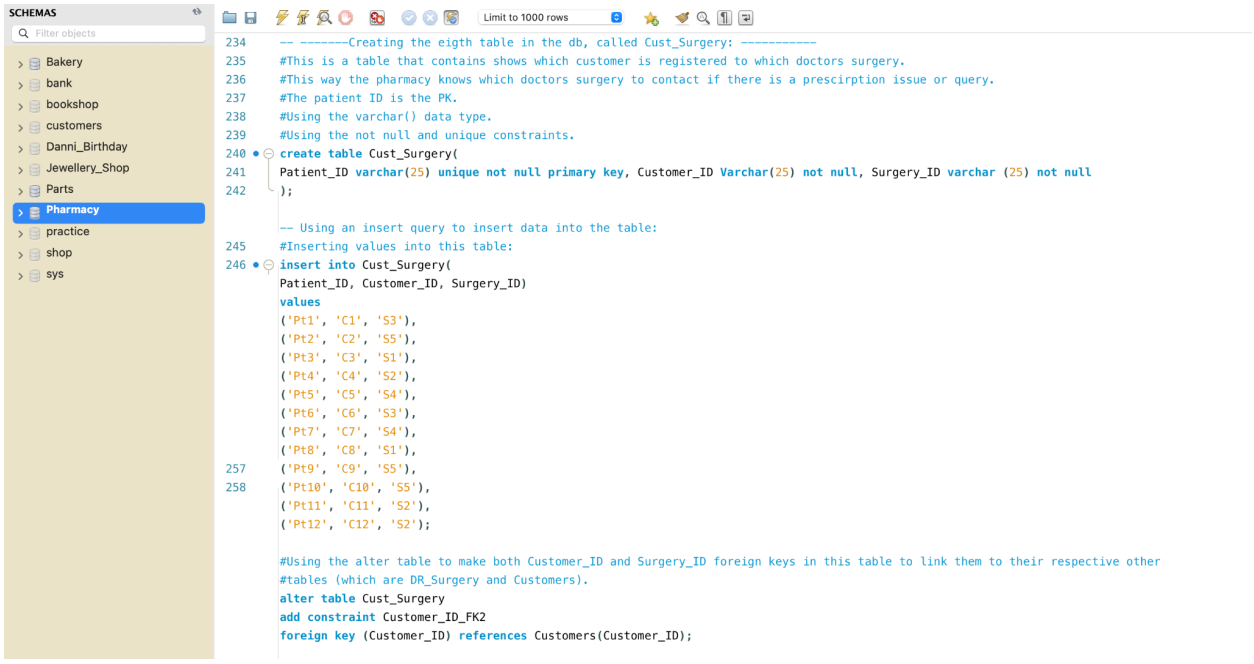


```
-- -------Creating the seventh table in the db, called Surgery_Info: -----------
#This table contains all the contact information for the doctors surgeries, incase the pharmacy needs to contact them about
#a cusotmers' prescription.
#Using varchar() for all columns - again for phone number incase of special characters like '+33' or '-' for example.
#Using not null and unique constraints where appropriate.

create table Surgery_Info(
Surgery_ID varchar(25) unique not null, Phone varchar(25) not null, Email varchar(50), Address varchar(50)
);

-- Using an insert query to insert data into the table:
insert into Surgery_Info(
Surgery_ID, Phone, Email, Address)
values
('S1', '02089596421', 'admin@enterpriseclinic.nhs.net', '5 Starway Rd, KT5 9RP'),
('S2', '02035659714', 'Centralperk.practice@admin.nhs.uk', '90 Newyark Way, KP9 1LD'),
('S3', '02058689532', 'info@NextGenerationPractice.nhs.com', '10 Galaxy Place, KW2 4ST'),
('S4', '02059867289', 'queries@illbethereforyou.nhs.com', '46 Brew St, K19 7PB'),
('S5', '02089567473', 'Prescriptions@CFGSurgery.nhs.net', '10 Devonshire Square, EC2M 4YP' );

#Using alter table to make the Surgery_ID the foreign key to link the tables properly together.
alter table Surgery_Info
add constraint Surgery_ID_FK
foreign key (Surgery_ID) references DR_Surgeries(Surgery_ID);
```

Output:

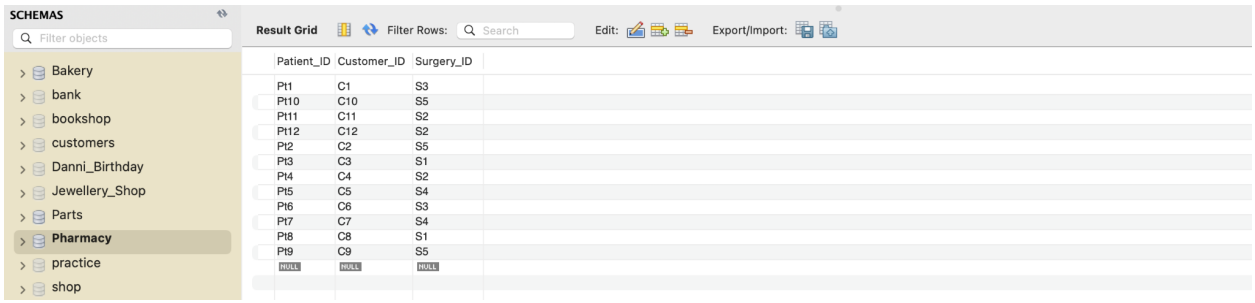


Creating eighth table, ‘Cust_Surgery’;



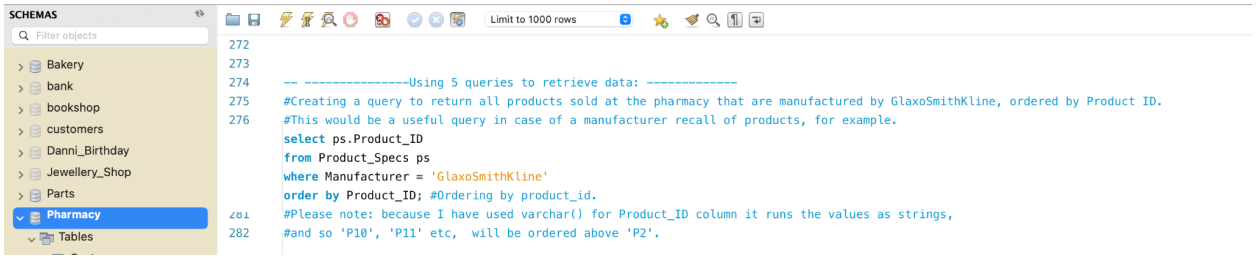

```
alter table Cust_Surgery
269 add constraint Surgery_ID_FK2
foreign key(Surgery_ID) references DR_Surgeries(Surgery_ID);
```

Output:

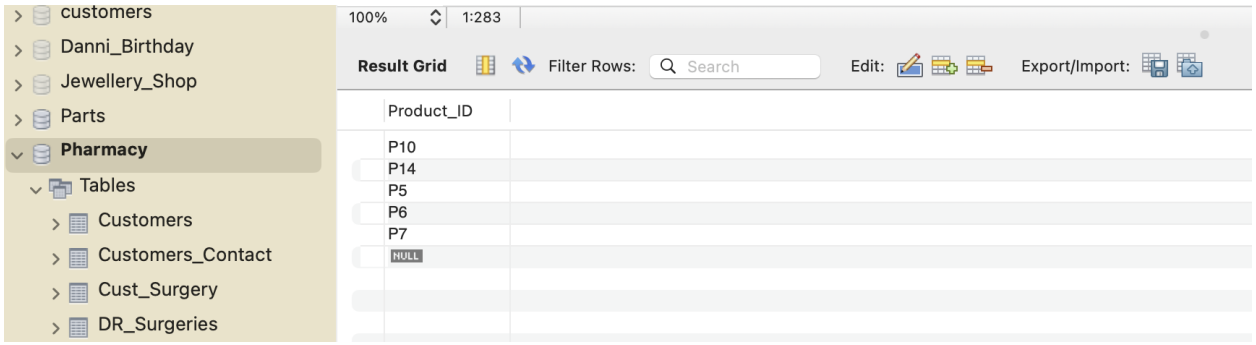


Using queries to retrieve data:

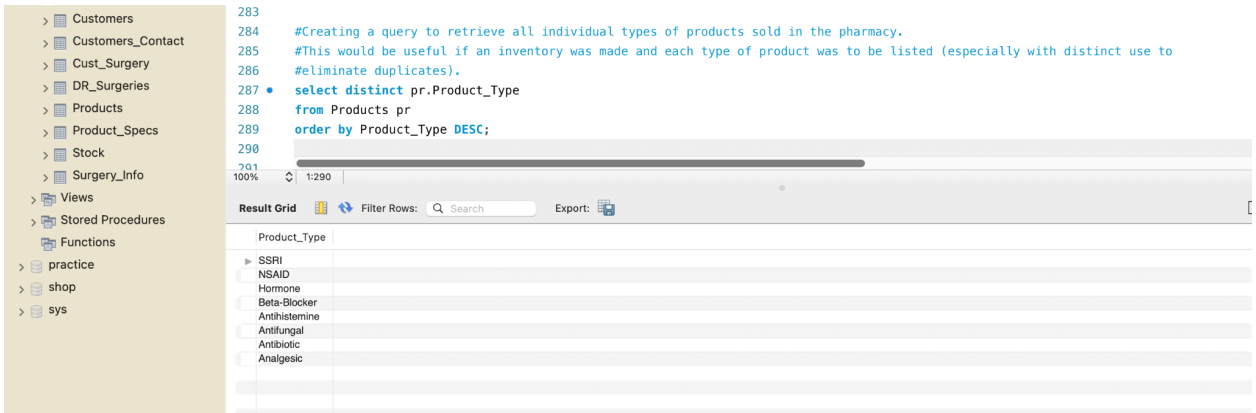
Query1 code:



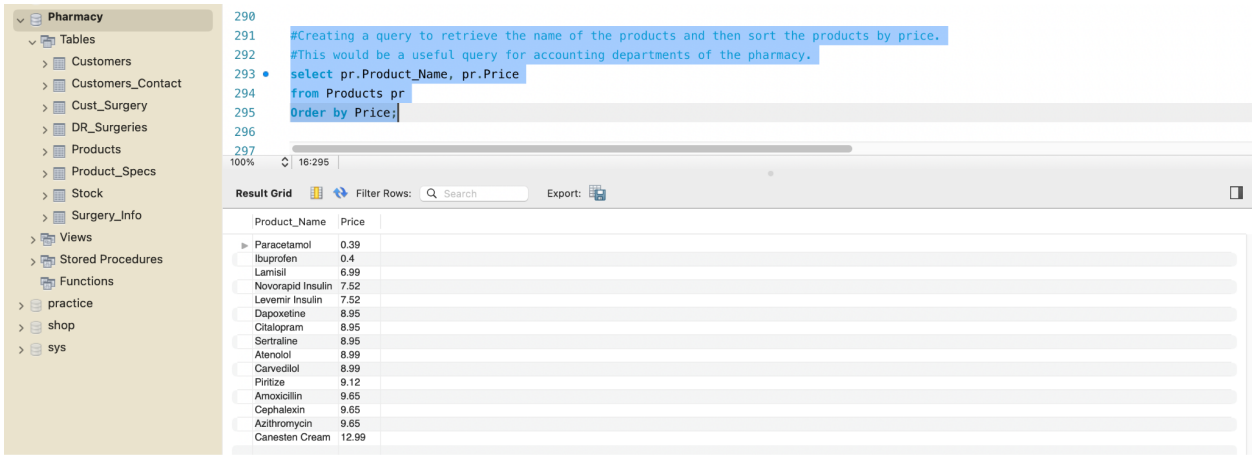
Output:



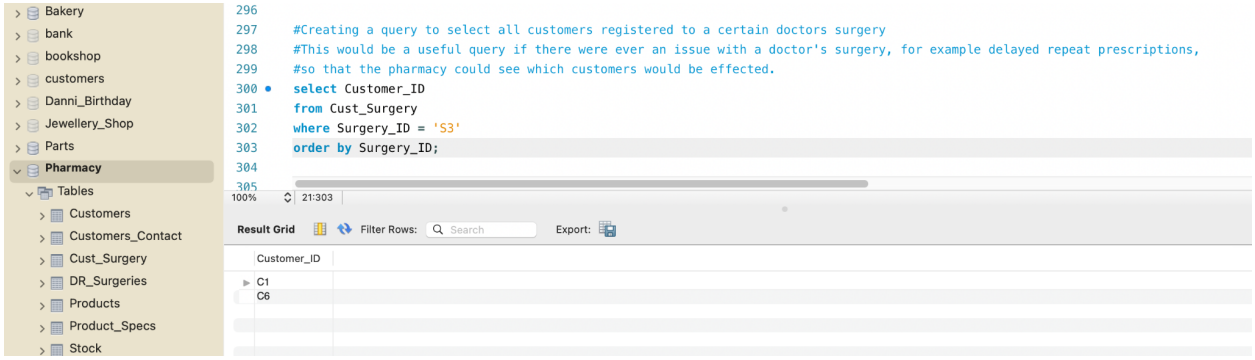
Query 2 code and output:



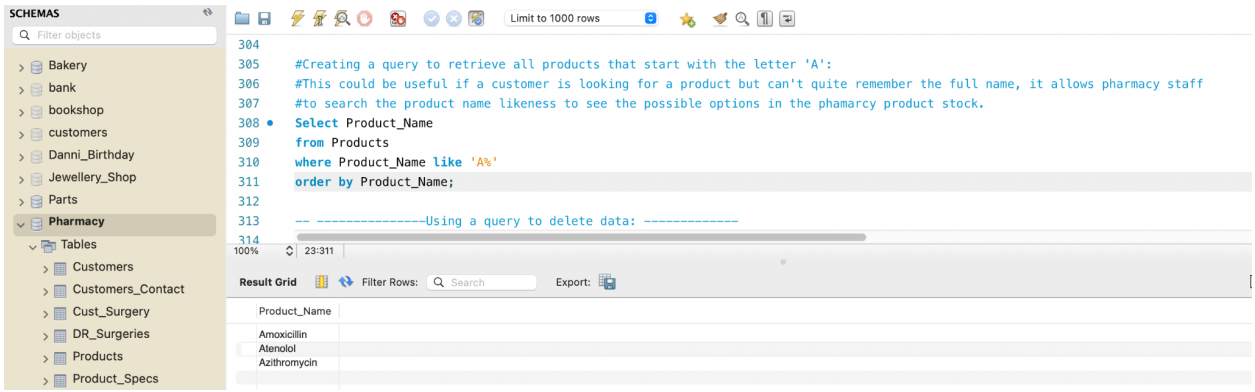
Query 3 code and output:



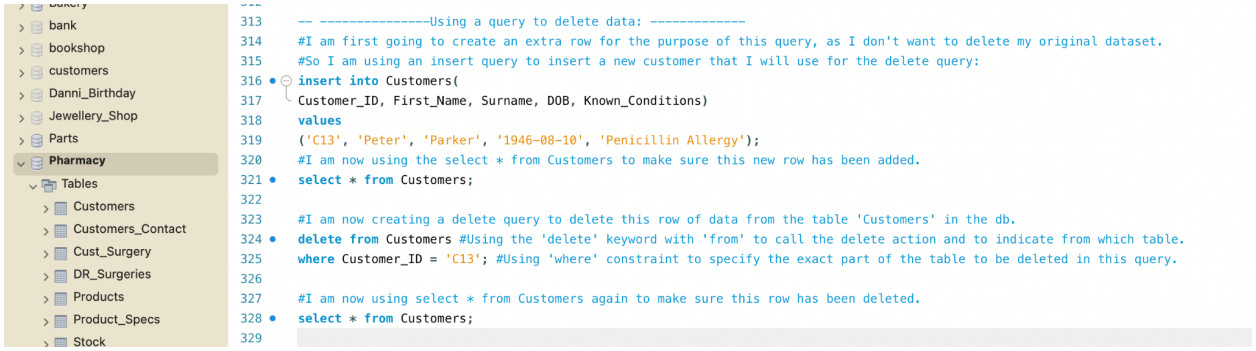
Query 4 code and output:



Query 5 code and output:

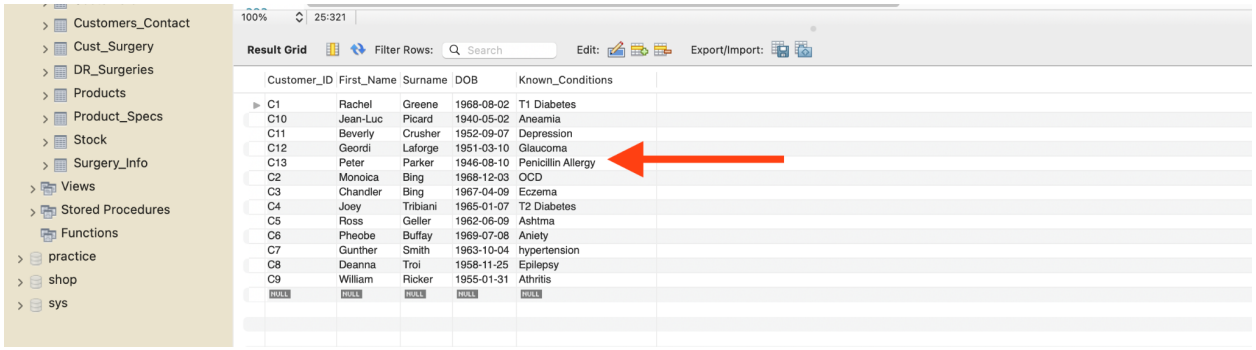


Using a query to delete data:



```
313  -- ---------------Using a query to delete data: -------------
314  #I am first going to create an extra row for the purpose of this query, as I don't want to delete my original dataset.
315  #So I am using an insert query to insert a new customer that I will use for the delete query:
316  insert into Customers(
317  Customer_ID, First_Name, Surname, DOB, Known_Conditions)
318  values
319  ('C13', 'Peter', 'Parker', '1946-08-10', 'Penicillin Allergy');
320  #I am now using the select * from Customers to make sure this new row has been added.
321  select * from Customers;
322
323  #I am now creating a delete query to delete this row of data from the table 'Customers' in the db.
324  delete from Customers #Using the 'delete' keyword with 'from' to call the delete action and to indicate from which table.
325  where Customer_ID = 'C13'; #Using 'where' constraint to specify the exact part of the table to be deleted in this query.
326
327  #I am now using select * from Customers again to make sure this row has been deleted.
328  select * from Customers;
329
```

Output before delete:

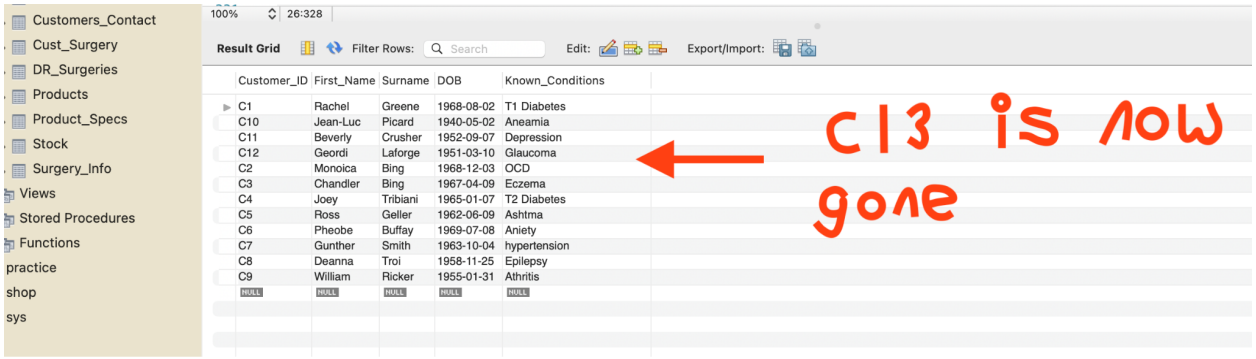


100% 25:321

Result Grid Filter Rows: Search Edit: Export/Import:

Customer_ID	First_Name	Surname	DOB	Known_Conditions
C1	Rachel	Greene	1968-08-02	T1 Diabetes
C10	Jean-Luc	Picard	1940-05-02	Aneamia
C11	Beverly	Crusher	1952-09-07	Depression
C12	Geordi	Laforge	1951-03-10	Glaucoma
C13	Peter	Parker	1946-08-10	Penicillin Allergy
C2	Monoica	Bing	1968-12-03	OCD
C3	Chandler	Bing	1967-04-09	Eczema
C4	Joey	Tribiani	1965-01-07	T2 Diabetes
C5	Ross	Geller	1962-06-09	Ashtma
C6	Pheobe	Buffay	1969-07-08	Aniety
C7	Gunther	Smith	1963-10-04	hypertension
C8	Deanna	Troi	1958-11-25	Epilepsy
C9	William	Ricker	1955-01-31	Athritis

Output after delete:

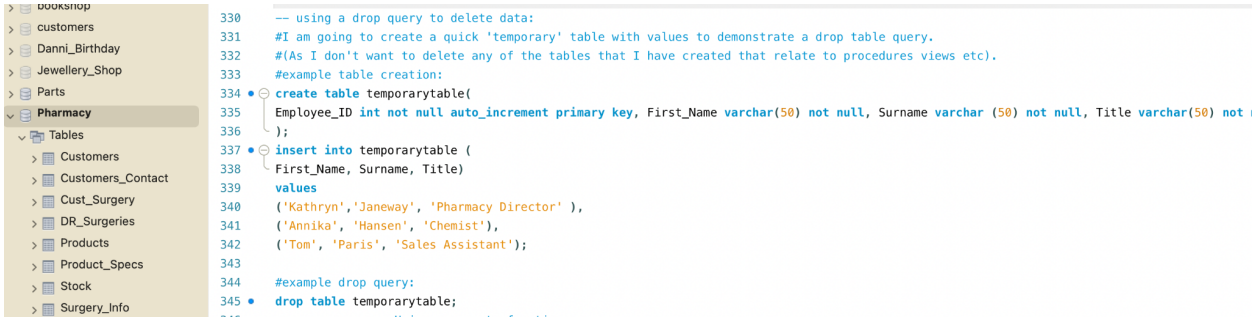


100% 26:328

Result Grid Filter Rows: Search Edit: Export/Import:

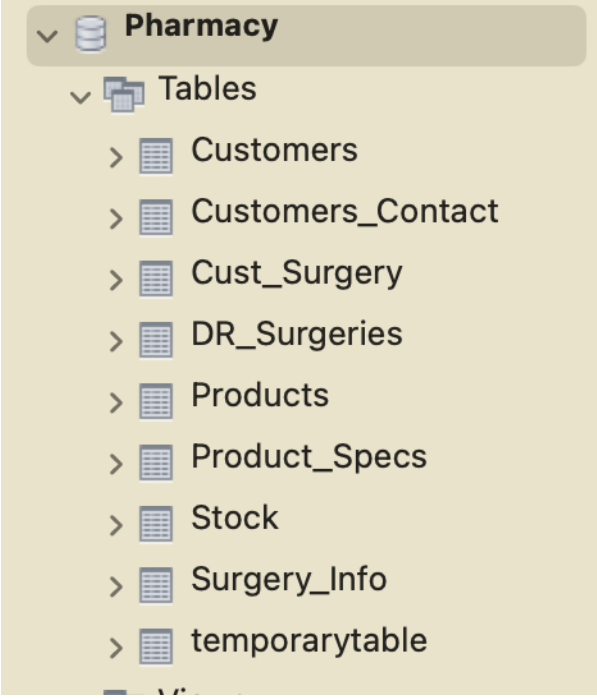
Customer_ID	First_Name	Surname	DOB	Known_Conditions
C1	Rachel	Greene	1968-08-02	T1 Diabetes
C10	Jean-Luc	Picard	1940-05-02	Aneamia
C11	Beverly	Crusher	1952-09-07	Depression
C12	Geordi	Laforge	1951-03-10	Glaucoma
C2	Monoica	Bing	1968-12-03	OCD
C3	Chandler	Bing	1967-04-09	Eczema
C4	Joey	Tribiani	1965-01-07	T2 Diabetes
C5	Ross	Geller	1962-06-09	Ashtma
C6	Pheobe	Buffay	1969-07-08	Aniety
C7	Gunther	Smith	1963-10-04	hypertension
C8	Deanna	Troi	1958-11-25	Epilepsy
C9	William	Ricker	1955-01-31	Athritis

Using a drop query to delete data:

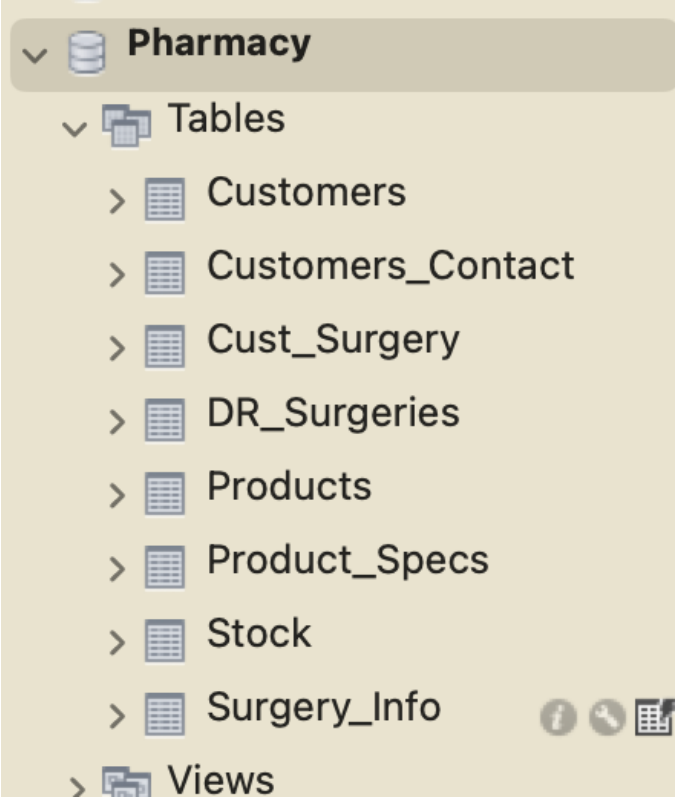


```
330  -- using a drop query to delete data:
331  #I am going to create a quick 'temporary' table with values to demonstrate a drop table query.
332  #(As I don't want to delete any of the tables that I have created that relate to procedures views etc).
333  #example table creation:
334  create table temporarytable(
335  Employee_ID int not null auto_increment primary key, First_Name varchar(50) not null, Surname varchar (50) not null, Title varchar(50) not
336  );
337  insert into temporarytable (
338  First_Name, Surname, Title)
339  values
340  ('Kathryn','Janeway', 'Pharmacy Director' ),
341  ('Annika', 'Hansen', 'Chemist'),
342  ('Tom', 'Paris', 'Sales Assistant');
343
344  #example drop query:
345  drop table temporarytable;
```

Output before query:

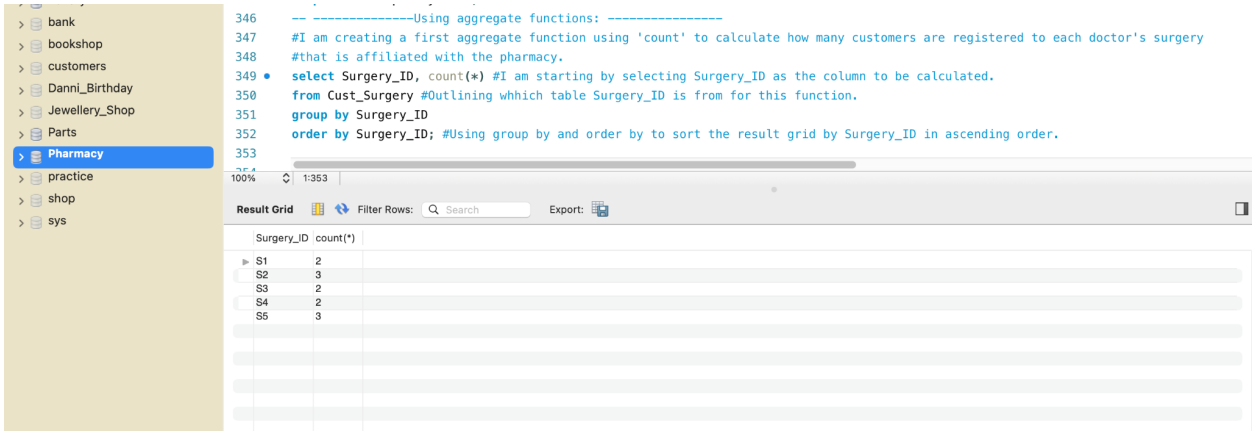


Output after query:

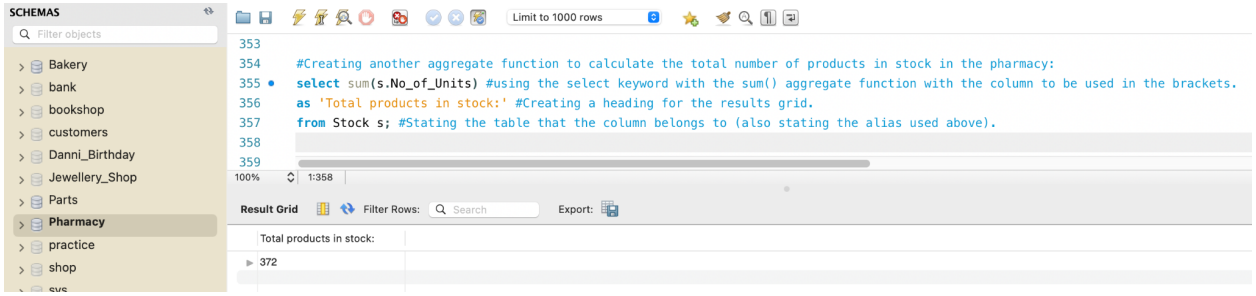


Aggregate Functions:

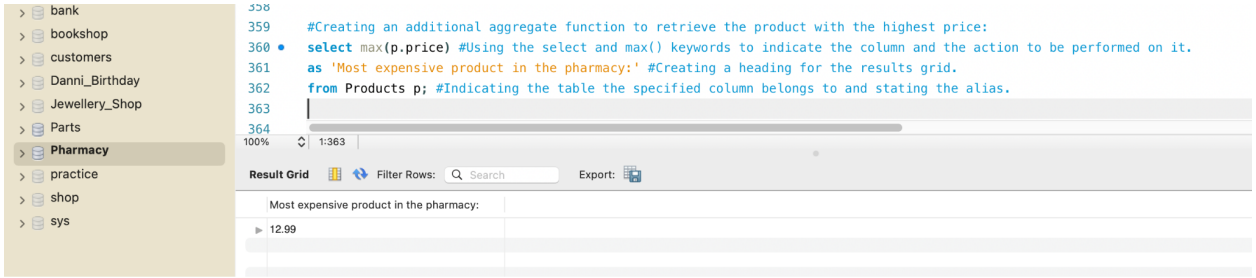
Aggregate function 1 and output:



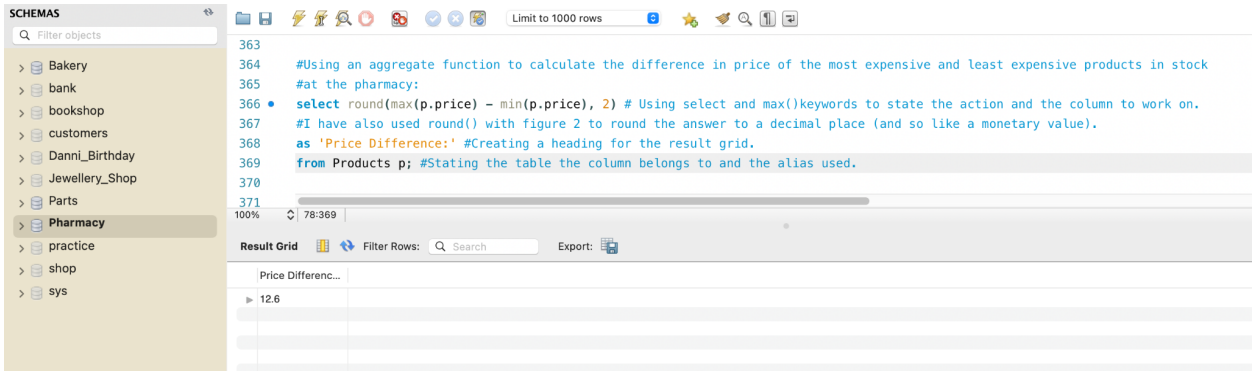
Aggregate function 2 and output:



Aggregate function 3 and output:

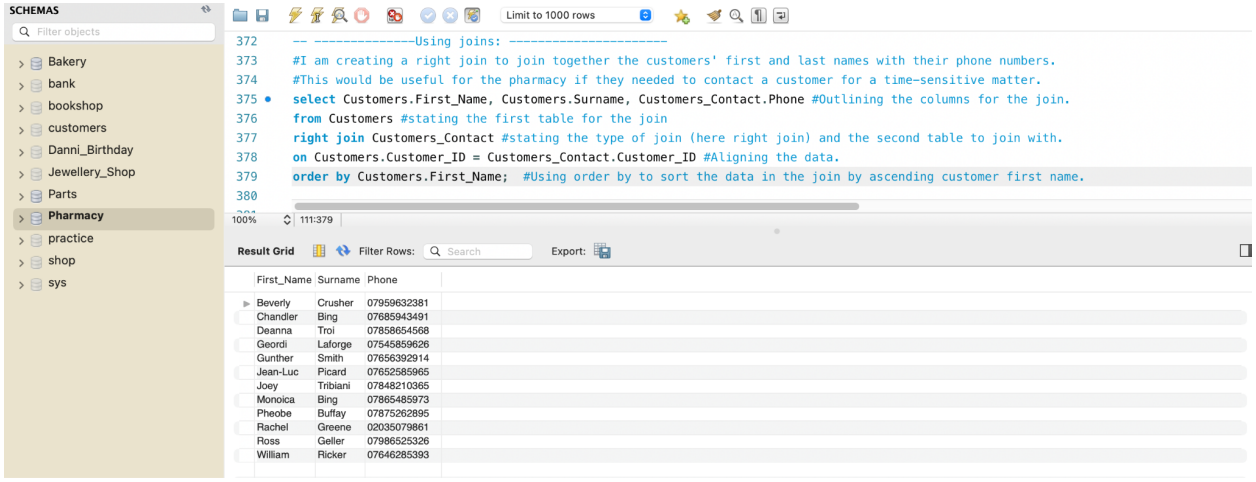


Aggregate function 4 and output:

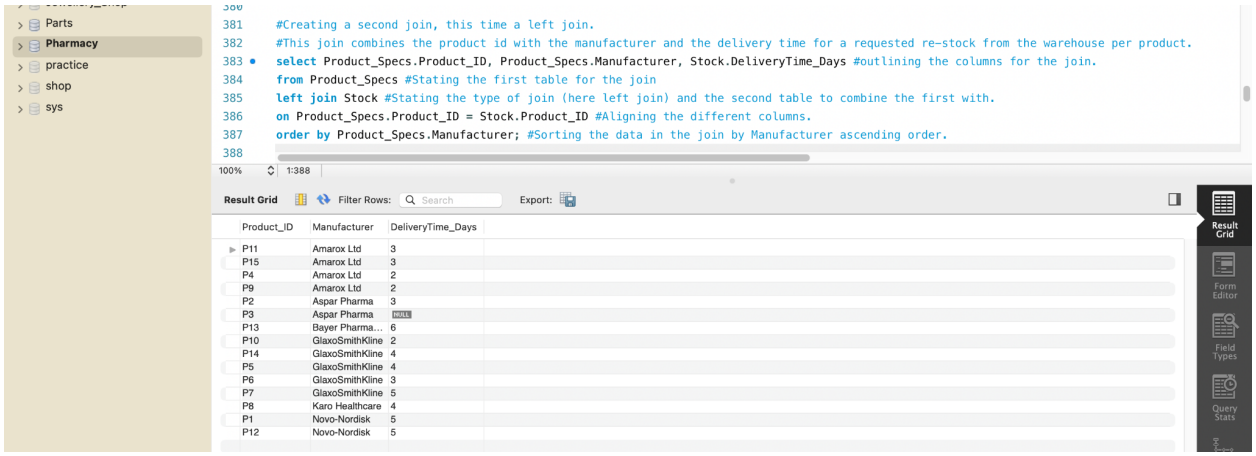


Using Joins:

Right join and output:

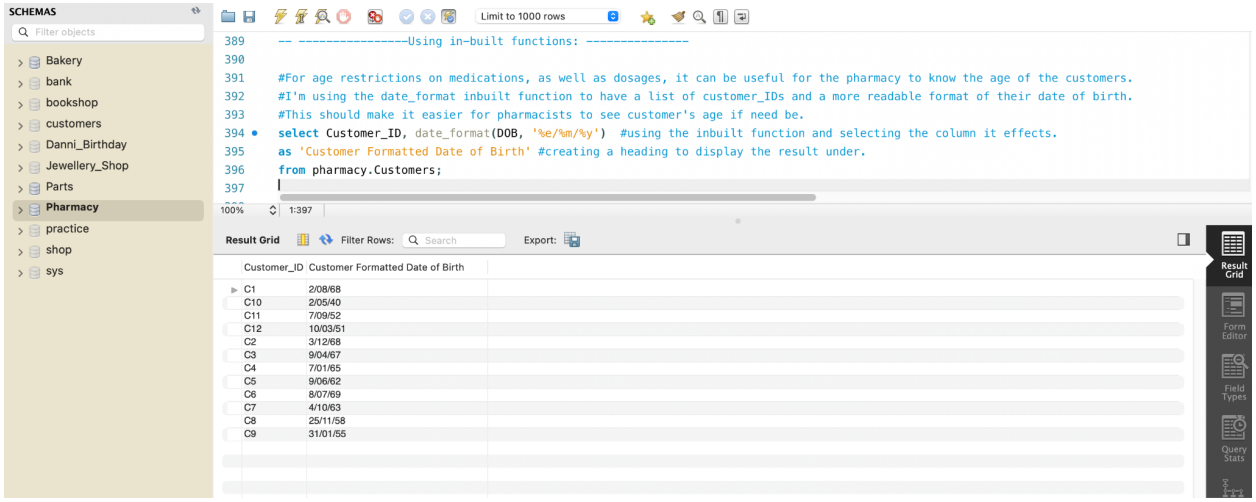


Left join and output:

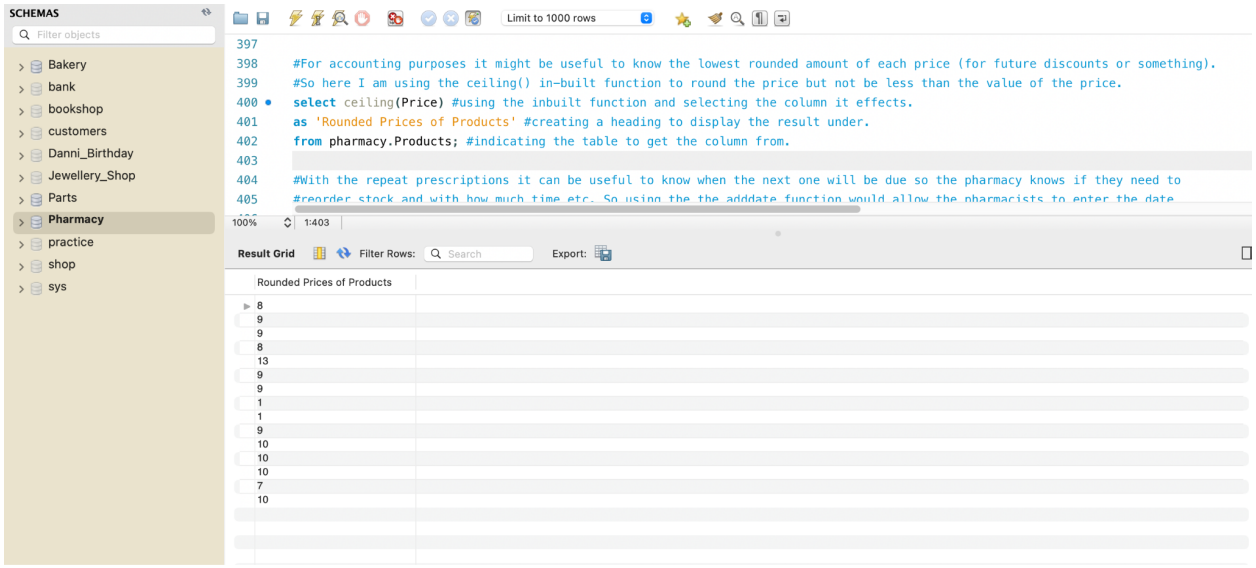


In-built functions:

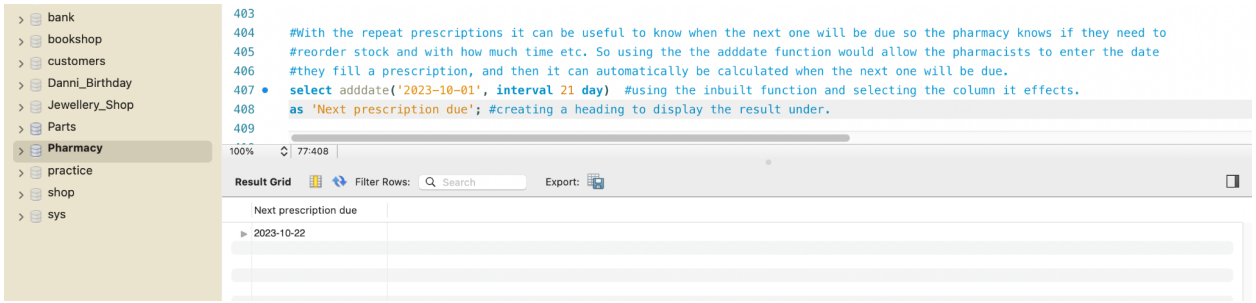
Date_format function and output:



Ceiling function and output:

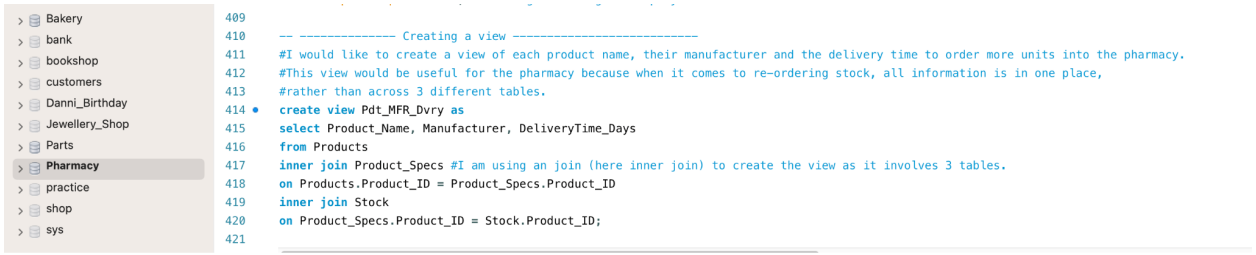


Adddate function and output:

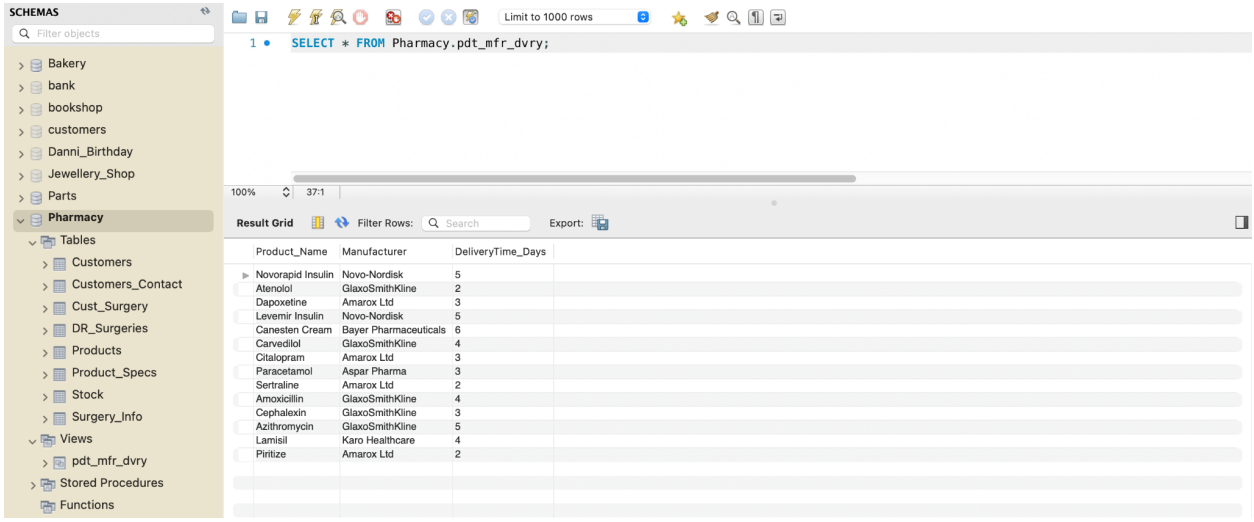


Creating a View:

View code:

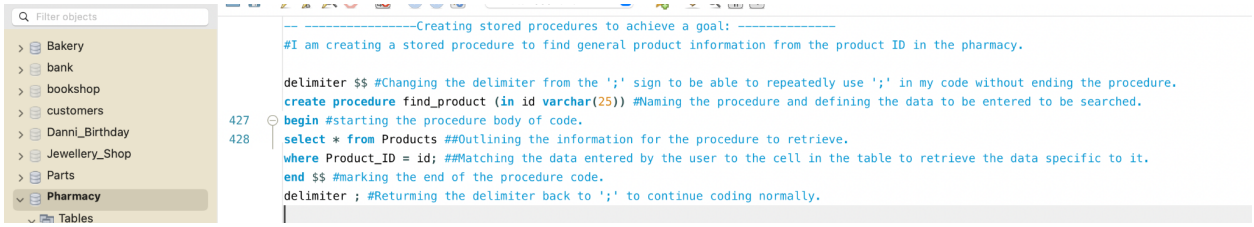


View output:

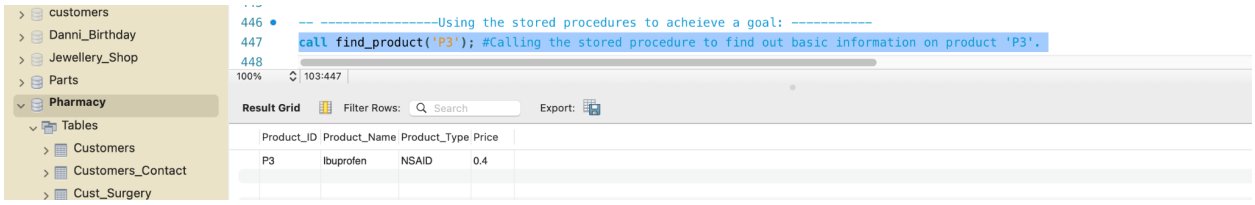


Stored Procedures:

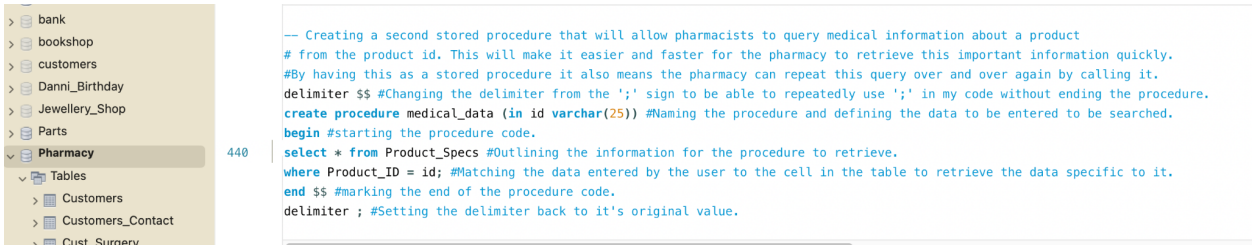
Procedure 1 code:



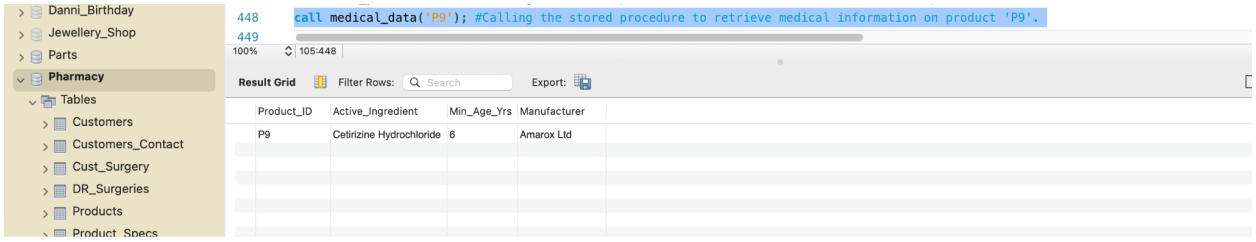
Calling Procedure 1:



Procedure 2 code:



Calling Procedure 2:



Updating 10 rows of mock data with dml command: