inf-2700 Oblig 1

Sigurd Uhre

August 2022

PART 1

1 Task 1. Database schema

Describe the inf2700 orders database schema. It is sufficient to show the CREATE TABLE statements that defined the schema.

The create table statements that define the schema is as follows: Customer, Employees, Offices, Order Details, Orders, Payments, Products and Product Lines. Each table mention above have underlying attributes that defines the table statements. The customer table statement have some underlying attributes such as: customer number, customer name, phone, address line, address line 2, state, city, and so on. Each table statements needs to have one or more primary keys. They can also have foreign keys. The primary key is used for indexing.

schema.txt

```
CREATE TABLE Customers (
  customerNumber INTEGER PRIMARY KEY,
  customerName TEXT NOT NULL,
  contactLastName TEXT NOT NULL,
  contactFirstName TEXT NOT NULL,
  phone TEXT NOT NULL,
  addressLine1 TEXT NOT NULL,
  addressLine2 TEXT NULL,
  city TEXT NOT NULL,
  state TEXT NULL,
  postalCode TEXT NULL,
  country TEXT NOT NULL,
  salesRepEmployeeNumber INTEGER NULL,
  creditLimit REAL NULL
);
CREATE TABLE Employees (
  employeeNumber INTEGER PRIMARY KEY,
  lastName TEXT NOT NULL,
  firstName TEXT NOT NULL,
  extension TEXT NOT NULL,
  email TEXT NOT NULL,
  officeCode TEXT NOT NULL,
  reportsTo INTEGER NULL,
  jobTitle TEXT NOT NULL,
  FOREIGN KEY (reportsTo) REFERENCES Employees(employeeNumber)
);
CREATE TABLE Offices (
  officeCode TEXT PRIMARY KEY,
  city TEXT NOT NULL,
  phone TEXT NOT NULL,
  addressLine1 TEXT NOT NULL,
  addressLine2 TEXT NULL,
  state TEXT NULL,
  country TEXT NOT NULL,
  postalCode TEXT NOT NULL,
  territory TEXT NOT NULL
);
CREATE TABLE OrderDetails (
  orderNumber INTEGER NOT NULL,
  productCode TEXT NOT NULL,
  quantityOrdered INTEGER NOT NULL,
  priceEach REAL NOT NULL,
  orderLineNumber INTEGER NOT NULL,
  PRIMARY KEY (orderNumber, productCode),
  FOREIGN KEY (productCode) REFERENCES Products
);
CREATE TABLE Orders (
  orderNumber INTEGER PRIMARY KEY,
  orderDate TEXT NOT NULL,
  requiredDate TEXT NOT NULL,
  shippedDate TEXT NULL,
  status TEXT NOT NULL,
  comments TEXT NULL,
  customerNumber INTEGER NOT NULL,
  FOREIGN KEY (customerNumber) REFERENCES Customers
);
CREATE TABLE Payments (
  customerNumber INTEGER NOT NULL,
  checkNumber TEXT NOT NULL,
  paymentDate TEXT NOT NULL,
  amount REAL NOT NULL,
  PRIMARY KEY (customerNumber, checkNumber),
  FOREIGN KEY (customerNumber) REFERENCES Customers
);
CREATE TABLE Products (
  productCode TEXT PRIMARY KEY,
  productName TEXT NOT NULL,
  productLine TEXT NOT NULL,
  productScale TEXT NOT NULL,
  productVendor TEXT NOT NULL,
  productDescription TEXT NOT NULL,
  quantityInStock INTEGER NOT NULL,
  buyPrice REAL NOT NULL,
  MSRP REAL NOT NULL,
  FOREIGN KEY (productLine) REFERENCES Productlines
);
CREATE TABLE ProductLines(
  productLine TEXT PRIMARY KEY,
  description TEXT NULL
```

Figure 1: database schema

2 Task 2. Run the given SQL queries

a)

Atelier graphique	Schmitt	Carine
Signal Gift Store	King	Jean
Australian Collec	Ferguson	Peter
La Rochelle Gifts	Labrune	Janine
Baane Mini Import	Bergulfsen	Jonas
Mini Gifts Distri	Nelson	Susan
Havel & Zbyszek C	Piestrzeniewicz	Zbyszek
Blauer See Auto,	Keitel	Roland
Mini Wheels Co.	Murphy	Julie
Land of Toys Inc.	Lee	Kwai

Figure 2: a).

For each query, write in your own words in a few lines that describe its purpose and how it works.

The first query selects attributes (customer name, contact name and contact first name) from the customers table. The tuples (rows) is set at a limit of 10. Resulting in the overlying figure 1.

b

The selection in this query is asterisk (*), which is the shorthand for all columns of the table as follows:. In this example the following table is Orders. In other words, all attributes from the order table where shippingdate is NULL will be displayed. This query would be relevant to check all orders that have not been sent yet.

c

The selection in this query is selected from a specific attribute that is referenced below in the table. Like SELECT C.customerName AS Customer, FROM Customer C, means that the customerName attribute is to be selected from the Customer table and set in a row with the name Customer presented in the terminal window.

The **as** clause can appear in both the **select** and **from** clauses. The as clause is particularly useful in renaming relations. One reason to rename a relation is to replace a long relation name with a shortened version that is more convenient to use elsewhere in the query.

The values to be summed in the preceding query include the quantityOrdered attribute from the Order details. This summary is to be presented as **Total** in the terminal table.

In the above query, **C**, **O** and **OD** can be thought of as copies of the relation instructor, but more precisely, they are declared as aliases, that is, as alternative names, for the relation instructor.

If we follow the **O** relation instructor to the **where section**, we can interpret this as *customerNumber* from **Orders** set to be equal as *customersNumbers* from **customers**. The same logic goes for *orderNumber* from **Orders** and **orderDetails**.

The **group by** clause is an optional clause of the **select** statement. The GROUP BY clause a selected group of rows into summary rows by values of one or more columns. In this example the grouping is set by *customerNumber* with the the relation instructor **O** set from **Orders**. Ranking is done with an order by specification. The following query gives the rank of customer under the summary **Total** in descending order.

d

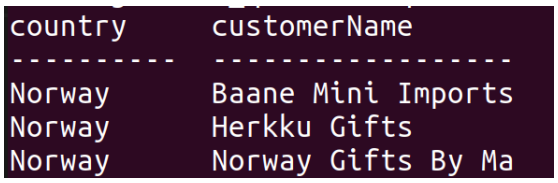
The **select** is set to *productName* with the relation instructor set to **products** in the **from** section and indicated with an *P*.

The secound **select** is *totalQuantityOrdered* with relation instructor set as *T*. In the **from** line the **products** are set in a *natural JOIN*. The natural join operation operates on two relations and produces a relation as the result. It considers only those pairs of tuples with the same value on those attributes that appear in the schemas of both relations. In this example it considers only those pairs of tuples where both the tuple from **products** and the tuple from the underlying **parenthesis** selected have the same value on the common attribute.

This parenthesis is an queries of its own and **select** the **productCode** an the SUM of **quantityOrdered** set as **totalQuantityOrdered** in the finale table. The selection in this parenthesis is done **from** the **OrderDetails** and **group by** for **productCode**. Then set as the relation instructor **T** for *totalQuantityOrdered*. To shorten the final table, the limit for the *totalQuantityOrdered* is set so 1000 ob above.

Task 3

Subtask 1



country	customerName
Norway	Baane Mini Imports
Norway	Herkku Gifts
Norway	Norway Gifts By Ma

Figure 3: Retrieve all customers in Norway.

Subtask 2

productName	productScale	productLine
1969 Harley Davidson Ultimate Chopper	1:10	Classic Cars
1952 Alpine Renault 1300	1:10	Classic Cars
1996 Moto Guzzi 1100i	1:10	Classic Cars
2003 Harley-Davidson Eagle Drag Bike	1:10	Classic Cars
1972 Alfa Romeo GTA	1:10	Classic Cars
1962 LanciaA Delta 16V	1:10	Classic Cars
1968 Ford Mustang	1:12	Classic Cars
2001 Ferrari Enzo	1:12	Classic Cars
1958 Setra Bus	1:12	Classic Cars
2002 Suzuki XREO	1:12	Classic Cars
1969 Corvair Monza	1:18	Classic Cars
1968 Dodge Charger	1:12	Classic Cars
1969 Ford Falcon	1:12	Classic Cars
1970 Plymouth Hemi Cuda	1:12	Classic Cars
1957 Chevy Pickup	1:12	Classic Cars
1969 Dodge Charger	1:12	Classic Cars
1940 Ford Pickup Truck	1:18	Classic Cars
1993 Mazda RX-7	1:18	Classic Cars
1937 Lincoln Berline	1:18	Classic Cars
1936 Mercedes-Benz 500K Special Roads	1:18	Classic Cars
1965 Aston Martin DB5	1:18	Classic Cars
1980s Black Hawk Helicopter	1:18	Classic Cars
1917 Grand Touring Sedan	1:18	Classic Cars
1948 Porsche 356-A Roadster	1:18	Classic Cars
1995 Honda Civic	1:18	Classic Cars
1998 Chrysler Plymouth Prowler	1:18	Classic Cars
1911 Ford Town Car	1:18	Classic Cars
1964 Mercedes Tour Bus	1:18	Classic Cars
1932 Model A Ford J-Coupe	1:18	Classic Cars
1926 Ford Fire Engine	1:18	Classic Cars
P-51-D Mustang	1:72	Classic Cars
1936 Harley Davidson El Knucklehead	1:18	Classic Cars
1928 Mercedes-Benz SSK	1:18	Classic Cars
1999 Indy 500 Monte Carlo SS	1:18	Classic Cars
1913 Ford Model T Speedster	1:18	Classic Cars
1934 Ford V8 Coupe	1:18	Classic Cars
1999 Yamaha Speed Boat	1:18	Classic Cars
18th Century Vintage Horse Carriage	1:18	Classic Cars
1903 Ford Model A	1:18	Classic Cars
1992 Ferrari 360 Spider red	1:18	Classic Cars
1985 Toyota Supra	1:18	Classic Cars
Collectable Wooden Train	1:18	Classic Cars
1969 Dodge Super Bee	1:18	Classic Cars
1917 Maxwell Touring Car	1:18	Classic Cars
1976 Ford Gran Torino	1:18	Classic Cars
1948 Porsche Type 356 Roadster	1:18	Classic Cars
1957 Vespa GS150	1:18	Classic Cars
1941 Chevrolet Special Deluxe Cabriol	1:18	Classic Cars
1970 Triumph Spitfire	1:18	Classic Cars
1932 Alfa Romeo 8C2300 Spider Sport	1:18	Classic Cars
1904 Buick Runabout	1:18	Classic Cars
1940s Ford truck	1:18	Classic Cars

Figure 4: Retrieve all classic car products and their scale. .

Subtask 3

orderNumber	status	requiredDate	productName	quantityInStock	quantityOrdered
10420	In Process	2005-06-07	1917 Grand Touring Sedan	2724	37
10420	In Process	2005-06-07	1911 Ford Town Car	540	36
10420	In Process	2005-06-07	1932 Model A Ford J-Coup	9354	45
10420	In Process	2005-06-07	1932 Alfa Romeo 8C2300 S	6553	66
10420	In Process	2005-06-07	1957 Ford Thunderbird	3209	36
10420	In Process	2005-06-07	1970 Chevy Chevelle SS 4	1005	60
10420	In Process	2005-06-07	1966 Shelby Cobra 427 S/	8197	37
10420	In Process	2005-06-07	1939 Chevrolet Deluxe Co	7332	45
10420	In Process	2005-06-07	1949 Jaguar XK 120	2350	39
10420	In Process	2005-06-07	1952 Citroen-15CV	1452	55
10420	In Process	2005-06-07	1969 Chevrolet Camaro Z2	4695	35
10420	In Process	2005-06-07	2002 Chevy Corvette	9446	26
10420	In Process	2005-06-07	1936 Mercedes Benz 500k	2081	15
10421	In Process	2005-06-06	1928 Mercedes-Benz SSK	548	35
10421	In Process	2005-06-06	1938 Cadillac V-16 Presi	2847	40
10422	In Process	2005-06-11	1937 Lincoln Berline	8693	51
10422	In Process	2005-06-11	1936 Mercedes-Benz 500K	8635	25
10423	In Process	2005-06-05	1913 Ford Model T Speeds	4189	10
10423	In Process	2005-06-05	1934 Ford V8 Coupe	5649	31
10423	In Process	2005-06-05	18th Century Vintage Hor	5992	21
10423	In Process	2005-06-05	1917 Maxwell Touring Car	7913	21
10423	In Process	2005-06-05	1936 Chrysler Airflow	4710	28
10424	In Process	2005-06-08	1952 Alpine Renault 1300	7305	50
10424	In Process	2005-06-08	1958 Setra Bus	1579	49
10424	In Process	2005-06-08	1940 Ford Pickup Truck	2613	54
10424	In Process	2005-06-08	1939 Cadillac Limousine	6645	26
10424	In Process	2005-06-08	1996 Peterbilt 379 Stake	814	44
10424	In Process	2005-06-08	1982 Camaro Z28	6934	46
10425	In Process	2005-06-07	1962 LanciaA Delta 16V	6791	38
10425	In Process	2005-06-07	1957 Chevy Pickup	6125	33
10425	In Process	2005-06-07	1998 Chrysler Plymouth P	4724	28
10425	In Process	2005-06-07	1964 Mercedes Tour Bus	8258	38
10425	In Process	2005-06-07	1926 Ford Fire Engine	2018	19
10425	In Process	2005-06-07	1992 Ferrari 360 Spider	8347	28
10425	In Process	2005-06-07	1940s Ford truck	3128	38
10425	In Process	2005-06-07	1970 Dodge Coronet	4074	55
10425	In Process	2005-06-07	1962 Volkswagen Microbus	2327	49
10425	In Process	2005-06-07	1958 Chevy Corvette Limi	2542	31
10425	In Process	2005-06-07	1980◆s GM Manhattan Expre	5099	41
10425	In Process	2005-06-07	1954 Greyhound Scenicrui	2874	11
10425	In Process	2005-06-07	Diamond T620 Semi-Skirte	1016	18

Figure 5: Create a list of incomplete orders (order status is "In process"). The list must contain orderNumber, requiredDate, productName, quantityOrdered and quantityInStock.

Subtask 4

customerNumber	customerName	creditLimit	TotalPrice	Diff
103	Atelier graphique	21000.0	22314.36	1314.36
112	Signal Gift Store	71800.0	80180.98	8380.98
114	Australian Collec	117300.0	180585.07	63285.07
121	Baane Mini Import	81700.0	104224.79	22524.79
124	Mini Gifts Distri	210500.0	584188.24	373688.24
128	Blauer See Auto,	59700.0	75937.76	16237.76
129	Mini Wheels Co.	64600.0	66710.56	2110.56
141	Euro+ Shopping Ch	227600.0	715738.98	488138.98
145	Danish Wholesale	83400.0	107446.5	24046.5
146	Saveley & Henriot	123900.0	130305.35	6405.35000
148	Dragon Souveniers	103800.0	156251.03	52451.03
151	Muscle Machine In	138500.0	177913.95	39413.95
161	Technics Stores I	84600.0	104545.22	19945.22
166	Handji Gifts& Co	97900.0	105420.57	7520.56999
167	Herkku Gifts	96800.0	97562.47	762.470000
172	La Corne D'abonda	84300.0	86553.52	2253.52
175	Gift Depot Inc.	84300.0	95424.63	11124.63
187	AV Stores, Co.	136800.0	148410.09	11610.09
205	Toys4GrownUps.com	90700.0	93803.3	3103.29999
209	Mini Caravy	53800.0	75859.32	22059.32
216	Enaco Distributor	60300.0	68520.47	8220.47
233	Qu◆bec Home Shopp	48700.0	68977.67	20277.67
276	Anna's Decoration	107800.0	137034.22	29234.22
278	Rovelli Gifts	119600.0	127529.69	7929.69
311	Oulu Toy Supplies	90500.0	95706.15	5206.14999
320	Mini Creations Lt	94500.0	101872.52	7372.51999
321	Corporate Gift Id	105000.0	132340.78	27340.78
323	Down Under Souven	88000.0	154622.08	66622.08
324	Stylish Desk Deco	77000.0	80556.73	3556.73
333	Australian Gift N	51600.0	55190.16	3590.16
334	Suominen Souvenie	98800.0	103896.74	5096.73999
350	Marseille Mini Au	65000.0	71547.53	6547.53
353	Reims Collectable	81100.0	126983.19	45883.19
363	Online Diecast Cr	114200.0	116449.29	2249.29000
379	Collectables For	70700.0	73533.65	2833.64999
381	Royale Belge	23500.0	29217.18	5717.18
382	Salzburg Collecta	71700.0	85060.0	13360.0
385	Cruz & Sons Co.	81500.0	87468.3	5968.3
398	Tokyo Collectable	94400.0	105548.73	11148.73
424	Classic Legends I	67500.0	69214.33	1714.33
447	Gift Ideas Corp.	49700.0	49967.78	267.779999
452	Mini Auto Werke	45300.0	51059.99	5759.99000
458	Corrida Auto Repl	104600.0	112440.09	7840.09
462	FunGiftIdeas.com	85800.0	88627.49	2827.49000
486	Motor Mint Distri	72600.0	77726.59	5126.59
496	Kelly's Gift Shop	110000.0	114497.19	4497.19

Figure 6: Create a list of customers where the difference between the total price of all ordered products and the total amount of all payments exceeds the credit limit. The list must contain the customer name, credit limit, total price, total payment and the difference between the two sums.

5

This queries was not completed.

PART 2

In this part we are going to implement a SQL tester in C. We will test it against the sample database test/messages.sqlite3.

The first implementation is to open the test database. The function "sqlite3 open()" is used for opening a new database connection with the test database filename as argument. If the process does not return a successfull result, the process will return "Could not open DB".

The next "TODO" process is to run the query. We run the "sqilite execute()" with the necessary arguments. Before presenting the arguments, we use a null terminating process to help the strncopy() function with copying the query-string to a new array. The query result is put in "query res". I f anything goes wrong, "querry err" will print the error messages form the "sqilite execute()" function.

Sqlite3 exec() contains a callback mechanism that provides a way to obtain results from select statements. this mechanism is implemented by the third an fourth arguments of the function, where the third argument is a pointer to the callback function. SQLite will call the function for each record processed in a select statement executed within the sql argument. The fourth argument to sqlite exec() is a void pointer to the application-specific data to be supplied to the callback function. The last argument in the function (err) is a pointer to an string which an error message can be written should an error occur. [1].

The sqlite exec() function allows us to issue a batch of commands, and we can collect all of the returned data by using the callback interface.[1]. The callback function manually formatting the results. Pipe formatting is done by inserting pipe before and after every query result. A newline is also added after each query.

The result of the implementation is given by running "./testsql test test queries messages" in the terminal.

```
suh000@ifilab87-lnx:~/suh000/inf2700/suh000/assignment-1/sql-teste
r$ ./testsql test test_queries_messages
0
 tekst

test "all_countries" succeeded
 tekst

test "messages_to_people_other_than_the_author" succeeded
 tekst

test "empty_result" succeeded
 tekst

test "wrong_result" failed
exepected:
|correct result|

got:
|wrong result|

 tekst

test "erroneous_query" failed
exepected:
|correct result|

got:

test "non_existent" not found.
```

Figure 7: Results.

3 References

1. Owens M, Allen G. The Definitive Guide to SQLite. New York: Apress L.P; 2010. Page: 208-211