# CS 500: Fundamentals of Databases

*Assignment #2*

**The database**

For this assignment, we will use the same database as assignment 1.

As a reminder, the database is based on a fictitious specialty foods import/export company. The customers of the company are other companies.

The table structures for the database include the following tables

The categories table describes the various product categories.

CREATE TABLE categories (

CategoryID smallint primary key,

CategoryName varchar(15) NOT NULL,

Description text

);

The products table has information about the products sold by the company.

CREATE TABLE products (

ProductID smallint primary key,

ProductName varchar(40) NOT NULL,

SupplierID smallint,

CategoryID smallint,

QuantityPerUnit varchar(20),

UnitPrice real,

UnitsInStock smallint,

UnitsOnOrder smallint,

ReorderLevel smallint,

Discontinued integer NOT NULL,

foreign key (CategoryID) references categories(CategoryID)

);

The customers table has information about the companies that buy the products. The customers of the import/export company are companies.

CREATE TABLE customers (

CustomerID char(8) primary key,

CompanyName varchar(40) NOT NULL,

CustomerName varchar(30),

CustomerTitle varchar(30),

Address varchar(60),

City varchar(15),

Region varchar(15),

PostalCode varchar(10),

Country varchar(15),

Phone varchar(24),

Fax varchar(24)

);

The orders table has information about orders placed by customers (companies) to the company. Orders are shipped via freight shipping.

CREATE TABLE orders (

OrderID smallint primary key,

CustomerID char(8),

EmployeeID smallint,

OrderDate date,

RequiredDate date,

ShippedDate date,

ShipVia smallint,

Freight real,

ShipName varchar(40),

ShipAddress varchar(60),

ShipCity varchar(15),

ShipRegion varchar(15),

ShipPostalCode varchar(10),

ShipCountry varchar(15),

foreign key (CustomerID) references customers(CustomerID)

);

The order\_details table links orders with products.

CREATE TABLE order\_details (

OrderID smallint NOT NULL,

ProductID smallint NOT NULL,

UnitPrice real NOT NULL,

Quantity smallint NOT NULL,

Discount real NOT NULL,

primary key (OrderID, ProductID),

foreign key (OrderID) references orders(OrderID),

foreign key (ProductID) references products(ProductID)

);

**The setup**

Create a file named a2.sql that contains the SQL statements for each query below. The format of the a2.sql file should be

-- Q1

select attributes

from tables

where conditions

-- Q2

select attributes

from tables

where conditions

and so on..

**The queries**

Q1 – For each category id, find the number of products in the category. List by category id ascending.

SELECT CategoryID, count(ProductID)

FROM products

GROUP BY CategoryID

ORDER BY CategoryID ASC;

Q2 – For each category name, find the number of products in the category. List the results alphabetically by category name.

SELECT C.CategoryName, count(ProductID)

FROM products P, categories C

WHERE P.CategoryID = C.CategoryID

GROUP BY CategoryName

ORDER BY CategoryName;

Q3 – Find the number of distinct suppliers for products.

SELECT COUNT(DISTINCT SupplierID) AS "Number of distinct suppliers"

FROM products;

Q4 – Find the total number of distinct customers.

select count (distinct CustomerID) as "Number of distinct customers" from customers;

Q5 – Find the company name that placed 10 or more orders in 1997. Order the result by the order count.

SELECT customers.CompanyName, COUNT(\*) as order\_count

FROM customers

INNER JOIN orders ON customers.CustomerID = orders.CustomerID

WHERE orders.OrderDate >= '1997-01-01'

AND orders.OrderDate < '1998-01-01'

GROUP BY CompanyName

HAVING COUNT(\*) >= 10

ORDER BY COUNT(\*);

Q6 – Find the number of unique products of every product that was ever offered for a discount of 25%.

select count(distinct order\_details.ProductID) as unique\_products\_count

from order\_details

where discount = '0.25';

Q7 – Count the customers that did not place an order in 1996.

SELECT COUNT(CustomerID) as total\_customers\_not\_placed\_order\_in\_1996

FROM customers S

WHERE S.CustomerID NOT IN (SELECT O.CustomerID

FROM orders O

WHERE OrderDate BETWEEN '1996-01-01' AND '1996-12-31');

Q8 – Cities with at least two companies. Order by number of companies descending, first and then by the city name ascending.

SELECT City, COUNT(DISTINCT CompanyName) as companycount

FROM customers

GROUP BY City

HAVING COUNT(DISTINCT CompanyName) >= 2

ORDER BY companycount DESC, City ASC;

Q9 – Total price for order id 10332. In the calculation of total cost, do not include freight costs just product costs. Show the results in 2 decimal points.

SELECT Sum(cast(D.UnitPrice\*D.Quantity-D.UnitPrice\*D.Quantity\*D.Discount as decimal(10,2)))

FROM orders O, order\_details D

WHERE O.OrderID = D.OrderID

AND O.OrderID = 10332;

Q10 – Find the city with the most companies. Show the city name and the company names.

select companyName

from customers

where city in (select city

from customers

group by city

having count(companyName) = (select Max(numberofcompanies)

from (select city, count(companyname) as numberofcompanies

from customers

group by city) as foo));

select city, companyname

from customers

where city =

(select city from

(select ROW\_NUMBER() OVER (order by count(companyname) desc) row\_num, city, count(companyname)

from customers

group by city

order by count(companyname) desc

) x

where x.row\_num = 1);

**How to submit**

Submit your file a2.sql to Blackboard Learn by the deadline.