# CS 500: Fundamentals of Databases

*Assignment #1*

**The database**

This assignment 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**

Download and install postgreSQL and, then, perform the following steps:

1. From a terminal window, create a new database named a1db using PostgreSQL’s command createdb. Do this from the command line of your working directory. You may have to use -U with the username that you used during installation (default: postgres)

createdb -U postgres a1d\b or

createdb a1db

To connect to this database, use PostgreSQL’s psql command line interface. It is an interactive terminal program that allows you to connect to a database and execute SQL commands.

To connect to the database, use

psql -U postgres a1db or

psql a1db

1. Using psql create and load the tables. The commands have been provided in the file set\_a1.sql. Use psql’s \i command to execute the commands in the file. From within psql run

\i set\_a1.sql

1. Create a file named a1.sql that contains the SQL statements for each query below. The format of the a1.sql file should be

-- Q1

select attributes

from tables

where conditions

-- Q2

select attributes

from tables

where conditions

and so on..

**The queries**

Q1 – Show all the different product categories. Order results alphabetically.

select distinct c.categoryname

from categories c, products p

where c.categoryid = p.categoryid

order by c.categoryname ;

Q2 – Show the name and title of all customers from Canada. Order the results by customer name.

select customername, customertitle

from customers

where country = 'Canada'

order by customername;

Q3 – Show the customer name, address, city and country of all customers not from Brazil, France, Germany or USA. Order the results by country first, then city, both ascending.

SELECT customername, address, city, country

FROM customers

WHERE country NOT IN ('Brazil', 'France', 'Germany', 'USA')

ORDER BY country asc;

SELECT customername, address, city, country

FROM customers

WHERE country NOT IN ('Brazil', 'France', 'Germany', 'USA')

ORDER BY city asc;

Q4 – Show the order date, shipped date, customer id and freight of all orders placed on July 25 1997.

SELECT orderdate, shippeddate, customerid, freight

FROM orders

WHERE orderdate = '1997-07-25';

Q5 – Show the product name and unit price of all products whose name starts with 'C'. Order by unit price descending.

SELECT productname, unitprice

FROM products

WHERE productname LIKE 'C%'

ORDER BY unitprice DESC;

Q6 – Show the product name, unit price and quantity per unity of all products whose quantity comes in bottles. Hint: look at the possible distinct values in the quantityperunit attribute. Order by product name.

SELECT productname, unitprice, quantityperunit

FROM products

WHERE quantityperunit LIKE '%bottles%'

ORDER BY productname;

Q7 – Show the product name and category name for all products in the 'Produce' category. Order by product\_name.

SELECT p.productname, c.categoryname

FROM products p,categories c

WHERE c.categoryname LIKE '%Produce%'

ORDER BY p.productname;

Q8 – Show the order id, the company name that placed the order, and the first and last name of the associated employee (customer name). Only show orders placed after January 1, 1998 that shipped after they were required. Sort by order id.

SELECT o.OrderID, c.CompanyName, c.CustomerName

FROM orders o, customers c

WHERE o.ShippedDate > o.RequiredDate AND o.OrderDate > '1998-01-01'

ORDER BY o.OrderID;

Q9 – If the cost of freight is greater than or equal to $500.00, it will be taxed by 10%. Show the order id, freight cost and freight cost plus tax (freight\_with\_tax) for all orders of freight $500 or more. Use 2 decimal points for the calculated number freight\_with\_tax. Use cast(X as decimal(10,2)) to show the value of X as a decimal number with 10 digits and 2 decimal points.

SELECT OrderID, Freight, CAST(Freight+(Freight\*0.1) AS decimal(10,2)) AS Freight\_with\_tax

FROM orders

WHERE Freight >= 500;

Q10 – Show the order id, product name, product quantity, product unit price, product discount and product total price for order ids 10248 and 10866. You must calculate the product total price given the other data. To understand the meaning of the discount attribute, use this: a discount value of 0.25 represents a 25% discount. Show the total price with only 2 decimal points. Order the results by order id.

SELECT o.orderid, p.productname, od.quantity, od.unitprice, od.discount, CAST((od.quantity\*od.unitprice\*(1-od.discount)) as decimal(10, 2)) as product\_total\_price

FROM orders o

JOIN order\_details od ON o.orderid = od.orderid

JOIN products p ON od.productid = p.productid

WHERE o.orderid IN (10248, 10866)

ORDER BY o.orderid;

**How to submit**

Submit your file a1.sql to Blackboard Learn by the deadline.