-- Question 1 (4 points)

/\* Rewrite the following query to present the same data in a horizontal format,

as listed below, using the SQL PIVOT command. Use the sample format for

formatting purposes only.

Please use AdventureWorks2008R2 for this question. \*/

select datepart(mm, OrderDate) Month,

SalesPersonID,

cast(sum(TotalDue) as int) as TotalSales

from Sales.SalesOrderHeader

where month(OrderDate) in (1,3,5,7,9,11) and SalesPersonID between 275 and 280

group by SalesPersonID, datepart(mm, OrderDate)

having sum(TotalDue) > 350000

/\*

Month SalesPerson 275 276 277 278 279 280

1 580101 565043 424524

3 717093 682796 823458 535338

5 891806 1733135 1191378 632674 956272

7 824701 482019 799084 508966

9 1180955 1045380 1143952 590792

11 1148670 1728309 1112659 703393 1043030 385788

\*/

-- Question 2 (5 points)

/\*

Using AdventureWorks2008R2, write a query to return the territory id,

number of unique products sold, highest order value, total sales amount,

and top 3 orders for each territory. Use TotalDue in SalesOrderHeader

when calculating the highest order value and total sales amount. The

top 3 orders have the 3 highest total order quantities. If there is a tie,

the tie must be retrieved. Return only the territories that had total sales

greater than $12000000.

Return the order value and total sales as int. Sort the

returned data by TerritoryID. The returned data should have a format

displayed below. Use the sample format for formatting purposes only.

\*/

/\*

TerritoryID TotalUniqueProducts OrderValue TotalSales Orders

1 263 126852 18061660 51721, 57046, 53518, 69422

4 262 187488 27150595 47355, 51090, 47369

6 262 170513 18398929 51160, 53465, 55297

\*/

-- Question 3 (6 points)

/\* A financial planning company has a business rule that it cannot have

more than 55 active financial planning seminars at the same time and

a financial planning seminar will not accept more than 50 clients.

An active financial planning seminar is a seminar which has

at least one participant and doesn't have an end date.

Given 3 tables as defined below, write a SINGLE table-level CHECK

constraint based on a SINGLE Function to implement the rule. \*/

CREATE TABLE Client

(ClientID INT PRIMARY KEY,

LastName VARCHAR(50),

FirsName VARCHAR(50),

Email VARCHAR(30),

Phone VARCHAR(20));

CREATE TABLE Seminar

(SeminarID INT PRIMARY KEY,

Name VARCHAR(50),

Description VARCHAR(500),

StartDate DATE,

EndDAte DATE);

CREATE TABLE Registration

(ClientID INT REFERENCES Client(ClientID),

SeminarID INT REFERENCES Seminar(SeminarID),

Notes VARCHAR(1000)

PRIMARY KEY (ClientID, SeminarID));

-- Question 1 (4 points)

/\* Using AdventureWorks2008R2, rewrite the following query to

present the same data in a horizontal format,

as listed below, based on the SQL PIVOT command. \*/

USE AdventureWorks2008R2

select (p.LastName + ', ' + p.FirstName) FullName, datepart(dw, sh.OrderDate) Weekday, count(SalesOrderID) TotalOrder

from Sales.SalesOrderHeader sh

join Person.Person p

on sh.SalesPersonID = p.BusinessEntityID

group by p.LastName + ', ' + p.FirstName, datepart(dw, sh.OrderDate)

order by FullName;

/\*

FullName Sun Mon Tue Wed Thr Fri Sat

Abbas, Syed 2 0 2 0 3 1 8

Alberts, Amy 2 2 5 7 8 7 8

Ansman-Wolfe, Pamela 11 19 20 10 7 12 16

Blythe, Michael 57 39 63 54 82 64 91

Campbell, David 28 19 33 20 35 20 34

Carson, Jillian 49 43 65 58 117 81 60

Ito, Shu 32 18 29 22 51 40 50

Jiang, Stephen 8 2 11 7 9 4 7

Mensa-Annan, Tete 20 7 17 16 30 20 30

Mitchell, Linda 40 36 67 58 94 62 61

Pak, Jae 47 16 52 35 66 62 70

Reiter, Tsvi 53 41 64 59 98 52 62

Saraiva, José 29 36 37 38 55 31 45

Tsoflias, Lynn 22 7 12 10 13 8 37

Valdez, Rachel 19 13 17 14 19 12 36

Vargas, Garrett 20 21 35 32 55 37 34

Varkey Chudukatil, Ranjit 22 8 21 24 44 29 27

\*/

-- Question 2 (5 points)

/\*

Using AdventureWorks2008R2, write a query to retrieve

the customers and their order info.

Return the customer id, a customer's total purchase,

and a customer's top 5 orders.

The top 5 orders have the 5 highest order values.

Use TotalDue as the order value. If there is a tie,

the tie must be retrieved.

Include only the customers who have had at least one order

which contained more than 70 unique products.

Sort the returned data by CustomerID. Return the data in

the format specified below.

\*/

/\*

CustomerID TotalPurchase Orders

29712 653973.76 51739, 46987, 69437, 57061, 50225

29722 954021.92 45529, 48306, 47365, 44750, 53465

30048 678828.84 51160, 46657, 67316, 49879, 55297

30107 650362.05 51721, 57046, 69422, 43869, 63157

\*/

-- Question 3 (6 points)

/\* Given the following tables, there is a $100

club annual membership fee per customer.

There is a business rule, if a customer has spent more

than $5000 for the current year, then the membership fee

is waived for the current year. But if the total spending

of the current year gets below $5000 after the fee has

been waived, the fee will be charged again. The total

spending may be reduced by a return.

Please write a trigger to implement the business rule.

The membership fee is stored in the Customer table. \*/

create table Customer

(CustomerID int primary key,

LastName varchar(50),

FirstName varchar(50),

MembershipFee money);

create table SalesOrder

(OrderID int primary key,

CustomerID int references Customer(CustomerID),

OrderDate date not null);

create table OrderDetail

(OrderID int references SalesOrder(OrderID),

ProductID int,

Quantity int not null,

UnitPrice money not null

primary key(OrderID, ProductID));