```sql
--Exercise -Dafesty pg 83

--1) Retrieve the different Rating from the Movie table
select distinct rating from movies

--2) Retrieve all VideoCode and MovieTitle from Movie table with Rating = ‘PG’ or
  ‘R’
select videocode,movietitle from movies where rating ='pg' or rating = 'r'
select videocode,movietitle from movies where rating in ('pg','r')

--3) Retrieve all VideoCode and MovieTitle from Movie table with TotalStock greater
   than 4
 select videocode, movietitle from movies where totalstock>4

 --4) Retrieve CustomerName who has made a transaction (IssueTran) • Hint: use
   subquery for Customer and Issuetran tables
 select distinct c.customername from issuetran i,customers c where
   c.CustomerID=i.customerid

-- OR (subquery)
 select customername from customers where customerid in (select customerid from
   issuetran)

 --5) Retrieve CustomerName who has made a transaction (IssueTran) for the
   videocode = 27
 select customername from customers where customerid in(select customerid from
   issuetran where videocode=27)

  -- Dafesty table

--1) Retrieve videocode, movietitle from movies
select videocode, movietitle from movies

--2) Retrieve Customerid from issuetran table with videocode = 27
select customerid,videocode from issuetran where videocode=27

--3) Retrieve all distinct countrycode from Producers table
select distinct countrycode from Producers

--4) Retrieve all details of Producers for Producer 'Columbia'
select * from producers where producer in ('columbia')
select * from producers where producer = 'columbia'

--5) Retrieve all details of Producers for Producer 'Columbia' or 'George'
select * from producers where producer ='columbia' or producer= 'george'
select * from producers where producer in ('columbia','george')

--6) Retrieve movietitle and Producer from Movie table for Rating ='PG' and Rental
  Price = 1.5
select movietitle,producer,rating,rentalprice from movies where rating ='pg' and
  rentalprice=1.5

select * from movies
```

```
--1)a. List all details of all Shippers that the company is dealing with.*/
select * from shippers

-- b. List all details of Shippers with the output presented in ascending order of 
  shipper names.
 select * from shippers order by companyname

-- 2)a. List all employees - you need to display only the details of their First 
  Name, Last Name, Title, Date of birth and their city of residence.
select firstname, lastname,title,birthdate,city from employees

--b. Based on the designations (Titles) available in the employees table, can you 
  extract the designation list?
select distinct title from employees

--3) Retrieve the details of all orders made on 19 May 1997
select * from orders where orderdate = '19 May 1997'
select * from orders where orderdate = '1997-05-19'
select * from orders where orderdate in ('1997-05-19')

--4) Retrieve details of all customers that are located in the cities of London or 
  Madrid.
select * from customers where city in ('london','madrid')
select * from customers where city ='london' or city='madrid'

--5) List all customers (Customer number and names) who are located in UK. The list
   should be produced in alphabetical order of customer names.
select customerid,contactname from customers where country ='uk' order by 
  contactname

--6) Provide a list of Orders (Order IDs and order dates) made by customer whose ID
   is 'Hanar'.
select customerid,orderid,orderdate from orders where customerid='hanar'

--7) List the Fully Qualified Names of All Northwind Employees as a single column. 
  Fully qualified Names should look like this: Dr. Venkat Raman OR Ms Esther Tan, 
  where Ms is the Title of courtesy Esther is first name and Tan is last name.
--Hint: You may need to use string concatenation. Is it possible that this is 
  listed in alphabetical order of the last names?
select titleofcourtesy + ' '+ firstname +' '+lastname from employees order by 
  lastname

-- 8) List all Orders (Order number and date) of the orders made by the Customer 
  "Maison Dewey" (column: company name). Note: Maison Dewey is the name of the 
  customer.
select o.orderid,o.orderdate,c.companyname
from orders o, customers c
where o.customerid=c.customerid and c.companyname = 'Maison Dewey'

--9) Retrieve the details of all Products' having the word "lager" in the product 
  name.
select *
from products
where productname like '%lager%'
```

```sql
--10) Retrieve the Customer IDs and contact names of all customers who have yet to ↩
  order any products.
select customerid, contactname
from customers
where customerid not in(select customerid from orders)

--11) Display the average product price.
select avg(unitprice)
from products

--12) Prepare a list of cities where customers reside in. The list should not ↩
  contain any duplicate cities.
select distinct city
from customers

--13) Retrieve the number of customers who has made orders.
select count(distinct customerid)
from orders

-- 14) Retrieve the company name and phone number of customers that do not have any↩
   fax number captured.
select companyname,phone,fax
from customers
where fax is null

-- 15) Retrieve the total sales made. Assume sales is equal to unit price * ↩
  quantity.
select sum(unitprice*quantity)
from [order details]

--16) List order ids made by customer 'Alan Out' and 'Blone Coy’
select o.orderid,c.CompanyName
from orders o,customers c
where o.CustomerID=c.CustomerID
and companyname in ('alan out','blone coy')

/* second part*/
--17) Prepare a list of customer ids and the number of orders made by the ↩
  customers.
select customerid,count(orderid) as 'num of orders'
from orders
group by CustomerID

--18) Retrieve company name for the customer id ‘BONAP’, and also order ids made by↩
   him.
select c.companyname,o.orderid,c.CustomerID
from customers c,orders o
where c.CustomerID=o.customerid
and c.CustomerID='bonap'

--19)
--a. Retrieve the number of orders made, IDs and company names of all customers ↩
  that have made more than 10 orders.
--The retrieved list should be display in the descending order of ‘number of orders↩
   made’.
```

```sql
select count(o.orderid) as 'no. of orders made',c.customerid,c.companyname
from orders o,customers c
where o.CustomerID=c.CustomerID
group by c.customerid,c.companyname
having count(o.orderid)>10
order by count(o.orderid) desc

--b. Retrieve the number of orders made, IDs and company names for the customer 
  with customer id ‘BONAP’.
select count(o.orderid),c.customerid,c.companyname
from orders o, customers c
where o.CustomerID=c.CustomerID and c.CustomerID='bonap'
group by c.customerid,c.companyname

--c. Retrieve the number of orders made, IDs and company names of all customers 
  that have more orders than customer with customer id ‘BONAP’.
select count(o.orderid),c.customerid,c.companyname
from customers c, orders o
where o.CustomerID=c.CustomerID
group by c.customerid,c.companyname
having count(*)>(select count(o.orderid) from orders o, customers c where 
  o.CustomerID=c.CustomerID and c.CustomerID='bonap')

--20)
--a. Prepare a Product list belonging to Beverages and Condiments categories (you 
  may use in your SQL statement Categories Codes 1 and 2). The list should be 
  sorted on Product code and Product Name.
select productid,productname
from products
where Categoryid in ('1','2')
order by productid,productname

--b. How would the above query change if you are not provided category codes but 
  only the names "Beverages", "Condiments" for retrieval. Would this require a join
   or subquery?
-- subquery:
select productid,productname
from products
where categoryid in (select categoryid from categories where categoryname in 
  ('beverages','condiments'))
order by productid,productname

-- PJE(last time): Yes need to do join
select p.productid,p.productname
from products p,Categories c
where p.CategoryID=c.CategoryID and c.CategoryName in ('beverages','condiments')
order by productid,productname

--21)
--a. How many employees do we have in our organization?
select count(*) as ' no. of employees'
from employees

--b. How many employees do we have in USA?
select count(*) as ' no. of employees'
from employees
```

```sql
where country='usa'

--22) Retrieve all details of Orders administered by persons who hold the designation
-- Sales Representative and shipped by United Package.
select *
from orders
where employeeid in (select employeeid from employees where title='sales representative')
and shipvia in (select shipperid from shippers where companyname in ('united package'))
--or

select o.*
from orders o, employees e,shippers s
where o.employeeid=e.employeeid
and o.shipvia=s.ShipperID
and e.title='sales representative'
and s.CompanyName='united package'

-- 23) Retrieve the names of all employee. For each employee list the name of his/her manager in adjacent columns.
select empe.lastname+' '+empe.firstname as employeename,empr.lastname+' '+empr.firstname as managername
from employees empe left outer join employees empr
on empe.reportsto=empr.EmployeeID

--24) Retrieve the five highest ranking discounted product. "Discounted Product" indicates products with the total largest discount (in dollars) given to customers.
--JE: wrong the discount amount is for one unit.
select top 5 p.productname,sum(od.unitprice*od.quantity*od.discount)as totaldiscount
from [order details] od,products p
where od.productid=p.productid
group by p.productname
order by sum(od.unitprice*od.quantity*od.discount) desc

--25) Retrieve a list of Northwind’s Customers (names) who are located in cities where there are no suppliers.
select companyname
from customers
where city not in (select city from suppliers)

--26) List all those cities that have both Northwind’s Supplier and Customers.
-- via subquery
select distinct city
from customers
where city in (select city from suppliers)

-- or innerjoin
select distinct c.city
from customers c,suppliers s
where c.city=s.city

--27)
```

```sql
--a. Northwind proposes to create a mailing list of its business associates. The
  mailing list would consist of all Suppliers and Customers
--collectively called Business Associates here. The details that need to be
  captured are the Business Associates' Names, Address and Phone.
select companyname as 'business associates names',address,phone from customers
union select companyname as 'business associates names',address,phone from
  suppliers

--b. Is it possible for you to add on to the same list Northwind's Shippers also.
--Since we do not have address of shippers, it is sufficient only phone is included
   leaving the address column blank.
select companyname as 'business associates names',address,phone from customers
union select companyname as 'business associates names',address,phone from
  suppliers
union select companyname as 'business associates names',null,phone from shippers
order by 'business associates names'

--28) Retrieve the manager’s name of the employee who has handled the order 10248.
--innerjoin and self join
select empr.lastname+empr.firstname as 'employer name',empe.lastname+empe.firstname
   as 'employee name',o.orderid
from employees empe,employees empr,orders o
where empe.ReportsTo=empr.EmployeeID and empe.EmployeeID=o.EmployeeID and
  o.orderid=10248

--subquery and self-join
select empr.lastname+empr.firstname as 'employer name',empe.lastname+empe.firstname
   as 'employee name'
from employees empe,employees empr
where empe.ReportsTo=empr.EmployeeID
and empe.EmployeeID in(select employeeid from orders where orderid=10248)

--29) List the product name and product id with unit price greater than average
  unit product price.
select productname, productid,unitprice
from products
where unitprice > (select avg(unitprice) from products)

-- 30) List all the orders (order number and amount) that exceed $10000 value per
  order. Amount means Quantity*Price.
select orderid,sum(quantity*(unitprice-discount)) as amount
from[order details]
group by orderid
having sum(quantity*(unitprice-discount))>10000

--31) List all the orders that exceed $10000 value per order. Your list should
  include order number and customer id.
select od.orderid,o.customerid,sum(od.quantity*(od.unitprice-od.discount)) as
  amount
from[order details] od, orders o
where o.orderid=od.orderid
group by od.orderid,o.customerid
having sum(od.quantity*(od.unitprice-od.discount))>10000

--32) List all the orders that exceed $10000 value per order. Your list should
  include order number and customer id and customer name.
```

```
select od.orderid,o.customerid,c.companyname,sum(od.quantity*(od.unitprice-
  od.discount)) as amount
from[order details] od, orders o, customers c
where o.orderid=od.orderid and c.customerid=o.customerid
group by od.orderid,o.customerid,c.companyname
having sum(od.quantity*(od.unitprice-od.discount))>10000

--33) List the total orders made by each customer. Your list should have customer
  id and Amount (Quantity * Price) for each customer.
select o.customerid,sum(od.quantity*od.unitprice) as amount
from [order details] od,orders o
where o.orderid=od.orderid
group by o.customerid

-- 34) Retrieve the Average Amount of business that a northwind customer provides.
  The Average Business is total amount for each customer divided by the number of
  customer.
select sum(od.unitprice*od.quantity)/count(distinct o.customerid) from orders o,
  [order details] od where o.orderid=od.orderid

--35) List all customers (Customer id, Customer name) who have placed orders more
  than the average business that a northwind customer provides.
select o.customerid,c.companyname,sum(od.unitprice*od.quantity) as amount
from orders o,customers c,[order details] od
where o.CustomerID=c.customerid and od.orderid=o.orderid
group by o.customerid,c.companyname
having sum(od.unitprice*od.quantity)>(select sum(od.unitprice*od.quantity)/count
  (distinct o.customerid) from orders o,[order details] od where
  o.orderid=od.orderid)
order by amount

--36) List the total orders made by each customer. Your list should have customer
  id and Amount (Quantity * Price) for each customer in the year 1997. (Use year
  (orderdate) to retrieve the year of the column orderdate)
select o.customerid,sum(od.quantity*od.unitprice)
from orders o,[order details] od
where o.orderid=od.orderid and year(orderdate) in ('1997')
group by o.customerid

-- so when using count (*), it counts the unique combination in the group by (as a
  PK related)
```

```sql
-- 06a Workshop on SQL DDL
/*
1. Create a Table called MemberCategories with the following fields
MemberCategory nvarchar(2)
MemberCatDescription nvarchar(200)
None of the fields above can be null. Set the MemberCategory as the Primary key.
*/
Create table MemberCategories
(MemberCategory nvarchar(2) not null,
MemberCatDescription nvarchar(200) not null,
primary key (membercategory))

/*
2. Add the following data into the MemberCategories Table:
MemberCategory MemberCatDescription
A Class A Members
B Class B Members
C Class C Members
*/
Insert into MemberCategories
(MemberCategory,MemberCatDescription)
values ('A','Class A Members')
Insert into MemberCategories
(MemberCategory,MemberCatDescription)
values ('B', 'Class B Members')
Insert into MemberCategories
(MemberCategory,MemberCatDescription)
values ('C', 'Class C Members')

select * from MemberCategories

/*
3. Create a Table called GoodCustomers with the following fields:
CustomerName nvarchar(50)
Address nvarchar(65)
PhoneNumber nvarchar(9)
MemberCategory nvarchar(2)
Only Customer Name and Phone Number is mandatory.
Since there could be two customers having the same name, make CustomerName and
  Phone Number as a composite primary key.
The MemberCategory should have a referential integrity to the MemberCategories
  Table so that only those categories that have been listed in MemberCategories
  Table could be entered.
*/
create table GoodCustomers
(CustomerName nvarchar(50)  not null,
Address nvarchar(65),
PhoneNumber nvarchar(9) not null,
MemberCategory nvarchar(2),
primary key (customername,phonenumber),
foreign key (membercategory) references membercategories(membercategory))

-- 4. Insert into GoodCustomer all records form Customer table with corresponding
  fields except Address, which is to be left Null.
--Only Customers having Member Category ‘A’ or ‘B’ are good customers hence the
  table should be inserted only those records from the Customers table.
```

```
insert into goodcustomers (customername,phonenumber,membercategory)
select customername,phonenumber,membercategory
from Customers
where MemberCategory in ('A','B')

/*
5. Insert into GoodCustomers the following new customer.
CustomerName = Tracy Tan
PhoneNumber = 736572
MemberCategory = 'B'
*/
insert into goodcustomers (customername,phonenumber,membercategory)
values ('Tracy Tan',736572,'B')

/*
6. Insert into GoodCustomers table the following information for a new customer
the column names.
CustomerName = Grace Leong
Address = 15 Bukit Purmei Road, Singapore 0904'
PhoneNumber = 278865
MemberCategory = 'A'
*/
insert into goodcustomers (customername,phonenumber,Address,membercategory)
values ('Grace Leong',278865,'15 Bukit Purmei Road, Singapore 0904','A')

/*
7. Insert into GoodCustomers table the following information for a new customer
Since all the columns are provided you may insert the record without specifying the
   column names.
CustomerName = Lynn Lim
Address = 15 Bukit Purmei Road, Singapore 0904'
PhoneNumber = 278865
MemberCategory = 'P'
Does the command go through – It should not since member category ‘P’ is not
  defined in MemberCategories Table.
(Violation of referential integrity)
*/
insert into goodcustomers
values ('Lynn Lim','15 Bukit Purmei Road, Singapore 0904',278865,'P')

-- foreign key constraints cannot enter this into the data.

--8. Change the Address of Grace Leong so that the new address is '22 Bukit Purmei
  Road, Singapore 0904' in GoodCustomers table.
Update GoodCustomers
set address='22 Bukit Purmei Road, Singapore 0904'
where Customername='Grace Leong'

--9. Change the Member Category to ‘B’ for customer whose Customer ID is 5108 in
  GoodCustomers table.
Update GoodCustomers
set MemberCategory='B'
where customername =(select customername from customers where customerid=5108)

-- JE: at this point the dependent table didnt amend the figures if i change the
  parent table directly.
```

```sql
--how to amend the subsequent table to make it efficient? need to do cascade


--10. Remove Grace Leong from GoodCustomers table.
Delete from goodcustomers
where customername='Grace Leong'

--11. Remove customers with ‘B’ member category in GoodCustomers table.
Delete from goodcustomers
where membercategory='B'

--12. Add column FaxNumber (nvarchar(25)) to GoodCustomers table.
alter table goodcustomers
add FaxNumber nvarchar(25)

--13. Alter the column Address to nvarchar(80) in GoodCustomers table.
alter table goodcustomers
alter column address nvarchar(80)

--14. Add column ICNumber (nvarchar(10)) to GoodCustomers table.
alter table goodcustomers
add ICnumber nvarchar(10)

--15. Create a unique index ICIndex on table GoodCustomers bases on ICNumber.
--Notice that the column ICNumber have no values. Can you create the unique index
  successfully? Why?
create unique index ICindex on goodcustomers(ICnumber)
-- JE: cannot create as the icnumber has duplicate values, not unique value for
  each icnumber.

--16. Create an index on table GoodCustomers based on FaxNumber.
create index faxindex on goodcustomers(faxnumber)

--17. Drop the index created on FaxNumber.
drop index faxindex on goodcustomers

--18. Remove the column FaxNumber from GoodCustomer table.
alter table goodcustomers
drop column faxnumber

--19. Delete all records from GoodCustomers.
delete from goodcustomers

---20. Drop the table GoodCustomers.
drop table GoodCustomers
```

```sql
--Exercise Userview 1
--•Create a View Customer1998 containing Customer IDs and names, Product IDs and 
  names for customers who have made orders on the year 1998.
create view Customer1998 as
select c.customerid,c.companyname,od.productid,p.productname
from customers c, [order details] od,orders o,products p
where c.CustomerID=o.CustomerID and o.orderid=od.OrderID and 
  od.ProductID=p.ProductID
and year(o.OrderDate)=1998
group by c.customerid,c.companyname,p.productid,p.productname,o.orderdate

select * from Customer1998

 --Exercise Userview 2
--•Using the View Customer1998, retrieve the Customer name, Product name and 
  supplier names for the Customers who have made orders on the year 1998 according 
  to Customer Name.
select c.companyname,c.productname,p.SupplierID
from Customer1998 c,products p,suppliers s
where c.productid=p.ProductID and p.supplierid=s.supplierid
group by c.companyname,c.productname,p.SupplierID

-- Exercise Userview 3
--•Retrieve the Customer name and the number of DISTINCT products ordered by them 
  in the year 1998.
select companyname,count(distinct productid) as 'no.of products ordered'
from Customer1998
group by companyname

--Exercise Userview 4
--a)Create an Userview to represent total business made by each customer. The 
  userview includes two columns:
---The sum of product’s unit price multiplied by quantity ordered by the customer
---Customer id
create view totalbusiness as
select sum(od.unitprice*od.quantity) as totalprice,o.customerid
from [Order Details] od,orders o
where od.OrderID=o.Orderid
group by o.CustomerID

select * from totalbusiness

--b)Using the userviewcreated, retrieve the Average Amount of business that a 
  northwindcustomer provides. The Average Business is total amount for each 
  customer divided by the number of customer.
select sum(tb.totalprice)/count(distinct CustomerID)
from totalbusiness tb

--Exercise Userview 5
--a)Create an Userview to represent employee details with employee id, last name 
  and title
create view employeedetails as
select employeeid, lastname,title
from Employees
group by employeeid,lastname,title
```

```
drop view employeedetails

select * from employeedetails
```

```
--Exercise Stored Procedure 1
--•Write a stored procedure that would list all members who belong to ‘A’ category.
create procedure amembers as
begin
select * from customers where MemberCategory='A'
end

--•Write statements to call this procedure.
exec amembers

--• Exercise Stored Procedure 2
--•Write stored procedure that would take as parameter (argument) a member category
   and list all the members belonging to that category.
create procedure searchmember (@var1 nvarchar(2)) as
select * from customers where MemberCategory= @var1;

--•Write calling Statements to call the procedure and test the stored procedure for
   various inputs.
--a)What is the output if the argument is ‘B’?
exec searchmember 'b'

--b)What is the output if the argument is ‘Z’?
exec searchmember 'z'

--Exercise Stored Procedure 3
--•Write a stored procedure that update customer address based on the customer
  name.
create procedure updateaddress (@var1 nvarchar(50)) as
begin
update Customers
set address= 'Blk 26, Telok Blangah Crescent #22-87, Singapore 0409'
where customername=@var1
end

--•Write statements to test this procedure. (change customername=Maria Anders, from
   Blk 30, Telok Blangah Crescent #22-87, Singapore 0409)
exec updateaddress 'maria Anders'

-- Exercise Stored Procedure 4
--•Write stored procedure that insert customer record, with input parameters as
  customer id, customer name, member category, address and postal code.
create procedure insertrecord (@id nvarchar(4),@name nvarchar(50),@membercat
  nvarchar(2),@add nvarchar(65),@postal nvarchar(6)) as
begin
insert into Customers (CustomerID,CustomerName,MemberCategory,Address,PostalCode)
values(@id,@name,@membercat,@add,@postal)
end


--•Write statements to test this procedure. (add in this: 0001,'Jayne','A','choa
  chu kang',689093)
exec insertrecord 0001,'Jayne','A','choa chu kang',689093

select* from customers order by CustomerID

-- Exercise Stored Procedure 5
```

```
--•Write a stored procedure that delete a customer record based on the customer id.
create procedure deletecustrecord (@id nvarchar(4)) as
begin
delete from Customers
where CustomerID=@id
end

--•Write statements to test this procedure.
exec deletecustrecord 0001

select* from customers order by CustomerID

-- Exercise Stored Procedure 6
-- •Write a stored procedure to retrieve all video code rented by a customer, with ⮒
  customer name as the input parameter.
create procedure retrievevideocode (@custname nvarchar(50))as
begin
select i.videocode,c.CustomerName
from IssueTran i,Customers c
where i.CustomerID=c.CustomerID and c.CustomerName=@custname
end

drop procedure retrievevideocode

--•Write statements to test this procedure. (check for patricia mckenna)
exec retrievevideocode 'patricia mckenna'
```