**There are four workouts**

**To be completed before Sunday 30-Jan-2022**

**Workout-1 (Postgresql DVDRENTAL Database)**

**Problem Statement**: Write an SQL query to fetch actors' names along with the count of films as well as pipe (|) separated film names.

**Hint**: Use tables – actor, film\_actor, film

**Column Names**:

1. Actor Name – This should be concatenation of actor's first name and last name separated by a space.
2. Films Done – This is the count of total films done by each actor.
3. Film Names – This is the concatenation, pipe-separated, of all films done by each actor.

select

temp.actor\_names,

count(temp.actor\_names) as films\_done

,string\_agg(temp.film\_name, '| ' order by temp.film\_name) as film\_names from

(

select

actor.first\_name || ' ' || actor.last\_name as actor\_names

,film.title as film\_name

from film

join film\_actor on film.film\_id = film\_actor.film\_id

join actor on film\_actor.actor\_id = actor.actor\_id

) as temp

group by temp.actor\_names;

**Workout-2 (Postgresql DVDRENTAL Database)**

**Problem Statement**: Write an SQL query to fetch customers' names and Staffs' names along with the total payment and maximum duration of rental.

**Hint**: Use tables – customer, payment, staff, rental

**Column Names**:

1. Customer Name – This should be concatenation of customer's first name and last name separated by a space.
2. Staff Name – This should be concatenation of staff's first name and last name separated by a space.
3. Total Payment – This should be the total payment per customer.
4. Maximum Duration – This is the maximum rental duration per customer per staff.

select distinct concat(customer.first\_name ,' ' , customer.last\_name) as "Customer Name" ,

concat(staff.first\_name , ' ', staff.last\_name) as "Staff Name",

sum(payment.amount) over (partition by customer.customer\_id) "Total Payment",

max(rental.return\_date -rental.rental\_date)over(partition by customer.customer\_id, staff.staff\_id) "Maximum duration"

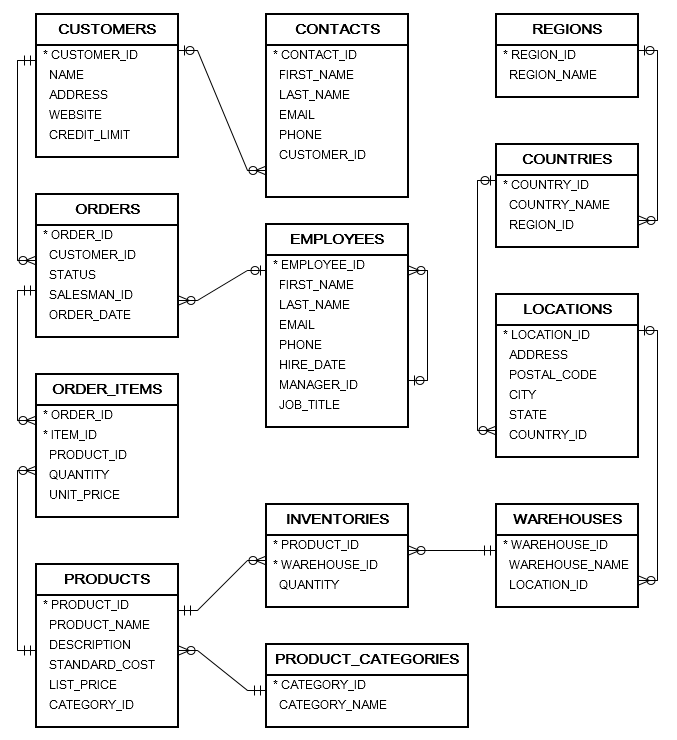
from customer join payment on customer.customer\_id = payment.customer\_id

join staff on payment.staff\_id = staff.staff\_id

join rental on payment.rental\_id = rental.rental\_id

order by 1,2;

**Use the below schema diagram for better understanding of oracle table associations.**



**Workout-3 (Oracle Sample Data, Oracle database)**

**Problem Statement**: Write an SQL query to fetch warehouse and product names whose total inventory amount is either less than or equal to 50 or more than 1000. Data should be ordered by total inventory in decreasing order.

**Hint**: Use tables – warehouses, inventories, products

**Column Names**:

1. Warehouse Name – This should be name of warehouse.
2. Product Name – This should be name of product.
3. Total Inventory – This is the total inventory amount per warehouse per product.

select "Warehouse Name"

,"Product Name"

,"Total Inventory"

from

(

select w.warehouse\_name "Warehouse Name"

,p.product\_name "Product Name"

,sum(i.quantity) "Total Inventory"

from Warehouses w

Join Inventories i on i.warehouse\_id=w.warehouse\_id

Join Products p on p.product\_id=i.product\_id

group by

w.warehouse\_name

,p.product\_name

) where

"Total Inventory" <= 50 or "Total Inventory" > 1000

order by "Total Inventory";

**Workout-4 (Oracle Sample Data, Oracle database)**

**Problem Statement**: Write an SQL query to fetch two largest order (by order amount) for each employee, where order is done for product category as CPU and order status is Shipped.

**Hint**: Use tables – employees, orders, order\_items, products, product\_categories

**Column Names**:

1. ManagerName – Name of the employee's manager.
2. EmployeeName – Name of the employee combining first name and last name.
3. OrderStatus – Status of the order.
4. OrderAmount – Multiply order quantity and list price.
5. ProductCategory – Category of the product.
6. ProductName – Name of the product.
7. ProductDescription – Description of the product.

SELECT employee\_name

,manager\_name

,status

,order\_amount

,product\_category

,product\_name

,description

FROM

(

SELECT employee\_name, manager\_name

,orders.status

,quantity\*list\_price AS order\_amount

,category\_name AS product\_category

,products.product\_name

,products.description,

row\_number() over (partition by employee\_id ORDER BY quantity\*list\_price desc) AS row\_num FROM

(SELECT A.employee\_id AS employee\_id,

A.first\_name || ' ' || A.last\_name AS employee\_name,

B.first\_name || ' ' || B.last\_name AS manager\_name

FROM employees A,employees B

WHERE

A.manager\_id = B.employee\_id(+)

) t, orders, order\_items, products, product\_categories

WHERE

employee\_id = orders.salesman\_id

AND orders.order\_id = order\_items.order\_id

AND order\_items.product\_id = products.product\_id

AND products.category\_id = product\_categories.category\_id

AND product\_categories.category\_name = 'CPU'

AND orders.status = 'Shipped'

) temp

WHERE row\_num in (1,2);