Chris Harms

DBMS Lab 4

* 1. select c.customer\_id, NVL((o.shipped\_date - o.order\_date),-99) as time\_to\_ship

from customers c right outer join orders o

on c.customer\_id = o.customer\_id

where c.customer\_city = 'Los Angeles';

A screenshot of a social media post

Description automatically generated

* 1. select o.customer\_id, NVL((o.shipped\_date - o.order\_date),-99) as time\_to\_ship

from orders o

where o.customer\_id in (select c.customer\_id

from customers c

where c.customer\_city = 'Los Angeles');

A screenshot of a social media post

Description automatically generated

* 1. select o.customer\_id, NVL((o.shipped\_date - o.order\_date),-99) as time\_to\_ship

from orders o

where o.customer\_id in (select c.customer\_id

from customers c

where c.customer\_city = 'Los Angeles'

and c.customer\_id = o.customer\_id);

A screenshot of a social media post

Description automatically generated

* 1. select o.customer\_id, NVL((o.shipped\_date - o.order\_date),-99) as time\_to\_ship

from orders o

where exists (select c.customer\_id

from customers c

where c.customer\_city = 'Los Angeles'

and c.customer\_id = o.customer\_id);

A screenshot of a social media post

Description automatically generated

* 1. select avg(invoice\_total) as quantity\_total\_avg

from invoices;

A screenshot of a cell phone

Description automatically generated

* 1. select vendor\_id, avg(invoice\_total) as vendor\_avg

from invoices

group by vendor\_id;

A screenshot of a cell phone

Description automatically generated

* 1. select vendor\_id, avg(invoice\_total) as vendor\_avg

from invoices

group by vendor\_id

having avg(invoice\_total) > (select avg(invoice\_total) from invoices);

A screenshot of a social media post

Description automatically generated

* 1. select distinct vendor\_id

from invoices

where invoice\_total > (select avg(invoice\_total) from invoices);

A screenshot of a social media post

Description automatically generated

* 1. select o.order\_id

from order\_details o inner join items i

on o.item\_id = i.item\_id

where i.artist = 'Burt Ruggles'

minus

select o.order\_id

from order\_details o inner join items i

on o.item\_id = i.item\_id

where i.artist = 'Jess & Odie';

A screenshot of a social media post

Description automatically generated

* 1. select o.order\_id

from order\_details o inner join items i

on o.item\_id = i.item\_id

where i.artist = 'Burt Ruggles'

intersect

select o.order\_id

from order\_details o inner join items i

on o.item\_id = i.item\_id

where i.artist = 'Jess & Odie';

A screenshot of a cell phone

Description automatically generated

* 1. select o.order\_id

from order\_details o inner join items i

on o.item\_id = i.item\_id

where i.artist = 'Burt Ruggles'

union

select o.order\_id

from order\_details o inner join items i

on o.item\_id = i.item\_id

where i.artist = 'Jess & Odie';

A screenshot of a cell phone

Description automatically generated

* 1. select o.order\_id

from order\_details o inner join items i

on o.item\_id = i.item\_id

where i.artist = 'Burt Ruggles'

union all

select o.order\_id

from order\_details o inner join items i

on o.item\_id = i.item\_id

where i.artist = 'Jess & Odie';

A screenshot of a social media post

Description automatically generated

1. select o.order\_id

from order\_details o inner join items i

on o.item\_id = i.item\_id

where i.artist = 'Burt Ruggles'

and o.order\_id in (select o.order\_id

from order\_details o inner join items i

on o.item\_id = i.item\_id

where i.artist = 'Jess & Odie');

I assumed that query ‘b’ was the 2nd query from question 3, which uses intersect, not union.

A screenshot of a social media post

Description automatically generated

1. select c.customer\_id, NVL(sum(d.order\_qty \* i.unit\_price),0) as total\_amount

from customers c full outer join orders o

on c.customer\_id = o.customer\_id

full outer join order\_details d

on o.order\_id = d.order\_id

full outer join items i

on d.item\_id = i.item\_id

group by c.customer\_id;

A screenshot of a social media post

Description automatically generated

1. select customer\_id, total\_amount, case

when total\_amount <= 50 then 'very bad'

when total\_amount > 50 and total\_amount <= 100 then 'bad'

when total\_amount > 100 and total\_amount <= 150 then 'ok'

when total\_amount > 150 and total\_amount <= 200 then 'good'

else 'very good'

end customer\_status

from (select c.customer\_id, NVL(sum(d.order\_qty \* i.unit\_price),0) as total\_amount

from customers c full outer join orders o

on c.customer\_id = o.customer\_id

full outer join order\_details d

on o.order\_id = d.order\_id

full outer join items i

on d.item\_id = i.item\_id

group by c.customer\_id)

order by customer\_id;

A screenshot of a social media post

Description automatically generated