Relational Divides - How they work

1.15 List the departments that have at least one sale of all the items delivered to them

This query is known as a Relational Divide take your time to look at the SQL and understand what the NOT EXISTS clause is saying in each variation of the query.

Attempt 1:

```
SELECT DISTINCT DepartmentID
FROM deliveryitem del1
WHERE NOT EXISTS
     (SELECT *
      FROM deliveryitem del2
      WHERE del2.DepartmentID = del1.DepartmentID
AND NOT EXISTS
     (SELECT *
       FROM saleitem natural join sale
       WHERE del2.ItemID = saleitem.ItemID
             AND del1.DepartmentID = sale.DepartmentID));
```

Firstly NOT EXISTS means if there are no rows that evaluates to true if there are rows it evaluates to false. Therefore if the departmentid from deliveritem d1 matches a row in deliveritem d2 a value is in the set and there not exists evaluates to FALSE. If we list the result sets for the departments that have received items

```
select distinct(departmentid), itemid
from deliveryitem
order by departmentid, itemid;
```

The result set is (departmentid, itemid)

{(2,3), (2,5), (2,6), (2,9), (2,12), (2,14), (2,17),

(3,1), (3,8), (3,12), (3,14),(3,17), (3,18), (3,22),(3,23),(3,24),(3,25),

(4,2), (4,3), (4,12), (4,14), (4,15), (4,16), (4,17),

(5,12), (5,14), (5,17),

(6,3), (6,5), (6,6), (6,9), (6,10), (6,11), (6,12), (6,13), (6,14), (6,17),

(7,5), (7,9), (7,14), (7,19), (7,19), (7,20), (7,21) }

This automatically tells us that departmentids 1,8,9,10 & 11 will not be in our result set because they have not received a delivery

We then look at the departments that have sold items

```
SELECT distinct(departmentid), saleitem.itemid
FROM saleitem inner join sale
on sale.saleid = saleitem.saleid
order by departmentid, itemid;
```

This result set is (departmentid, itemid)

{(2,1), (2,3), (2,5), (2,6), (2,9), (2,12), (2,14), (2,17),
(3,8), (3,12), (3,14), (3,18),(3,22), (3,23), (3,24), (3,25),
(4,3), (4,12), (4,14), (4,15), (4,16), (4,17),
(5, 12), (5,14), (5,17) ,
(6,3), (6,6), (6,9), (6,10), (6,11), (6,12), (6, 14), (6,17),
(7,14), (7,19), (7,20), (7,21)}

Let's look at the result sets side by side

DeliveryItem (departmnetid, itemid)		Sale/SaleItem (departmentid, itemid)
(2,3), (2,5), (2,6), (2,9), (2,12), (2,14), (2,17)		(2,1), (**2,3**), (**2,5**), (**2,6**), (**2,9**), (**2,12**), (**2,14**), (**2,17**),
(**3,1**), (3,8), (3,12), (3,14),(3,17), (3,18), (3,22),(3,23),(3,24),(3,25)		(3,8), (3,12), (3,14), (3,18),(3,22), (3,23), (3,24), (3,25),
(**4,2**), (4,3), (4,12), (4,14), (4,15), (4,16), (4,17)		(4,3), (4,12), (4,14), (4,15), (4,16), (4,17),
(5,12), (5,14), (5,17)		(**5, 12**), (**5,14**), (**5,17**)
(6,3), (6,5), (6,6), (6,9), (6,10), (6,11), (6,12), (**6,13**), (6,14), (6,17)		(6,3), (6,6), (6,9), (6,10), (6,11), (6,12), (6, 14), (6,17)
(**7,5**), (**7,9**), (7,14), (7,19), (7,19), (7,20), (7,21)		(7,14), (7,19), (7,20), (7,21)

Table 1: The result set in DeliveryItem must be found for the department result set for Sale/SaleItem.This is true for Departments 2 & 5 only (note the DeliveryItem ItemID is a subset of the Sale/SaleItem result set for department id 2, as item id 1 was in stock and sold but has not been delivered)

Remember the query wants to return the departments that have sold one item of every item delivered. The only departments where this is true are departments 2 & 5.

Consider the department 3 (row 3) result sets.
The SELECT clause is selecting department 3 from the deliveryitem table it then joins to the delivery item in the first subquery and finds DepartmentID 3, ItemID 1 the result set (3,1). As the record is found and the NOT EXISTS condition is evaluated to FALSE as a record exists.

Now we need to find a FALSE record for the Sale, however result set (3,1) does NOT EXIST in the Sale, SaleItem subquery - and evaluates to TRUE. As TRUE != FALSE the result set is not returned.

As this has an AND condition (the department ID must be present in the deliveryitem and sale tables)

This process repeats for every result set returned by the queries. Only when TRUE=TRUE will a result set be returned this is because of the join to the table DeliveryItem (aliased as del1) in both subqueries.