INFO90002 Week 5 Tutorial Solutions

Objectives:

This week the focus in labs is to write, execute and understand how SQL works

- SQL Sub Queries 5 min
- Multi (3+)-table joins 5 min
- HAVING SQL clause 5 min
- LEARNING SQL by Example 35 min (including the UNION clause)
- HOMEWORK: Using the MySQL reference manual and SQL Functions

Section 1 Sub Queries,

Sub Queries

Sometimes we need to find a value and then use it. For example, if we wanted to find out the DepartmentID of the department which has the lowest salary.

First, we would find the lowest salary across all of the department store.
You might be tempted to write a query like this

```
SELECT MIN(Salary), DepartmentID
FROM employee;
```

But if we check this result set, we will see that this result is wrong.

```
SELECT MIN(salary)
FROM employee
WHERE DepartmentID =1;
```

OR

```
SELECT MIN(Salary), DepartmentID
FROM employee
GROUP BY DepartmentID;
```

Department 1 does not have the lowest salary! While we can use LIMIT keyword – it is not always wise to do so.

So, we break the problem into two components. Find the lowest salary, then find the DepartmentID rows where the salary matches the value of the first query.

```
SELECT MIN(Salary)
FROM employee;
```

The result set is one value {41000} we then use that value in our next query

```
SELECT DepartmentID
FROM employee
WHERE Salary = 41000;
```

This can be done in the one statement:

```
SELECT DepartmentID
FROM employee
WHERE Salary =
    (SELECT MIN(Salary)
     FROM employee);
```

DepartmentID
4

The query in parenthesis is known as the "inner query" and the other query is known as the "outer query".

This is doing exactly the same as the two individual SQL statements above it but in one statement. The query in parenthesis is run first and the result returned (41000). Then each row in the outer query evaluates the salary value with the value returned in the inner query (41000). The only match is where Department ID is 4.

Important: if the subquery returns more than one result "=" will not work you must use "IN"

How many employees work in departments are on the fifth floor?

To approach this question, we break it into separate components.

1. What department ID's are on the fifth floor
2. Count the number of employees whose DepartmentID matches the results returned in the inner query who are in the same department as the first query

Query 1 The "inner" query

```
SELECT DepartmentID
FROM department
WHERE Floor = 5;
```

DepartmentID
1
8
9
10
11

Returns the result set {1,8,9,10,11} that is five rows.

Now we need to count each row in the Employee table where the DepartmentID matches 1 or 8 or 9 or 10 or 11.

As the result set has more than one row we need to use the keyword IN

```
SELECT COUNT(EmployeeID) as EMP_COUNT_FLOOR5
FROM employee
```

```
WHERE DepartmentID IN
    (SELECT DepartmentID
     FROM department
     WHERE Floor = 5);
```

	EMP_COUNT_FLOOR5
	9

TRY: If you have time replace the keyword IN with" =" and observe the error in the query window

Error Code: 1242. Subquery returns more than 1 row

Multiple table joins

Just remember to do a NATURAL J OIN between two tables they must share the same column name and data type.

As all tables in the New Department store schema (see Appendix New Department Store Physical ER Model)

Using a NATURAL JOIN

```
SELECT item.Name, sale.SaleDate, SUM(saleitem.Quantity)
FROM item NATURAL JOIN saleitem NATURAL JOIN sale
WHERE item.Name like 'TENT%'
GROUP BY item.Name, sale. SaleDate;
```

Remember, using a NATURAL JOIN both tables need to have columns of the same name, data type and purpose.

And written using INNER JOIN

```
SELECT item.Name, sale.SaleDate, SUM(saleitem.Quantity)
FROM item INNER JOIN saleitem INNER JOIN sale
ON item.ItemID = saleitem.ItemID
AND saleitem. SaleId = sale.SaleID
WHERE item.Name LIKE 'TENT%'
GROUP BY item.Name, sale. SaleDate;
```

Name	SaleDate	SUM(saleitem.Quantity)
Tent - 2 person	2017-10-14	1
Tent - 2 person	2017-10-15	1
Tent - 2 person	2017-10-24	1
Tent - 2 person	2017-10-25	2
Tent - 8 person	2017-08-19	1
Tent - 8 person	2017-12-14	1
Tent - 4 person	2017-10-15	1

HAVING clause

The HAVING clause acts like a WHERE clause but it identifies groups meeting a criterion, rather than rows. A HAVING clause usually follows a GROUP BY clause

```
SELECT department.Name, COUNT(employee.EmployeeID)
```

```
FROM department NATURAL JOIN employee
GROUP BY department.Name
HAVING COUNT(employee. EmployeeID) > 2;
```

Name	COUNT(employee.EmployeeID)
Marketing	3

Remember that WHERE works on rows, HAVING works on groups (e.g. COUNT, AVG, SUM etc.)

Section 2 SQL By Example

1.1 Find the item IDs sold by at least two departments on the second floor

```
SELECT DISTINCT ItemID
FROM sale INNER JOIN saleitem inner Join department
ON sale.SaleID = saleitem.SaleID and sale.DepartmentID =
department.DepartmentID
WHERE department.Floor=2
GROUP BY ItemID
HAVING Count(Distinct(department. DepartmentID)) > 1;
```

Alternatively

```
SELECT DISTINCT ItemID
FROM sale INNER JOIN saleitem INNER JOIN department
ON Sale.SaleID = SaleItem.SaleID
AND department.DepartmentID = sale.DepartmentID
WHERE department.DepartmentID IN
      (SELECT DepartmentID
      FROM department
      WHERE department.Floor=2)
GROUP BY ItemID
HAVING Count(Distinct(department.Departmentid)) > 1
ORDER BY ItemID;
```

There are other ways of achieving the same result set.

	ItemID
	14

Remember: If necessary, review the slides from the week 3 lecture on the types of joins.

1.2 List the departments having an average salary of over 55000

```
SELECT DepartmentID, AVG(employee.Salary)
```

```
FROM employee
GROUP BY DepartmentID
HAVING AVG(employee.Salary) > 55000;
```

You can format the output for better readability:

```
SELECT DepartmentID,
FORMAT(AVG(employee.Salary),2) AS AverageSalary
FROM employee
GROUP BY DepartmentID
HAVING AVG(employee.Salary) > 55000;
```

DepartmentID	AverageSalary
1	125.000.00
8	60.000.00
9	79.500.00
10	75.000.00
11	64.000.00

Remember that HAVING is the way to use a condition for any column that has an aggregate function used on it (e.g. AVG MAX SUM COUNT etc)

1.3 Name the items which have only been delivered by exactly one supplier

HINT: You will need to join three tables (Item, Sale and Department) together and make sure the ambiguous columns in your select statement are fully qualified e.g.:
SELECT Employee.Lastname, Department.Name
FROM Employee Natural Join Department

```
SELECT item.Name, count(distinct(SupplierID)) as SupplierCount
FROM deliveryitem INNER JOIN delivery INNER JOIN item
ON deliveryitem.DeliveryID = delivery.DeliveryID
AND deliveryitem.ItemID = item.ItemID
GROUP BY item.Name
HAVING Count(distinct(SupplierID)) = 1
ORDER BY item.Name;
```

Name	SupplierCount
BBQ - Jumbuk	1
Boots - Mens Hiking	1
Boots - Womens Goretex	1
Boots - Womens Hiking	1
Boots Riding	1
Camping chair	1
Cowboy Hat	1
Horse saddle	1
Polar Fleece Beanie	1
Sun Hat	1
Tent - 2 person	1
Tent - 4 person	1
Tent - 8 person	1

1.4 List the suppliers that have delivered at least 10 distinct items. List the supplier name and id

```
SELECT Supplier.SupplierID, Supplier.Name
FROM deliveryitem INNER JOIN supplier INNER JOIN delivery
ON deliveryitem.DeliveryID = delivery.DeliveryID
AND delivery.SupplierID = supplier.SupplierID
GROUP BY supplier.SupplierID
HAVING COUNT(DISTINCT deliveryitem.ItemID) >= 10;
```

	SupplierID	Name
	102	Nepalese Corp.
	105	All Points Inc.

1.5 Type the SQL that for each item, gives its type, the departments that sell the item, and the floor location of these departments.

> *HINT: You will need to join four tables (Item, SaleItem, Sale and Department) together and make sure each ambiguous column in your select statement is fully qualified e.g.:*
> *SELECT Employee.lastname, Department.Name*
> *FROM Employee Natural Join Department*

```
SELECT distinct(item.Name), item.Type, sale.DepartmentID, department.Floor
FROM item INNER JOIN saleitem INNER JOIN sale INNER JOIN department
on item.ItemID = saleitem.ItemID and saleitem.SaleID = sale.SaleID and
sale.DepartmentID = department.DepartmentID
ORDER BY item.Name, sale.DepartmentID;
```

Your result set should look something like this:

Name	Type	DepartmentID	Floor
BBQ - Jumbuk	F	4	3
Boots - Mens Hiking	C	3	2
Boots - Womens Goretex	C	3	2
Boots - Womens Hiking	C	3	2
Boots Riding	C	2	1
Camping chair	F	4	3
Compass - Silva	N	2	1
Compass - Silva	N	4	3
Compass - Silva	N	6	1
Cowboy Hat	C	3	2
Exploring in 10 Easy Les...	B	2	1
Geo positioning system	N	2	1
Geo positioning system	N	6	1
Gortex Rain Coat	C	2	1
Gortex Rain Coat	C	3	2
Gortex Rain Coat	C	4	3
Gortex Rain Coat	C	5	4
Gortex Rain Coat	C	6	1
How to Win Foreign Frie...	B	2	1
How to Win Foreign Frie...	B	6	1
Map case	E	6	1
Map measure	N	6	1
Pocket knife - Essential	E	2	1
Pocket knife - Essential	E	3	2
Pocket knife - Essential	E	4	3
Pocket knife - Essential	E	5	4
Pocket knife - Essential	E	6	1
Pocket knife - Essential	E	7	2
Polar Fleece Beanie	C	3	2
Sun Hat	C	3	2
Tent - 2 person	F	7	2
Tent - 4 person	F	7	2
Tent - 8 person	F	7	2
Torch	E	2	1
Torch	E	3	2
Torch	E	4	3
Torch	E	5	4
Torch	E	6	1

1.6 Name the items that are delivered by Nepalese Corp **or** sold in the Navigation department

```
SELECT DISTINCT item.Name
FROM item
WHERE ItemID IN
      (SELECT ItemID
       FROM deliveryitem NATURAL JOIN delivery NATURAL JOIN supplier
       WHERE supplier.Name = 'Nepalese Corp.')
OR ItemID IN
      (SELECT ItemID
       FROM saleitem NATURAL JOIN sale NATURAL JOIN department
       WHERE department.Name = 'Navigation');
```

UNION

You could also use the UNION clause

A UNION Join includes all data from each table that is joined. The columns selected in a UNION join must be the same.

```
SELECT DISTINCT item.Name
FROM item
WHERE ItemID IN
      (SELECT ItemID
       FROM deliveryitem NATURAL JOIN delivery NATURAL JOIN supplier
       WHERE supplier.Name = 'Nepalese Corp.')
UNION
      (SELECT item.Name
       FROM saleitem NATURAL JOIN sale
        NATURAL JOIN department NATURAL JOIN item
       WHERE department.Name = 'Navigation');
```

Note the second SELECT clause matches the first SELECT clause both choosing item.name to use separate column information use UNION ALL

	Name
	Compass - Silva
	Exploring in 10 Easy Lessons
	Geo positioning system
	How to Win Foreign Friends
	Map case
	Map measure
	Gortex Rain Coat
	Pocket knife - Steadfast
	Pocket knife - Essential
	Camping chair
	BBQ - Jumbuk
	Torch
	Tent - 2 person
	Tent - 8 person
	Tent - 4 person

1.7 Find the items that are not sold by departments on the second floor but are sold on other floors within the store

When solving problems like this, work in steps

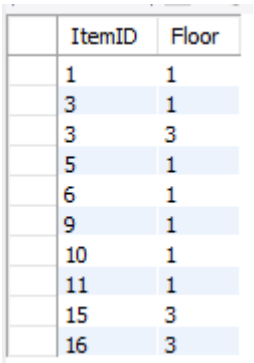
1. Identify the items sold on the second floor,

2. Then find the items that are not in the result from part 1

```
SELECT DISTINCT ItemID
FROM sale INNER JOIN saleitem INNER JOIN department
ON sale.SaleID = saleitem.SaleID AND sale.DepartmentID =
department.DepartmentID
```

```
WHERE ItemID NOT IN
   (SELECT DISTINCT ItemID
    FROM sale INNER JOIN saleitem inner Join department
    ON sale.SaleID = saleitem.SaleID and sale.DepartmentID =
department.DepartmentID
     WHERE department.Floor =2);
```

The inner query identifies the itemIDs that ARE sold on the second floor. The outer query then finds all itemIDs which have been sold but are not in the inner query but only for departments not located on the second floor.

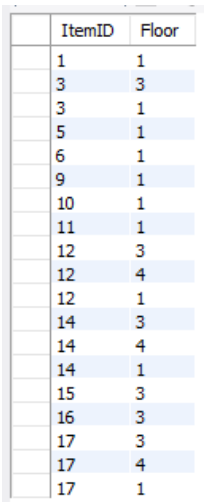


	ItemID	Floor
	1	1
	3	1
	3	3
	5	1
	6	1
	9	1
	10	1
	11	1
	15	3
	16	3

Compare that to this query

```
SELECT distinct ItemID, department.Floor
FROM sale INNER JOIN saleitem inner Join department
ON sale.SaleID = saleitem.SaleID AND sale.DepartmentID =
department.DepartmentID
WHERE department.Floor!=2
ORDER BY ItemID;
```

This query only finds items sold on floors other than the second floor – but this includes items which also happen to have been sold on the second floor.



	ItemID	Floor
	1	1
	3	3
	3	1
	5	1
	6	1
	9	1
	10	1
	11	1
	12	3
	12	4
	12	1
	14	3
	14	4
	14	1
	15	3
	16	3
	17	3
	17	4
	17	1

The additional itemIDs are 12 14 and 17. A final query will confirm which floors items 12, 14 and 17 are sold on which department floor:

```
SELECT distinct(item.ItemID), department.Floor
FROM item INNER JOIN saleitem INNER JOIN sale INNER JOIN department
ON item.ItemID = saleitem.ItemID and
saleitem.SaleID = sale.SaleID and
sale.DepartmentID = department.DepartmentID
WHERE saleitem.ItemID in (12,14,17)
ORDER BY item.itemID, department.Floor;
```

ItemID	Floor
12	1
12	2
12	3
12	4
14	1
14	2
14	3
14	4
17	1
17	2
17	3
17	4

Be sure you understand the question being asked.

1.8 Find the numbers and names of the employees who earn more than their manager.

```
SELECT emp.EmployeeID, emp.FirstName, emp.LastName , emp.Salary as empSal,
boss.Salary as BossSal
FROM employee emp INNER JOIN employee boss
ON emp.BossID = boss.EmployeeID
WHERE boss.Salary < emp.Salary;
```

Note: This is a unary join – we have created aliases for the Employee table. The first alias is 'emp' the second 'boss'

EmployeeID	FirstName	LastName	empSal	BossSal
8	Sarah	Fergusson	86000.00	73000.00

1.9 Type the query to find the average salary of employees in the Clothes department

Your result set should look something like this:

```
SELECT format(avg(salary),2) as AVG_SALARY
FROM employee NATURAL JOIN department
WHERE department.Name = 'Clothes';
```

	AVG_SALARY
	46.000.00

HINT: Note the formatting of the result

1.10 Type the query to find, for each department, the average salary of employees

```
SELECT department.Name, format(avg(employee.Salary),0) as AVG_SALARY
FROM employee NATURAL JOIN department
GROUP BY department.Name;
```

Your result set should look something like this:

	Name	AVG_SALARY
▶	Management	125,000
	Books	45,000
	Clothes	46,000
	Equipment	43,000
	Furniture	45,000
	Navigation	45,000
	Recreation	45,000
	Accounting	60,000
	Purchasing	79,500
	Personnel	75,000
	Marketing	64,000

1.11 Find, for each department on the second floor, the average salary of the employees

```
SELECT department.Name, FORMAT(AVG(employee.Salary),2) as AverageSalary
FROM employee NATURAL JOIN department
WHERE department.Floor = 2
GROUP BY department.Name;
```

	Name	AverageSalary
	Clothes	46.000.00
	Recreation	45.000.00

1.12 List suppliers that deliver a total quantity of items of types C and N that is greater than 40

In this SQL query we are building the query in stages

i) First let's find the items that are of type C and N

```
SELECT item.Name, item.Type
FROM item
WHERE item.Type in ('C','N')
ORDER BY item.Name;
```

ii) Then find out how many of those items have been delivered

```
SELECT item.Name, SUM(deliveryitem.Quantity)
FROM deliveryitem INNER JOIN item
ON deliveryitem.ItemID = item.ItemID
WHERE item.Type in ('C','N')
GROUP BY item.Name;
```

iii) And if the quantity delivered is greater than 40

```
SELECT item.Name, SUM(deliveryitem.Quantity)
FROM deliveryitem INNER JOIN item
ON deliveryitem.ItemID = item.ItemID
WHERE item.Type in ('C','N')
GROUP BY item.Name
HAVING Sum(deliveryitem.Quantity) > 40;
```

iv) Now let's find the Supplier Names and IDs:

Placed below are three different approaches to solving this task

```
SELECT delivery.SupplierID, supplier.Name , SUM(deliveryitem.Quantity)
FROM supplier INNER JOIN delivery INNER JOIN deliveryitem Inner Join item
ON supplier.SupplierID = delivery.SupplierID
AND delivery.DeliveryID = deliveryitem.DeliveryID
AND deliveryitem.ItemID = item.ItemID
WHERE item.Type IN ('C','N')
GROUP BY delivery.SupplierID, supplier.Name
HAVING SUM(deliveryitem.Quantity) > 40;
```

Notice the difference in the WHERE statement using an OR:

```
SELECT delivery.SupplierID, supplier.Name , SUM(deliveryitem.Quantity)
FROM supplier INNER JOIN delivery INNER JOIN deliveryitem Inner Join item
ON supplier.SupplierID = delivery.SupplierID
AND delivery.DeliveryID = deliveryitem.DeliveryID
AND deliveryitem.ItemID = item.ItemID
WHERE (item.Type = 'C' OR item.Type = 'N')
GROUP BY delivery.SupplierID, supplier.Name
HAVING SUM(deliveryitem.Quantity) > 40;
```

And the WHERE x OR y condition can be written without parenthesis

```
SELECT delivery.SupplierID, supplier.Name , SUM(deliveryitem.Quantity)
FROM supplier INNER JOIN delivery INNER JOIN deliveryitem Inner Join item
ON supplier.SupplierID = delivery.SupplierID
```

```
AND delivery.DeliveryID = deliveryitem.DeliveryID
AND deliveryitem.ItemID = item.ItemID
WHERE item.Type = 'C'
OR item.Type = 'N'
GROUP BY delivery.SupplierID, supplier.Name
HAVING SUM(deliveryitem.Quantity) > 40;
```

The result is the same:

SupplierID	Name
101	Global Books & Maps
105	All Points Inc.

1.13 What is the average delivery quantity of items of type N made by each company who delivers them. Be sure to list the Item name in your answer.

```
SELECT delivery.SupplierID, supplier.Name, item.Type, item.Name,
FORMAT(AVG(deliveryitem.Quantity),2) AS AvgDelQty
FROM supplier INNER JOIN delivery INNER JOIN deliveryitem INNER JOIN item
ON supplier.SupplierID = delivery.SupplierID
AND delivery.DeliveryID = deliveryitem.DeliveryID
AND deliveryitem.ItemID = item.ItemID
WHERE item.Type = 'N'
GROUP BY delivery.SupplierID,supplier.Name,item.Name;
```

SupplierID	Name	Name	AvgDelQty
101	Global Books & Maps	Compass - Silva	4.67
101	Global Books & Maps	Geo positioning system	3.00
101	Global Books & Maps	Map measure	10.00
102	Nepalese Corp.	Compass - Silva	3.00
102	Nepalese Corp.	Geo positioning system	4.00
102	Nepalese Corp.	Map measure	10.00
103	All Sports Manufacturing	Compass - Silva	8.00
103	All Sports Manufacturing	Geo positioning system	1.50
103	All Sports Manufacturing	Map measure	10.00
105	All Points Inc.	Compass - Silva	1.00

1.14 List the name and salary of the managers with more than 2 employees

```
SELECT employee.FirstName, employee.LastName, employee.Salary
FROM employee
WHERE employeeID IN
     (SELECT BossID
      FROM employee
      GROUP BY BossID
      HAVING COUNT(*) > 2);
```

	FirstName	LastName	Salary
	Alice	Munro	125000.00
	Andrew	Jackson	55000.00
	Clare	Underwood	52000.00

1.15 Type the names and salaries of the employees who earn more than any employee in the marketing department

HINT: Split the problem up into 2 components – calculate max salary of people in marketing (inner query) and then who earns more than that (outer query)…

```
SELECT employee.FirstName, employee.LastName, employee.Salary
FROM employee
WHERE employee.Salary >
  (SELECT MAX(employee.Salary)
   FROM employee NATURAL JOIN department
   WHERE department.Name = "Marketing ");
```

	FirstName	LastName	Salary
	Alice	Munro	125000.00
	Sarah	Fergusson	86000.00

1.16 Find the supplier id and supplier names that deliver compasses

```
SELECT DISTINCT supplier.SupplierID, supplier.Name
FROM supplier INNER JOIN delivery INNER JOIN deliveryitem INNER JOIN item
ON supplier.SupplierID = delivery.SupplierID
AND delivery.DeliveryID = deliveryitem.DeliveryID
AND deliveryitem.ItemID = item.ItemID
WHERE item.Name LIKE 'Compass%';
```

	SupplierID	Name
	101	Global Books & Maps
	102	Nepalese Corp.
	103	All Sports Manufacturing
	105	All Points Inc.

Note, the LIKE condition if you use the following query the result returns no rows:

```
SELECT DISTINCT supplier.SupplierID, supplier.Name
FROM supplier INNER JOIN delivery INNER JOIN deliveryitem INNER JOIN item
ON supplier.SupplierID = delivery.SupplierID
AND delivery.DeliveryID = deliveryitem.DeliveryID
AND deliveryitem.ItemID = item.ItemID
WHERE item.Name = 'Compass';
```

1.17 List item names that are delivered by Nepalese Corp and sold in the Navigation department

```
SELECT DISTINCT item.Name
```

```
FROM item
WHERE ItemID IN
  (SELECT ItemID
   FROM deliveryitem NATURAL JOIN delivery NATURAL JOIN supplier
   WHERE supplier.Name = 'Nepalese corp.')
AND ItemID IN
  (SELECT ItemID
   FROM saleitem NATURAL JOIN sale NATURAL JOIN department
   WHERE department.Name = 'Navigation');
```

	Name
	Geo positioning system
	Torch
	Gortex Rain Coat
	Pocket knife - Essential
	Compass - Silva
	Map case
	Map measure
	How to Win Foreign Friends

1.18 Type the query that gives the overall average of the salaries of all employees.

```
SELECT AVG(employee.Salary)
FROM employee;
```

	AVG_SALARY
	60529.411765

1.19 Type the query that finds the item id of the items sold on the second floor

```
SELECT DISTINCT saleitem.ItemID
FROM saleitem INNER JOIN sale INNER JOIN department
ON saleitem.SaleID = sale.SaleID
AND sale.DepartmentID = department.DepartmentID
WHERE department.Floor = 2
ORDER BY ItemID
```

	ItemID
	8
	12
	14
	17
	18
	19
	20
	21
	22
	23
	24
	25

1.20 Type the query that finds the name and salary of Clare Underwood's manager

```
SELECT employee.FirstName, employee.LastName, employee.Salary
FROM employee
WHERE employeeID IN
   (SELECT BossID
     FROM employee
    WHERE employee.FirstName = 'Clare'
     AND employee.LastName = 'Underwood');
```

	FirstName	LastName	Salary
	Ned	Kellv	85000.00

1.21 List the ids of the departments where all of the employees earn less than their manager

```
SELECT DISTINCT DepartmentID
FROM employee
WHERE DepartmentID NOT IN
     (SELECT wrk.DepartmentID
      FROM employee wrk INNER JOIN employee boss
      ON wrk.BossID = boss.employeeID
      WHERE wrk.Salary >= boss.Salary)
AND employee.BossID IS NOT NULL
ORDER BY DepartmentID;
```

Hint: Notice that the inner query uses a unary join to create a result set that lists all departmentids where an employee earns more than their boss. That is why the condition is NOT IN

	DepartmentID
	2
	3
	4
	5
	6
	7
	8
	10
	11

1.22 Find the names of employees who are in the same department as their manager (as an employee). Report the name of the employee, the department, and the boss's name

```
SELECT concat (wrk.FirstName, ' ',wrk.LastName) as employee,
wrk.DepartmentID, CONCAT (boss.FirstName, ' ',boss.LastName) as Boss
FROM employee wrk INNER JOIN employee boss
ON wrk.BossID = boss.employeeID
WHERE wrk.DepartmentID = boss.DepartmentID;
```

	Employee	DepartmentID	Boss
	Andrew Jackson	11	Ned Kelly
	Clare Underwood	11	Ned Kelly
	Nancy Cartwright	8	Todd Beamer
	Sarah Ferguson	9	Brier Patch

1.23 Type the query to find the name, salary, and managers of the employees of department id 11 who have a salary over $55,000

Hint: This will require a unary join to find the managers and employees. You will need to filter your query to limit it to department id 11, and who have a salary over $25,000. Break this query up into its component parts

```
SELECT concat (wrk.FirstName, ' ', wrk.LastName) as employeeName, wrk.Salary,
concat(boss.FirstName, ' ',boss.LastName) as Manager
FROM employee wrk INNER JOIN employee boss
ON wrk.BossID = boss.employeeID
WHERE wrk.DepartmentID = 11
AND wrk.Salary > 55000;
```

	EmployeeName	Salary	Manager
	Ned Kelly	85000.00	Alice Munro

1.24. Among all the departments with total salary greater than $55,000, find the department names that sell item Number 17

```
SELECT DISTINCT(department.Name)
FROM saleitem INNER JOIN sale INNER JOIN department
ON saleitem.SaleID = sale.SaleID
AND sale.DepartmentID = department.DepartmentID
WHERE saleitem.ItemID = 17
AND department.DepartmentID IN
     (SELECT DepartmentID
      FROM employee
      GROUP BY DepartmentID
      HAVING SUM(employee.Salary) > 55000);
```

	Name
	Clothes
	Equipment

1.25 Find the supplier id and supplier names that deliver both compasses and an item other than compasses

Attempt 1:

```
SELECT DISTINCT delivery.SupplierID, supplier.Name
FROM supplier INNER JOIN delivery INNER JOIN deliveryitem INNER JOIN item
ON supplier.SupplierID = delivery.SupplierID
```

```
AND delivery.DeliveryID = deliveryitem.DeliveryID
AND deliveryitem.ItemID = item.ItemID
WHERE item.Name NOT LIKE 'Compass%'
AND delivery.SupplierID IN
     (SELECT SupplierID
      FROM delivery NATURAL JOIN item NATURAL JOIN deliveryitem
      WHERE item.Name LIKE 'Compass%')
ORDER BY delivery.SupplierID;
```

Attempt 2:

```
SELECT DISTINCT delivery.SupplierID, supplier.Name
FROM supplier INNER JOIN delivery INNER JOIN deliveryitem INNER JOIN item
ON supplier.SupplierID = delivery.SupplierID
AND delivery.DeliveryID = deliveryitem.DeliveryID
AND deliveryitem.ItemID = item.ItemID
WHERE delivery.SupplierID IN
     (SELECT SupplierID
      FROM delivery NATURAL JOIN deliveryitem Natural JOIN item
      WHERE item.Name LIKE'Compass%')
GROUP BY delivery.SupplierID, supplier.Name
HAVING COUNT(DISTINCT item.Name) > 1
order by delivery.SupplierID;
```

	SupplierID	Name
	101	Global Books & Maps
	102	Nepalese Corp.
	103	All Sports Manufacturing
	105	All Points Inc.

Note: Attempt 1 uses the approach to find those suppliers that supply things other than compasses and also supply compasses (sub query).

Attempt 2 uses a more generalizable approach. The generalizable approach is better as it allows queries such as "Find suppliers that deliver two items other than compasses" – change the >1 to >2 in the HAVING clause in Attempt 2 to do this. (Attempt 2 uses DISTINCT to handle multiple deliveries of compasses for the same supplier.)

SQL Homework - Functions

If you have time you can attempt the following SQL questions or do them at home against the department store schema you have installed on your computer.
Most of this week's homework requires you to read the manual. That is the functions section of the MySQL reference manual https://dev.mysql.com/doc/refman/5.7/en/functions.html

H1 How many deliveries have there been in the month of July?

Hint: the only information you have been given is the month name

```
SELECT COUNT(DeliveryID)
FROM delivery
WHERE Monthname(deliverydate) = 'July';
```

H2 List the names of the tents available for sale

```
SELECT Name
FROM item
WHERE Name like '%Tent%';
```

H3 What month has had the highest number of sales?

```
SELECT COUNT(SaleID), Monthname(SaleDate)
FROM sale
Group By Monthname(SaleDate)
Order by COUNT(SaleID) DESC
LIMIT 1;
```

H4 List the salary total and employee count for each DepartmentID. Order by the smallest salary total to largest.

```
SELECT DepartmentID, COUNT(EmployeeID), SUM(Salary)
FROM employee
GROUP BY DepartmentID
ORDER BY Sum(Salary);
```

H5 How many sales have been on a Sunday?

```
SELECT COUNT(saleID)
FROM sale
Where DayName(Saledate) = 'Sunday';
```

H6 How many days have elapsed between the first delivery date and most recent delivery date for each supplier?

```
SELECT SupplierID, Datediff(max(deliverydate),min(deliverydate)),
COUNT(distinct(deliverydate))
```

```
FROM delivery
Group by SupplierID;
```

H7 Produce the following output by writing a SQL statement

	Where is each department?
	The Management department is on floor number 5
	The Books department is on floor number 1
	The Clothes department is on floor number 2
	The Equipment department is on floor number 3
	The Furniture department is on floor number 4
	The Navigation department is on floor number 1
	The Recreation department is on floor number 2
	The Accounting department is on floor number 5
	The Purchasing department is on floor number 5
	The Personnel department is on floor number 5
	The Marketing department is on floor number 5

Note there is no _ in the title it is "Where is each department?" not Where_is_each_department?

```
SELECT concat('The ', name, ' department is on floor number ',floor) as "Where
is each department?"
FROM department;
```

H8 Find the minimum, maximum, average and standard deviation for salaries in each department

```
SELECT DepartmentID, MIN(Salary), MAX(Salary), STDDEV(Salary)
FROM employee
GROUP BY DepartmentID;
```

Appendix New Department Store Physical ER Model