INFO90002 Week 6 Lab

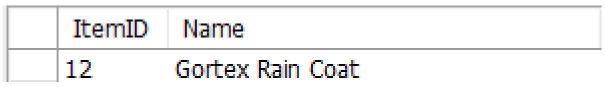
Objectives

- Practice SQL
- Outer Joins
- Views

Section 1 SQL

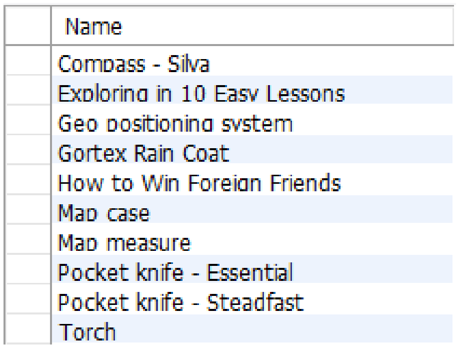
Connect to your MySQL database on the engineering server

1.1. Type the query to list the green items of type C



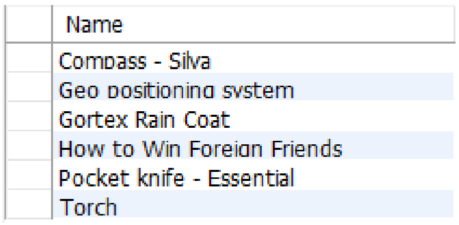
ItemID	Name
12	Gortex Rain Coat

1.2 Type the query to find the items delivered by at least two suppliers



Name
Compass - Silva
Exploring in 10 Easy Lessons
Geo positioning system
Gortex Rain Coat
How to Win Foreign Friends
Map case
Map measure
Pocket knife - Essential
Pocket knife - Steadfast
Torch

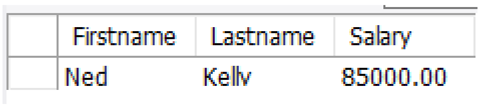
1.3 Type the query to find the items that have been sold by at least two departments



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

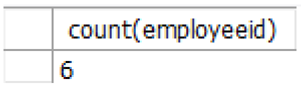
1.4 Find the name of the highest-paid employee in the Marketing department

```
SELECT employee.Firstname, employee.Lastname, employee.Salary
FROM employee natural join department
WHERE department.Name = 'Marketing'
AND employee.Salary =
    (SELECT max(salary)
    FROM employee natural join department
    WHERE department.Name = 'Marketing');
```



Firstname	Lastname	Salary
Ned	Kelly	85000.00

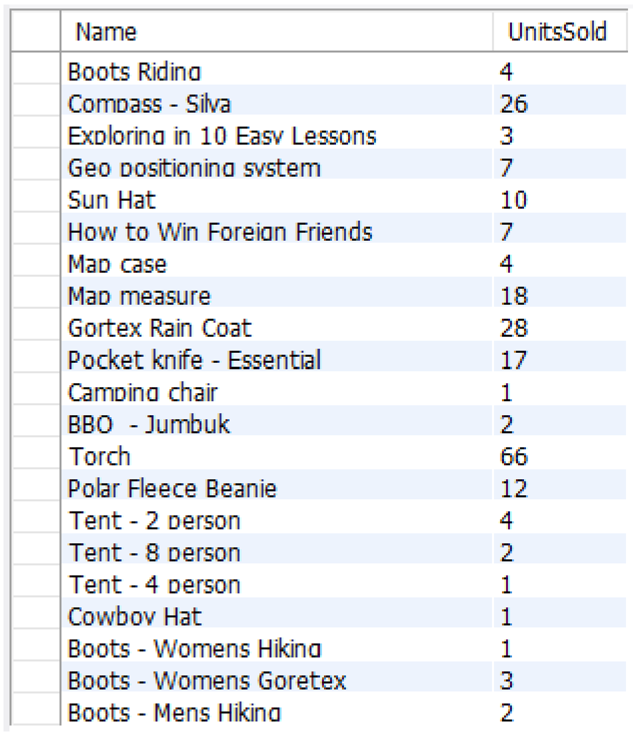
1.5 Type the query to find the number of employees with a salary equal to or less than $45,000



count(employeeid)
6

1.6 Find the number of units sold of each item

```
SELECT item.Name, Count(saleitem.itemID) * saleitem.Quantity
as UnitsSold
FROM saleitem  natural join item
GROUP BY ItemID
ORDER BY ItemID;
```



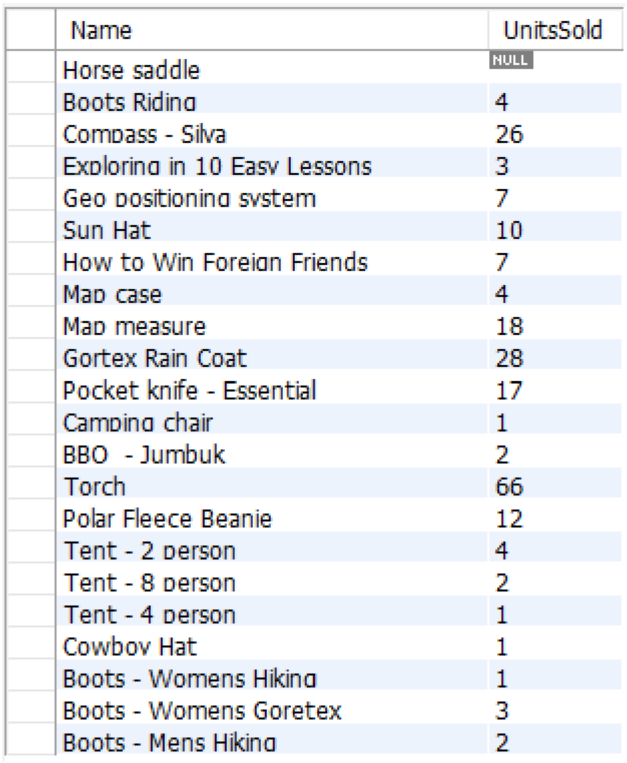
Name	UnitsSold
Boots Riding	4
Compass - Silva	26
Exploring in 10 Easy Lessons	3
Geo positioning system	7
Sun Hat	10
How to Win Foreign Friends	7
Map case	4
Map measure	18
Gortex Rain Coat	28
Pocket knife - Essential	17
Camping chair	1
BBQ - Jumbuk	2
Torch	66
Polar Fleece Beanie	12
Tent - 2 person	4
Tent - 8 person	2
Tent - 4 person	1
Cowboy Hat	1
Boots - Womens Hiking	1
Boots - Womens Goretex	3
Boots - Mens Hiking	2

However this query does not return the fact that the Horse Saddle (ItemID 1) has not been sold! As such for all of the information you may need to use either a Left Join or a Right Join (dependent on where the 'Null' column resides in the query)

In the example below there is not itemID =1 in the saleitem "Units Sold")

Outer Joins

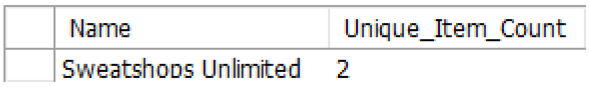
```
SELECT item.Name, Count(saleitem.itemID) * Quantity as
UnitsSold
FROM saleitem right join item
on saleitem.ItemID = item.ItemID
GROUP BY saleitem.ItemID
ORDER BY saleitem.ItemID;
```



Name	UnitsSold
Horse saddle	NULL
Boots Riding	4
Compass - Silva	26
Exploring in 10 Easy Lessons	3
Geo positioning system	7
Sun Hat	10
How to Win Foreign Friends	7
Map case	4
Map measure	18
Gortex Rain Coat	28
Pocket knife - Essential	17
Camping chair	1
BBQ - Jumbuk	2
Torch	66
Polar Fleece Beanie	12
Tent - 2 person	4
Tent - 8 person	2
Tent - 4 person	1
Cowboy Hat	1
Boots - Womens Hiking	1
Boots - Womens Goretex	3
Boots - Mens Hiking	2

1.7 Find any suppliers that deliver no more than two unique items. List the suppliers in alphabetical order

```
SELECT supplier.Name, COUNT(DISTINCT deliveryitem.ItemID) as
Unique_item_count
FROM delivery INNER JOIN supplier INNER JOIN deliveryitem
ON supplier.SupplierID = delivery.SupplierID
AND delivery.DeliveryID = deliveryitem.DeliveryID
GROUP BY supplier.SupplierID
HAVING COUNT(DISTINCT deliveryitem.ItemID) <= 2
ORDER BY supplier.Name;
```



Name	Unique_Item_Count
Sweatshops Unlimited	2

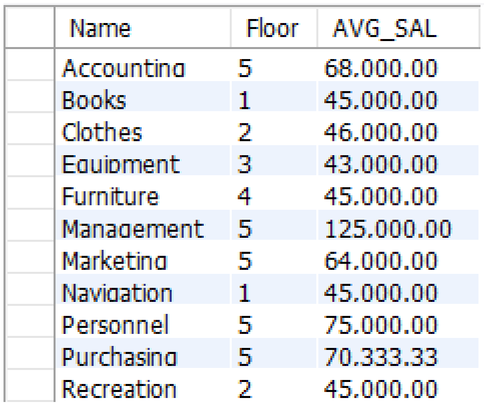
1.8 Find the names of suppliers that have never delivered a Compass

```
SELECT DISTINCT supplier.Name
FROM supplier
  WHERE supplier.SupplierID NOT IN
    (SELECT SupplierID
    FROM delivery NATURAL JOIN deliveryitem NATURAL JOIN ITEM
     WHERE item.Name like 'Compass%');
```

This question was challenging in that you were not given the complete data. The item name for the Compass in the database is 'Compass – Silva' but you only had part of the information. In this case you can not get an exact match – so a condition such as "= 'Compass' " would return no rows. To find rows that match – you must use the like condition and the wildcard %

Therefore it is always good to query reference tables like item to know how the item name is actually stored in the database.

1.9 Find, for each department, its floor and the average salary in the department.

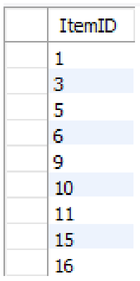


Name	Floor	AVG_SAL
Accounting	5	68.000.00
Books	1	45.000.00
Clothes	2	46.000.00
Equipment	3	43.000.00
Furniture	4	45.000.00
Management	5	125.000.00
Marketing	5	64.000.00
Navigation	1	45.000.00
Personnel	5	75.000.00
Purchasing	5	70.333.33
Recreation	2	45.000.00

```
SELECT Name, Floor, FORMAT(AVG(SALARY),2) as AVG_SAL
FROM department natural Join employee
Group by Name, Floor
ORDER BY department.Name;
```

1.10 Type the query to find the Items (ItemID) sold on floors other than the second floor

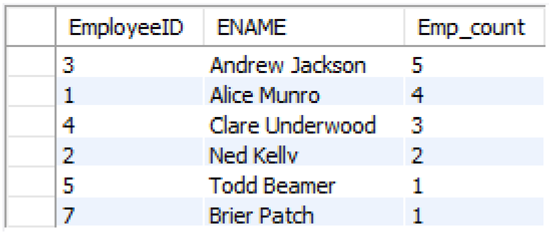
The result set should look similar to this:



ItemID
1
3
5
6
9
10
11
15
16

1.11 Type the query to count the number of direct employees of each manager, List the EmployeeID, Manager Name and number of employees.

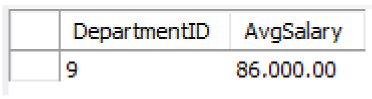
Your result set should look similar to this:



EmployeeID	ENAME	Emp_count
3	Andrew Jackson	5
1	Alice Munro	4
4	Clare Underwood	3
2	Ned Kellv	2
5	Todd Beamer	1
7	Brier Patch	1

1.12 List the department id and average salary where the average salary of the employees of each manager is more than $55000

```
SELECT wrk.DepartmentID, FORMAT(AVG(wrk.Salary),2) AS
AvgSalary
FROM Employee wrk
WHERE wrk.EmployeeID NOT IN
     (SELECT Department.ManagerID
      FROM Department
      WHERE wrk.EmployeeID = Department.ManagerID
      AND wrk.DepartmentID = Department.DepartmentID)
GROUP BY wrk.DepartmentID
HAVING AVG(wrk.Salary) > 55000;
```



DepartmentID	AvgSalary
9	86,000.00

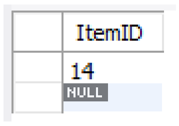
1.13 Find the items (itemID) sold by ALL departments located on the second floor

```
SELECT SaleItem.ItemID
FROM SaleItem NATURAL JOIN Sale NATURAL JOIN Department
WHERE Department.Floor = 2
GROUP BY SaleItem.ItemID
```

```
HAVING count(DISTINCT Department.DepartmentID) =
             (SELECT count(DISTINCT DepartmentID)
              FROM Department
              WHERE Department.Floor = 2
              )
ORDER BY SaleItem.ItemID;
```

And using a different method

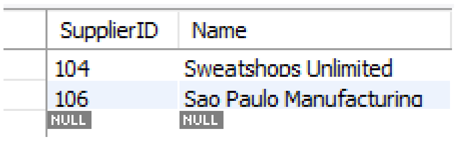
```
SELECT DISTINCT ItemID
FROM Item
WHERE NOT EXISTS
    (SELECT *
     FROM Department
     WHERE Department.Floor = 2
     AND NOT EXISTS
        (SELECT *
         FROM SaleItem natural join Sale
         WHERE SaleItem.ItemID = Item.ItemID
         AND Sale.DepartmentID = Department.DepartmentID
         )
      )
ORDER BY ItemID;
```



	ItemID
	14
	NULL

1.14 Find the supplier id and supplier names that do not deliver compasses

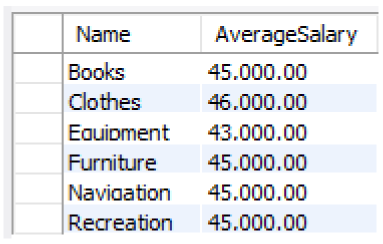
```
SELECT SupplierID, Supplier.Name
FROM Supplier
WHERE SupplierID NOT IN
      (SELECT SupplierID
       FROM Delivery NATURAL JOIN DeliveryItem NATURAL JOIN
ITEM
       WHERE Item.Name Like 'Compass%');
```



SupplierID	Name
104	Sweatshops Unlimited
106	Sao Paulo Manufacturing
NULL	NULL

1.15 Find, for each department that has sold items of type E. List the department name and the average salary of the employees

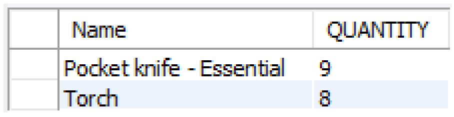
```
SELECT Department.Name, FORMAT(AVG(Employee.Salary),2) AS
AverageSalary
FROM Employee INNER JOIN Department INNER JOIN Sale
INNER JOIN SaleItem INNER JOIN Item
ON Employee.DepartmentID = Department.DepartmentID
AND Department.DepartmentID = Sale.DepartmentID
AND Sale.SaleID = SaleItem.SaleID
AND SaleItem.ItemID = Item.ItemID
WHERE Item.Type = 'E'
GROUP BY Department.Name;
```



Name	AverageSalary
Books	45.000.00
Clothes	46.000.00
Equipment	43.000.00
Furniture	45.000.00
Navigation	45.000.00
Recreation	45.000.00

1.16 Find the total number of items (list the item and sale quantity) of type E sold by the departments on the second floor

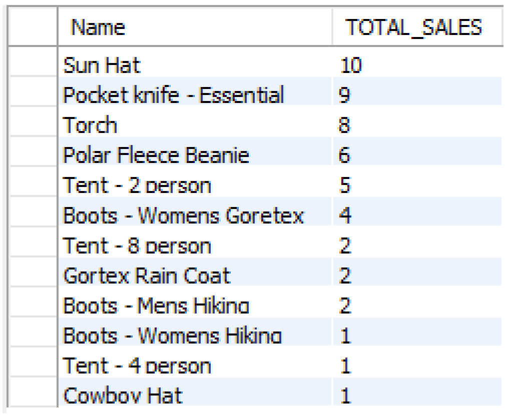
```
SELECT ITEM.Name, SUM(SaleItem.Quantity) AS QUANTITY
FROM Item INNER JOIN SaleItem INNER JOIN Sale INNER JOIN
Department
ON Item.ItemID = SaleItem.ItemID
AND Sale.SaleID = SaleItem.SaleID
AND Department.DepartmentID = Sale.DepartmentID
WHERE Item.Type = 'E'
AND Department.Floor = 2
GROUP BY ITEM.ITEMID;
```



Name	QUANTITY
Pocket knife - Essential	9
Torch	8

1.17 Type the query to find the total quantity sold of each item by the departments on the second floor

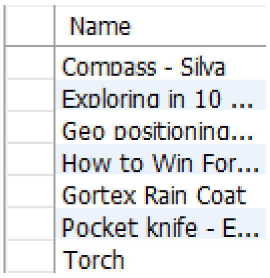
The result set should look similar to this:



Name	TOTAL_SALES
Sun Hat	10
Pocket knife - Essential	9
Torch	8
Polar Fleece Beanie	6
Tent - 2 person	5
Boots - Womens Goretex	4
Tent - 8 person	2
Gortex Rain Coat	2
Boots - Mens Hiking	2
Boots - Womens Hiking	1
Tent - 4 person	1
Cowboy Hat	1

1.18 List each item (ItemName) delivered to at least two departments by each supplier that delivers it

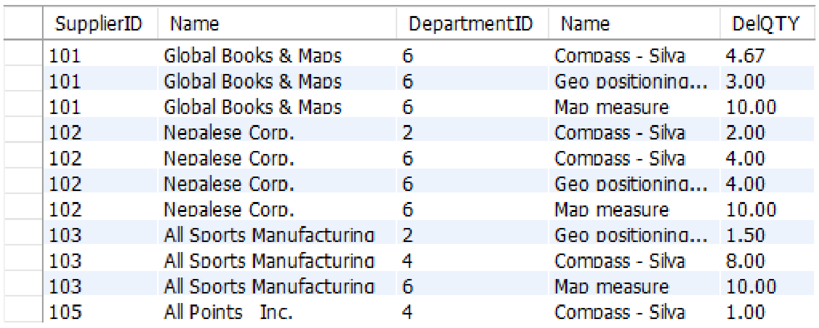
```
Select Distinct(Item.Name)
FROM Item natural join DeliveryItem
Where ItemID NOT IN (
 Select distinct(itemID)
 FRom DeliveryItem
 group by ItemID
 Having Count(distinct(departmentid)) < 2
 order by ItemID);
```



Name
Compass - Silva
Exploring in 10 ...
Geo positioning...
How to Win For...
Gortex Rain Coat
Pocket knife - E...
Torch

1.19 What is the average delivery quantity of items of type N delivered by each supplier to each department (given that the supplier delivers items of type N to the department)?

```
SELECT Delivery.SupplierID, Supplier.Name, DepartmentID,
Item.Name,
FORMAT(AVG(DeliveryItem.Quantity),2) AS DelQTY
FROM Delivery INNER JOIN Supplier INNER JOIN Item INNER JOIN
DeliveryItem
ON Delivery.SupplierID =  Supplier.SupplierID
AND DeliveryItem.ItemID = Item.ItemID
AND DeliveryItem.DeliveryID = Delivery.DeliveryID
WHERE  Item.Type = 'N'
GROUP BY Delivery.SupplierID,  Supplier.Name,
DepartmentID,  Item.Name;
```



SupplierID	Name	DepartmentID	Name	DelQTY
101	Global Books & Maps	6	Compass - Silva	4.67
101	Global Books & Maps	6	Geo positioning...	3.00
101	Global Books & Maps	6	Map measure	10.00
102	Nepalese Corp.	2	Compass - Silva	2.00
102	Nepalese Corp.	6	Compass - Silva	4.00
102	Nepalese Corp.	6	Geo positioning...	4.00
102	Nepalese Corp.	6	Map measure	10.00
103	All Sports Manufacturing	2	Geo positioning...	1.50
103	All Sports Manufacturing	4	Compass - Silva	8.00
103	All Sports Manufacturing	6	Map measure	10.00
105	All Points Inc.	4	Compass - Silva	1.00

Views

Views are a table whose rows are not explicitly stored in the database but are returned as needed from a stored view definition.

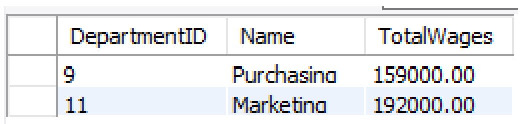
Consider the following view

```
CREATE VIEW vDepartment_Wages AS
SELECT DepartmentID, Name, SUM(Salary) as TotalWages
FROM Department NATURAL JOIN Employee
GROUP BY DepartmentID, Name
ORDER BY DepartmentID;
```

This creates a view called vDepartment_wages. I can use this view like any table in my schema.

```
SELECT *
```

```
FROM vDepartment_Wages
WHERE TotalWages > 150000;
```



DepartmentID	Name	TotalWages
9	Purchasing	159000.00
11	Marketing	192000.00

However what is really going on is the following query:

```
SELECT *
FROM
    (SELECT DepartmentID, Name, SUM(Salary) as TotalWages
     FROM Department NATURAL JOIN Employee
     GROUP BY DepartmentID, Name
     ORDER BY DepartmentID) as vDepartment_Wages
WHERE TotalWages > 150000;
```

The SELECT statement for the View is being used in the FROM clause of SQL

At any time the SQL that makes up the view definition can be queried from the Data Dictionary:

```
SELECT table_name, view_definition
FROM Information_schema.views
-- WHERE Table_SCHEMA= 'labs2018' – remove comment for BYOD
devices
;
```

1.20 List the employees in the Accounting department and the difference between their salaries and the average salary of the department

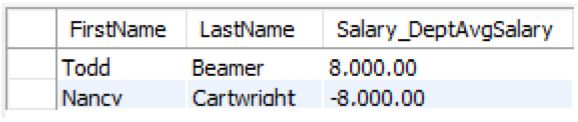
First create a view of the Department Name and average Salary

```
CREATE VIEW VDepartmentSalary (DepartmentID, dpavgsal) AS
SELECT DepartmentID, AVG(Salary)
FROM Employee
GROUP BY DepartmentID;
```

Now use the view in the query to answer the question

```
SELECT FirstName, LastName, FORMAT((Salary - dpavgsal),2)
AS Salary_DeptAvgSalary
```

```
FROM vDepartmentSalary INNER JOIN Employee INNER JOIN
Department
ON vDepartmentSalary.DepartmentID = Employee.DepartmentID
AND Department.DepartmentID = Employee.DepartmentID
WHERE Department.Name = 'Accounting';
```

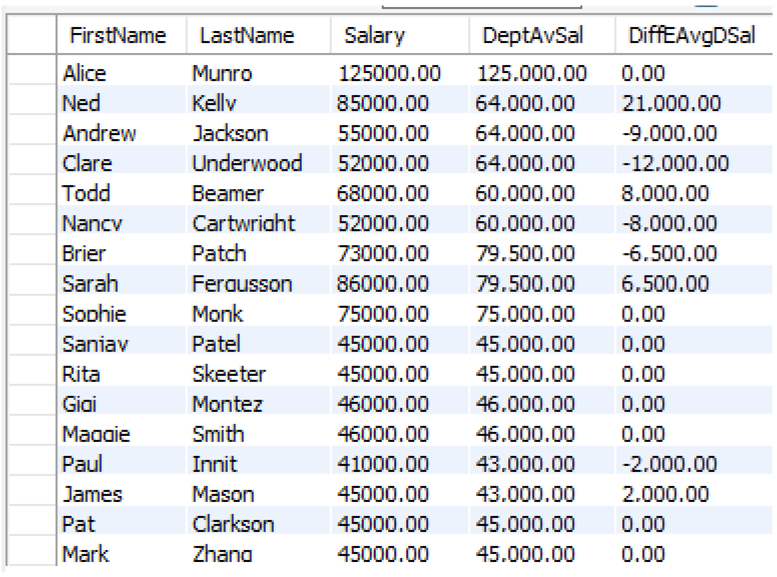


FirstName	LastName	Salary_DeptAvgSalary
Todd	Beamer	8.000.00
Nancv	Cartwriaht	-8.000.00

1.21 List each employee's salary, the average salary within that person's department, and the difference between the employees' salaries and the average salary of the department

Using the vDepartmentSalary view …

```
SELECT FirstName, LastName, Salary, FORMAT(dpavgsal,2) AS
DeptAvSal,
FORMAT(Salary - dpavgsal,2) AS DiffEAvgDSal
FROM vDepartmentSalary NATURAL JOIN Employee
WHERE vDepartmentSalary.DepartmentID = Employee.DepartmentID;
```



FirstName	LastName	Salary	DeptAvSal	DiffEAvgDSal
Alice	Munro	125000.00	125.000.00	0.00
Ned	Kellv	85000.00	64.000.00	21.000.00
Andrew	Jackson	55000.00	64.000.00	-9.000.00
Clare	Underwood	52000.00	64.000.00	-12.000.00
Todd	Beamer	68000.00	60.000.00	8.000.00
Nancv	Cartwriaht	52000.00	60.000.00	-8.000.00
Brier	Patch	73000.00	79.500.00	-6.500.00
Sarah	Ferausson	86000.00	79.500.00	6.500.00
Sophie	Monk	75000.00	75.000.00	0.00
Saniav	Patel	45000.00	45.000.00	0.00
Rita	Skeeter	45000.00	45.000.00	0.00
Giai	Montez	46000.00	46.000.00	0.00
Maaaie	Smith	46000.00	46.000.00	0.00
Paul	Innit	41000.00	43.000.00	-2.000.00
James	Mason	45000.00	43.000.00	2.000.00
Pat	Clarkson	45000.00	45.000.00	0.00
Mark	Zhana	45000.00	45.000.00	0.00

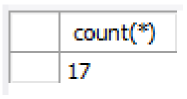
1.22 How many supplier – department pairs exist in which the supplier delivers at least one item of type E to the department?

Frist the view:

```
CREATE VIEW vSupplierDepartment AS
(SELECT DISTINCT SupplierID, DepartmentID
FROM Delivery NATURAL JOIN DeliveryItem Natural Join Item
WHERE Item.Type = 'E' );
```

And now just count the rows in the view

```
SELECT count(*)
FROM vSupplierDepartment;
```



count(*)
17

Using Views

1.23 Which department (departmentid) has more than five sales?

```
CREATE VIEW vDepartmentSales AS
SELECT departmentid, departmentname, COUNT(*) as numSales
FROM Department NATURAL JOIN Sale
GROUP BY departmentid;
```

And to use the view:

```
SELECT *
FROM vDepartmentSales
WHERE numSales > 5;
```

1.24 Create a view to list the maximum salary, average salary, minimum salary, total salary and number of staff in each department. Find the lowest salary in the department with the highest headcount.

```
CREATE VIEW VEmployeeSalary
AS
SELECT Department.DepartmentID, DepartmentName,
MAX(EmployeeSalary) as Maxsal, AVG(EmployeeSalary) as Avgsal,
MIN(EmployeeSalary) as Minsal, SUM(EmployeeSalary) as
Totalsal, COUNT(Employee.EmployeeID) as Headcount
FROM Department natural join Employee
GROUP BY Department.DepartmentID, DepartmentName;
```

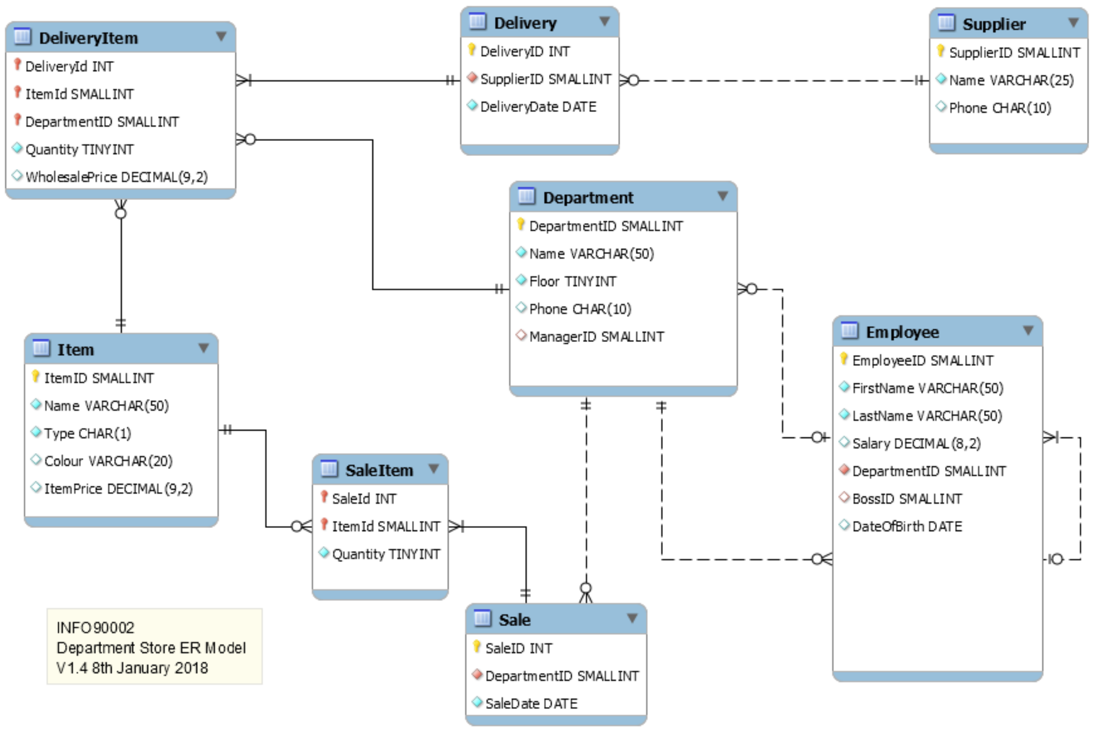
A more complex use of a view:

```
SELECT DepartmentID, DepartmentName, minsal
FROM Vemployeesalary
where Headcount in
    (Select max(Headcount)
    FROM Vemployeesalary);
```

You will notice for clarity each view name uses a "v" prefix to indicate it is a view. This is good practice in writing SQL but is not mandatory.

End of Week 6 Lab

Appendix: The New Department Store Physical ER Model


DeliveryItem
DeliveryId INT
ItemId SMALLINT
DepartmentID SMALLINT
Quantity TINYINT
WholesalePrice DECIMAL(9,2)
Delivery
DeliveryID INT
SupplierID SMALLINT
DeliveryDate DATE
Supplier
SupplierID SMALLINT
Name VARCHAR(25)
Phone CHAR(10)
Department
DepartmentID SMALLINT
Name VARCHAR(50)
Floor TINYINT
Phone CHAR(10)
ManagerID SMALLINT
Employee
EmployeeID SMALLINT
FirstName VARCHAR(50)
LastName VARCHAR(50)
Salary DECIMAL(8,2)
DepartmentID SMALLINT
BossID SMALLINT
DateOfBirth DATE
Item
ItemID SMALLINT
Name VARCHAR(50)
Type CHAR(1)
Colour VARCHAR(20)
ItemPrice DECIMAL(9,2)
SaleItem
SaleId INT
ItemId SMALLINT
Quantity TINYINT
Sale
SaleID INT
DepartmentID SMALLINT
SaleDate DATE
INFO90002
Department Store ER Model
V1.4 8th January 2018