SQL Assignment

Seosamh Ó Cinnéide

Request 1:

```
--List the last name, first name and employee number of all programmers who were hired
on or before 21 May 1991 in ascending order of last name.
```

Expected Result:

Last_Name	First_Name	Employee_No
A…	aFirstName	anEmployeeNo
Etc.	Etc.	Etc.

Query:

```
SELECT Last_Name, First_Name, Employee_No FROM Employees WHERE Job_ID = (SELECT Job_ID
FROM Jobs WHERE Job_Title = 'programmer') AND Hire_Date <= '1991/05/21' ORDER BY
Last_Name ASC
```

Actual Result:

Last_Name	First_Name	Employee_No
Acreman	Lucy	172
Court	Peter	155
Hadlow	Nelson	133
Monroe	Justin	174
Repetto	Joanna	166

Request 2:

```
--List the department number, last name and salary of all employees who were hired
between 16/09/87 and 12/05/96 sorted in ascending order of last name within department
number.
```

Expected Result:

Department_No	Last_Name	Annual_Salary
LowestDeptNo	A…	anAnnualSalary
Etc.	Etc.	Etc.

Query:

```
SELECT Department_No, Last_Name, Annual_Salary FROM Employees WHERE Hire_Date BETWEEN
'1987/09/16' AND '1996/05/12' ORDER BY Department_No, Last_Name
```

Actual Result:

Department_No	Last_Name	Annual_Salary
10	Flowers	10500
10	Powers	10900
10	Sharma	9500
20	Merton	25000
20	Wise	14200
30	Christy	7500
30	Clifford	20000
30	Wilkins	23000
50	Acreman	18000
50	Court	17950
50	Hadlow	16500
50	Mace	22000
50	Monroe	17500
50	Repetto	16000
50	Trotter	18000
50	Williams	45000
60	Avery	9000
60	Barnes	9800
60	Davis	9000
60	Hart	12000
60	Miles	10000
60	Reismaster	10500
70	Laslo	11800
80	Holland	8500
80	Lambert	10200
90	Marsh	150000
90	Molavi	45000
110	Klein	14200
110	Rogers	15000
190	Elliott	9800
190	Goodwin	10000
190	Lugini	7800
190	Macffrey	9800
190	Watson	10000

Request 3:

```
--List all the data for all jobs where the maximum salary is greater than 15000 sorted
in decending order of maximum salary.
```

Expected Result:

Job_ID	Job_Title	Min_Salary	Max_Salary
aJobID	aJobTitle	aMinSalry	HighestMaxSalary
Etc.	Etc.	Etc.	Etc.

Query:

```
SELECT * FROM Jobs WHERE Max_Salary > 15000 ORDER BY Max_Salary DESC
```

Actual Result:

Job_ID	Job_Title	Min_Salary	Max_Salary
CO_CHAIR	Company Chairman	100000	200000
CO_DIR	Company Director	30000	100000
IT_MGR	IT Manager	40000	60000
HR_MGR	Human Resources Manager	30000	50000
AD_MGR	Administration Manager	20000	30000
IT_PROG	Programmer	15000	25000
FN_MGR	Finance Manager	13000	25000
AC_ACCOUNT	Accountant	8500	25000
AC_MGR	Accounts Manager	10000	25000
SH_MGR	Shipping Manager	20000	25000
SA_MGR	Sales Manager	10000	20000
MK_MGR	Marketing Manager	10500	17500

Request 4:

```
--List the last name, first name, job id and commission of employees who earn
commission sorted in ascending order of first name within last name.
```

Expected Result:

Last_Name	First_Name	Job_ID	Commission_Percent
Avery	William	SA_REP	0.2
Etc.	Etc.	Etc.	Etc.

Query:

```
SELECT Last_Name, First_Name, Job_ID, Commission_Percent FROM Employees WHERE
Commission_Percent IS NOT NULL ORDER BY Last_Name, First_Name
```

Actual Result:

Last_Name	First_Name	Job_ID	Commission_Percent
Avery	William	SA_REP	0.2
Barnes	Gillian	SA_REP	0.25
Berry	Jayne	SA_REP	0.3
Bitton	Joshua	SA_REP	0.15
Caprice	Michelle	SA_REP	0.1
Connor	Patrick	SA_REP	0.2
Davis	Adrian	SA_REP	0.2
Davis	Graeme	SA_REP	0.25
French	Natalie	SA_REP	0.15
Hart	Tracey	SA_REP	0.3
Hartmore	Wayne	SA_MGR	0.35
Miles	Everton	SA_REP	0.2
Miles	James	SA_REP	0.25
Neilson	Joanna	SA_REP	0.1
Nolan	Ben	SA_REP	0.2
Pench	John	SA_REP	0.15
Reismaster	Janet	SA_REP	0.25
Weston	Colin	SA_REP	0.3
Wilmott	Jennifer	SA_REP	0.25

Request 5:

```
--Which jobs are found in the IT and Sales departments?
```

Expected Result:

Job_Title
aJobTitle
Etc.

Query:

```
SELECT DISTINCT Job_Title FROM Jobs WHERE Job_ID IN (SELECT Job_ID FROM Employees
WHERE Department_No IN (SELECT Department_No FROM Departments WHERE Department_Name IN
('IT', 'Sales')))
```

Actual Result:

Job_Title
IT Manager
Programmer
Sales Manager
Sales Representative

Request 6:

```
--List the last name of all employees in departments 50 and 90 together with their
monthly salaries (rounded to 2 decimal places), sorted in ascending order of last
name.
```

Expected Result:

Last_Name	Monthly Salary
A…	aMonthlySalary
Etc.	Etc.

Query:

```
SELECT Last_Name, CAST(ROUND(Annual_Salary/12,2) AS DECIMAL(10,2)) AS 'Monthly Salary'
FROM Employees ORDER BY Last_Name ASC
```

Actual Result:

Last_Name	Monthly Salary
Acreman	1500
Betteridge	1458.33
Bown	1583.33
Court	1495.83
Dambridge	4166.67
Hadlow	1375
Mace	1833.33
Marsh	12500
Martinez	1333.33
Matthews	2000
Molavi	3750
Monroe	1458.33
Repetto	1333.33
Tetbury	1750
Trotter	1500
Watts	1833.33
Williams	3750

Request 7:

```
--Show the total salaries figure for one month displayed with no decimal places
```

Expected Result:

Total Monthly Salary
TheTotalMonthlySalary

Query:

```
SELECT CAST(SUM(Annual_Salary)/12 AS INTEGER) AS 'Total Monthly Salary' FROM Employees
```

Actual Result:

Total Monthly Salary
96970

Request 8:

```
--Show the total number of employees.
```

Expected Result:

Total Number of employees
The total

Query:

```
SELECT COUNT(*) AS 'Total Number of employees' FROM Employees
```

Actual Result:

Total Number of employees
71

Request 9:

```
--List the department number, department name and the number of employees for each
department that has more than 2 employees grouping by department number and department
name.
```

Expected Result:

Department_No	Department_Name	Number Of Employees
DepartmentNo	DepartmentName	Number Of Employees
Etc.	Etc.	Etc.

Query:

```
SELECT Departments.Department_No, Department_Name, COUNT(Employee_No) AS 'Number Of
Employees' FROM Departments JOIN Employees ON
(Departments.Department_No=Employees.Department_No) GROUP BY
Departments.Department_No, Department_Name HAVING COUNT(Employee_No)>2
```

Actual Result:

Department_No	Department_Name	Number Of Employees
10	Marketing	5
20	Administration	3
30	Accounting	5
50	IT	14
60	Sales	19
80	Finance	3
90	Management	3
110	Human Resources	4
190	Manufacturing	13

Request 10:

```
--List the department number, department name and the number of employees for the
department that has the highest number of employees using appropriate grouping
```

Expected Result:

Department_No	Department_Name	Number Of Employees
DepartmentNo	DepartmentName	Highest Number Of Employees

Query:

```
SELECT TOP 1 Departments.Department_No, Department_Name, COUNT(Employee_No) AS 'Number
Of Employees' FROM Departments JOIN Employees ON
(Departments.Department_No=Employees.Department_No)
GROUP BY Departments.Department_No, Department_Name HAVING COUNT(Employee_No)>2 ORDER
BY [Number Of Employees] DESC
```

Actual Result:

Department_No	Department_Name	Number Of Employees
60	Sales	19

Request 11:

```
--List the department number and name for all departments where no programmers work.
```

Expected Result:

Department_No	Department_Name
Department_No	Department_Name
Etc.	Etc.

Query:

```
SELECT Department_No, Department_Name FROM Departments WHERE Department_No NOT IN
(SELECT Department_No FROM Employees WHERE Job_ID = (SELECT Job_ID FROM Jobs WHERE
Job_Title = 'programmer'))
```

Actual Result:

Department_No	Department_Name
10	Marketing
20	Administration
30	Accounting
60	Sales
70	Shipping
80	Finance
90	Management
110	Human Resources
190	Manufacturing

Request 12:

```
--Add the following new job IT_ANAL, System Analyst, 10000, 15000
```

Expected Result:

The following will be added to the Jobs table

Job_ID	Job_Title	Min_Salary	Max_Salary
IT_ANAL	System Analyst	10000	15000

Query:

```
INSERT INTO Jobs VALUES('IT_ANAL', 'System Analyst', 10000, 15000)
```

Actual Result:

The following was added to the Jobs table

Job_ID	Job_Title	Min_Salary	Max_Salary
IT_ANAL	System Analyst	10000	15000

Request 13:

```
--Update all the maximum salaries for jobs with an increase of 1000.
```

Expected Result:

Before:	Max_Salary	After:	Max_Salary
	Max_Salary		Max_Salary+1000
	Etc.		Etc.

Query:

```
UPDATE Jobs SET Max_Salary += 1000;
```

Actual Result:

Before:	Max_Salary	After:	Max_Salary
	25000		26000
	11500		12500
	25000		26000
	15000		16000
	12000		13000
	30000		31000
	200000		201000
	100000		101000
	11000		12000
	25000		26000
	15000		16000
	50000		51000
	15000		16000
	60000		61000
	25000		26000
	17500		18500
	11000		12000
	8000		9000
	15000		16000
	20000		21000
	13000		14000
	12500		13500
	25000		26000
	10500		11500
	12000		13000

Request 14:

```
--List all the data for jobs sorted in ascending order of job id.
```

Expected Result:

Job_ID	Job_Title	Min_Salary	Max_Salary
AC_ACCOUNT	Accountant	8500	26000
Etc.	Etc.	Etc.	Etc.

Query:

```
SELECT * FROM Jobs ORDER BY Job_ID ASC
```

Actual Result:

Job_ID	Job_Title	Min_Salary	Max_Salary
AC_ACCOUNT	Accountant	8500	26000
AC_CLERK	Accounts Clerk	5000	12500
AC_MGR	Accounts Manager	10000	26000
AD_ASST	Administration Assistant	4500	16000
AD_CLERK	Administration Clerk	4500	13000
AD_MGR	Administration Manager	20000	31000
CO_CHAIR	Company Chairman	100000	201000
CO_DIR	Company Director	30000	101000
FN_CLERK	Finance Clerk	6000	12000
FN_MGR	Finance Manager	13000	26000
HR_CLERK	Human Resources Clerk	10000	16000
HR_MGR	Human Resources Manager	30000	51000
IT_ANAL	System Analyst	10000	16000
IT_MGR	IT Manager	40000	61000
IT_PROG	Programmer	15000	26000
MK_MGR	Marketing Manager	10500	18500
MK_REP	Marketing Representative	4490	12000
PC_CLERK	Purchase Clerk	4000	9000
PC_MGR	Purchase Manager	4600	16000
SA_MGR	Sales Manager	10000	21000
SA_REP	Sales Representative	4000	14000
SH_CLERK	Shipping Clerk	8000	13500
SH_MGR	Shipping Manager	20000	26000
ST_CLERK	Stock Clerk	4400	11500
ST_MGR	Stock Manager	6900	13000

Request 15:

```
--a) The job history for employee number 102 is no longer required. Delete this record
--b) List all the data for job history sorted in ascending order of employee number.
```

Expected Result:

Job History for employee number 102 will be deleted and the following result will be shown

Employee_No	Start_Date	End_Date	Job_ID	Department_No
Employee_No	Start_Date	End_Date	Job_ID	Department_No
Etc.	Etc.	Etc.	Etc.	Etc.

Query:

```
DELETE FROM Job_History WHERE Employee_No = 102;
SELECT * FROM Job_History ORDER BY Employee_No ASC
```

Actual Result:

Employee_No	Start_Date	End_Date	Job_ID	Department_No
100	16/11/1999 00:00:00	20/09/2000 00:00:00	MK_REP	10
107	13/03/1993 00:00:00	29/01/1999 00:00:00	IT-PROG	50
110	12/05/1996 00:00:00	06/12/1998 00:00:00	FN_CLERK	80
122	10/11/1986 00:00:00	31/07/1989 00:00:00	HR_CLERK	110
125	19/02/1986 00:00:00	12/05/1998 00:00:00	FN_CLERK	80
130	01/03/1988 00:00:00	04/12/1996 00:00:00	FN_MGR	80
130	04/12/1996 00:00:00	23/06/1999 00:00:00	CO_DIR	90
150	16/03/1998 00:00:00	25/01/1998 00:00:00	SA_REP	60
160	01/01/1986 00:00:00	25/03/1993 00:00:00	ST_CLERK	190
200	16/09/1987 00:00:00	29/10/1996 00:00:00	AD_CLERK	20
200	29/10/1996 00:00:00	12/04/1999 00:00:00	AD_ASST	20
250	02/12/1985 00:00:00	29/01/1990 00:00:00	SH_CLERK	70

Request 16:

```
--Produce a list of employees showing percentage raises, employee numbers and old and
new salaries.
--Employees in departments 20 and 10 are given a 5% rise,
--employees in departments 50, 80, 90 and 110 are given a 10% rise and employees in
other departments are not given a rise.
```

Expected Result:

Percentage Raise	Employee_No	Old Salary	New Salary
Percentage Raise	Employee_No	Old Salary	New Salary
Etc.	Etc.	Etc.	Etc.

Query:

```
SELECT
CASE WHEN Department_No IN (20, 10) THEN '5%' WHEN Department_No IN (50, 80, 90 , 110)
THEN '10%' ELSE NULL END AS 'Percentage Raise',
Employee_No,
Annual_Salary AS 'Old Salary',
CASE WHEN Department_No IN (20, 10) THEN CAST(Annual_Salary*1.05 AS decimal(10,2))
WHEN Department_No IN (50, 80, 90 , 110) THEN CAST(Annual_Salary*1.1 AS decimal(10,2))
ELSE Annual_Salary END AS 'New Salary'
FROM Employees
```

Actual Result:

Percentage Raise	Employee_No	Old Salary	New Salary
5%	100	17000	17850
10%	102	45000	49500
10%	107	45000	49500
NULL	110	20000	20000
5%	112	8000	8400
NULL	115	7250	7250
NULL	119	10000	10000
NULL	120	10750	10750
10%	122	35000	38500
NULL	123	9000	9000
10%	125	24500	26950
NULL	126	8200	8200
10%	130	150000	165000
NULL	131	8000	8000
NULL	132	8200	8200
10%	133	16500	18150
NULL	135	7800	7800
NULL	138	9500	9500
NULL	139	7500	7500
NULL	141	8000	8000
NULL	142	7000	7000
NULL	143	8400	8400
NULL	145	10200	10200
NULL	146	9100	9100
NULL	147	12000	12000
NULL	148	11500	11500
10%	149	22000	24200
NULL	150	10000	10000
NULL	152	9500	9500
NULL	154	8900	8900
10%	155	17950	19745
NULL	156	8100	8100
10%	159	18000	19800
NULL	160	11000	11000
10%	166	16000	17600
NULL	169	7100	7100
10%	170	24000	26400
10%	172	18000	19800
10%	174	17500	19250
10%	175	16000	17600
10%	177	17500	19250
10%	179	50000	55000

5%	180	6500	6825
NULL	182	23000	23000
NULL	185	9500	9500
NULL	195	6500	6500
5%	196	9500	9975
10%	198	14990	16489
5%	200	25000	26250
5%	202	14200	14910
10%	204	10200	11220
NULL	205	7500	7500
NULL	206	11800	11800
NULL	208	10000	10000
NULL	210	9800	9800
NULL	212	10000	10000
10%	214	22000	24200
10%	220	21000	23100
10%	222	19000	20900
NULL	224	13000	13000
NULL	236	9000	9000
NULL	240	9800	9800
NULL	245	9800	9800
NULL	246	10500	10500
NULL	250	25000	25000
NULL	255	12000	12000
5%	260	10500	11025
10%	266	15000	16500
10%	267	14200	15620
10%	274	8500	9350
5%	275	10900	11445

Request 17:

```
--Create a new view for manager's details only using all the fields from the employee
table. Apply a CHECK constraint.
```

Expected Result:

A view called Managers containing the managers' details from the employee table with a check constraint will be created.

Query:

```
CREATE VIEW Managers AS SELECT * FROM Employees WHERE Job_ID LIKE '%MGR' WITH CHECK
OPTION
```

Actual Result:

A view called Managers containing the managers' details from the employee table with a check constraint was created.

Request 18:

--Show all the fields and all the managers using the view for managers sorted in ascending order of employee number.

Expected Result:

Employee_No	First_Name	Last_Name	Email	Phone_Number	Hire_Date	Job_ID	Annual_Salary	Commission_Percent	Manager_ID	Department_No
Employee_No	First_Name	Last_Name	Email	Phone_Number	Hire_Date	Job_ID	Annual_Salary	Commission_Percent	Manager_ID	Department_No
Etc.	Etc.	Etc.	Etc.	Etc.	Etc.	Etc.	Etc.	Etc.	Etc.	Etc.

Query:

```
SELECT * FROM Managers ORDER BY Employee_No ASC
```

Actual Result:

Employee_No	First_Name	Last_Name	Email	Phone_Number	Hire_Date	Job_ID	Annual_Salary	Commission_Percent	Manager_ID	Department_No
100	David	Polanski	DPOLANSKI	555 324 190	16/11/1999 00:00:00	MK_MGR	17000	NULL	100	10
107	Megan	Williams	MWILLIAMS	111 987 245	13/03/1993 00:00:00	IT_MGR	45000	NULL	107	50
110	William	Clifford	WCLIFFORD	222 456 666	12/05/1996 00:00:00	AC_MGR	20000	NULL	110	30
122	Julie	Whittaker	JWHITTAKER	532 555 160	10/11/1986 00:00:00	HR_MGR	35000	NULL	122	110
125	Linda	Yorke	LYORKE	888 666 111	19/02/1986 00:00:00	FN_MGR	24500	NULL	125	80
150	Wayne	Hartmore	WHARTMORE	444 232 135	16/03/1998 00:00:00	SA_MGR	10000	0.35	150	60
160	Trevor	Mills	TMILLS	345 222 255	01/01/1986 00:00:00	ST_MGR	11000	NULL	160	190
200	Loraine	Merton	LMERTON	665 239 989	16/09/1987 00:00:00	AD_MGR	25000	NULL	200	20
250	Tony	Luigi	TLUIGI	333 111 120	02/12/1985 00:00:00	SH_MGR	25000	NULL	250	70

Request 19:

--Grant the authority to all other users to access the view for managers for SELECT statments only.

Expected Result:

All users will be able to use SELECT statements on the view for managers

Query:

```
GRANT SELECT ON Managers TO PUBLIC
```

Actual Result:

All users are able to use SELECT statements on the view for managers

Request 20:

```
--Create an index named LOC_POSTAL_CODE on the Postal Code in the locations table.
Provide a printout showing that the index has been created.
```

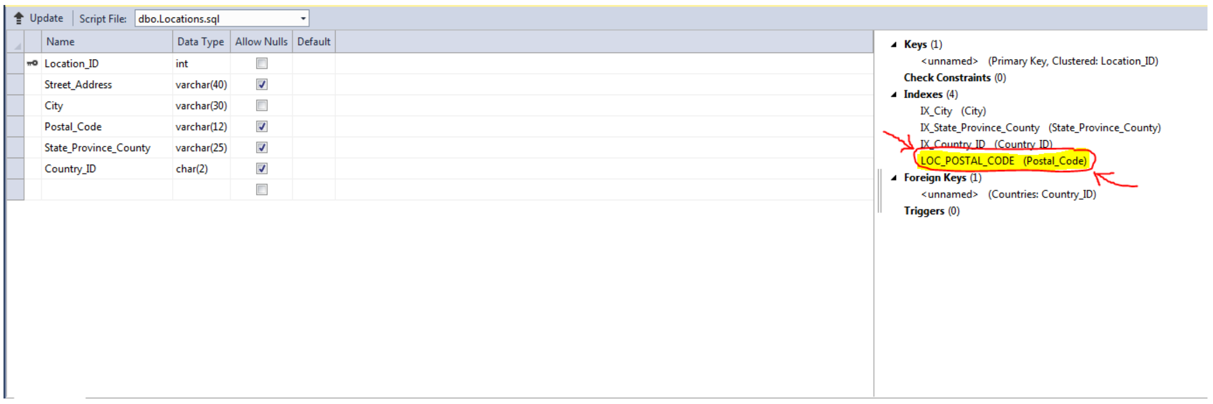
Expected Result:

An index will be created called LOC_POSTAL_CODE in the locations table.

Query:

```
CREATE INDEX LOC_POSTAL_CODE ON Locations (Postal_Code)
```

Actual Result:



Name	Data Type	Allow Nulls	Default
Location_ID	int	☐	
Street_Address	varchar(40)	✓	
City	varchar(30)	☐	
Postal_Code	varchar(12)	✓	
State_Province_County	varchar(25)	✓	
Country_ID	char(2)	✓	
		☐	

```
Keys (1)
    <unnamed>  (Primary Key, Clustered: Location_ID)
Check Constraints (0)
Indexes (4)
    IX_City  (City)
    IX_State_Province_County  (State_Province_County)
    IX_Country_ID  (Country_ID)
    LOC_POSTAL_CODE  (Postal_Code)
Foreign Keys (1)
    <unnamed>  (Countries: Country_ID)
Triggers (0)
```