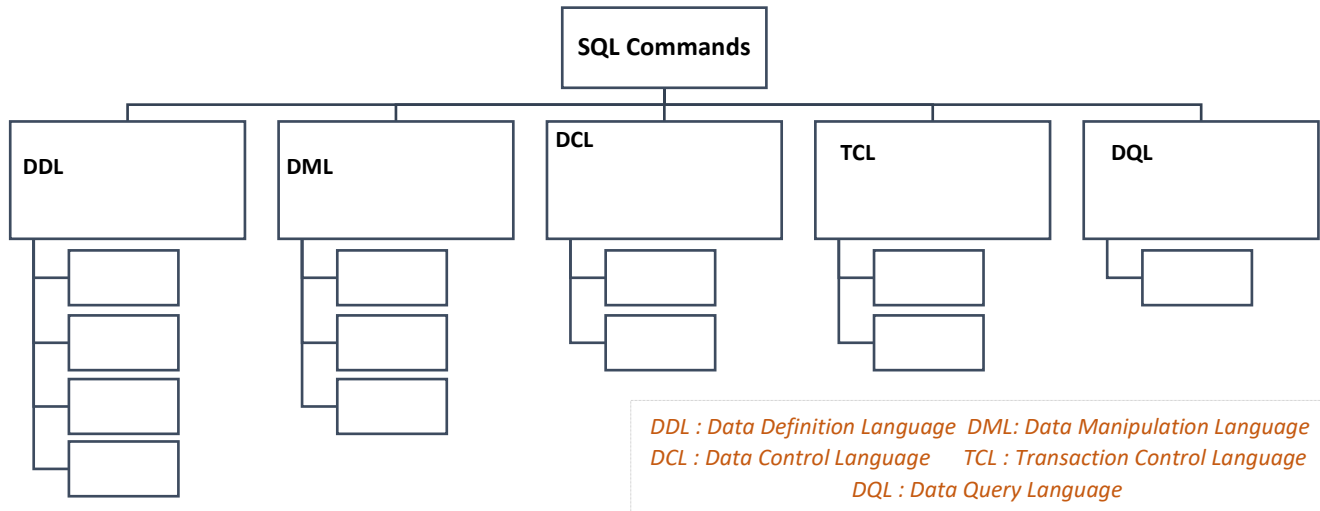


Structured Query language (SQL)



	<pre>select * from customer</pre>
	<pre>select customerid, customernumber, lastname, firstname from customer</pre>
	<pre>alter table customer add phonenummer varchar(20)</pre>
	<pre>update customer set phonenummer='1234545346' where customerid=1 update customer set phonenummer='45554654' where customerid=2</pre>
	<pre>alter table customer drop column phonenummer</pre>
	<pre>delete from customer where country='Thailand'</pre>
	<pre>drop table customer</pre>
	<pre>alter table customer alter column phonenummer varchar(10)</pre>

[illegible]

	one character multiple characters

	<pre>SELECT JobTitle, COUNT(JobTitle) FROM EmployeeDemographics ED JOIN EmployeeSalary ES ON ED.EmployeeID = ES.EmployeeID GROUP BY JobTitle HAVING COUNT(JobTitle) > 1 SELECT JobTitle, AVG(Salary) FROM EmployeeDemographics ED JOIN EmployeeSalary ES ON ED.EmployeeID = ES.EmployeeID GROUP BY JobTitle HAVING AVG(Salary) > 45000 ORDER BY AVG(Salary)</pre>																																																						
	<pre>SELECT CAST('2017-08-25 00:00:00.000' AS date) SELECT CONVERT(date, '2017-08-25 00:00:00.000')</pre>																																																						
	<pre>SELECT FirstName, LastName, Age, CASE WHEN Age > 30 THEN 'Old' WHEN Age BETWEEN 27 AND 30 THEN 'Young' ELSE 'Baby' END FROM EmployeeDemographics ED WHERE Age IS NOT NULL ORDER BY Age -- SELECT FirstName, LastName, JobTitle, Salary, CASE WHEN JobTitle = 'Salesman' THEN Salary + (Salary *.10) WHEN JobTitle = 'Accountant' THEN Salary + (Salary *.05) WHEN JobTitle = 'HR' THEN Salary + (Salary *.000001) ELSE Salary + (Salary *.03) END AS SalaryAfterRaise FROM EmployeeDemographics ED JOIN EmployeeSalary ES ON ED.EmployeeID = ES.EmployeeID</pre>																																																						
	<pre>SELECT FirstName, LastName, Gender, Salary, COUNT(Gender) OVER (PARTITION BY Gender) AS TotalGender FROM EmployeeDemographics ED JOIN EmployeeSalary ES ON ED.EmployeeID = ES.EmployeeID</pre> <table><tr><th></th><th>FirstName</th><th>LastName</th><th>Gender</th><th>Salary</th><th>TotalGender</th></tr><tr><td>1</td><td>Pam</td><td>Beasley</td><td>Female</td><td>36000</td><td>3</td></tr><tr><td>2</td><td>Angela</td><td>Martin</td><td>Female</td><td>47000</td><td>3</td></tr><tr><td>3</td><td>Meredith</td><td>Palmer</td><td>Female</td><td>41000</td><td>3</td></tr><tr><td>4</td><td>Stanley</td><td>Hudson</td><td>Male</td><td>48000</td><td>5</td></tr><tr><td>5</td><td>Kevin</td><td>Malone</td><td>Male</td><td>42000</td><td>5</td></tr><tr><td>6</td><td>Michael</td><td>Scott</td><td>Male</td><td>65000</td><td>5</td></tr><tr><td>7</td><td>Dwight</td><td>Schrute</td><td>Male</td><td>63000</td><td>5</td></tr><tr><td>8</td><td>Jim</td><td>Halpert</td><td>Male</td><td>45000</td><td>5</td></tr></table>		FirstName	LastName	Gender	Salary	TotalGender	1	Pam	Beasley	Female	36000	3	2	Angela	Martin	Female	47000	3	3	Meredith	Palmer	Female	41000	3	4	Stanley	Hudson	Male	48000	5	5	Kevin	Malone	Male	42000	5	6	Michael	Scott	Male	65000	5	7	Dwight	Schrute	Male	63000	5	8	Jim	Halpert	Male	45000	5
	FirstName	LastName	Gender	Salary	TotalGender																																																		
1	Pam	Beasley	Female	36000	3																																																		
2	Angela	Martin	Female	47000	3																																																		
3	Meredith	Palmer	Female	41000	3																																																		
4	Stanley	Hudson	Male	48000	5																																																		
5	Kevin	Malone	Male	42000	5																																																		
6	Michael	Scott	Male	65000	5																																																		
7	Dwight	Schrute	Male	63000	5																																																		
8	Jim	Halpert	Male	45000	5																																																		

	<pre> -- Remove space Select EmployeeID, TRIM(EmployeeID) AS IDTRIM FROM EmployeeErrors Select EmployeeID, RTRIM(EmployeeID) as IDRTRIM FROM EmployeeErrors Select EmployeeID, LTRIM(EmployeeID) as IDLTRIM FROM EmployeeErrors -- Replace Select LastName, REPLACE(LastName, '- Fired', '') as LastNameFixed FROM EmployeeErrors -- Substring Select Substring(err.FirstName,1,3), Substring(dem.FirstName,1,3), Substring(err.LastName,1,3), Substring(dem.LastName,1,3) FROM EmployeeErrors err JOIN EmployeeDemographics dem on Substring(err.FirstName,1,3) = Substring(dem.FirstName,1,3) and Substring(err.LastName,1,3) = Substring(dem.LastName,1,3) -- UPPER and LOWER CASE Select firstname, LOWER(firstname) from EmployeeErrors Select Firstname, UPPER(FirstName) from EmployeeErrors" </pre>
	<pre> CREATE PROCEDURE Temp_Employee @JobTitle nvarchar(100) AS DROP TABLE IF EXISTS #temp_employee Create table #temp_employee (JobTitle varchar(100), EmployeesPerJob int , AvgAge int, AvgSalary int) Insert into #temp_employee SELECT JobTitle, Count(JobTitle), Avg(Age), AVG(salary) FROM EmployeeDemographics emp JOIN EmployeeSalary sal ON emp.EmployeeID = sal.EmployeeID where JobTitle = @JobTitle --- make sure to change this in this script from original above group by JobTitle Select * From #temp_employee GO; </pre>

```
--- only need to run this on next time
EXEC Temp_Employee @JobTitle = 'Salesman'
```

```
-- Subquery in Select
SELECT EmployeeID, Salary, (SELECT AVG(Salary) FROM
EmployeeSalary) AS AllAvgSalary
FROM EmployeeSalary
```

```
-- with Partition By
SELECT EmployeeID, Salary, AVG(Salary) OVER () AS
AllAvgSalary
FROM EmployeeSalary
```

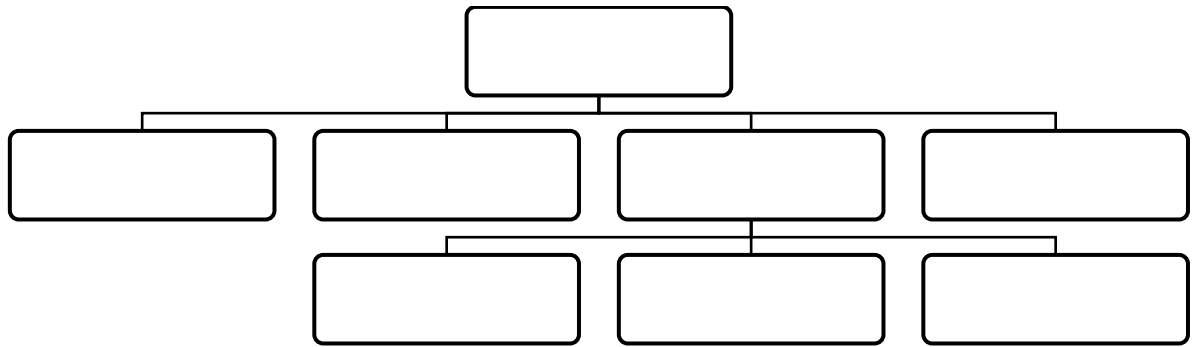
	EmployeeID	Salary	AllAvgSalary
1	1001	45000	47909
2	1002	36000	47909
3	1003	63000	47909
4	1004	47000	47909
5	1005	50000	47909

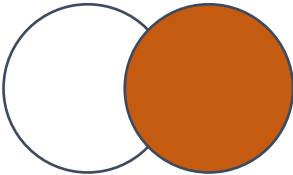
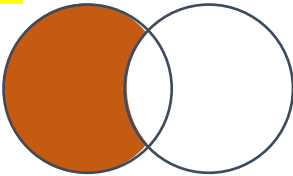
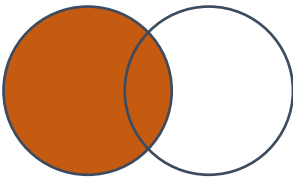
```
-- Subquery in From
SELECT a.EmployeeID, AllAvgSalary
FROM (SELECT EmployeeID, Salary, AVG(Salary) OVER () AS
AllAvgSalary
      FROM EmployeeSalary) a
ORDER BY a.EmployeeID
```

	EmployeeID	AllAvgSalary
1	NULL	47909
2	1001	47909
3	1002	47909
4	1003	47909
5	1004	47909
6	1005	47909

```
-- Subquery in Where
SELECT EmployeeID, JobTitle, Salary
FROM EmployeeSalary
WHERE EmployeeID in (SELECT EmployeeID FROM
EmployeeDemographics
                     WHERE Age > 30)
```

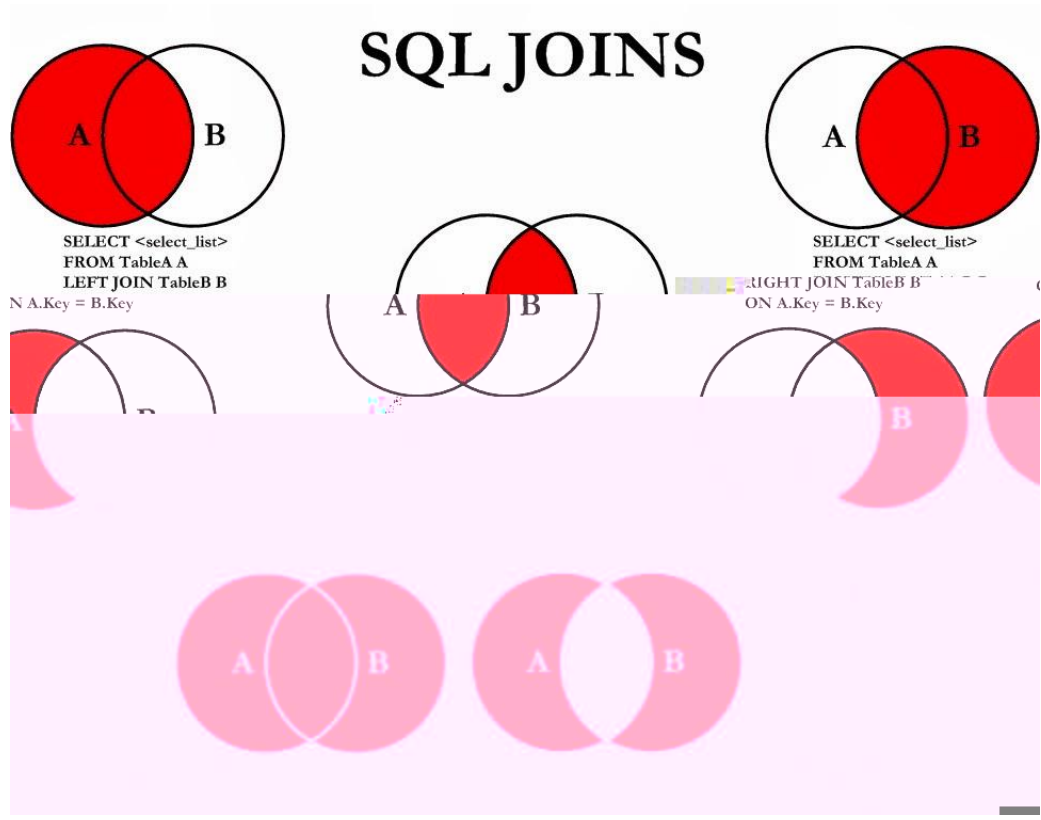
```
SELECT EmployeeID, JobTitle, Salary
FROM EmployeeSalary
WHERE Salary in (SELECT Max(Salary) FROM EmployeeSalary)
```





employeeID	employeefirstname	employeelastname	managerID
1001			
1002			
1003			

	<table><tr><th>employeeID</th><th>Full Name</th><th>managerID</th><th>managerName</th></tr><tr><td>1002</td><td></td><td></td><td></td></tr><tr><td>1003</td><td></td><td></td><td></td></tr></table>	employeeID	Full Name	managerID	managerName	1002				1003							
	employeeID	Full Name	managerID	managerName													
	1002																
	1003																
	<table><tr><th>employeeID</th><th>Full Name</th><th>managerID</th><th>managerName</th></tr><tr><td>1001</td><td></td><td></td><td></td></tr><tr><td>1002</td><td></td><td></td><td></td></tr><tr><td>1003</td><td></td><td></td><td></td></tr></table>	employeeID	Full Name	managerID	managerName	1001				1002				1003			
	employeeID	Full Name	managerID	managerName													
1001																	
1002																	
1003																	



SQL UNIONS

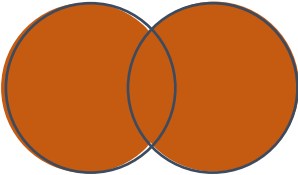
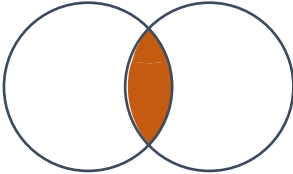
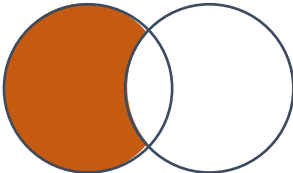
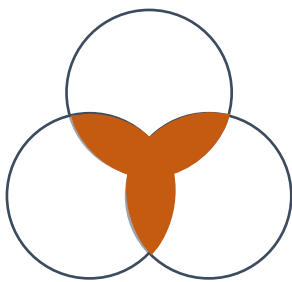
	
	
	

Table & View

	<div></div> <div></div>
	<pre>DROP TABLE IF EXISTS #temp_Employee Create table #temp_Employee (JobTitle varchar(100), EmployeesPerJob int, AvgAge int, AvgSalary int) Insert INTO #temp_Employee SELECT JobTitle, Count(JobTitle), Avg(Age), AVG(salary) FROM EmployeeDemographics emp JOIN EmployeeSalary sal ON emp.EmployeeID = sal.EmployeeID group by JobTitle SELECT * FROM #temp_Employee</pre>
	<pre>WITH CTE_Employee AS (SELECT FirstName, LastName, Gender, Salary, COUNT(Gender) OVER (PARTITION BY Gender) AS TotalGender FROM EmployeeDemographics ED JOIN EmployeeSalary ES ON ED.EmployeeID = ES.EmployeeID WHERE Salary > '45000') SELECT FirstName, LastName, Gender, TotalGender FROM CTE_Employee WHERE TotalGender = (SELECT MIN(TotalGender) FROM CTE_Employee)</pre>
	<div></div>



SQL RANKS

```
SELECT *,
       ROW_NUMBER() OVER(ORDER BY Salary DESC) SalaryRank
FROM EmployeeSalary
```

	EmployeeID	JobTitle	Salary	SalaryRank
1	1006	Regional Manager	65000	1
2	1003	Salesman	63000	2
3	1005	HR	50000	3
4	1008	Salesman	48000	4
5	1004	Accountant	47000	5
6	1010	NULL	47000	6
7	1001	Salesman	45000	7
8	NULL	Salesman	43000	8
9	1009	Accountant	42000	9
10	1007	Supplier Relations	41000	10
11	1002	Receptionist	36000	11

USING PARTITION BY

```
SELECT *,
       RANK() OVER(PARTITION BY JobTitle ORDER BY Salary DESC)
SalaryRank
FROM EmployeeSalary
ORDER BY JobTitle, SalaryRank
```

	EmployeeID	JobTitle	Salary	SalaryRank
1	1010	NULL	47000	1
2	1004	Accountant	47000	1
3	1009	Accountant	42000	2
4	1005	HR	50000	1
5	1002	Receptionist	36000	1
6	1006	Regional Manager	65000	1
7	1003	Salesman	63000	1
8	1008	Salesman	48000	2
9	1001	Salesman	45000	3
10	NULL	Salesman	43000	4
11	1007	Supplier Relations	41000	1

NOT USING PARTITION BY

```
SELECT *,
       RANK() OVER(ORDER BY Salary DESC) SalaryRank
FROM EmployeeSalary
ORDER BY SalaryRank
```

	EmployeeID	JobTitle	Salary	SalaryRank
1	1006	Regional Manager	65000	1
2	1003	Salesman	63000	2
3	1005	HR	50000	3
4	1008	Salesman	48000	4
5	1004	Accountant	47000	5
6	1010	NULL	47000	5
7	1001	Salesman	45000	7
8	NULL	Salesman	43000	8
9	1009	Accountant	42000	9
10	1007	Supplier Relations	41000	10
11	1002	Receptionist	36000	11



```
SELECT *,
       DENSE_RANK() OVER(ORDER BY Salary DESC) SalaryRank
FROM EmployeeSalary
ORDER BY SalaryRank
```

	EmployeeID	JobTitle	Salary	SalaryRank
1	1006	Regional Manager	65000	1
2	1003	Salesman	63000	2
3	1005	HR	50000	3
4	1008	Salesman	48000	4
5	1004	Accountant	47000	5
6	1010	NULL	47000	5
7	1001	Salesman	45000	6
8	NULL	Salesman	43000	7
9	1009	Accountant	42000	8
10	1007	Supplier Relations	41000	9
11	1002	Receptionist	36000	10

RANK()

```
SELECT *,
       RANK() OVER(PARTITION BY JobTitle ORDER
BY Salary DESC) SalaryRank
FROM EmployeeSalary
ORDER BY JobTitle, SalaryRank
```

	EmployeeID	JobTitle	Salary	SalaryRank
1	1010	NULL	47000	1
2	1004	Accountant	47000	1
3	1009	Accountant	42000	2
4	1005	HR	50000	1
5	1002	Receptionist	36000	1
6	1006	Regional Manager	65000	1
7	1003	Salesman	63000	1
8	1001	Salesman	48000	2
9	1008	Salesman	48000	2
10	NULL	Salesman	43000	4
11	1007	Supplier Relations	41000	1

-- skip a rank if have similar values

DENSE_RANK()

```
SELECT *,
       DENSE_RANK() OVER(PARTITION BY JobTitle
ORDER BY Salary DESC) SalaryRank
FROM EmployeeSalary
ORDER BY JobTitle, SalaryRank
```

	EmployeeID	JobTitle	Salary	SalaryRank
1	1010	NULL	47000	1
2	1004	Accountant	47000	1
3	1009	Accountant	42000	2
4	1005	HR	50000	1
5	1002	Receptionist	36000	1
6	1006	Regional Manager	65000	1
7	1003	Salesman	63000	1
8	1001	Salesman	48000	2
9	1008	Salesman	48000	2
10	NULL	Salesman	43000	3
11	1007	Supplier Relations	41000	1

-- maintains the rank and does not give any gap for the values



```
SELECT *,  
       NTILE(3) OVER(ORDER BY Salary DESC) SalaryRank  
FROM EmployeeSalary  
ORDER BY SalaryRank;
```

	EmployeeID	JobTitle	Salary	SalaryRank
1	1006	Regional Manager	65000	1
2	1003	Salesman	63000	1
3	1005	HR	50000	1
4	1001	Salesman	48000	1
5	1008	Salesman	48000	2
6	1004	Accountant	47000	2
7	1010	NULL	47000	2
8	NULL	Salesman	43000	2
9	1009	Accountant	42000	3
10	1007	Supplier Relations	41000	3
11	1002	Receptionist	36000	3

Group 1

Group 2

Group 3

```
SELECT *,  
       NTILE(3) OVER(PARTITION BY JobTitle ORDER BY Salary DESC)  
SalaryRank  
FROM EmployeeSalary  
ORDER BY JobTitle, SalaryRank;
```



Group 1

Group 2

Group 3



	<pre>Select a.ParcelID, a.PropertyAddress, b.ParcelID, b.PropertyAddress, ISNULL(a.PropertyAddress,b.PropertyAddress) From NashvilleHousing a JOIN NashvilleHousing b on a.ParcelID = b.ParcelID AND a.[UniqueID] <> b.[UniqueID] Where a.PropertyAddress is null</pre> <table><tr><th>ParcelID</th><th>PropertyAddress</th><th>ParcelID</th><th>PropertyAddress</th><th>(No column name)</th></tr><tr><td>1</td><td>025 07 0 031.00 NULL</td><td>025 07 0 031.00</td><td>410 ROSEHILL CT, GOODLETTSVILLE</td><td>410 ROSEHILL CT, GOODLETTSVILLE</td></tr><tr><td>2</td><td>026 01 0 069.00 NULL</td><td>026 01 0 069.00</td><td>141 TWO MILE PIKE, GOODLETTSVILLE</td><td>141 TWO MILE PIKE, GOODLETTSVILLE</td></tr><tr><td>4</td><td>026 06 0A 038.00 NULL</td><td>026 06 0A 038.00</td><td>109 CANTON CT, GOODLETTSVILLE</td><td>109 CANTON CT, GOODLETTSVILLE</td></tr><tr><td>5</td><td>033 06 0 041.00 NULL</td><td>033 06 0 041.00</td><td>1129 CAMPBELL RD, GOODLETTSVILLE</td><td>1129 CAMPBELL RD, GOODLETTSVILLE</td></tr><tr><td>6</td><td>033 06 0A 002.00 NULL</td><td>033 06 0A 002.00</td><td>1116 CAMPBELL RD, GOODLETTSVILLE</td><td>1116 CAMPBELL RD, GOODLETTSVILLE</td></tr><tr><td>7</td><td>033 15 0 123.00 NULL</td><td>033 15 0 123.00</td><td>438 W CAMPBELL RD, GOODLETTSVILLE</td><td>438 W CAMPBELL RD, GOODLETTSVILLE</td></tr></table> <pre>Update a SET PropertyAddress = ISNULL(a.PropertyAddress,b.PropertyAddress) From NashvilleHousing a JOIN NashvilleHousing b on a.ParcelID = b.ParcelID AND a.[UniqueID] <> b.[UniqueID] Where a.PropertyAddress is null</pre>	ParcelID	PropertyAddress	ParcelID	PropertyAddress	(No column name)	1	025 07 0 031.00 NULL	025 07 0 031.00	410 ROSEHILL CT, GOODLETTSVILLE	410 ROSEHILL CT, GOODLETTSVILLE	2	026 01 0 069.00 NULL	026 01 0 069.00	141 TWO MILE PIKE, GOODLETTSVILLE	141 TWO MILE PIKE, GOODLETTSVILLE	4	026 06 0A 038.00 NULL	026 06 0A 038.00	109 CANTON CT, GOODLETTSVILLE	109 CANTON CT, GOODLETTSVILLE	5	033 06 0 041.00 NULL	033 06 0 041.00	1129 CAMPBELL RD, GOODLETTSVILLE	1129 CAMPBELL RD, GOODLETTSVILLE	6	033 06 0A 002.00 NULL	033 06 0A 002.00	1116 CAMPBELL RD, GOODLETTSVILLE	1116 CAMPBELL RD, GOODLETTSVILLE	7	033 15 0 123.00 NULL	033 15 0 123.00	438 W CAMPBELL RD, GOODLETTSVILLE	438 W CAMPBELL RD, GOODLETTSVILLE
ParcelID	PropertyAddress	ParcelID	PropertyAddress	(No column name)																																
1	025 07 0 031.00 NULL	025 07 0 031.00	410 ROSEHILL CT, GOODLETTSVILLE	410 ROSEHILL CT, GOODLETTSVILLE																																
2	026 01 0 069.00 NULL	026 01 0 069.00	141 TWO MILE PIKE, GOODLETTSVILLE	141 TWO MILE PIKE, GOODLETTSVILLE																																
4	026 06 0A 038.00 NULL	026 06 0A 038.00	109 CANTON CT, GOODLETTSVILLE	109 CANTON CT, GOODLETTSVILLE																																
5	033 06 0 041.00 NULL	033 06 0 041.00	1129 CAMPBELL RD, GOODLETTSVILLE	1129 CAMPBELL RD, GOODLETTSVILLE																																
6	033 06 0A 002.00 NULL	033 06 0A 002.00	1116 CAMPBELL RD, GOODLETTSVILLE	1116 CAMPBELL RD, GOODLETTSVILLE																																
7	033 15 0 123.00 NULL	033 15 0 123.00	438 W CAMPBELL RD, GOODLETTSVILLE	438 W CAMPBELL RD, GOODLETTSVILLE																																
❖ ❖ ❖	<pre>SELECT PropertyAddress, SUBSTRING(PropertyAddress, 1, CHARINDEX(',', PropertyAddress) -1) as Address , SUBSTRING(PropertyAddress, CHARINDEX(',', PropertyAddress) + 1 , LEN(PropertyAddress)) as City From NashvilleHousing</pre> <table><tr><th></th><th>PropertyAddress</th><th>Address</th><th>City</th></tr><tr><td>1</td><td>1808 FOX CHASE DR, GOODLETTSVILLE</td><td>1808 FOX CHASE DR</td><td>GOODLETTSVILLE</td></tr><tr><td>2</td><td>1832 FOX CHASE DR, GOODLETTSVILLE</td><td>1832 FOX CHASE DR</td><td>GOODLETTSVILLE</td></tr><tr><td>3</td><td>1864 FOX CHASE DR, GOODLETTSVILLE</td><td>1864 FOX CHASE DR</td><td>GOODLETTSVILLE</td></tr><tr><td>4</td><td>1853 FOX CHASE DR, GOODLETTSVILLE</td><td>1853 FOX CHASE DR</td><td>GOODLETTSVILLE</td></tr><tr><td>5</td><td>1829 FOX CHASE DR, GOODLETTSVILLE</td><td>1829 FOX CHASE DR</td><td>GOODLETTSVILLE</td></tr></table> <pre>ALTER TABLE NashvilleHousing Add PropertySplitAddress Nvarchar(255); ALTER TABLE NashvilleHousing Add PropertySplitCity Nvarchar(255);</pre>		PropertyAddress	Address	City	1	1808 FOX CHASE DR, GOODLETTSVILLE	1808 FOX CHASE DR	GOODLETTSVILLE	2	1832 FOX CHASE DR, GOODLETTSVILLE	1832 FOX CHASE DR	GOODLETTSVILLE	3	1864 FOX CHASE DR, GOODLETTSVILLE	1864 FOX CHASE DR	GOODLETTSVILLE	4	1853 FOX CHASE DR, GOODLETTSVILLE	1853 FOX CHASE DR	GOODLETTSVILLE	5	1829 FOX CHASE DR, GOODLETTSVILLE	1829 FOX CHASE DR	GOODLETTSVILLE											
	PropertyAddress	Address	City																																	
1	1808 FOX CHASE DR, GOODLETTSVILLE	1808 FOX CHASE DR	GOODLETTSVILLE																																	
2	1832 FOX CHASE DR, GOODLETTSVILLE	1832 FOX CHASE DR	GOODLETTSVILLE																																	
3	1864 FOX CHASE DR, GOODLETTSVILLE	1864 FOX CHASE DR	GOODLETTSVILLE																																	
4	1853 FOX CHASE DR, GOODLETTSVILLE	1853 FOX CHASE DR	GOODLETTSVILLE																																	
5	1829 FOX CHASE DR, GOODLETTSVILLE	1829 FOX CHASE DR	GOODLETTSVILLE																																	


```
ALTER TABLE NashvilleHousing
Add OwnerSplitAddress Nvarchar(255);
ALTER TABLE NashvilleHousing
Add OwnerSplitCity Nvarchar(255);
ALTER TABLE NashvilleHousing
Add OwnerSplitState Nvarchar(255);

Update NashvilleHousing
SET OwnerSplitAddress = PARSENAME(REPLACE(OwnerAddress,
',', '.'), 3)

Update NashvilleHousing
SET OwnerSplitCity = PARSENAME(REPLACE(OwnerAddress, ',',
'.') , 2)

Update NashvilleHousing
SET OwnerSplitState = PARSENAME(REPLACE(OwnerAddress, ',',
'.') , 1)
```

```
WITH RowNumCTE AS(
Select *,
        ROW_NUMBER() OVER (
            PARTITION BY ParcelID,
                        PropertyAddress,
                        SalePrice,
                        SaleDate,
                        LegalReference
            ORDER BY UniqueID) as row_num
From NashvilleHousing
order by ParcelID
)
--DELETE
Select * From RowNumCTE
Where row_num > 1
Order by PropertyAddress
```