GROUP BY and Aggregate Functions

- Count
- Sum
- Min
- Max
- Avg

- Group By
- Having
- Aggregates and Nulls

Aggregates and Grouping

- The GROUP BY statement is used in conjunction with the aggregate functions to group the result-set by one or more columns.
- GROUP BY comes after the WHERE clause (if there is one) and before the ORDER BY clause
- Items in the SELECT statement that aren't aggregated must be included in the GROUP BY clause

```
SELECT
    C.FirstName
    ,C.LastName
    ,I.Total
FROM Customer C
JOIN Invoice I
    ON I.CustomerID = C.CustomerId
WHERE Country = 'usa'
ORDER BY LastName
```

	FirstName	LastName	Total
1	Julia	Barnett	13.86
2	Julia	Barnett	1.98
3	Julia	Barnett	8.91
4	Julia	Barnett	1.98
5	Julia	Barnett	3.96
6	Julia	Barnett	11.94
7	Julia	Barnett	0.99
8	Michelle	Brooks	1.98
9	Michelle	Brooks	13.86
10	Michelle	Brooks	8.91
11	Michelle	Brooks	0.99
12	Michelle	Brooks	5.94

```
SELECT
    C.FirstName
    ,C.LastName
    --,I.Total
FROM Customer C
JOIN Invoice I
    ON I.CustomerID = C.CustomerId
WHERE Country = 'usa'
GROUP BY LastName, FirstName
ORDER BY LastName
```

	FirstName	LastName
1	Julia	Barnett
2	Michelle	Brooks
3	Kathy	Chase
4	Richard	Cunningham
5	John	Gordon
6	Tim	Goyer
7	Patrick	Gray
8	Frank	Harris
9	Heather	Leacock
10	Dan	Miller
11	Frank	Ralston
12	Jack	Smith
13	Victor	Stevens

Aggregates and Grouping 2

- Multiple aggregate functions can be applied to a single query
- Different columns can be aggregated in the same query
- If you use an ORDER BY clause the column(s) must be included in the GROUP BY clause

```
SELECT
    C.FirstName
    ,C.LastName
    ,SUM(I.Total) SumTotal
    ,AVG(I.Total) AvgTotal
    ,MIN(I.Total) MinTotal
    ,MAX(I.Total) MaxTotal
    ,COUNT(I.Total) CountTotal
FROM Customer C
JOIN Invoice I
    ON I.CustomerID = C.CustomerId
WHERE Country = 'usa'
GROUP BY LastName, FirstName
ORDER BY LastName
```

	FirstName	LastName	SumTotal	AvgTotal	MinTotal	MaxTotal	CountTotal
1	Julia	Barnett	43.62	6.231428	0.99	13.86	7
2	Michelle	Brooks	37.62	5.374285	0.99	13.86	7
3	Kathy	Chase	37.62	5.374285	0.99	13.86	7
4	Richard	Cunningham	47.62	6.802857	0.99	23.86	7
5	John	Gordon	37.62	5.374285	0.99	13.86	7
6	Tim	Goyer	38.62	5.517142	1.98	13.86	7
7	Patrick	Gray	37.62	5.374285	0.99	13.86	7
8	Frank	Harris	37.62	5.374285	0.99	13.86	7
9	Heather	Leacock	39.62	5.660000	0.99	13.86	7
10	Dan	Miller	39.62	5.660000	0.99	13.86	7
11	Frank	Ralston	43.62	6.231428	0.99	15.86	7
12	Jack	Smith	39.62	5.660000	0.99	13.86	7
13	Victor	Stevens	42.62	6.088571	0.99	18.86	7

Count() Function

- The COUNT function counts the total occurrences of the specified column in a record set
- If an asterisk is used then the function counts the total number of rows
- You insert the DISTINCT keyword if you want to only count distinct values

```
SELECT
    Country
    ,COUNT(*) CountryTotal
FROM Customer
GROUP BY Country
ORDER BY CountryTotal DESC
```

	Country	CountryTotal
1	USA	13
2	Canada	8
3	France	5
4	Brazil	5
5	Germany	4
6	United Kingdom	3
7	Portugal	2
8	India	2
9	Czech Republic	2
10	Denmark	1
11	Finland	1
12	Chile	1
13	Argentina	1

```
SELECT
    COUNT(*) Records
    ,COUNT(State) StateRecords
    ,COUNT(DISTINCT State) DistinctStateRecords
FROM Customer
```

	Records	StateRecords	DistinctStateRecords
1	59	30	25

```
SELECT
    CustomerId
    ,State
FROM Customer
```

	CustomerId	State
8	8	NULL
9	9	NULL
10	10	SP
11	11	SP
12	12	RJ
13	13	DF

Sum() Function

- The SUM function adds together the numbers in a grouping
- You insert the DISTINCT keyword if you want to only add distinct values
- SUM will add both positive and negative values

```
SELECT
    C.FirstName
    ,C.LastName
    ,I.Total
FROM Customer C
JOIN Invoice I
    ON I.CustomerID = C.CustomerId
WHERE Country = 'USA'
    AND State = 'CA'
```

	FirstName	LastName	Total
1	Frank	Harris	0.99
2	Frank	Harris	1.98
3	Frank	Harris	13.86
4	Frank	Harris	8.91
5	Frank	Harris	1.98
6	Frank	Harris	3.96
7	Frank	Harris	5.94
8	Tim	Goyer	1.98
9	Tim	Goyer	13.86
10	Tim	Goyer	8.91
11	Tim	Goyer	1.98
12	Tim	Goyer	3.96
13	Tim	Goyer	5.94
14	Tim	Goyer	1.99
15	Dan	Miller	1.98

```
SELECT
    C.FirstName
    ,C.LastName
    ,SUM(I.Total) SumTotal
FROM Customer C
JOIN Invoice I
    ON I.CustomerID = C.CustomerId
WHERE Country = 'USA'
    AND State = 'CA'
GROUP BY LastName, FirstName
ORDER BY SumTotal DESC
```

	FirstName	LastName	SumTotal
1	Dan	Miller	39.62
2	Tim	Goyer	38.62
3	Frank	Harris	37.62

Min() and Max()

- The MIN function returns the minimum value in a column
- The MAX function returns the maximum value in a column
- Both functions can work on string as well as numeric values

```
SELECT
    C.FirstName
    ,C.LastName
    ,MIN(I.Total) MinTotal
    ,MAX(I.Total) MaxTotal
FROM Customer C
JOIN Invoice I
    ON I.CustomerID = C.CustomerId
WHERE Country = 'usa'
GROUP BY LastName, FirstName
ORDER BY MaxTotal DESC
```

	FirstName	LastName	MinTotal	MaxTotal
1	Richard	Cunningham	0.99	23.86
2	Victor	Stevens	0.99	18.86
3	Frank	Ralston	0.99	15.86
4	Heather	Leacock	0.99	13.86
5	Jack	Smith	0.99	13.86
6	John	Gordon	0.99	13.86
7	Julia	Barnett	0.99	13.86
8	Kathy	Chase	0.99	13.86
9	Michelle	Brooks	0.99	13.86
10	Patrick	Gray	0.99	13.86
11	Tim	Goyer	1.98	13.86

```
SELECT
    C.FirstName
    ,C.LastName
    ,I.Total
FROM Customer C
JOIN Invoice I
    ON I.CustomerID = C.CustomerId
WHERE C.LastName = 'Cunningham'
ORDER BY Total
```

	FirstName	LastName	Total
1	Richard	Cunningham	0.99
2	Richard	Cunningham	1.98
3	Richard	Cunningham	1.98
4	Richard	Cunningham	3.96
5	Richard	Cunningham	5.94
6	Richard	Cunningham	8.91
7	Richard	Cunningham	23.86

Avg() Function

- The AVG function returns the average of values in a group
- Null values are ignored

```
SELECT
    C.FirstName
    ,C.LastName
    ,AVG(I.Total) Average
FROM Customer C
JOIN Invoice I
    ON I.CustomerID = C.CustomerId
WHERE Country = 'usa'
GROUP BY LastName, FirstName
```

	FirstName	LastName	Average
1	Dan	Miller	5.660000
2	Frank	Harris	5.374285
3	Frank	Ralston	6.231428
4	Heather	Leacock	5.660000
5	Jack	Smith	5.660000
6	John	Gordon	5.374285
7	Julia	Barnett	6.231428
8	Kathy	Chase	5.374285
9	Michelle	Brooks	5.374285
10	Patrick	Gray	5.374285
11	Richard	Cunningham	6.802857
12	Tim	Goyer	5.517142
13	Victor	Stevens	6.088571

Aggregates without GROUP BY

- The GROUP BY clause is not used if you are grouping against an entire result set or table
- If there are no non-aggregated columns then you don't need the GROUP BY clause

```
SELECT COUNT(*) CountInvoice
FROM Invoice

SELECT MAX(Total) MaxInvoice
FROM Invoice

SELECT MIN(Total) MinInvoice
FROM Invoice

SELECT AVG(Total) AvgInvoice
FROM Invoice
```

Results | Messages

	CountInvoice
1	412

	MaxInvoice
1	25.86

	MinInvoice
1	0.99

	AvgInvoice
1	5.651941

Aggregates and NULL

- Aggregate functions will exclude NULL values from their calculations
- If you want to include NULL values then you will need to enclose the column in an ISNULL scalar function

```
SELECT
    LastName
    ,Company
FROM Customer


SELECT
    COUNT(LastName) CountLastName
    ,COUNT(Company) CountCompany
    ,COUNT(ISNULL(Company,''))
        AS CountCompanyWithNull
FROM Customer

SELECT
    MIN(Company) MinCompany
FROM Customer
```

Results | Messages

	LastName	Company
1	Gonçalves	Embraer - Empresa Brasileira de Aeronáutica S.A.
2	Köhler	NULL
3	Tremblay	NULL
4	Hansen	NULL
5	Wichterlová	JetBrains s.r.o.
6	Holý	NULL
7	Gruber	NULL
8	Peeters	NULL
9	Nielsen	NULL
10	Martins	Woodstock Discos

	CountLastName	CountCompany	CountCompanyWithNull
1	59	10	59

	MinCompany
1	Apple Inc.

Group By with HAVING

- HAVING keyword is used to filter on aggregate values
- It functions similarly to a WHERE clause except aggregates are allowed
- Can only use HAVING when a GROUP BY clause exists

```
SELECT
    Country
    ,COUNT(*) CountCountry
FROM Customer
GROUP BY Country
ORDER BY CountCountry DESC
```

	Country	CountCountry
1	USA	13
2	Canada	8
3	France	5
4	Brazil	5
5	Germany	4
6	United Kingdom	3
7	Portugal	2
8	India	2
9	Czech Republic	2

```
SELECT
    Country
    ,COUNT(*) CountCountry
FROM Customer
GROUP BY Country
HAVING COUNT(*) >= 5
ORDER BY CountCountry DESC
```

	Country	CountCountry
1	USA	13
2	Canada	8
3	France	5
4	Brazil	5

GROUP BY Examples

- Example One returns the total sales by Artist where the sales are greater than 50
- Example Two returns the totals sales of the “Lost” Artist name broken down by Track title

```
SELECT
    AT.Name
    ,SUM(IL.UnitPrice) TotalSales
FROM Artist AT
JOIN Album AL
    ON AT.ArtistId = AL.ArtistId
JOIN Track T
    ON T.AlbumId = AL.AlbumId
JOIN InvoiceLine IL
    ON IL.TrackId = T.TrackId
GROUP BY
    AT.Name
HAVING SUM(IL.UnitPrice) > 50
ORDER BY TotalSales DESC
```

	Name	TotalSales
1	Iron Maiden	138.60
2	U2	105.93
3	Metallica	90.09
4	Led Zeppelin	86.13
5	Lost	81.59

```
SELECT
    AT.Name
    ,AL.Title
    ,SUM(IL.UnitPrice) TotalSales
FROM Artist AT
JOIN Album AL
    ON AT.ArtistId = AL.ArtistId
JOIN Track T
    ON T.AlbumId = AL.AlbumId
JOIN InvoiceLine IL
    ON IL.TrackId = T.TrackId
WHERE AT.Name = 'Lost'
GROUP BY
    AT.Name
    ,AL.Title
ORDER BY TotalSales DESC
```

	Name	Title	TotalSales
1	Lost	Lost, Season 2	25.87
2	Lost	Lost, Season 3	21.89
3	Lost	Lost, Season 1	19.90
4	Lost	LOST, Season 4	13.93

Summary

- Count
- Sum
- Min
- Max
- Avg

- Group By
- Having
- Aggregates and Nulls