SQL for Data Science

SELECTING AND RETRIEVING DATA 3

Retrieving Data with a SELECT Statement 3
Retrieving Multiple Columns 3
Retrieving Multiple Columns Using a Wildcard 3
Limiting Result Using Different Syntaxes 3
Creating Table 3
Creating Your Own Table 3
Adding Data to Table 3
Creating a Temporary Table 4
Adding Comments 4

FILTERING, SORTING AND CALCULATING DATA 5

Basis of Filtering 5
Operator Values 5
WHERE Clause Operators 5
Advanced Filtering: IN, OR, and NOT 5
IN Operator 5
OR Operator 6
NOT Operator 6
Using Wildcards 6
% Wildcards 6
Underscore (__) Wildcard 6
Bracket ([ ]) Wildcard 6
Sorting with ORDER BY 7
Sorting by Column Position 7
Sort Direction 7
Maths Operators 7
Example 7
Order of Operators 7
Combining Math Operators 7
Aggregate Functions 7
AVERAGE Function 7
COUNT Function 8
MAX and MIN Functions 8
SUM Function 8
Using DISTINCT on Aggregate Functions 8
Grouping Data 8
Key SQL Clauses 8

SUBQUERIES AND JOINS 9

Subqueries 9
Subquery Best Practices and Considerations 9
Subquery in a Subquery 9
Subqueries for Calculations 9
Joining Tables 10

Aliases 10
Cartesian (Cross) Joins 10
Inner Joins 10
Self Joins 10
Advanced Joins: Left, Right, and Full Outer Joins 11
Unions 11

MODIFYING AND ANALYSING DATA 12

Text Strings 12
Concatenations 12
Trimming Strings 12
Substring (SUBSTR) 12
UPPER and LOWER 12
Date and Time Strings 13
STRFTIME 13
Compute Current Date 13
Compute Year, Month, Day for the Current Date 13
Compute the Hour, Minute and Second and Milliseconds from Current DATETIME 13
Compute Age Using Birthdate 13
Case Statements 13
Example – Simple Case Statement 14
Example – Search Case Statement 14
Views 15
Create a View 15
Drop a View 15

Selecting and Retrieving Data

Retrieving Data with a SELECT Statement

Retrieving Multiple Columns

Add multiple column names, be sure to use a comma

```
SELECT prod_name, prod_id, prod_price
FROM Products;
```

```
SELECT prod_name,
prod_id,
prod_price
FROM Products;
```

Retrieving Multiple Columns Using a Wildcard

Request all columns by using the asterisk (*) wildcard character instead of column names

```
SELECT *
FROM Products;
```

Limiting Result Using Different Syntaxes

SQLite	*Oracle*	*DB2*

SELECT prod_name
FROM Products
LIMIT 5;
	SELECT prod_name
	FROM Products
	WHERE ROWNUM <=5;
		SELECT prod_name
		FROM Products
		FETCH FIRST 5 ROWS ONLY;

Creating Table

Creating Your Own Table

```
CREATE TABLE Shoes
(
Id              char (10)          PRIMARY KEY,
Brand           char (10)          NOT NULL,
Type            char (250)         NOT NULL,
Color           char (250)         NOT NULL,
Price           decimal (8,2)      NOT NULL,
Desc            Varchar (750)      NULL
);
```

Adding Data to Table

```
INSERT INTO Shoes
(Id,
Brand,
Type,
Color,
Price,
Desc
)
VALUES
('14535974',
'Gucci',
'Slippers',
'Pink',
'695.00',
NULL
);
```

Creating a Temporary Table

```
CREATE TEMPORARY TABLE Sandals AS
(
SELECT *
FROM shoes
WHERE shoe_type = 'sandals'
)
```

Adding Comments

Single Line	*Section*
SELECT shoe_id	SELECT shoe_id
- -,	/*, brand_id
brand_id,	,shoe_name
shoe_name	*/
FROM shoes	FROM shoes

Filtering, Sorting and Calculating data

Basis of Filtering

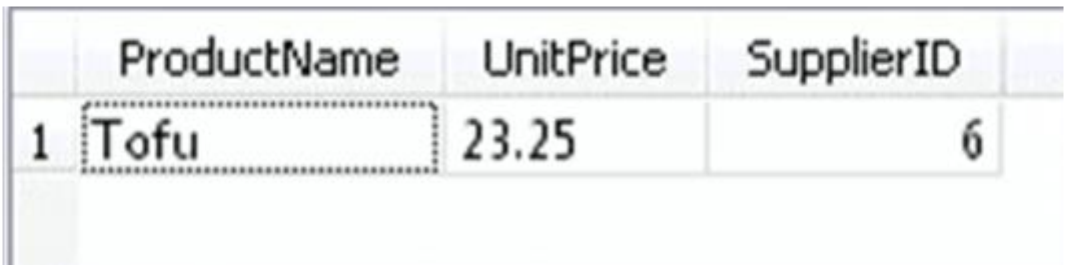
Operator Values

Operator	Description
=	Equal
<>	Not equal. Note: In some versions of SQL this operator may be written as !=
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
IS NULL	is a null value

WHERE Clause Operators

```
SELECT column_name, column_name
FROM table_name
WHERE column_name operator value;
```

Example – Filtering on a Single Condition

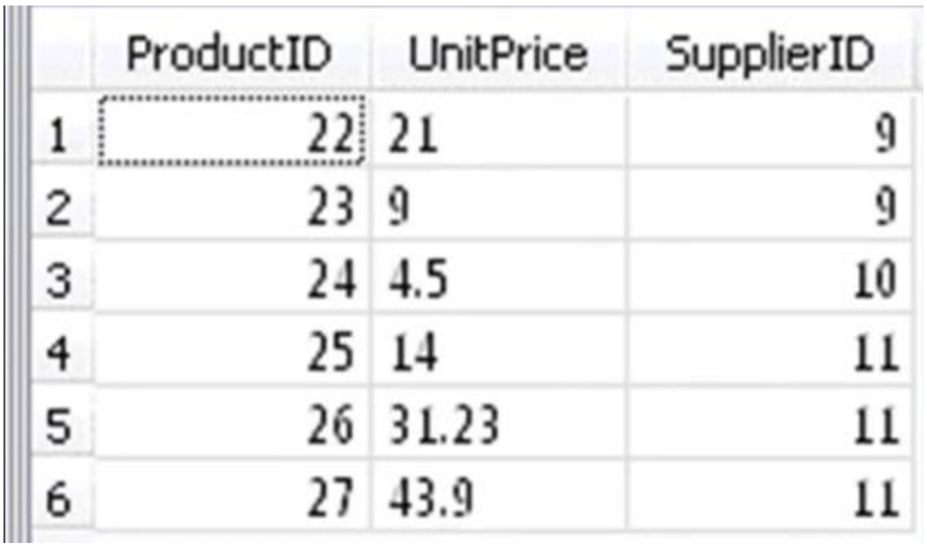


	ProductName	UnitPrice	SupplierID
1	Tofu	23.25	6

```
SELECT ProductName,
UnitPrice,
SupplierID
FROM Products
WHERE ProductName = 'Tofu';
```

Advanced Filtering: IN, OR, and NOT

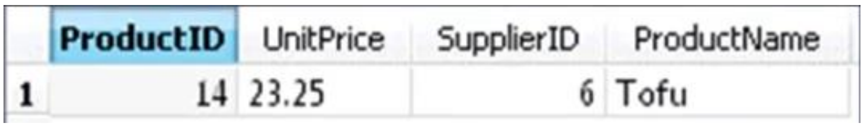
IN Operator



	ProductID	UnitPrice	SupplierID
1	22	21	9
2	23	9	9
3	24	4.5	10
4	25	14	11
5	26	31.23	11
6	27	43.9	11

```
SELECT ProductID,
UnitPrice,
SupplierID
From Products
WHERE SupplierID IN (9, 10, 11);
```

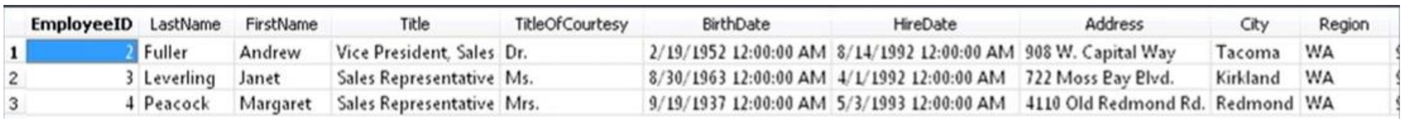
OR Operator



	ProductID	UnitPrice	SupplierID	ProductName
1	14	23.25	6	Tofu

```
SELECT ProductName
,ProductID
,UnitPrice
,SupplierID
,ProductName
From Products
Where ProductName = 'Tofu' OR 'Konbu';
```

NOT Operator

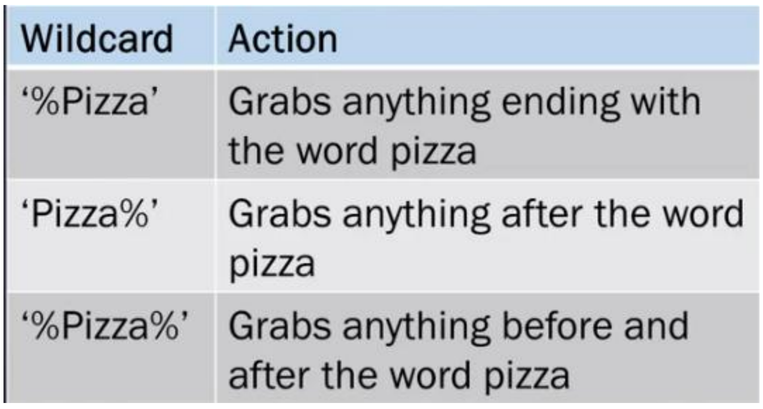


	EmployeeID	LastName	FirstName	Title	TitleOfCourtesy	BirthDate	HireDate	Address	City	Region
1	2	Fuller	Andrew	Vice President, Sales	Dr.	2/19/1952 12:00:00 AM	8/14/1992 12:00:00 AM	908 W. Capital Way	Tacoma	WA
2	3	Leverling	Janet	Sales Representative	Ms.	8/30/1963 12:00:00 AM	4/1/1992 12:00:00 AM	722 Moss Bay Blvd.	Kirkland	WA
3	4	Peacock	Margaret	Sales Representative	Mrs.	9/19/1937 12:00:00 AM	5/3/1993 12:00:00 AM	4110 Old Redmond Rd.	Redmond	WA

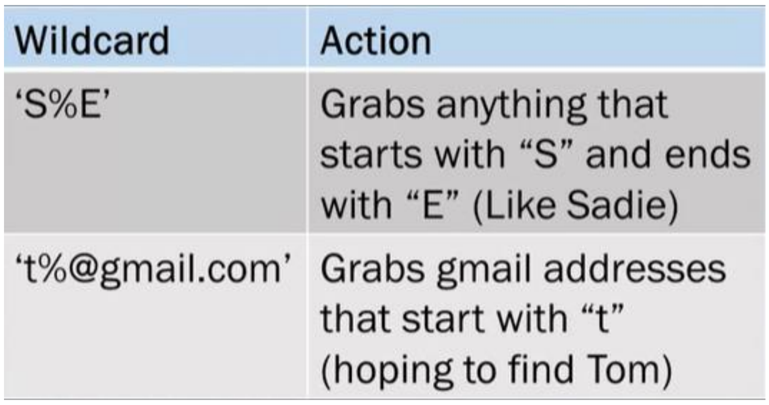
```
SELECT *
FROM Employees
WHERE NOT City='London' AND
NOT City=’Seattle';
```

Using Wildcards

% Wildcards



Wildcard	Action
‘%Pizza’	Grabs anything ending with the word pizza
‘Pizza%’	Grabs anything after the word pizza
‘%Pizza%’	Grabs anything before and after the word pizza



Wildcard	Action
‘S%E’	Grabs anything that starts with “S” and ends with “E” (Like Sadie)
‘t%@gmail.com’	Grabs gmail addresses that start with “t” (hoping to find Tom)

Underscore (__) Wildcard

```
WHERE size LIKE '_pizza'
```

Output:
spizza
mpizza
Is not supported by DB2

Bracket ([ ]) Wildcard

Does not work with all DBMS, SQLite

Sorting with ORDER BY

```
SELECT something
FROM database
ORDER BY characteristic
```

Sorting by Column Position

```
ORDER BY 2,3
```

Sort Direction

Operators	Description
DESC	Descending Order
ASC	Ascending Order

Ascending Order

```
SELECT something
FROM database
ORDER BY characteristic ASC
```

Descending Order

```
SELECT something
FROM database
ORDER BY characteristic DESC
```

Maths Operators

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division

Example

```
SELECT
ProductID
,UnitsOnOrder
,UnitPrice
,UnitsOnOrder * UnitPrice AS Total_Order_Cost
FROM Products
```

Order of Operators

- Parentheses
- Exponents
- Multiplication
- Division
- Addition
- Subtraction

Combining Math Operators

```
SELECT
ProductID
,Quantity
,UnitPrice
,Discount
,(UnitPrice – Discount) * Quantity AS
Total_Cost
FROM OrderDetails;
```

Aggregate Functions

Functions	Description
AVG ()	Averages a column of values
COUNT ()	Counts the number of values
MIN ()	Finds the minimum value
MAX ()	Finds the maximum value
SUM ()	Sums the column values

AVERAGE Function

```
SELECT AVG(UnitPrice) AS avg_price
FROM products
```

COUNT Function

COUNT (*) – Counts all the rows in a table containing values or NULL values

```
SELECT COUNT (*) AS
total_customers
FROM Customers;
```

Count (column) - Counts all the rows in a specific column ignoring NULL values

```
SELECT COUNT(CustomerID) AS
total_customers
FROM Customers
```

MAX and MIN Functions

MAX

```
SELECT MAX(UnitPrice) AS max_prod_price
FROM Products
```

MAX and MIN

```
SELECT MAX(UnitPrice) AS max_prod_price
,MIN (UnitPrice) AS min_prod_price
FROM Products
```

SUM Function

Sum

```
SELECT SUM (UnitPrice) AS
total_prod_price
FROM Products
```

Sum with Conditions (Sum, If)

```
SELECT SUM(UnitPrice*UnitsInStock)
AS total_price
FROM Products
WHERE SupplierID = 23;
```

Using DISTINCT on Aggregate Functions

- If DISTINCT is not specified, ALL is assumed
- Cannot use DISTINCT on COUNT(*)
- No value to use with MIN and MAX functions

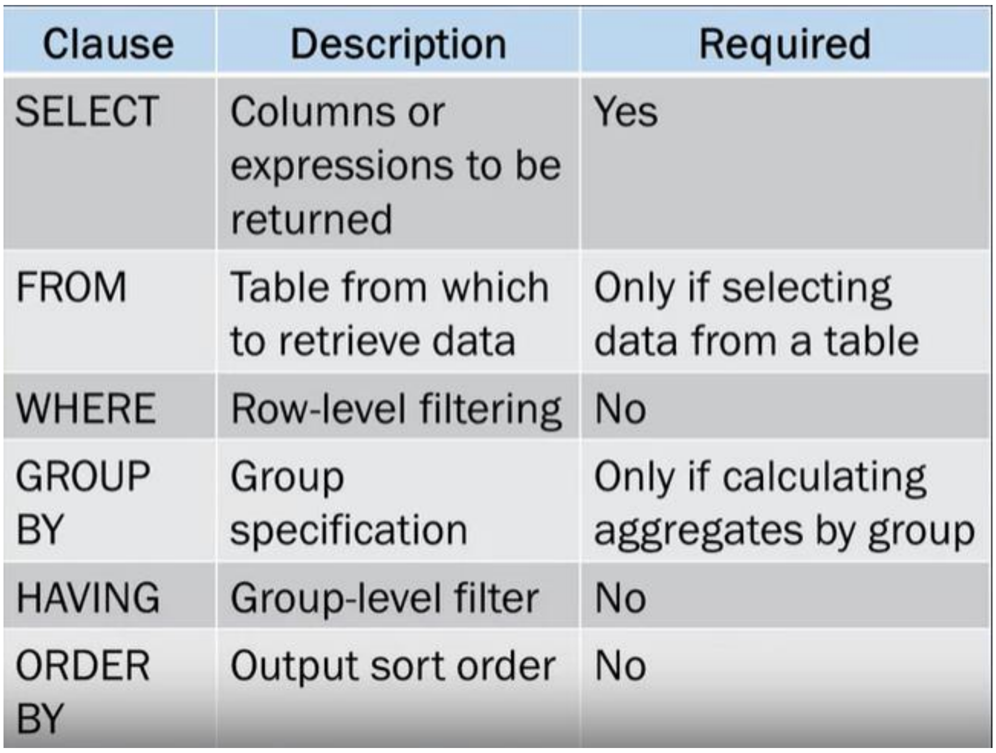
```
SELECT COUNT (DISTINCT CustomerID)
FROM Customers
```

Grouping Data

WHERE filters before the data is grouped, HAVING filters after the data is grouped.

```
SELECT
CustomerID
,COUNT(*) AS orders
FROM Orders
GROUP BY CustomerID
HAVING COUNT (*) >=2;
```

Key SQL Clauses



Clause	Description	Required
SELECT	Columns or expressions to be returned	Yes
FROM	Table from which to retrieve data	Only if selecting data from a table
WHERE	Row-level filtering	No
GROUP BY	Group specification	Only if calculating aggregates by group
HAVING	Group-level filter	No
ORDER BY	Output sort order	No

Subqueries and Joins

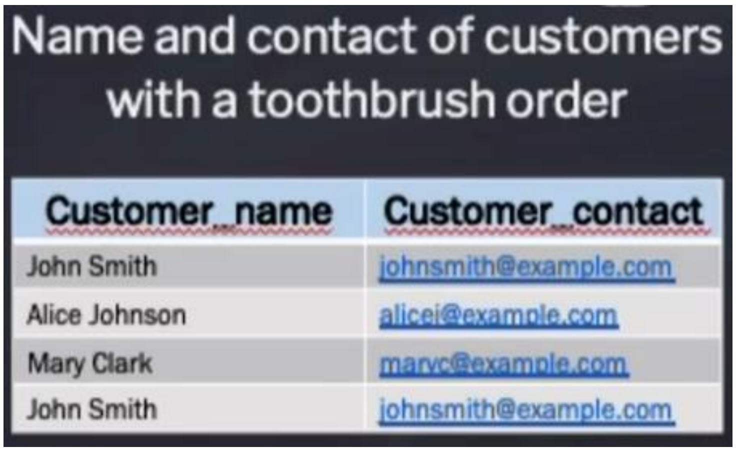
Subqueries

```
SELECT
CustomerID
,CompanyName
,Region
FROM Customers
WHERE customerID IN (SELECT customerID
        FROM Orders
        WHERE Freight > 100);
```

Subquery Best Practices and Considerations

Subquery in a Subquery

```
SELECT Customer_name, Customer_contact
FROM Customers
WHERE cust_id IN
        SELECT customer_id
        FROM Orders
        WHERE order_number IN
                (SELECT order_number
                FROM OrderItems
                WHERE prod.name = 'Toothbrush');
```

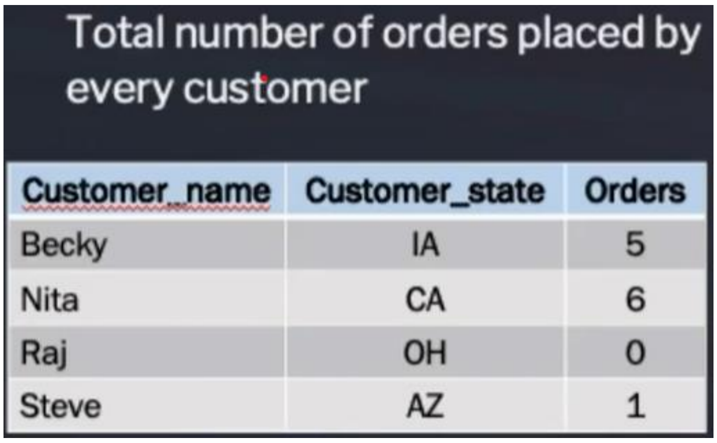


Name and contact of customers with a toothbrush order

Customer_name	Customer_contact
John Smith	johnsmith@example.com
Alice Johnson	alicej@example.com
Mary Clark	maryc@example.com
John Smith	johnsmith@example.com

Subqueries for Calculations

```
SELECT customer_name
,customer_state
        ,(SELECT COUNT(*) AS orders
        FROM Orders
        WHERE orders.customer_id = customer.customer_id) AS orders
FROM customers
ORDER BY customer_name
```



Total number of orders placed by every customer

Customer_name	Customer_state	Orders
Becky	IA	5
Nita	CA	6
Raj	OH	0
Steve	AZ	1

Joining Tables

Aliases

Assigning Names Temporarily

```
SELECT vendor_name
,product_name
,product_price
FROM Vendors AS v, Products AS p
WHERE v.vendor_id = p.vendor_id;
```

Cartesian (Cross) Joins

Multiplying one table with other

```
SELECT product_name
,unit_price
,company_name
FROM suppliers CROSS JOIN products;
```

Inner Joins

Selects records that have matching values in both tables.

```
SELECT suppliers.CompanyName
,ProductName
,UnitPrice
FROM Suppliers INNER JOIN Products
ON Suppliers.supplierid =
Products.supplierid
```

Inner join with multiple tables

```
SELECT o.OrderID, c.CompanyName, e. LastName
FROM ((Orders o INNER JOIN Customers c ON o.CustomerID = c.CustomerID)
INNER JOIN Employees e ON o.EmployeeID = e.EmployeeID);
```

Self Joins

```
SELECT
e1.FirstName || ' ' || e1. LastName
AS EmployeeName,
e2.FirstName || ' ‘ || e2.LastName
AS ManagerName
FROM Employee e1
LEFT JOIN Employee e2 ON e1.ReportsTo
= e2.EmployeeId
ORDER BY EmployeeName;
```

Advanced Joins: Left, Right, and Full Outer Joins

Left	*Right*	*Full Outer Join*
Returns all records from the left table (table1), and the matched records from the right table (table2).	**Returns all records from the right table (table2), and the matched records from the left table (table1).**	**Return all records when there is a match in either left (table1) or right (table2) table records**
SELECT **C.CustomerName, O.OrderID** FROM **Customers C** LEFT JOIN **Orders O** ON **C. CustomerID = O.CustomerID** ORDER BY **C.CustomerName;**	SELECT **Orders.OrderID, Employees.LastName, Employees.FirstName** FROM **Orders** RIGHT JOIN **Employees** ON **Orders.EmployeeID = Employees.EmployeeID** ORDER BY **Orders.Order ID;**	SELECT **Customers.CustomerName, Orders.OrderID** FROM **Customers** FULL OUTER JOIN **Orders ON Customers.CustomerID = Orders.CustomerID** ORDER BY **Customers.CustomerName;**

- *SQLite only supports Left Joints only*
- *We can reverse table orders to use LEFT to RIGHT joins or vice-versa*

Unions

```
SELECT City, Country
FROM Customers
WHERE Country = 'Germany’
UNION
SELECT City, Country
FROM Suppliers
WHERE Country = 'Germany'
ORDER BY City;
```

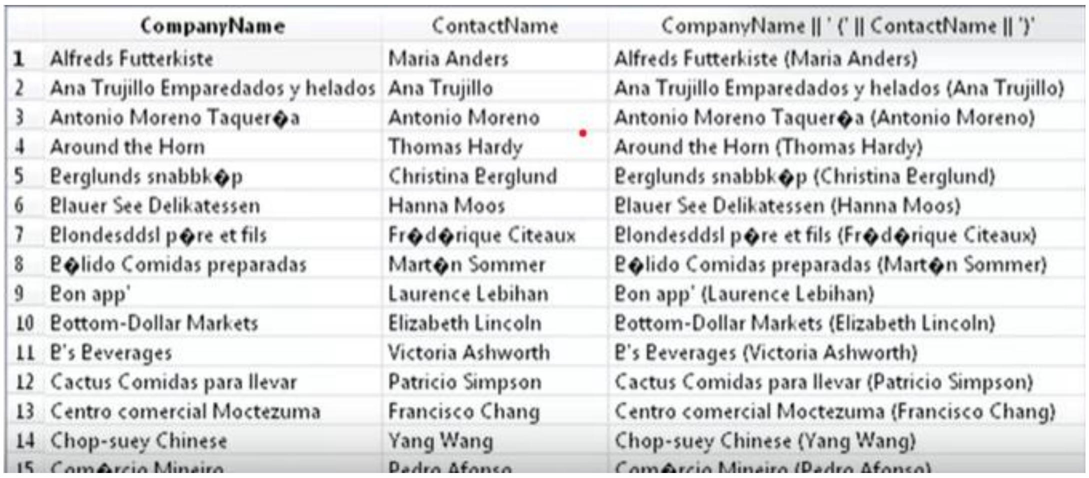
Modifying and Analysing Data

Text Strings

Concatenations

```
SELECT
CompanyName,
ContactName,
CompanyName || ' ('|| ContactName ||’) ‘
FROM customers
```

SQL Server supports + instead of ||



	CompanyName	ContactName	CompanyName || ' (' || ContactName || ')'
1	Alfreds Futterkiste	Maria Anders	Alfreds Futterkiste (Maria Anders)
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Ana Trujillo Emparedados y helados (Ana Trujillo)
3	Antonio Moreno Taquer�a	Antonio Moreno	Antonio Moreno Taquer�a (Antonio Moreno)
4	Around the Horn	Thomas Hardy	Around the Horn (Thomas Hardy)
5	Berglunds snabbk�p	Christina Berglund	Berglunds snabbk�p (Christina Berglund)
6	Blauer See Delikatessen	Hanna Moos	Blauer See Delikatessen (Hanna Moos)
7	Blondesddsl p�re et fils	Fr�d�rique Citeaux	Blondesddsl p�re et fils (Fr�d�rique Citeaux)
8	B�lido Comidas preparadas	Mart�n Sommer	B�lido Comidas preparadas (Mart�n Sommer)
9	Bon app'	Laurence Lebihan	Bon app' (Laurence Lebihan)
10	Bottom-Dollar Markets	Elizabeth Lincoln	Bottom-Dollar Markets (Elizabeth Lincoln)
11	B's Beverages	Victoria Ashworth	B's Beverages (Victoria Ashworth)
12	Cactus Comidas para llevar	Patricio Simpson	Cactus Comidas para llevar (Patricio Simpson)
13	Centro comercial Moctezuma	Francisco Chang	Centro comercial Moctezuma (Francisco Chang)
14	Chop-suey Chinese	Yang Wang	Chop-suey Chinese (Yang Wang)
15	Com�rcio Mineiro	Pedro Afonso	Com�rcio Mineiro (Pedro Afonso)

Trimming Strings

Three functions –

- TRIM
- LTRIM
- RTRIM

```
SELECT TRIM("    You the best.    ")
AS TrimmedString;
```

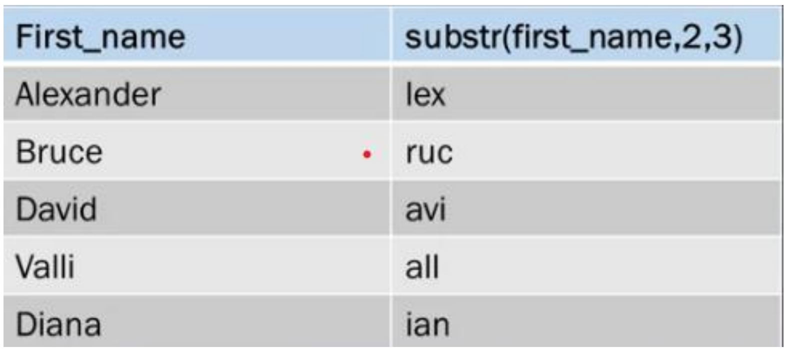
Substring (SUBSTR)

Usage

```
SUBSTR (string name, string position, number of
characters to be returned);
```

Example

```
SELECT first_name,
SUBSTR (first_name,2,3)
FROM employees
WHERE department_id = 60;
```



First_name	substr(first_name,2,3)
Alexander	lex
Bruce	ruc
David	avi
Valli	all
Diana	ian

UPPER and LOWER

UPPER

```
SELECT UPPER (column_name)
FROM table_name;
```

LOWER

```
SELECT LOWER (column_name)
FROM table_name;
```

UCASE

```
SELECT UCASE (column_name)
FROM table_name;
```

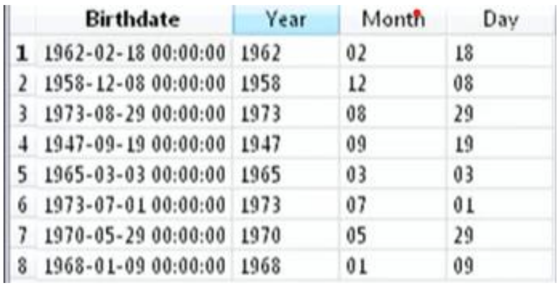
Date and Time Strings

- **DATE (timestring, modifier, modifier, ...)**
- **TIME (timestring, modifier, modifier, ...)**
- **DATETIME (timestring, modifier, modifier, ...)**
- **JULIANDAY (timestring, modifier, modifier, ...)**
- **STRFTIME (format, timestring, modifier, modifier, ...)**

Identify the database management system (DBMS) you are using and then look up the variety of date type/s used in that system.

STRFTIME

```
SELECT Birthdate
,STRFTIME('%Y', Birthdate) AS Year
,STRFTIME('%m', Birthdate) AS Month
,STRFTIME('%d', Birthdate) AS Day
FROM employees
```



	Birthdate	Year	Month	Day
1	1962-02-18 00:00:00	1962	02	18
2	1958-12-08 00:00:00	1958	12	08
3	1973-08-29 00:00:00	1973	08	29
4	1947-09-19 00:00:00	1947	09	19
5	1965-03-03 00:00:00	1965	03	03
6	1973-07-01 00:00:00	1973	07	01
7	1970-05-29 00:00:00	1970	05	29
8	1968-01-09 00:00:00	1968	01	09

Compute Current Date

```
SELECT DATE ('now')
```

Compute Year, Month, Day for the Current Date

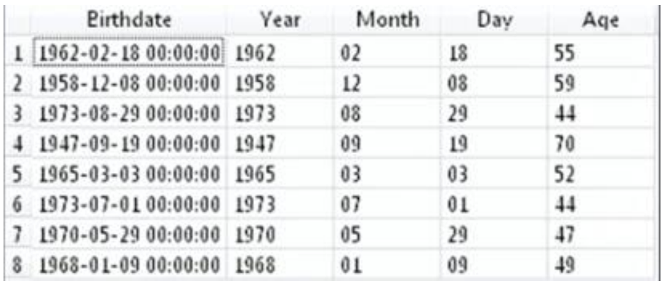
```
SELECT STRFTIME ('%Y %m %d','now')
```

Compute the Hour, Minute and Second and Milliseconds from Current DATETIME

```
SELECT STRFTIME ('%H %M %S %s', 'now');
```

Compute Age Using Birthdate

```
SELECT Birthdate,
STRFTIME ('%Y', Birthdate) AS Year,
STRFTIME ('%m', Birthdate) AS Month,
STRFTIME ('%d', Birthdate) AS Day,
STRFTIME (('now') - Birthdate) AS Age
FROM employees
```



	Birthdate	Year	Month	Day	Age
1	1962-02-18 00:00:00	1962	02	18	55
2	1958-12-08 00:00:00	1958	12	08	59
3	1973-08-29 00:00:00	1973	08	29	44
4	1947-09-19 00:00:00	1947	09	19	70
5	1965-03-03 00:00:00	1965	03	03	52
6	1973-07-01 00:00:00	1973	07	01	44
7	1970-05-29 00:00:00	1970	05	29	47
8	1968-01-09 00:00:00	1968	01	09	49

Case Statements

Simple Case Statement

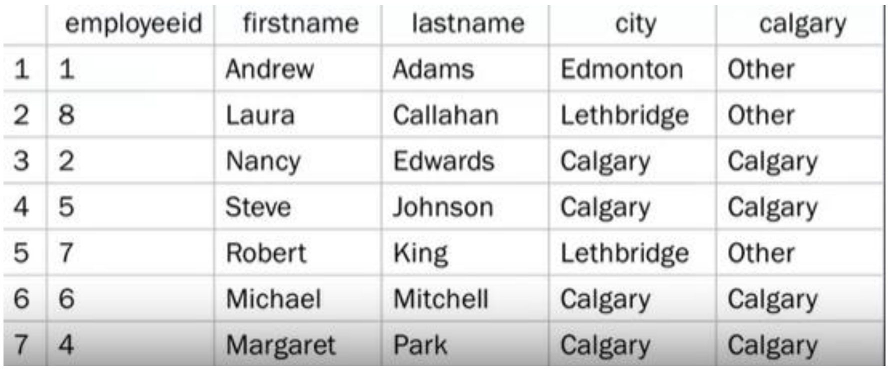
```
CASE
WHEN C1 THEN E1
WHEN C2 THEN E2
ELSE [result else]
END
```

Search Case Statement

```
CASE input_expression
WHEN when_expression THEN result_expression [...n]
[ ELSE else_result_expression ]
END
```

Example – Simple Case Statement

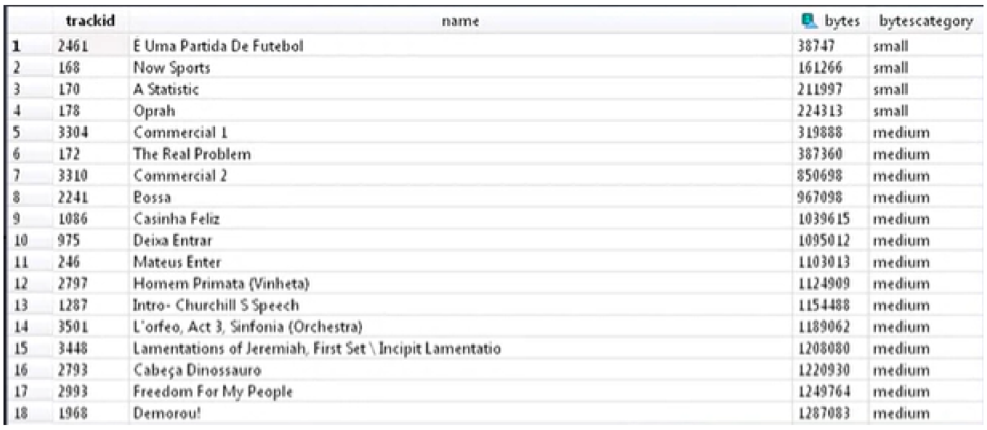
```
SELECT
Employeeid,
firstname,
lastname,
city
,CASE City
            WHEN 'Calgary' THEN 'Calgary'
ELSE 'Other'
            END calgary
FROM Employees
ORDER BY LastName, FirstName;
```



	employeeid	firstname	lastname	city	calgary
1	1	Andrew	Adams	Edmonton	Other
2	8	Laura	Callahan	Lethbridge	Other
3	2	Nancy	Edwards	Calgary	Calgary
4	5	Steve	Johnson	Calgary	Calgary
5	7	Robert	King	Lethbridge	Other
6	6	Michael	Mitchell	Calgary	Calgary
7	4	Margaret	Park	Calgary	Calgary

Example – Search Case Statement

```
SELECT trackid,
name,
bytes
,CASE
WHEN bytes < 300000 THEN 'small'
WHEN bytes >= 300001 AND bytes <= 500000 THEN 'medium'
WHEN bytes >= 500001 THEN 'large'
ELSE 'Other'
END bytescategory
FROM tracks;
```

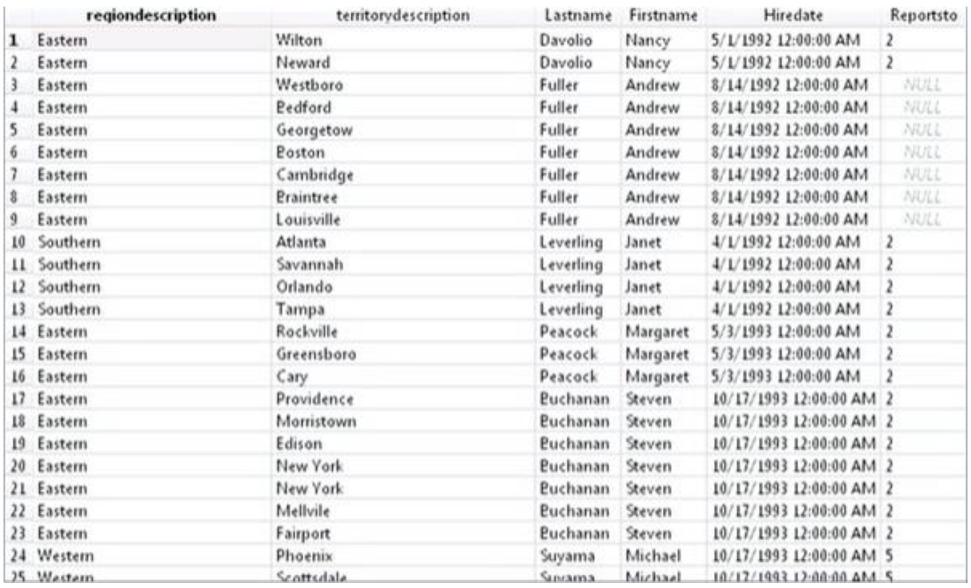


	trackid	name	bytes	bytescategory
1	2461	É Uma Partida De Futebol	38747	small
2	168	Now Sports	161266	small
3	170	A Statistic	211997	small
4	178	Oprah	224313	small
5	3304	Commercial 1	319888	medium
6	172	The Real Problem	387360	medium
7	3310	Commercial 2	850698	medium
8	2241	Bossa	967098	medium
9	1086	Casinha Feliz	1039615	medium
10	975	Deixa Entrar	1095012	medium
11	246	Mateus Enter	1103013	medium
12	2797	Homem Primata (Vinheta)	1124909	medium
13	1287	Intro- Churchill S Speech	1154488	medium
14	3501	L'orfeo, Act 3, Sinfonia (Orchestra)	1189062	medium
15	3448	Lamentations of Jeremiah, First Set \ Incipit Lamentatio	1208080	medium
16	2793	Cabeça Dinossauro	1220930	medium
17	2993	Freedom For My People	1249764	medium
18	1968	Demorou!	1287083	medium

Views

Create a View

```
CREATE VIEW my_view
AS
SELECT
r.regiondescription,
t.territorydescription,
e.Lastname,
e.Firstname,
e.Hiredate,
e.Reportsto
FROM Region r
INNER JOIN Territories t ON r.regionid = t.regionid
INNER JOIN Employeeterritories et ON t.TerritoryID = et.Territory ID
INNER JOIN Employees e ON et.employeeid = e.EmployeeID
```



	regiondescription	territorydescription	Lastname	Firstname	Hiredate	Reportsto
1	Eastern	Wilton	Davolio	Nancy	5/1/1992 12:00:00 AM	2
2	Eastern	Neward	Davolio	Nancy	5/1/1992 12:00:00 AM	2
3	Eastern	Westboro	Fuller	Andrew	8/14/1992 12:00:00 AM	NULL
4	Eastern	Bedford	Fuller	Andrew	8/14/1992 12:00:00 AM	NULL
5	Eastern	Georgetow	Fuller	Andrew	8/14/1992 12:00:00 AM	NULL
6	Eastern	Boston	Fuller	Andrew	8/14/1992 12:00:00 AM	NULL
7	Eastern	Cambridge	Fuller	Andrew	8/14/1992 12:00:00 AM	NULL
8	Eastern	Braintree	Fuller	Andrew	8/14/1992 12:00:00 AM	NULL
9	Eastern	Louisville	Fuller	Andrew	8/14/1992 12:00:00 AM	NULL
10	Southern	Atlanta	Leverling	Janet	4/1/1992 12:00:00 AM	2
11	Southern	Savannah	Leverling	Janet	4/1/1992 12:00:00 AM	2
12	Southern	Orlando	Leverling	Janet	4/1/1992 12:00:00 AM	2
13	Southern	Tampa	Leverling	Janet	4/1/1992 12:00:00 AM	2
14	Eastern	Rockville	Peacock	Margaret	5/3/1993 12:00:00 AM	2
15	Eastern	Greensboro	Peacock	Margaret	5/3/1993 12:00:00 AM	2
16	Eastern	Cary	Peacock	Margaret	5/3/1993 12:00:00 AM	2
17	Eastern	Providence	Buchanan	Steven	10/17/1993 12:00:00 AM	2
18	Eastern	Morristown	Buchanan	Steven	10/17/1993 12:00:00 AM	2
19	Eastern	Edison	Buchanan	Steven	10/17/1993 12:00:00 AM	2
20	Eastern	New York	Buchanan	Steven	10/17/1993 12:00:00 AM	2
21	Eastern	New York	Buchanan	Steven	10/17/1993 12:00:00 AM	2
22	Eastern	Mellvile	Buchanan	Steven	10/17/1993 12:00:00 AM	2
23	Eastern	Fairport	Buchanan	Steven	10/17/1993 12:00:00 AM	2
24	Western	Phoenix	Suyama	Michael	10/17/1993 12:00:00 AM	5
25	Western	Scottsdale	Suyama	Michael	10/17/1993 12:00:00 AM	5

Drop a View

```
SELECT *
FROM my_view
DROP VIEW my_view;
```