learnit

Note :

All course guides are in PDF format and you must have an appropriate PDF reader installed on your computer to open and print these course guides. If you do not already have one, you can find and download a free copy of Acrobat Reader at: *https://get.adobe.com/reader/*

Course Reference Handout

SQL QUERY - FUNDAMENTALS

Contact us at Learn iT!
(415) 693-0250
www.learnit.com

SQL QUERY

learnit

What is SQL?

- SQL stands for Structured Query Language.
- SQL lets you access and manipulate databases.
- SQL is an AMSI (American National Standards Institute) standard.

What Can SQL Do?

- Execute queries from a database.
- Retrieve data from a database.
- Insert records into a database.
- Update records in a database.
- Delete records in a database.
- Create new databases.
- Create tables in a database.
- Create stored procedures (queries) in a database.
- Create views to control how users see query results.
- Set permissions for database access.

Common SQL High-Level Action Commands:

SELECT:
Extracts Data from a Database.

UPDATE:
Updates Data in a Database.

DELETE:
Deletes Data from a Database.

INSERT INTO:
Inserts new Data into a Database.

CREATE DATABASE:
Creates a new Database.

ALTER DATABASE:
Modifies a Database.

CREATE TABLE:
Creates a new Table.

ALTER TABLE:
Modifies a Table.

DROP TABLE:
Deletes a Table.

CREATE INDEX:
Creates an Index (search key).

DROP INDEX:
Deletes an Index.

Hierarchy of Basic Select Clause:

Select	(column1, column 2, etc.)
From	(table name)
Where	(used for criteria/conditional statement)
Group By	(groups data)
Having	(used for criteria for grouping)
Order By	(sets the sort order of the result)

Syntax and Example of a Select Statement:

SQL SELECT Syntax

```
SELECT column_name,column_name
FROM table_name;
```

and

```
SELECT * FROM table_name;
```

Example

```
SELECT CustomerName,City FROM Customers;
```

Applying Criteria with the Where Clause:

The SQL WHERE Clause

The WHERE clause is used to extract only those records that fulfill a specified criterion.

SQL WHERE Syntax

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

Example

```
SELECT * FROM Customers
WHERE Country='Mexico';
```

Text Fields vs. Numeric Fields

SQL requires single quotes around text values (most database systems will also allow double quotes).

However, numeric fields should not be enclosed in quotes:

Example

```
SELECT * FROM Customers
WHERE CustomerID=1;
```

Operators in SQL Where Clause:

The following operators can be used in the WHERE clause:

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
LIKE	Search for a pattern
IN	To specify multiple possible values for a column

Using And/Or Conditions in SQL Where Clause:

Example

```
SELECT * FROM Customers
WHERE Country='Germany'
AND City='Berlin';
```

Example

```
SELECT * FROM Customers
WHERE City='Berlin'
OR City='München';
```

Example

```
SELECT * FROM Customers
WHERE Country='Germany'
AND (City='Berlin' OR City='München');
```

Using the In Operator in SQL Query Criteria (works like OR Operator):

Example

```
SELECT * FROM Customers
WHERE City IN ('Paris','London');
```

Applying Wildcard Operators in Criteria:

In SQL, wildcard characters are used with the SQL LIKE operator.

SQL wildcards are used to search for data within a table.

With SQL, the wildcards are:

Wildcard	Description
%	A substitute for zero or more characters
_	A substitute for a single character
[*charlist*]	Sets and ranges of characters to match
[^*charlist*] or [!*charlist*]	Matches only a character NOT specified within the brackets

Examples of Wildcard Criteria:

Using the SQL _ Wildcard

The following SQL statement selects all customers with a City starting with any character, followed by "erlin":

Example

```
SELECT * FROM Customers
WHERE City LIKE '_erlin';
```

The following SQL statement selects all customers with a City starting with "a", "b", or "c":

Example

```
SELECT * FROM Customers
WHERE City LIKE '[a-c]%';
```

The following SQL statement selects all customers with a City NOT starting with "b", "s", or "p":

Example

```
SELECT * FROM Customers
WHERE City LIKE '[!bsp]%';

or

SELECT * FROM Customers
WHERE City NOT LIKE '[bsp]%';
```

Using Between Operator in SQL Criteria:

Example

```
SELECT * FROM Products
WHERE Price BETWEEN 10 AND 20;
```

Example

```
SELECT * FROM Products
WHERE Price NOT BETWEEN 10 AND 20;
```

Example

```
SELECT * FROM Products
WHERE (Price BETWEEN 10 AND 20)
AND NOT CategoryID IN (1,2,3);
```

Formatting Query Results:

```
SELECT bktitle
  , FORMAT(devcost, 'C', 'en-us') AS 'DevCost Currency Format'
  , FORMAT(slprice, 'C', 'en-us') AS 'Sales Price Currency Format'
  , FORMAT( pubdate, 'd', 'en-US' ) AS 'DateTime Result'
FROM Titles
```

Results without formatting:

Results | Messages

	partnum	bktitle	devcost	slprice	pubdate
1	39843	Clear Cupboards	15055.50	49.95	2012-08-19 00:00:00
2	39905	Developing Mobile Apps	19990.00	45.00	2013-01-01 00:00:00
3	40121	Boating Safety	15421.81	36.50	2013-05-18 00:00:00
4	40122	Sailing	9932.96	29.15	2013-05-03 00:00:00
5	40123	The Sport of Windsurfing	12798.32	38.50	2012-07-13 00:00:00
6	40124	The Sport of Hang Gliding	15421.81	49.68	2013-01-06 00:00:00
7	40125	The Complete Football Reference	15032.41	49.99	2012-08-03 00:00:00
8	40231	How to Play Piano (Beginner)	9917.75	25.00	2012-06-11 00:00:00
9	40232	How to Play Piano (Intermediate)	8565.35	20.50	2012-10-22 00:00:00
10	40233	How to Play Piano (Advanced)	7971.02	20.50	2012-12-01 00:00:00
11	40234	How to Play Piano (Professional)	9901.42	25.00	2007-11-13 00:00:00
12	40251	How to Play Guitar (Beginner)	9727.80	25.00	2012-09-14 00:00:00

Results with formatting:

Results | Messages

	bktitle	DevCost Currency Format	Sales Price Currency Format	DateTime Result
1	Clear Cupboards	$15,055.50	$49.95	8/19/2012
2	Developing Mobile Apps	$19,990.00	$45.00	1/1/2013
3	Boating Safety	$15,421.81	$36.50	5/18/2013
4	Sailing	$9,932.96	$29.15	5/3/2013
5	The Sport of Windsurfing	$12,798.32	$38.50	7/13/2012
6	The Sport of Hang Gliding	$15,421.81	$49.68	1/6/2013
7	The Complete Football Reference	$15,032.41	$49.99	8/3/2012
8	How to Play Piano (Beginner)	$9,917.75	$25.00	6/11/2012
9	How to Play Piano (Intermediate)	$8,565.35	$20.50	10/22/2012
10	How to Play Piano (Advanced)	$7,971.02	$20.50	12/1/2012
11	How to Play Piano (Professional)	$9,901.42	$25.00	11/13/2007
12	How to Play Guitar (Beginner)	$9,727.80	$25.00	9/14/2012

SQL Table Joins:

A SQL Join clause is used to combine rows from two of more tables, based on a common field between them.

The most common type of join is the SQL Inner Join (simple join).

A SQL Inner Join returns all rows from multiple tables where the join condition (common field value) is met.

Different SQL Table Join Types:

INNER JOIN:
Returns all rows when there is at least one match in BOTH Tables.

LEFT JOIN:
Returns all rows from the left table, and the matched rows from the right table.

RIGHT JOIN:
Returns all rows from the right table, and the matched rows from the left table.

FULL JOIN:
Returns all rows when there is a match in One of the Tables.

Example of an Inner Join in a query using the Customers and the Orders tables:

Example

```
SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate
FROM Orders
INNER JOIN Customers
ON Orders.CustomerID=Customers.CustomerID;
```

Join 3 or More Tables:

The following SQL statement selects all orders with customer and shipper information:

Example

```
SELECT Orders.OrderID, Customers.CustomerName, Shippers.ShipperName
FROM ((Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID)
INNER JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID);
```

SQL Outer Joins:

SQL LEFT JOIN Keyword

The LEFT JOIN keyword returns all rows from the left table (table1), with the matching rows in the right table (table2). The result is NULL in the right side when there is no match.

SQL LEFT JOIN Syntax

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name=table2.column_name;
```

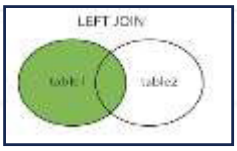
or:

```
SELECT column_name(s)
FROM table1
LEFT OUTER JOIN table2
ON table1.column_name=table2.column_name;
```

PS! In some databases LEFT JOIN is called LEFT OUTER JOIN.

Example

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
LEFT JOIN Orders
ON Customers.CustomerID=Orders.CustomerID
ORDER BY Customers.CustomerName;
```



SQL RIGHT JOIN Keyword

The RIGHT JOIN keyword returns all rows from the right table (table2), with the matching rows in the left table (table1). The result is NULL in the left side when there is no match.

SQL RIGHT JOIN Syntax

```
SELECT column_name(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name=table2.column_name;
```

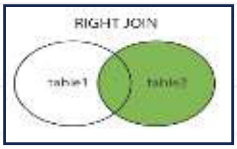
or:

```
SELECT column_name(s)
FROM table1
RIGHT OUTER JOIN table2
ON table1.column_name=table2.column_name;
```

PS! In some databases RIGHT JOIN is called RIGHT OUTER JOIN.

Example

```
SELECT Orders.OrderID, Employees.FirstName
FROM Orders
RIGHT JOIN Employees
ON Orders.EmployeeID=Employees.EmployeeID
ORDER BY Orders.OrderID;
```



Using SQL Functions to Summarize Data:

Aggregate functions like Sum, Average, and Count can be used to calculate and summarize query results.

```
-- Sums total quantity field in the Order Details table

select sum(Quantity) as 'Total Units Sold',
AVG(quantity) as 'Average Units Per Order',
COUNT(quantity) as 'Total Number of Orders'
from [Order Details]
```

100 %

Results | Messages

	Total Units Sold	Average Units Per Order	Total Number of Orders
1	51317	23	2155

SQL Date Functions:

Year Function:

SQLQuery16.sql - loc...ona990\student (52))*

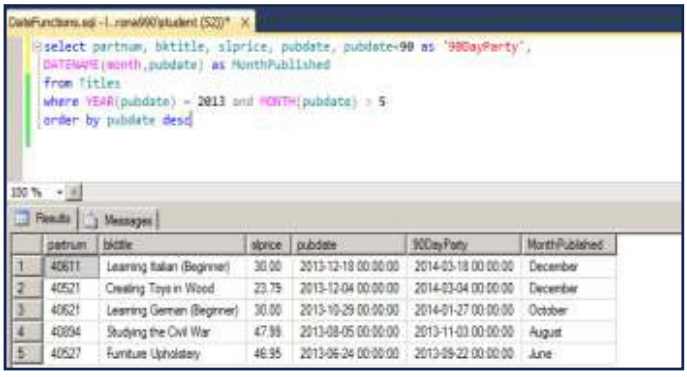
```
select partnum, bktitle, pubdate
from Titles
where YEAR(pubdate) = 2013 and MONTH(pubdate) > 5
order by pubdate desc
```

100 %

Results | Messages

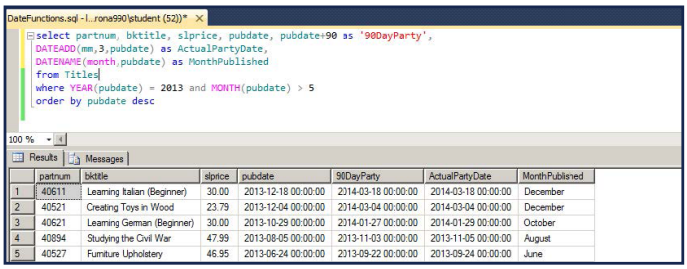
	partnum	bktitle	pubdate
1	40611	Learning Italian (Beginner)	2013-12-18 00:00:00
2	40521	Creating Toys in Wood	2013-12-04 00:00:00
3	40621	Learning German (Beginner)	2013-10-29 00:00:00
4	40894	Studying the Civil War	2013-08-05 00:00:00
5	40527	Furniture Upholstery	2013-06-24 00:00:00

DateName Function:



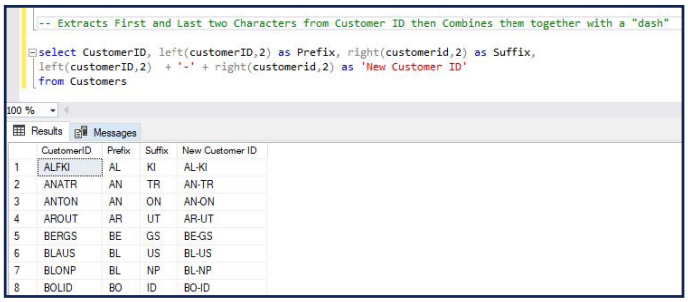
DateAdd Function:

Returns a future date based on number of days, months, or years added to another date.



SQL Text Functions:

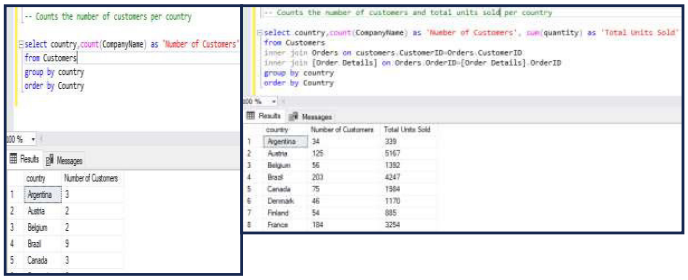
Examples of Left, Right, and Combine Functions:



Sort Records Using Order By Clause:

Organizing Records with Group By Clause:

Grouping records in SQL allows you to organize and summarize records from a table.



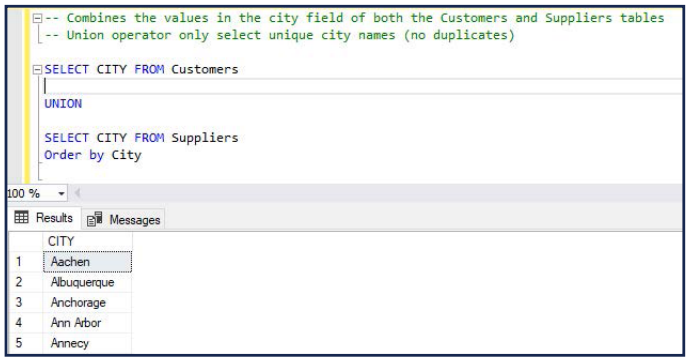
Combine the Results of Two Queries:

The Union operator is used to combine the result-set of two or more SELECT statements.

Each SELECT statement within the Union must have the same number of columns. They must also have similar data types.

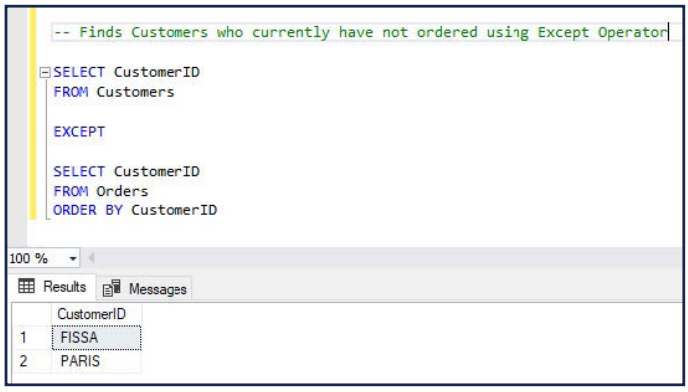
The columns in each SELECT statement must be in the same order.

The UNION operator selects only distinct values by default. To allow duplicate value, use the ALL keyword with UNION.



Compare the Results of Two Queries:

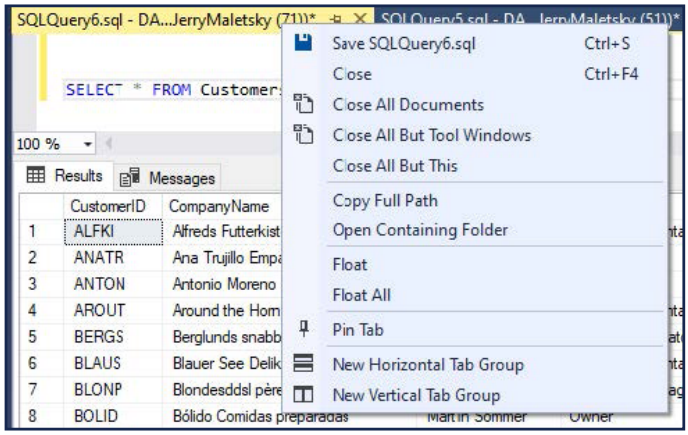
The Except operator looks for records that appear in one table that do not appear in the other.



Combine the Results of Two Queries:

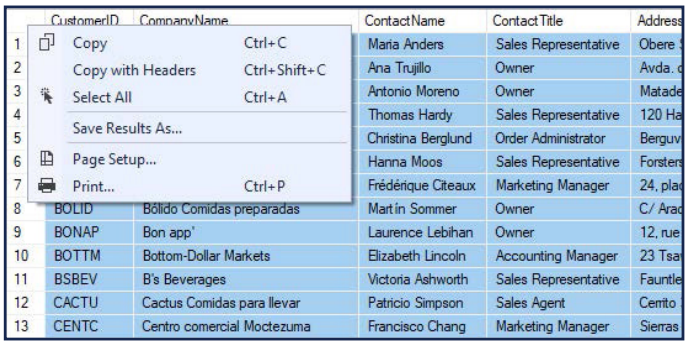
Save SQL Queries:

You can save SQL queries that you want to use in the future. The query is saved as a ".sql" file and when opened launches SQL Server. The query can also be opened using the File menu in SQL server.



Export to Excel:

Query results can be exported to Excel in several ways. This method may be different depending on your version of SQL server. Copy Records With Headers and Paste into Microsoft Excel.



Export as a Text File:

You can export to Text or CSV file by using the Save Results As... (seen in above image).