PCWorkshops

SQL Beginners

SQL 1-Day Beginners Course
An anthology of practical, illustrative SQL Query examples

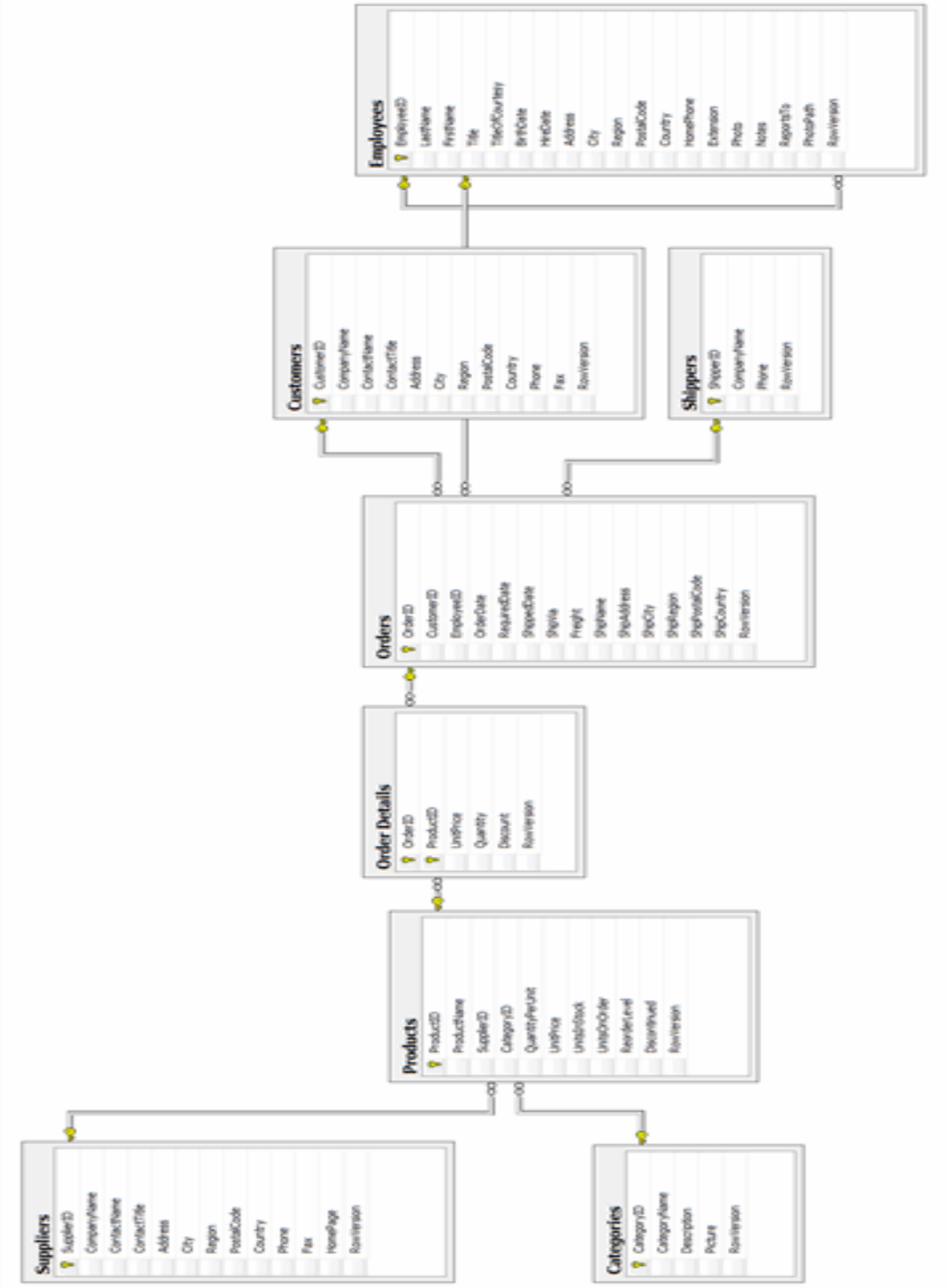
By: Sarah Barnard

SQL Beginners 1-Day Course

Thank you to my assistant, Dr Mary Smith., for assisting with formatting and editing this document.

SQL 1-Day Beginners Course

Table of Content


Employees
EmployeeID
LastName
FirstName
Title
TitleOfCourtesy
BirthDate
HireDate
Address
City
Region
PostalCode
Country
HomePhone
Extension
Photo
Notes
ReportsTo
PhotoPath
RowVersion
Customers
CustomerID
CompanyName
ContactName
ContactTitle
Address
City
Region
PostalCode
Country
Phone
Fax
RowVersion
Shippers
ShipperID
CompanyName
Phone
RowVersion
Orders
OrderID
CustomerID
EmployeeID
OrderDate
RequiredDate
ShippedDate
Shipvia
Freight
ShipName
ShipAddress
ShipCity
ShipRegion
ShipPostalCode
ShipCountry
RowVersion
Order Details
OrderID
ProductID
UnitPrice
Quantity
Discount
RowVersion
Products
ProductID
ProductName
SupplierID
CategoryID
QuantityPerUnit
UnitPrice
UnitsInStock
UnitsOnOrder
ReorderLevel
Discontinued
RowVersion
Suppliers
SupplierID
CompanyName
ContactName
ContactTitle
Address
City
Region
PostalCode
Country
Phone
Fax
HomePage
RowVersion
Categories
CategoryID
CategoryName
Description
Picture
RowVersion

«

Course Outline

SQL Language Essentials	
The Select Statement, the basics of a SQL query: select, from, where.	
How to refer to fields	
How to use aliases	
The Select and From Clauses	
In	
Between	
Like	
And/or	
Top	
Distinct	
Coalesce	
The Order By Clause ordering output	
Arithmetic Operations	
Column Aliases	
Summarizing and Grouping Data	
Aggregate Functions , how to perform aggregations: count(*), sum, max, min, group by	
The Group By Clause	
The Having Clause	
case statements, incl. in combination with aggregations	
Querying Multiple Tables	
Joining Tables	
Inner Joins	
Left Joins	
Right Joins	
Full Joins	
Additional SQL Features	
Combining Queries - views	
Basic Sub Queries	
How to create and drop tables	

Simple Queries

Select

```
SELECT *
```

```
SELECT *
FROM table_name
```

Browse All the Tables

Show all the columns of the Customers Table-

How many records are in this table? _____

Name the primary Key_____

Show all the columns of the Categories Table

How many records are in this table? _____

Name the primary Key_____

Show all the columns of the Employees Table

How many records are in this table? _____

Name the primary Key_____

Show all the columns of the OrderDetails Table

How many records are in this table? _____

Name the primary Key_____

Show all the columns of the Orders Table

How many records are in this table? _____

Name the primary Key_____

Show all the columns of the Products Table

How many records are in this table? _____

Name the primary Key_____

Show all the columns of the Shippers Table

How many records are in this table? _____

Name the primary Key_____

Show all the columns of the Suppliers Table

How many records are in this table? _____

Name the primary Key_____

Select, order By

SELECT *	SELECT * FROM table_name ORDER BY column_name [ASC|DESC]

Products Table

- Show all the columns of the Products Table, sorted by ProductID
- Show all the columns of the Products Table, sorted by ProductName
- Show all the columns of the Products Table, sorted by SupplierID, ProductName
- Show all the columns of the Products Table, sorted by CategoryID, SupplierID, ProductName
- Show all the columns of the Products Table, sorted by CategoryID , ProductID
- Show all the columns of the Products Table, sorted by CategoryID in Descending Order , ProductID

Select, order By

SELECT	**SELECT column_name(s)** **FROM table_name** ORDER BY column_name [ASC|DESC]

- Show SupplierID, Productname, ProductID, Price of the Products Table, sorted by SupplierID, ProductName
- Show CategoryID, SupplierID, ProductNamePrice of the Products Table, sorted by CategoryID, ProductName
- Show CategoryID, SupplierID, Productname, ProductID, Price of the Products Table, sorted by CategoryID, SupplierID, ProductName
- Show SupplierID, Productname, ProductID, Price of the Products Table, sorted by SupplierID, Price

Select where Queries

```
WHERE       SELECT column_name(s)
            FROM table_name
            WHERE column_name operator value
            ORDER BY column_name [ASC|DESC]
```

Operators:

=
>=
<=
>
<
<> (Not equals)

Suppliers Table

- Show all the columns , for records where the SupplierID is not equals to 5, sorted by SupplierID
- Show all the columns , for records where the SupplierID = 5 , sorted by SupplierID
- Show all the columns , for records where the SupplierID> 5 , sorted by SupplierID
- Show all the columns , for records where the SupplierID<5, sorted by SupplierID
- Show all the columns , for records where the SupplierID<=5, sorted by SupplierID
- Show all the columns , for records where the SupplierID is not equals to 20, sorted by SupplierID
- Show all the columns , for records where the SupplierID>= 10, sorted by SupplierID

Customer Table

Available columns :CustomerID,Customername , ContactName, Addres,City, PostalCode,Country

Select where Queries (operator =)

- Show all the columns , for records where the country is France
- Show all the columns , for records where the country is Germany

Select and Oder by

- Show all the columns , for records where the country is UK , sorted by Customername
- Show all the columns , for records where the country is Finland, sorted by Country, City
- Show all the columns , for records where the country is USA, sorted by Country, City , Customername

Wildcards / Like

(Comparing to string values)

LIKE

```
SELECT column_name(s)
FROM table_name
WHERE column_name LIKE '%land%'
ORDER BY column_name [ASC|DESC]

Patterns:
starts with: 'a%'
Ends with '%land'
Contains: '%land%'

Example:
Select Customername  from customers where customername  like 'a%'
```

Customer Table

Wildcards

- Show all the columns , for records where the Customername start with a
- Show all the columns , for records where the Customername start with c
- Show all the columns , for records where the Customername start with z

- Show only the columns City, Country, Customername , ContactName, address
    - for records where the Customername contains Comidas
    - For records where the ADDRESS starts with str
    - For records where the ADDRESS contains str
    - For records where the ADDRESS contains 'street'

    - for records where the Customername ends with iste
    - for records where the Customername ends with essen
    - for records where the Customername contains essen
    - for records where the Customername contains grocer

    - for records where the Customername start with a,
        - order by Country, Customername
    - for records where the Customername start with m,
        - order by Country, Customername

In

IN	SELECT column_name(s) FROM table_name WHERE column_name IN (value1,value2,..) ORDER BY column_name(s)

Orders table

- Show OrderId and Orderdate for shipperID in 1,2,3

Products table

- Show all the columns where the SupplierID is 1,5 or 7
- Show all the columns where the Productname is Chais , Chang or Ikura

Between

BETWEEN	SELECT column_name(s) FROM table_name WHERE column_name BETWEEN value1 AND value2 ORDER BY column_name(s)

Decide first: Which table am I working with?

- Show only the columns City, Country, Customername , ContactName
    - for records where the CustomerID is between 10 and 15
    - for records where the Customername is between a and b
    - for records where the Customername is between a and c
    - for records where the country is between a and m

And /or

AND / OR	SELECT column_name(s) FROM table_name WHERE column operator value AND|OR column operator value ORDER BY column_name(s)
example	SELECT * FROM Products WHERE price > 10 AND categoryid = 1 ORDER BY price desc
example	SELECT * FROM Products WHERE price > 10 or categoryid = 1 ORDER BY price desc

Orders Table

- Show records where the OrderID <10250 or OrderID >10440
- Show records where the CustomerId in 1,5, or 8(using OR)
- Show records where the CustomerId in 1,5, or 8(using IN)

SuppliersTable

- Show records for suppliers in the USA, Germany OR Uk (using IN)
- Show records for suppliers in the USA, Germany OR Uk (using OR)
- Show records for suppliers in the usa OR the city is elgin

Customers Table

- Show records for customers in the usa OR the city is elgin
- Show records for customers in the usa and the city is elgin or the country is france and the city is paris

Suppliers Table

- Show records for suppliers in the usa and the city is boston or the country is france and the city is paris

Suppliers Table

Show records for suppliers in the UK and the suppliername contains ltd or the contactname contains pete

Top

```
SELECT TOP          SELECT
                    TOP number|percent
                    column_name(s)
                    FROM table_name
                    WHERE column operator value
                    AND|OR column operator value
                    ORDER BY column_name(s)
```

Top

Orderdetails Table

- Show the top 10 rows, (all columns) row ordered by ORDERID
- Show the top 10 PERCENT of rows, (all columns) row ordered by ORDERID

Orders Table

- Show the first 5 orders based on the Orderdate
- Show the last 5 orders based on the Orderdate

Products table

- Show the 5 most expensive products
- Show the 5 least expensive

in mySQL

```
SELECT limit      SELECT column_name(s)
FROM table_name
ORDER BY column_name(s)
Limit 10;
```

```
Limit by percent:
SELECT column_name(s) FROM table ORDER BY column_name LIMIT COUNT(*) / 10
```

ORACLE

```
    SELECT          SELECT column_name(s)
    ROWNUM          FROM table_name
                    WHERE  ROWNUM < 11;
```

Distinct

SELECT DISTINCT	SELECT DISTINCT column_name(s) FROM table_name WHERE column operator value AND|OR column operator value ORDER BY column_name(s)

- Show country in the Customer Table order by country
  - – Number of rows: _____
- Show distinct country in the Customer Table order by country
  - – Number of rows: _____

- Show country, city in the Customer Table order by country
  - – Number of rows: _____
- Show distinct country, city in the Customer Table order by country
  - – Number of rows: _____

- Show customername, country, city in the Customer Table order by country
  - – Number of rows: _____
- Show distinct customername, country, city in the Customer Table order by country
  - – Number of rows: _____

- Show all the countries in the Supplier table

- Show the distinct countries in the Supplier table

IS Null and IS NOT NULL

SQL IS NULL

```
SELECT column(s) FROM Table
WHERE column IS NULL
```

Example

```
Select *
From Products
Where price is null
```

SQL IS NOT NULL

```
SELECT column(s) FROM Table
WHERE column IS NOT NULL
```

Example

```
Select *
From Products
Where price is not null
```

Exercise:

Show allthe products on the Products Table where the price is null
Show all the columns on the Customers Table where the address is null

Expression Queries

Use a query to perform arithmetic calculations on columns, and display the result when the query is run without adding new columns to the table

```
SELECT column_name(s), Expression AS alias_name
FROM table_name

Example : Training Database;
SELECT ProductID, ProductName, Price, Price * 1.2 as Amount
from PRODUCTS

Example Northwind database:
SELECT ProductID, ProductName, Price,  unitprice * 1.2 as Amount
from PRODUCTS
```

For each question, First decide which table (s) to use

- Show the ProductID, Product Name, Price, ***Vat*** on the Price from the products table
- Show the ProductID, Product Name, ***Price + Vat*** from the products table
- Show the ProductID, Product Name, Price, ***price plus 10% of the price*** from the products table

CASE

```
Syntax

Select column(s),
CASE
WHEN column operator value THEN value
WHEN column operator value THEN value
        ELSE value
END as Alias
From Table
```

```
Select shipperID (or shipvia) , phone, shippername  ,
CASE    WHEN shippername  IN ('mr shipper','mr delivery','deliveroo')
                THEN 'preferred shippers'
         ELSE  'shippers on standby Only'
END As Alternative_Shippername
from shippers


Select ProductName, Price,
CASE
      WHEN Price Is null or Price <= 0 THEN 'Item not for resale'
      WHEN Price < 10 THEN 'Under $10'
      WHEN price between 10 and 50 THEN 'Under $50'
      WHEN Price > 50 and Price < 100 THEN 'OVER $50'
      ELSE 'Over $100'
    END as Classification
 From Products


Select shippername   , phone
 , CASE
WHEN shippername  =  'deliveroo' THEN 'Covers Chelsey'
WHEN shippername   =  'mr delivery' THEN'Covers Pimlico'
WHEN shippername   =  'mr shipper' THEN'Covers Victoria'
ELSE 'UPS'
END as Alternative_Shippername
from shippers


Select shippername , phone
, CASE shippername
WHEN 'deliveroo'   THEN 'Covers Chelsey'
WHEN 'mr delivery'  THEN'Covers Pimlico'
WHEN 'mr shipper'  THEN'Covers Victoria'
ELSE 'UPS'
END  as Alternative_Shippername    from shippers
```

Exercise

* List the suppliername , city , country ,and a new column with the alias REGION :
  For countries in Europe, display EUROPE
  Or if the country is UK then display'UK'
  Or'Rest of the World' for all other countries

Solution:

```
selectsuppliername( orcompanyname) , country,
Case      when country in( 'denmark','france','finland','germany','italy',
          'spain','sweden','Netherlands','Norway')then'Europe'
          when country ='uk'then'UK'
          else'Rest of the world'
endas'Region'from suppliersorderby Region,country
```

ISNULL/ COALESCE

ISNULL(Column_name, replacement)
Or
COALESCE(Column_name,replacement)

What should show In this case where there is a NULL values.

Example (MSSQL Server):

```
SELECT ProductID, Productname, Price, ISNULL(price,0)
FROM Products
```

Example in mySQL:

In MySQL we can use the IFNULL() function, like this:
or we can use the COALESCE() function, like this:

```
SELECT ProductID, Productname, Price, IFNULL(price,0)
FROM Products
```

ORACLE

```
SELECT ProductID, Productname, Price, COALESCE(price,0)
FROM Products
SELECT ProductID, Productname, COALESCE(productname,'no name')
FROM Products
```

Aggregate Queries

Avg(Column name) :

- SELECT avg(column_name) FROM table_name;

Sum(Column name)

- SELECT sum(column_name) FROM table_name;

Min(Column name)

- SELECT min(column_name) FROM table_name;

Max(Column name)

- SELECT max(column_name) FROM table_name;

Count(Column name)

- SELECT COUNT(column_name) FROM table_name;

– countshow many **Non-null**values in a specified column:

COUNT(*)

- SELECT COUNT(*) FROM table_name; - COUNT(*)returns the number of records in a table:

COUNT(DISTINCT column_name)

- SELECT COUNT(DISTINCT column_name) FROM table_name;

- counts how many distinct values in a specified column

Aggregate

Count

- How many records are on the orderdetails table
- How many records are on the suppliers table
- Count distinct prices on the products table

Sum, Avg

- What is the sum of prices of all products
- What is the average of price of all products
- What is the maximum of price of all products
- What is the minimum of price of all products

Group by

GROUP BY	SELECT column_name, aggregate_function(column_name) FROM table_name WHERE column_name operator value GROUP BY column_name ORDER BY column_name

Sum, Avg

Which Table will you use?

Tip: Use the AS (alias) Clause to name the aggregate column)

- What is the average of the price (or unitprice) column per Category
- What is the average price (or unitprice) per Supplier
- What is the maximum price (or unitprice) per Category
- What is the minimum price (or unitprice) per Supplier

Orders Table

OrderID, CustomerID, EmployeeID, OrderDate, ShipperID (or shipvia)

- How many Orders per Customer
- How many Orders per employee
- How many Orders per shipper

Group by and Aggregates Using WHERE

Tip: Use the AS (alias) Clause to name the aggregate column

Count using the Customer Table and WHERE

- How many Customers per country for all countries
- How many Customers per city
- How many Customers per country, per city
- How many Customers are in France
- How many Customers are in Berlin

Group by and having

```
SELECT column_name, aggregate_function(column_name)
FROM table_name
WHERE column_name operator value
GROUP BY column_name
HAVING aggregate_function(column_name) operator value
ORDER BY column_name
```

Which Table will you use?

In the Products table,
For every Category , what is the average of Price
where the ***average***of the Price per category is more than 50

In the Products table,
perSupplier, What is the average of Price
where the ***count*** of the SupplierIDis more than 1

What is the avg of Price in the Products
perSupplier
where the ***minimum*** of the Price is more than 10

in the Orders table
for every Customer , what is the count of Orders
Where the ***count*** of Orders is more than the 3
(ie customers who placed more than 3 orders)

in the Orders table
for every Employee , what is the count of Orders
where the ***count*** of Orders is more than the 3
,ie employees who placed more than 3 orders

Joins

Queries over more than one table : Joins

INNER JOIN	SELECT column_name(s) FROM table_name1 INNER JOIN table_name2 ON table_name1.column_name = table_name2.column_name

Inner joins

Tables Product and Supplier

- Common field?_____
- Show the ProductID, ProductName, Price, Suppliername , Country order by Productname
- How many records are in the result ____

Tables Product and Category

- Common field?_____
- Show the ProductID, ProductName, Price, categoryName order by Productname
- How many records are in the result ____

Tables Product and OrderDetail

- Common field?_____
- Show the ProductID, ProductName, Price, Qty, order by Productname
- How many records are in the result ____

Tables Orders and Customer

- Common field?_____
- Customername , OrderDate, OrderID, order by Customername
- How many records are in the result___

Tables Orders and Shipper

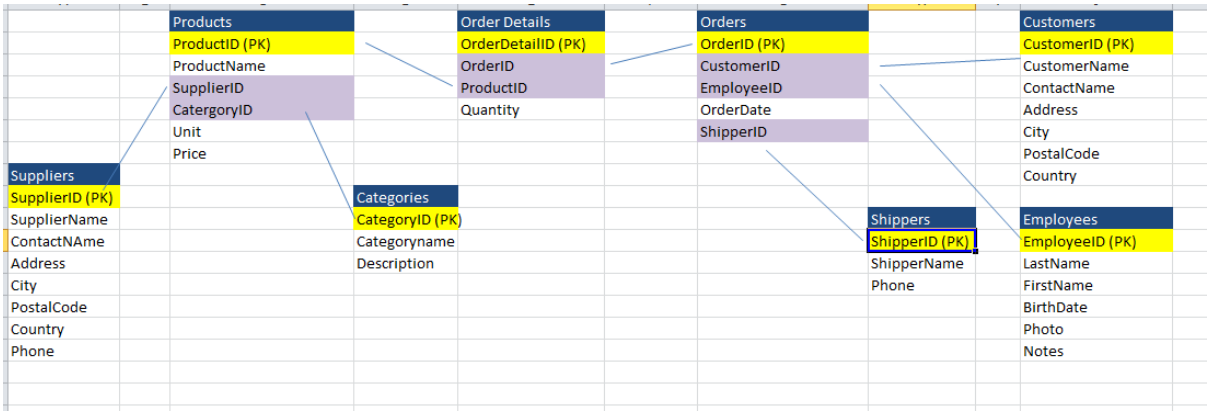
- Common field?_____
- OrderDate, Shippername order by Orderdate
- How many records are in the result ____

Tables Orders and Employee

- Common field?_____
- OrderDate, EmployeeFirstName, EmployeeLastName order by EmployeeFirstName
- How many records are in the result Inner Join____

Multiple table Inner Join

1. Show the Orderdate, OrderId and Customername ordered by Orderdate, OrderID
    - Tables : Orders , Customer
2. Show the Orderdate, OrderId , Customername , **FirstName** ordered by Orderdate, OrderID
    - Tables: Orders , Customer , **Employee**
3. Show the Orderdate, OrderId , Customername , EmployeeFirstName, **Shippername** ordered by Orderdate, OrderID
    - **Tables**: Orders , Customer , Employee, **Shippers**
4. Show the Orderdate, OrderId , Customername , EmployeeFirstName, Shippername , **OrderDetailID**, **Quantity** ordered by Orderdate, OrderID
    - **Tables**: Orders , Customer, Employee, Shippers , **OrderDetails**
5. Show the Orderdate, OrderId , Customername , EmployeeFirstName, Shippername , OrderDetailID, Quantity, **Price**, **Productname** ordered by Orderdate, OrderID
    - **Tables**: Orders , Customer , Employee, Shippers , OrderDetails, **Products**
6. Show the Orderdate, OrderId , Customername , EmployeeFirstName, Shippername , OrderDetailID, Productname , Quantity , Price, **CategoryName** ordered by Orderdate, OrderID
    - **Tables**: Orders , Customer , Employee, Shippers , OrderDetails, Products, **Category**
7. Show the Orderdate, OrderId , Customername , EmployeeFirstName, Shippername , OrderDetailID, Productname , Quantity , Price, CategoryName, **Suppliername** ordered by Orderdate, OrderID
    - **Tables**: Orders , Customer , Employee, Shippers , OrderDetails , Products, Category, **Supplier**



LEFT Join

LEFT JOIN	SELECT column_name(s) FROM table_name1 LEFT JOIN table_name2 ON table_name1.column_name=table_name2.column_name

Tables Customers and Orders Use Inner Join

- Customername , OrderDate , OrderID, order by Customername
- How many records are in the result ____

Tables Customers and Orders Use LEFT Join

- Customername , OrderDate , OrderID, order by Customername
- How many records are in the result ____

Tables Orders and Customers Use LEFT Join

- Customername , OrderDate , OrderNumber, order by Customername where ORDERID is null
- How many records are in the result ____

Exercise

1. Show Customers with no orders

   Show Customer_name, orderID using INNER Join
   Show Customer_name, orderID using LEFT Join
   Show Customer_name, orderID using LEFT Join where OrderID is Null
2. Show Employees with no Orders
3. Show Shippers with no Orders
4. Show Suppliers with no Products
5. Show Categories with no Products

RIGHT Join

RIGHT JOIN	SELECT column_name(s) FROM table_name1 RIGHT JOIN table_name2 ON table_name1.column_name=table_name2.column_name

Tables Customers and Orders Use RIGHT Join

- Customername , OrderDate , OrderNumber, order by Customername
- How many records are in the result ____

FULL Join

FULL JOIN	SELECT column_name(s) FROM table_name1 FULL JOIN table_name2 ON table_name1.column_name=table_name2.column_name

Tables Customers and Orders Use FULL Join

- Customername , OrderDate , OrderNumber, order by Customername
- How many records are in the result ____

MySQL : no full join , you will have to create a union query

Revision Exercise

Query exercise – Inner Join and Order By

8. Identify employees who are graduates, have a degree or went to college or university

9. Show the Suppliername , CategoryName, ProductName, Unit and Price for suppliers in the UK only, ordered by Suppliername , CategoryName, then ProductName

10. for records where the Customername ends with iste
11. for records where the Customername ends with essen
12. for records where the Customername contains essen
13. for records where the Customername contains grocer
14. Show all the columns where the Productname is Chais , Chang or Ikura (using IN)
15. for records where the CustomerID is between 10 and 15
16. for records where the Customername is between a and b
17. for records where the country is between a and m

18. Show records for suppliers in the UK and the suppliername contains ltd or the contactname contains pete

19. Show all the products on the Products Table where the price is null

20. Show the CategoryName, Suppliername , ProductName , Unit and Price
    - for beverages only and suppliers in the UK,
    - ordered by CatgoryName, then Suppliername , then ProductName

21. Show the Suppliername , ProductName, OrderDetailID, Price, Quantity
    - ordered by Suppliername , then ProductName

22. Show the Suppliername , OrderID, ProductName, Price, Quantity
    - ordered by Suppliername , then OrderID

23. Show EmployeeID,Productname
    - Category Beverages
    - ordered by EmployeeID, then Product

Revision Exercise Continued

Query exercise – Join and aggregate

- Show the number of orders per customer
- Show the number of orders per employee
- Show the number of orders per shipper

Query exercise – Join and aggregate and Order By

Show the OrderID, sum of Price, sum of Quantity per OrderID

- Show the average product price per supplier
- Show the number of orders for beverages
- Show the number of orders for beverages from suppliers in the US

- Show the Number of Orders per Employee , per Product
- Show the Number of Orders per Employee, per Supplier, per Product
- Show the Number of Orders per Employee, per Category, per Product
- Show Employees where the sum of quantity on an order is more than 1000

Left Joins

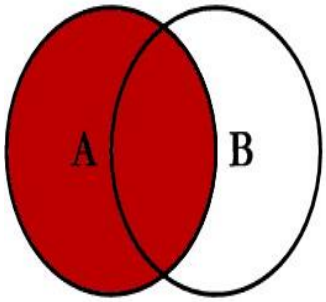
- List the customers with no orders
- List the employees with no orders
- List the suppliers with no products
- List the categories with no products

And or with join and dates

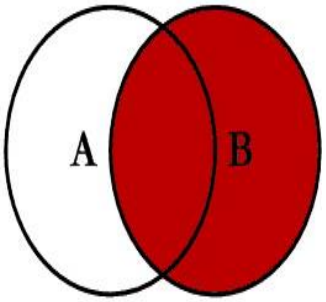
- Select
  categoryname, firstname , orderdate, orders.orderid, productname, price
  For orderdates in 1996 and
  categories could be beverages, condiments or contains conf

8. Which customers buy from suppliers in the same country
9. Which employee made the most orders
10. Which 5 employees made the most revenue
11. Which employee sold the most beverages

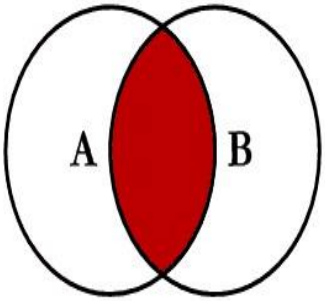
SQL JOINS



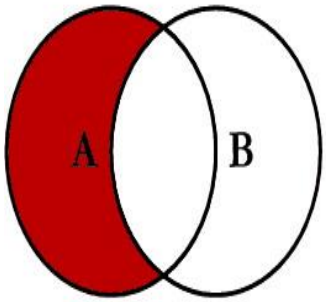
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key



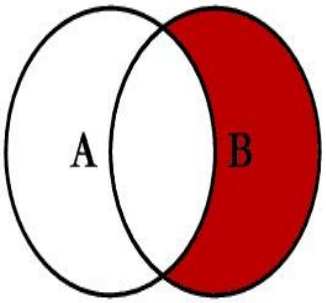
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key



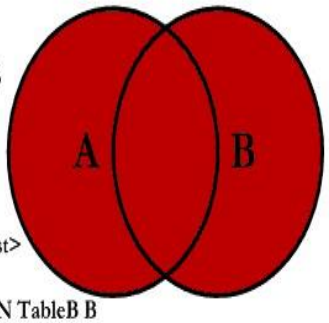
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key



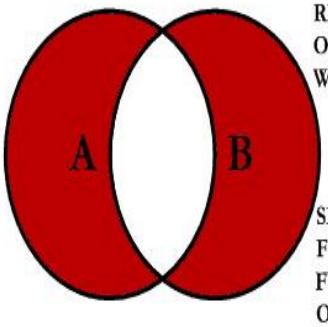
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL



SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL



SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key



SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL

Exact Numeric Data Types

Data type	Range	Storage
bigint	-2^63 (-9,223,372,036,854,775,808) to 2^63-1 (9,223,372,036,854,775,807)	8 Bytes
int	-2^31 (-2,147,483,648) to 2^31-1 (2,147,483,647)	4 Bytes
smallint	-2^15 (-32,768) to 2^15-1 (32,767)	2 Bytes
tinyint	0 to 255	1 Byte
Data type	Range	Storage
money	-922,337,203,685,477.5808 to 922,337,203,685,477.5807 (-922,337,203,685,477.58 to 922,337,203,685,477.58 for Informatica. Informatica only supports two decimals, not four.)	8 bytes
smallmoney	- 214,748.3648 to 214,748.3647	4 bytes

Approximate Numeric Data Types

float [**(***n***)**] Where *n* is the number of bits that are used to store the mantissa of the **float** number in scientific notation and, therefore, dictates the precision and storage size. If *n* is specified, it must be a value between **1** and **53**. The default value of *n* is **53**.

n value	Precision	Storage size
1-24	7 digits	4 bytes
25-53	15 digits	8 bytes

Note

SQL Server treats *n* as one of two possible values. If **1**<=n<=**24**, *n* is treated as **24**. If **25**<=n<=**53**, *n* is treated as **53**.

The SQL Server **float[(n)]** data type complies with the ISO standard for all values of *n* from **1** through **53**. The synonym for **double precision** is **float(53)**.

Remarks

Data type	Range	Storage
float	- 1.79E+308 to -2.23E-308, 0 and 2.23E-308 to 1.79E+308	Depends on the value of *n*
real	- 3.40E + 38 to -1.18E - 38, 0 and 1.18E - 38 to 3.40E + 38	4 Bytes

decimal[**(***p*[**,***s*] **)**] and **numeric**[**(***p*[**,***s*] **)**]
Fixed precision and scale numbers. When maximum precision is used, valid values are from - 10^38 +1 through 10^38 - 1. The ISO synonyms for **decimal** are **dec** and **dec(***p*, *s***)**. **numeric** is functionally equivalent to **decimal**.

p (precision)
The maximum total number of decimal digits that will be stored, both to the left and to the right of the decimal point. The precision must be a value from 1 through the maximum precision of 38. The default precision is 18.

Note

Informatica only supports 16 significant digits, regardless of the precision and scale specified.

s (scale)
The number of decimal digits that will be stored to the right of the decimal point. This number is subtracted from *p* to determine the maximum number of digits to the left of the decimal point. Scale must be a value from 0 through *p*. Scale can be specified only if precision is specified. The default scale is 0; therefore, 0 <= *s*<= *p*. Maximum storage sizes vary, based on the precision.

Precision	Storage bytes
1 - 9	5
10-19	9
20-28	13
29-38	17

Date and Time Data Types

The Transact-SQL date and time data types are listed in the following table:

Data type	Format	Range	Accuracy	Storage size (bytes)	User-defined fractional second precision	Time zone offset
time	hh:mm:ss[.nnnnnnn]	00:00:00.0000000 through 23:59:59.9999999	100 nanoseconds	3 to 5	Yes	No
date	YYYY-MM-DD	0001-01-01 through 9999-12-31	1 day	3	No	No
smalldatetime	YYYY-MM-DD hh:mm:ss	1900-01-01 through 2079-06-06	1 minute	4	No	No
datetime	YYYY-MM-DD hh:mm:ss[.nnn]	1753-01-01 through 9999-12-31	0.00333 second	8	No	No
datetime2	YYYY-MM-DD hh:mm:ss[.nnnnnnn]	0001-01-01 00:00:00.0000000 through 9999-12-31 23:59:59.9999999	100 nanoseconds	6 to 8	Yes	No

Character Strings Data Types

Sr.No.	DATA TYPE & Description
1	**char** Maximum length of 8,000 characters.(Fixed length non-Unicode characters)
2	**varchar** Maximum of 8,000 characters.(Variable-length non-Unicode data).
3	**varchar(max)** Maximum length of 2E + 31 characters, Variable-length non-Unicode data (SQL Server 2005 only).
4	**text** Variable-length non-Unicode data with a maximum length of 2,147,483,647 characters.

Unicode Character Strings Data Types

Sr.No.	DATA TYPE & Description
1	**nchar** Maximum length of 4,000 characters.(Fixed length Unicode)
2	**nvarchar** Maximum length of 4,000 characters.(Variable length Unicode)
3	**nvarchar(max)** Maximum length of 2E + 31 characters (SQL Server 2005 only).(Variable length Unicode)
4	**ntext** Maximum length of 1,073,741,823 characters. (Variable length Unicode)