สยามชั้นนำวิทยา
SIAM CHANNAN KIT

Shu Ha Ri

SEAL

防 25 止 62

We Love Bug

BASIC SQL WORKSHOP

สยามชั้นนายกิจ
SIAM CHANNAN KIT

Shu Ha Ri

SEAL

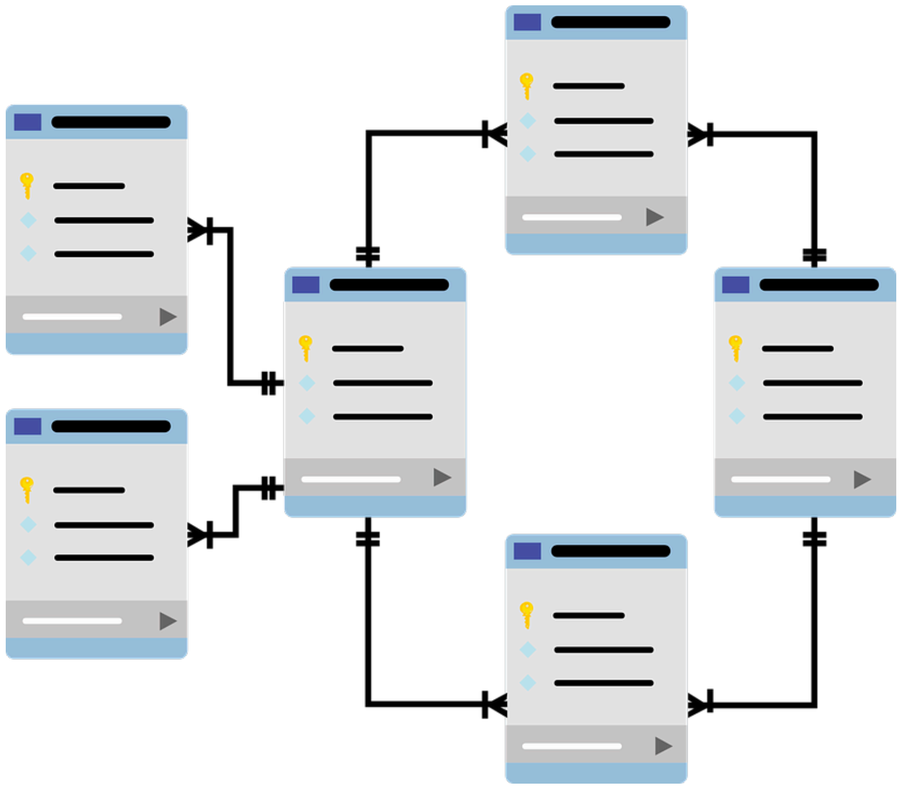
We Love Bug

Agenda

1. What is SQL and SQL syntax
2. SQL basic commands
3. SQL functions
4. Different type of SQL JOINs

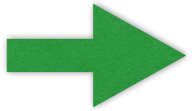
What is SQL and SQL syntax

SQL (Structured Query Language) is a standard language for accessing and manipulating databases.



What is SQL and SQL syntax

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden



```
CREATE TABLE `customers` (
  `CustomerID` int(11) NOT NULL,
  `CustomerName` varchar(255) DEFAULT NULL,
  `ContactName` varchar(255) DEFAULT NULL,
  `Address` varchar(255) DEFAULT NULL,
  `City` varchar(255) DEFAULT NULL,
  `PostalCode` varchar(255) DEFAULT NULL,
  `Country` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `customers` (`CustomerID`, `CustomerName`, `ContactName`, `Address`, `City`, `PostalCode`, `Country`) VA
(1, 'Alfreds Futterkiste', 'Maria Anders', 'Obere Str. 57', 'Berlin', '12209', 'Germany'),
(2, 'Ana Trujillo Emparedados y helados', 'Ana Trujillo', 'Avda. de la Constitución 2222', 'México D.F.', '05021', '
(3, 'Antonio Moreno Taquería', 'Antonio Moreno', 'Mataderos 2312', 'México D.F.', '05023', 'Mexico'),
(4, 'Around the Horn', 'Thomas Hardy', '120 Hanover Sq.', 'London', 'WA1 1DP', 'UK'),
(5, 'Berglunds snabbköp', 'Christina Berglund', 'Berguvsvägen 8', 'Luleå', 'S-958 22', 'Sweden'),
(6, 'Blauer See Delikatessen', 'Hanna Moos', 'Forsterstr. 57', 'Mannheim', '68306', 'Germany'),
(7, 'Blondel père et fils', 'Frédérique Citeaux', '24, place Kléber', 'Strasbourg', '67000', 'France'),
(8, 'Bólido Comidas preparadas', 'Martín Sommer', 'C/ Araquil, 67', 'Madrid', '28023', 'Spain'),
(9, 'Bon app\'', 'Laurence Lebihans', '12, rue des Bouchers', 'Marseille', '13008', 'France'),
(10, 'Bottom-Dollar Marketse', 'Elizabeth Lincoln', '23 Tsawassen Blvd.', 'Tsawassen', 'T2F 8M4', 'Canada'),
(11, 'B\'s Beverages', 'Victoria Ashworth', 'Fauntleroy Circus', 'London', 'EC2 5NT', 'UK'),
(12, 'Cactus Comidas para llevar', 'Patricio Simpson', 'Cerrito 333', 'Buenos Aires', '1010', 'Argentina'),
(13, 'Centro comercial Moctezuma', 'Francisco Chang', 'Sierras de Granada 9993', 'México D.F.', '05022', 'Mexico'),
(14, 'Chop-suey Chinese', 'Yang Wang', 'Hauptstr. 29', 'Bern', '3012', 'Switzerland'),
(15, 'Comércio Mineiro', 'Pedro Afonso', 'Av. dos Lusíadas, 23', 'São Paulo', '05432-043', 'Brazil'),
(16, 'Consolidated Holdings', 'Elizabeth Brown', 'Berkeley Gardens 12 Brewery', 'London', 'WX1 6LT', 'UK'),
(17, 'Drachenblut Delikatessend', 'Sven Ottlieb', 'Walserweg 21', 'Aachen', '52066', 'Germany'),
(18, 'Du monde entier', 'Janine Labrune', '67, rue des Cinquante Otages', 'Nantes', '44000', 'France'),
(19, 'Eastern Connection', 'Ann Devon', '35 King George', 'London', 'WX3 6FW', 'UK'),
(20, 'Ernst Handel', 'Roland Mendel', 'Kirchgasse 6', 'Graz', '8010', 'Austria'),
(21, 'Familia Arquibaldo', 'Aria Cruz', 'Rua Orós, 92', 'São Paulo', '05442-030', 'Brazil'),
(22, 'FISSA Fabrica Inter. Salchichas S.A.', 'Diego Roel', 'C/ Moralzarzal, 86', 'Madrid', '28034', 'Spain'),
(23, 'Folies gourmandes', 'Martine Rancé', '184, chaussée de Tournai', 'Lille', '59000', 'France'),
(24, 'Folk och fä HB', 'Maria Larsson', 'Åkergatan 24', 'Bräcke', 'S-844 67', 'Sweden'),
```

What is SQL and SQL syntax

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden

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

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany

What is SQL and SQL syntax

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden

‘Table’

‘Record’

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

‘Query’

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany

‘Result’

SQL basic commands : SELECT FROM

The **SELECT** statement is used to select data from a database.
The data returned is stored in a result table, called the result-set.

```
SELECT column1, column2, ...
FROM table_name;
```

```
SELECT * FROM table_name;
```

SQL basic commands : SELECT FROM

In example

```
SELECT CustomerName,Country FROM Customers;
SELECT * FROM Customers WHERE Country = 'Germany' AND CustomerName = 'Frankenversand';
```

With **ORDER BY**

```
SELECT * FROM Customers WHERE Country = 'Germany' ORDER BY City ASC;
SELECT * FROM Products ORDER BY Price DESC;
```

SELECT **DISTINCT**

```
SELECT DISTINCT Country FROM Customers;
```

SELECT **Aliases (AS)**

```
SELECT CustomerName as Customer , Address + ', ' + City + ', ' + Country as Address FROM Customers
```

Quiz#1 : SELECT FROM

1. Find all **cities** of customers in `**Brazil** `
2. Find all **products** that have **price** more than 20
3. Find **shipping address** for a customer name `**Wilman Kala**`
4. Find all **customers** that in country `**Germany**` and `**Mexico**`
5. Find all **customers** that in country `**Germany**` and `**Berlin**`

SQL basic commands : Insert Into , Update , Delete

The **INSERT INTO** statement is used to insert new records in a table.

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

```
INSERT INTO table_name
VALUES (value1, value2, value3, ...);
```

The **UPDATE** statement is used to modify the existing records in a table.

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

The **DELETE** statement is used to delete existing records in a table.

```
DELETE FROM table_name WHERE condition;
```

SQL basic commands : Insert Into , Update , Delete

In example

```
INSERT  INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)
VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'London', 'WA1 1DP', 'UK');

INSERT  INTO Customers VALUES
(111,'Karnawat', 'Karn', '3 Ladphrao PromptPhan', 'Bangkok', '10003', 'Thailand'),
(112,'Panumas', 'Lek', '3 Ladphrao PromptPhan', 'Bangkok', '10003', 'Thailand');

UPDATE Products SET CategoryID = 2 WHERE Price > 1;

DELETE FROM Customers WHERE City = 'London' AND PostalCode = 'WA1 1DP';
```

Quiz #2 : Insert Into , Update , Delete

1. Insert `**Your**` information into `**Customer** `table
2. Update`**Country**` of the customers who live in `**London**` and `**Berlin**` to `**Bangkok**`
3. Delete **products** that have price over `**20**`

BREAK;

SQL functions : Min , Max

The **MIN**() function returns the **smallest** value of the selected column.
The **MAX**() function returns the **largest** value of the selected column.

```
SELECT MIN(column_name)
FROM table_name
WHERE condition;
```

```
SELECT MAX(column_name)
FROM table_name
WHERE condition;
```

In example

```
SELECT MIN(Price) AS SmallestPrice
FROM Products;
```

Quiz#3 : Min , Max

1. Find the **product** in **CategoryID** `2` that have **highest** price
2. Update **name** of the **product** that have **lowest** price to `Cheapest Item`

SQL functions : Count , AVG , Sum

The **COUNT**() function returns the **number of rows** that matches a specified criterion.
The **AVG**() function returns the **average** value of a numeric column.
The **SUM**() function returns the **total sum** of a numeric column.

```sql
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

```sql
SELECT AVG(column_name)
FROM table_name
WHERE condition;
```

```sql
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

In example

```sql
SELECT COUNT(ProductID)
FROM Products;
```

```sql
SELECT AVG(Price)
FROM Products;
```

```sql
SELECT SUM(Quantity)
FROM OrderDetails;
```

Quiz#4 : Count , AVG , Sum

1. Find how many customer in `**Argentina**`
2. Find the **average price** of **products** in **CategoryID** `2`
3. Find how many **customer** of each **country**
4. Find the **EmployeeID** that have highest **order**

SQL functions : In , Between

The **IN** operator allows you to **specify multiple values** in a WHERE clause, shorthand for multiple **OR** conditions.

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1, value2, ...);
```

Compare with **OR** conditions

```
SELECT * FROM Customers
WHERE Country IN ('Germany', 'France', 'UK');
```

```
SELECT * FROM Customers
WHERE Country = 'Germany' OR Country = 'France' OR Country = 'UK';
```

SQL functions : In , Between

The **BETWEEN** operator selects values **within a given range**. The values can be **numbers**, **text**, or **dates**

```
SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN value1 AND value2;
```

Compare with **AND** conditions

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

```
SELECT * FROM Products
WHERE Price >= 10 AND Price <= 20;
```

Quiz#5 : In , Between

1. Find the **products** that have **price** between **15** and **20**
2. Find the **customers** that **not in** `**Argentina**`, `**Canada**` and `**Denmark**`
3. Find the employeeID that have highest **order** between **01/07/1996** and **31/07/1996 and** how many order that he have

BREAK;

Different type of SQL JOINs

Orders Table

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10308	2	7	9/18/1996	3
10365	3	3	11/27/1996	2
10383	4	8	12/16/1996	3
10355	4	6	11/15/1996	1
10278	5	8	8/12/1996	2

Customer Table

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden

Different type of SQL JOINs

Orders Table

PK FK

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10308	2	7	9/18/1996	3
10365	3	3	11/27/1996	2
10383	4	8	12/16/1996	3
10355	4	6	11/15/1996	1
10278	5	8	8/12/1996	2

Customer Table

PK

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden

PK : Primary Key
FK : Foreign Key

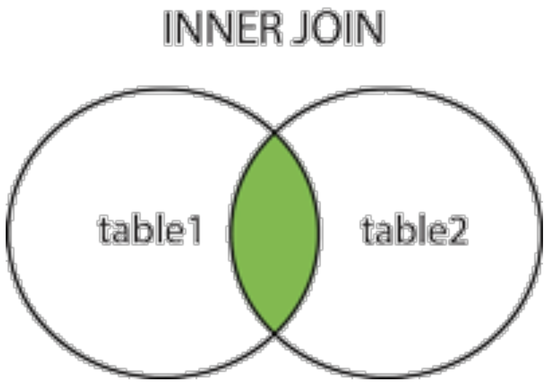
Different type of SQL JOINs

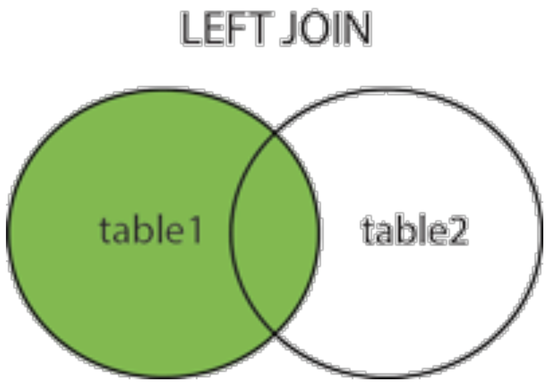
```sql
SELECT Orders.OrderID, Customers.CustomerName
FROM Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

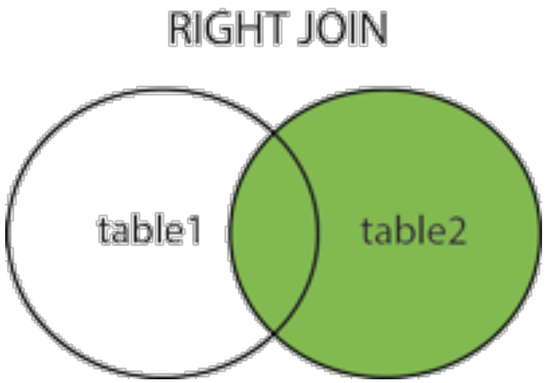
OrderID	CustomerName
10248	Wilman Kala
10249	Tradição Hipermercados
10250	Hanari Carnes
10251	Victuailles en stock
10252	Suprêmes délices
10253	Hanari Carnes
10254	Chop-suey Chinese

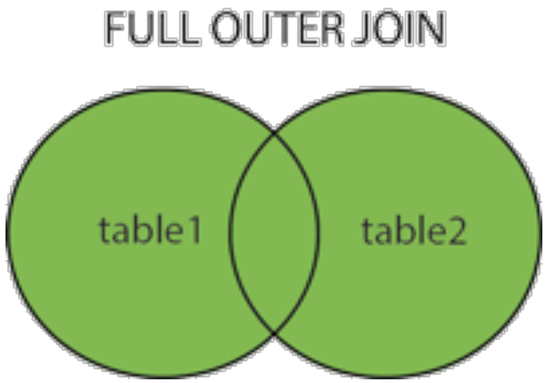
Different type of SQL JOINs

(INNER) JOIN: Returns records that have matching values in both tables
LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table
RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table
FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table









Different type of SQL JOINs

In 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);
```

OrderID	CustomerName	ShipperName
10248	Wilman Kala	Federal Shipping
10249	Tradição Hipermercados	Speedy Express
10250	Hanari Carnes	United Package
10251	Victuailles en stock	Speedy Express
10252	Suprêmes délices	United Package
10253	Hanari Carnes	United Package
10254	Chop-suey Chinese	United Package
10255	Richter Supermarkt	Federal Shipping
10256	Wellington Importadora	United Package

Quiz#6 Different type of SQL JOINs

CustomerID	CustomerName	OrderID
1	Alfreds Futterkiste	*null*
2	Ana Trujillo Emparedados y helados	10308
3	Antonio Moreno Taquería	10365
4	Around the Horn	10355
4	Around the Horn	10383
5	Berglunds snabbköp	10278
5	Berglunds snabbköp	10280
5	Berglunds snabbköp	10384
6	Blauer See Delikatessen	*null*
7	Blondel père et fils	10265
7	Blondel père et fils	10297
7	Blondel père et fils	10360
7	Blondel père et fils	10436

Quiz#7 Different type of SQL JOINs

OrderID	CustomerName	CustomerID	TotalProduct	TotalQuantity	TotalPrice
10248	Wilman Kala	90	3	27	566
10249	Tradição Hipermercados	81	2	49	2329.25
10250	Hanari Carnes	34	3	60	2267.25
10251	Victuailles en stock	84	3	41	839.5
10252	Suprêmes délices	76	3	105	4662.5
10253	Hanari Carnes	34	3	102	1806
10254	Chop-suey Chinese	14	3	57	781.5
10255	Richter Supermarkt	68	4	110	3115.75
10256	Wellington Importadora	88	2	27	648
10257	HILARIÓN-Abastos	35	3	46	1400.5
10258	Ernst Handel	20	3	121	2529.75
10259	Centro comercial Moctezuma	13	2	11	126
10260	Old World Delicatessen	55	4	102	2183.9
10261	Que Delícia	61	2	40	560
10262	Rattlesnake Canyon Grocery	65	3	29	782.2