Database Fundamentals

Lecture 3

SQL (Part2)

Like operator in where clause

- The LIKE search condition uses **wildcards** to search for patterns within a string.

Description	SQL Wildcard	MS-DOS Wildcard	Example
Any number (zero or more) of arbitrary characters	%	*	'Able' LIKE 'A%'
One arbitrary character	_	?	'Able' LIKE 'Abl_'

Syntax

```
SELECT Columns
FROM tables
WHERE column_in _condition
LIKE ‘values + wildcards';
```

```
SELECT fname
FROM dbo.employee
WHERE fname LIKE 'A%';
```

Using the Between search condition

- Greater than or equal to the first value, and less than or equal to the second value.

```
Select …..
From……..
Where column_name [ NOT ] BETWEEN
begin_expression AND end_expression
```

```
Select * from employee
where salary between 1000 and 2000
```

Comparing with a list

- The WHERE condition can compare the test value against the values in a list using IN. IN operator can also be mixed with a NOT to reverse the condition.

Select …..

From……..

Where column_name [NOT] IN (subquery | expression [,...n])

```
Select * from employee
Where dno in (10, 30)
```

AND Condition / OR condition

AND

Returns:

True If Both Conditions are True

```
Select Employeename, salary
From Employee
Where salary >3000
And employeename like 'M%'
```

OR

Returns:

True If One Condition is True

```
Select fname, lname, salary, dno
From Employee
Where salary >1000
or dno = 50
```

NOT Condition

Returns the opposite of the Conditions result

	NOT
TRUE	FALSE
FALSE	TRUE
UNKNOWN	UNKNOWN

```
Select fname, salary
 From Employee
 Where fname not like ('M%')
```

Ordering the Result Set

- SQL Server usually returns the data in the order of the primary key (because that's probably the clustered index), but there's no logical guarantee of that order.
- The only correct way to sort the results is with an ORDER BY clause.
- Sort order can be specified as ASC (ascending) or DESC (descending) for each column.

Ordering the Result Set

Order by Options:

- Column _name
- Alias
- Column_position

```
Select dependent_name , Bdate
From Dependent
order by Dependent_name
```

Join Types

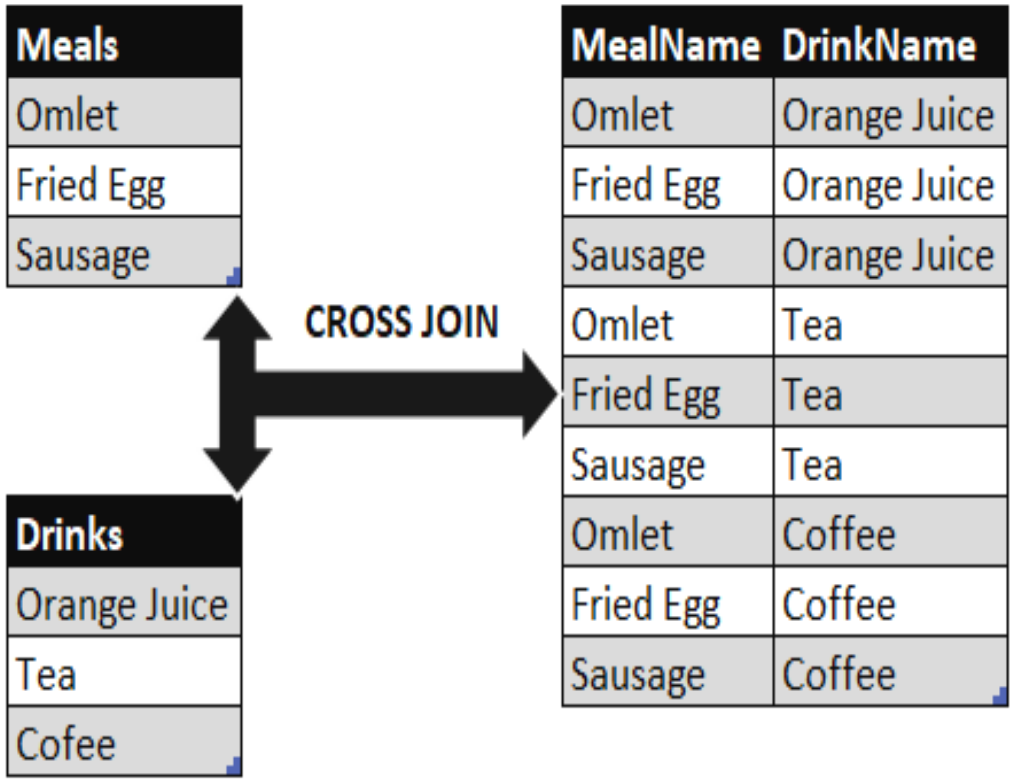
Joining tables to obtaining data From more than one Table this needs to Join Between these Tables.

- Cross Join/Cartesian product
- Inner Join/ Equi-Join
- Outer Join (left , Right, full)
- Self Join / Recursive Join
- Not Equal Join

Cross Join

The CROSS JOIN is used to generate a paired combination of each row of the first table with each row of the second table. This join type is also known as cartesian join.

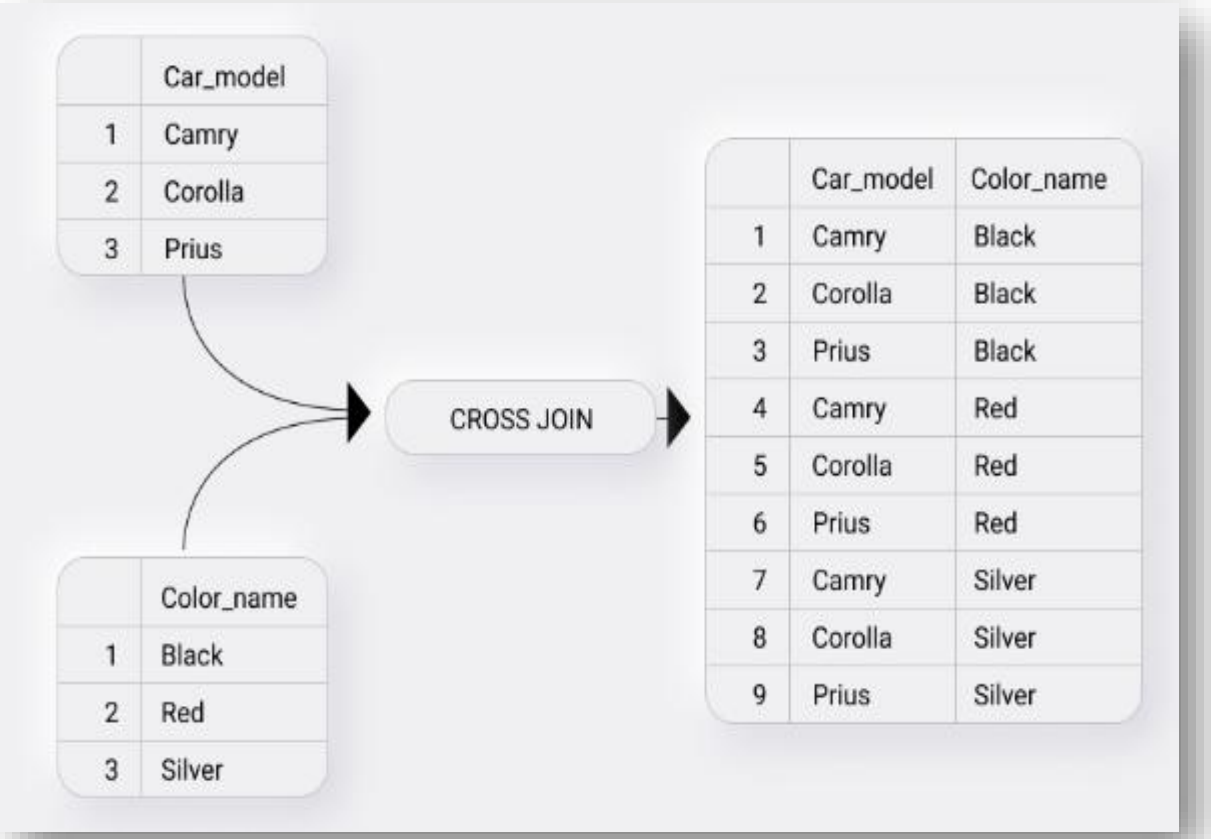
The **result set row count will equal to multiplication of tables row counts that will be joined**



Cross Join

The CROSS JOIN is used to generate a paired combination of each row of the first table with each row of the second table. This join type is also known as cartesian join.

The **result set row count will equal to multiplication of tables row counts that will be joined**



Joins and tables

Customers

Customer_Id	Customer Name	Phone
100	Ahmed Ali	012……
200	Fatema Khaled	015…..
300	Hany Mohamed	011…..
400	Mohamed Ahmed	011…..
500	Mona Ali	010….
600	Hamza M.	012…….

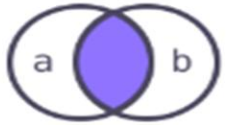
PK

Sales

Order_number	Product	No.of Units	Cust_id
1000	Milk	3	**100**
1001	Juice	5	**500**
1002	Bread	6	
1004	Milk	2	**500**
1005	Bread	4	**300**
1006	Suger	10	

FK

Inner Join/ Equi-Join

Join type	Visually	Example usage
Inner join		`a JOIN b ON a.id = b.id`

- It Select rows from the **two tables that have equal values** in matched columns. Two Columns **Must have same Data type** in two tables

Model1

```
Select customer_id, customer_name,
Order_number, product
From customer, purchase
Where customer_id = Cust_id
```

Model2

```
Select customer_id, customer_name,
Order_number, product
From customer[inner]join purchase
ON customer_id = Cust_id
```

Joins and tables

Customers

Customer_Id	Customer Name	Phone
100	Ahmed Ali	012……
200	Fatema Khaled	015…..
300	Hany Mohamed	011…..
400	Mohamed Ahmed	011…..
500	Mona Ali	010….
600	Hamza M.	012…….

Sales

Order_number	Product	No.of Units	Cust_id
1000	Milk	3	100
1001	Juice	5	500
1002	Bread	6	
1004	Milk	2	500
1005	Bread	4	300
1006	Suger	10	

Inner Join

Customer table

Sales table

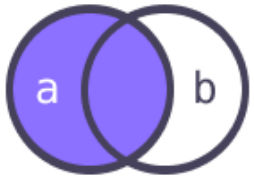
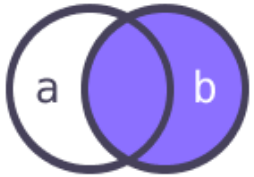
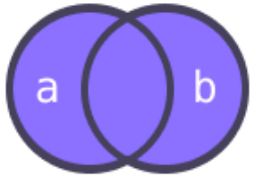
Customer_Id	Customer Name	Order_number	Product
100	Ahmed Ali	**1000**	Milk
300	Hany Mohamed	**1005**	Bread
500	Mona Ali	**1004**	Milk
500	Mona Ali	**1001**	Juice

But

What if I need all customers registering even if they did not do orders??

What if I need all the orders even if I don't know which customer did ????

Outer Joins

Join type	Visually	Example usage
Left join		`a LEFT JOIN b ON a.id = b.id`
Right join		`a RIGHT JOIN b ON a.id = b.id`
Full outer join		`a FULL OUTER JOIN b ON a.id = b.id`

Outer Join

Left Outer Join

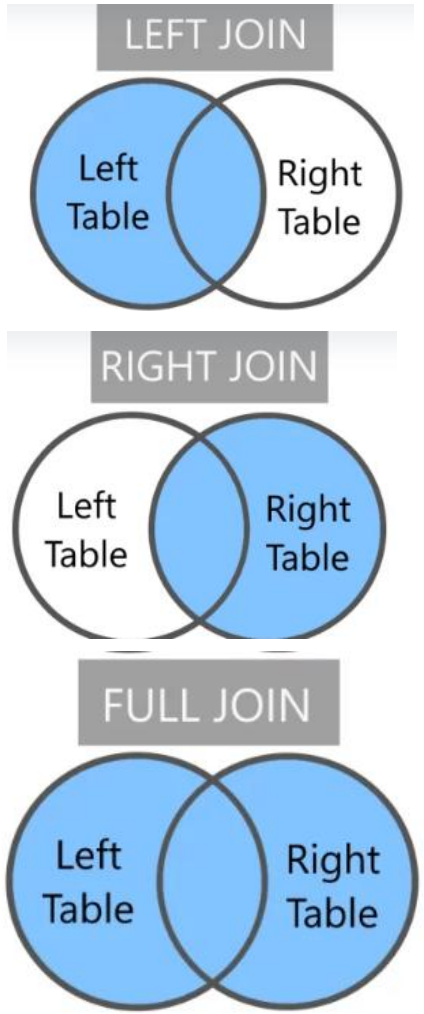
```
Select customer_id, customer_name, Order_number, product
From customer left outer join purchase
On customer_id = Cust_id
```

Right Outer Join

```
Select customer_id, customer_name, Order_number, product
From customer right outer join purchase
On customer_id = Cust_id
```

Full Outer Join

```
Select customer_id, customer_name, Order_number, product
From customer full outer join purchase
On customer_id = Cust_id
```



Left outer Join

Customer_Id	Customer Name	Phone	Order_number	Product	No.of Units	Customer_id
100	Ahmed Ali	012……	**1000**	Milk	3	**100**
300	Hany Mohamed	011……	**1005**	Bread	4	**300**
500	Mona Ali	010….	**1004**	Milk	2	**500**
500	Mona Ali	010….	**1001**	Juice	5	**500**
600	Hamza M.	012……	**Null**	**Null**	**Null**	**Null**
200	Fatema Khaled	015……	**Null**	**Null**	**Null**	**Null**
400	Mohamed Ahmed	011……	**Null**	**Null**	**Null**	**Null**

Right outer Join

Customer_Id	Customer Name	Phone	Order_number	Product	No.of Units	Customer_id
100	Ahmed Ali	012……	**1000**	Milk	3	**100**
300	Hany Mohamed	011……	**1005**	Bread	4	**300**
500	Mona Ali	010….	**1004**	Milk	2	**500**
500	Mona Ali	010….	**1001**	Juice	5	**500**
Null	**Null**	**Null**	**1002**	Bread	6	**null**
Null	**Null**	**Null**	**1006**	Suger	10	**null**

Full outer Join

Inner

Left

Right

Customer_Id	Customer Name	Phone	Order_number	Product	No.of Units	Customer_id
100	Ahmed Ali	012……	**1000**	Milk	3	**100**
300	Hany Mohamed	011……	**1005**	Bread	4	**300**
500	Mona Ali	010….	**1004**	Milk	2	**500**
500	Mona Ali	010….	**1001**	Juice	5	**500**
600	Hamza M.	012……	**Null**	**Null**	**Null**	**Null**
200	Fatema Khaled	015……	**Null**	**Null**	**Null**	**Null**
400	Mohamed Ahmed	011……	**Null**	**Null**	**Null**	**Null**
Null	**Null**	**Null**	**1002**	Bread	6	**null**
Null	**Null**	**Null**	**1006**	Suger	10	**null**

Qualifying Ambiguous Column Names

- Use Table Prefix To qualify column names that are in multiple tables
- Use table prefixes to improve performance
- Use column aliases to distinguish column that have identical names but reside in different tables

Wrong Example

```
Select customer_id, customer_name, Order_Order_number, product
From customer, purchase
Where customer_id = customer_id
```

Solution

```
Select customer.customer_id, customer_name, Order_Order_number,
product
From customer, purchase
Where customer.customer_id = customer.customer_id
```

Using Table Aliases

- Use Table Aliases to simplify queries
- Use table aliases to improve performance

```
Select C.customer_id, customer_name, Order_number, product
From customer C, purchase P
Where C.customer_id = P.customer_id
```

Self Join Syntax

- To find Name Of employee's manager , you need to join Employees table to itself

- Select name, salary, name, salary

  From Employee Emp , Employee Mng

  Where…….

Self Join Syntax

Employee

name	Ssn (PK)	Add	salary	Gender	Bdate	Deptid(FK)	SupID(FK)
A	11111	jhcsjah	2000	M	12542	100	
B	2222	jdhsjhjk	5000	M	12554	100	11111
C	1212	jhdkjsh	8000	F	1254		
E	3333	ndjsdn	2000	F		300	11111
G	5050	dsadsa	5000	M		200	

name	Ssn (PK)	salary	Gender	Deptid (FK)	Manager(FK)
Ahmed	1	20000	M	100	
Belal	2	15000	M	400	1
Ola	3	12000	F	500	1
Eman	4	8000	F	400	2
Gamal	5	5000	M	400	2

Employee

AS Emp

Employee
AS Mng

name	Ssn (PK)	salary	Gender	Deptid (FK)	Manager(FK)
Ahmed	1	20000	M	100	
Belal	2	15000	M	400	1
Ola	3	12000	F	500	1
Eman	4	8000	F	400	2
Gamal	5	5000	M	400	2

NonEquiJoins

- A nonequijoin is a join condition containing something other than an equality operator

- Select E.name , E.salary, G.grade
  from Employee E , grades G
  where E.salary between G.lowsal and G.highsal

jGrades

LowSal	HighSal	Grade
1000	4000	B
4000	7000	A

Employee

EmpNo	Name	Salary
100	Ahmed	5000
200	Mai	3000

Union and union all

- UNION combines the results of two or more queries into a single result set that includes all the rows that belong to all queries in the union
- UNION removes duplicate records (where all columns in the results are the same), UNION ALL does not.
- The column names, or aliases, are determined by the first SELECT.
- The order by clause sorts the results of all the SELECTs and must go on the last SELECT, but it uses the column names from the first SELECT.

Union and union all

UNION

```
Select  name, city, phone
From customers
Where  city in ('Cairo', 'Aswan')

UNION

Select Sname, place, mobile
From Supplyer
 Where  place in ('Cairo','Aswan)

Order by city
```

Exclude redundant data

UNION All

```
Select  name, city, phone
From customers
Where  city in ('Cairo', 'Aswan')

UNION All

Select Sname, place, mobile
From Supplyer
 Where  place in ('Cairo','Aswan)

Order by city
```

include redundant data

elect * from Student
here St_Age<=23

Intersect and except

- **INTERSECT** returns any distinct values that are returned by both the query on the left (up) and right (down) sides of the INTERSECT operand
- **EXCEPT** returns any distinct values from the query to the left (up)of the EXCEPT operand that are not also returned from the right query

Intersect and except

Intersect

```
Select  name, city, phone
From customers
Where  city in ('Cairo', 'Aswan')

Intersect

Select Sname, place, mobile
From Supplyer
 Where  place in ('Cairo','Aswan)
```

Returns customers who are **Also** one of our suppliers in the company

Except

```
Select  name, city, phone
From customers
Where  city in ('Cairo', 'Aswan')

Except

Select Sname, place, mobile
From Supplyer
 Where  place in ('Cairo','Aswan)
```

Returns customers who are **NOT** one of our suppliers in the company

Questions ?