

SQL

DBMS

# IN Operator

- The IN operator allows you to specify multiple values in a WHERE clause.
- The values can be **numbers, text, or dates**.

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

# IN Operator

`SELECT * FROM Persons`

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune

`SELECT * FROM Persons  
WHERE City IN ('Delhi', 'Pune');`

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
3	Sharma	Gaurav	Pune

# IN Operator

**SELECT \* FROM** Persons

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune

**SELECT \* FROM** Persons  
**WHERE** City **NOT IN** ('Delhi', 'Pune');

ID	FirstName	LastName	City
8	Singh	Ravindra	Mumbai

# BETWEEN Operator

- The BETWEEN operator selects values within a given range.
- The values can be **numbers, text, or dates**.
- The BETWEEN operator is inclusive: **begin and end values are included**.
- If any expression is NULL then BETWEEN returns NULL.

```
SELECT column_name(s)  
FROM table_name  
WHERE column_name BETWEEN begin_expr AND end_expr;
```

# BETWEEN Operator

**SELECT \* FROM Persons**

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune

**SELECT \* FROM Persons  
WHERE City BETWEEN 'Delho' AND  
'Pune';**

ID	FirstName	LastName	City
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune

**SELECT \* FROM Persons  
WHERE ID BETWEEN 0 AND 3;**

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
3	Sharma	Gaurav	Pune

# BETWEEN Operator

SELECT \* FROM Persons

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune

SELECT \* FROM Persons

WHERE City NOT BETWEEN 'Delhi'  
AND 'Pune';

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi

SELECT \* FROM Persons

WHERE ID NOT BETWEEN 0 AND 3;

ID	FirstName	LastName	City
8	Singh	Ravindra	Mumbai

# ORDER BY Clause

`SELECT * FROM Persons;`

ID	LastName	FirstName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune
4	Singh	Gaurav	Mumbai

ASC - Ascending Order

DESC - Descending Order

**NOTE:** Default order is Ascending Order.

`SELECT FirstName, City FROM Persons ORDER BY City DESC;`

FirstName	City
Gaurav	Pune
Ravindra	Mumbai
Gaurav	Mumbai
Pramod	Delhi

Q: Draw resulting table for following query.

`SELECT * FROM Persons WHERE City = 'Mumbai' OR ID > 2 ORDER BY City;`



# ORDER BY Clause

**SELECT \* FROM** Persons;

ID	LastName	FirstName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune
4	Singh	Gaurav	Mumbai

**SELECT** FirstName, City **FROM**  
Persons **ORDER BY** City **DESC**, ID  
**ASC**;

FirstName	City
Gaurav	Pune
Gaurav	Mumbai
Ravindra	Mumbai
Pramod	Delhi

# Finding Top Elements

`SELECT * FROM Persons;`

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune
4	Singh	Gaurav	Mumbai

➤ Different DBMS use different clause for this: TOP, **LIMIT**, ROWNUM

➤ MySql uses LIMIT; MS SQL uses TOP & Oracle uses ROWNUM.

`SELECT TOP 2 * FROM Persons;`

`SELECT * FROM Persons LIMIT 2;`

`SELECT * FROM Persons  
ROWNUM<= 2;`

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai

# Finding Top Elements (MySQL)

`SELECT * FROM Persons;`

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune
4	Singh	Gaurav	Mumbai

## Offset:

- specifies the offset of the first row to return.
- The offset of the first row is 0, not 1.

`SELECT * FROM Persons LIMIT 2;`

`SELECT * FROM Persons LIMIT 0,2;`

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai

## LIMIT count;

or

## LIMIT offset , count;

## Count:

- specifies the maximum number of rows to be returned.

# Finding Top Elements (MySQL)


## Offset:

- specifies the offset of the first row to return.
- The offset of the first row is 0, not 1.

## Count:

- specifies the maximum number of rows to be returned.

Offset      Count



LIMIT 3, 2 => Means Leave  
first 3 records and then  
take next 2 records.

## Offset Value      Table row

0	row 1
1	row 2
2	row 3
3	row 4
4	row 5
5	row 6
6	row 7
7	row 8

# Finding Top Elements (MySQL)

```
SELECT * FROM Persons;
```

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune
4	Singh	Gaurav	Mumbai

```
SELECT * FROM Persons LIMIT 2, 1;
```

ID	FirstName	LastName	City
3	Sharma	Gaurav	Pune

Question: Find the  $n^{\text{th}}$  highest ID row.

# Subquery

A subquery is a query nested within another query.

Example: `SELECT FirstName, LastName, City FROM Persons WHERE Id IN (SELECT Id FROM Persons WHERE City = 'Mumbai' OR City = 'Pune')`

A subquery which is inside another query is called an **inner query** while the query that contains the subquery is called an **outer query**.

A subquery can be nested inside another subquery.

# Subquery

Example:

```
SELECT FirstName, LastName, City FROM Persons WHERE Id IN (SELECT  
Id FROM Persons WHERE City = 'Mumbai' OR City = 'Pune')
```

Here, first the inner query is resolved.

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune
4	Singh	Gaurav	Mumbai



ID
8
3
4

# Subquery

Example:

```
SELECT FirstName, LastName, City FROM Persons WHERE Id IN (SELECT  
Id FROM Persons WHERE City = 'Mumbai' OR City = 'Pune')
```

In the actual query, inner query is replaced by the values returned by the inner query.



ID
8
3
4

```
SELECT FirstName, LastName, City FROM Persons WHERE Id IN (8, 3, 4)
```



# Subquery

Example:

**SELECT** FirstName, LastName, City **FROM** Persons **WHERE** Id **IN** (8, 3, 4)

Now, the outer query is resolved.

ID	FirstName	LastName	City
2	Kumar	Pramod	Delhi
8	Singh	Ravindra	Mumbai
3	Sharma	Gaurav	Pune
4	Singh	Gaurav	Mumbai

Persons Table



FirstName	LastName	City
Singh	Ravindra	Mumbai
Sharma	Gaurav	Pune
Singh	Gaurav	Mumbai

Final Results

# Subquery Example

- **Customer** (cust\_id, cust\_name, city, country)
- **Orders** (ord\_no, pur\_amt, ord\_date, cust\_id, salesman\_id)
- **Salesman** (salesman\_id, name, city, commission)
- Write a query to list all customers with order amount greater than 1000.

# Subquery Example

- **Customer** (cust\_id, cust\_name, city, country)
- **Orders** (ord\_no, pur\_amt, ord\_date, cust\_id, salesman\_id)
- **Salesman** (salesman\_id, name, city, commission)
- Write a query to list all customers with order amount greater than 1000.
- Sol:  

```
SELECT cust_name FROM  
customer WHERE cust_id IN  
(SELECT cust_id FROM order  
WHERE pur_amt > 1000)
```