**SQL Notes**

* Normally two databases are related by ID as ID is something which is not updated in future times while other fields might update.

Eg – User placed an order. In orders we store user id who placed the order, if we store address and then if that user changes the address it won’t reflect in orders

**OPERATORS**

* **IN operator**

Imagine doing

WHERE id = 23 OR id = 28 OR id = 98… so on

There is a sol for this – IN operator

**Eg – WHERE** id **IN (23, 28, 98, …)**

It is totally equivalent to that, i.e. there is no ordering based on sequences. Also we can use NOT with IN .. **NOT IN (13, 14, 15)** etc

* **BETWEEN operator**

WHERE id **BETWEEN** 23 **AND** 98

1. Boundary inclusive
2. Don’t forget **AND**

* **LIKE operator**

It is used for querring inside a string, **CASE INSENSITIVE**

Eg-

1. **%** - if we have to look for string containing XYZ in a column

Query – **WHERE** address **LIKE** ‘%XYZ%’ | true for AmmmmXYZ , JdkXYZlkjkljfldkfj, XYZjhgkjdf

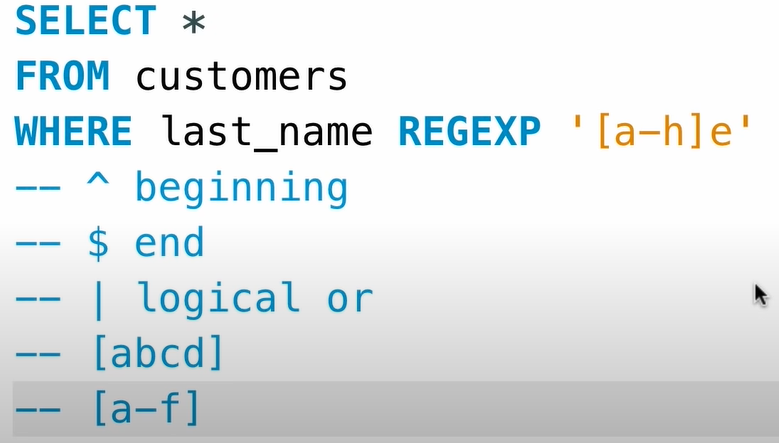
% - any no of char

\_ - one char

1. **\_\_**  - if we want to look after certain fixed no of char, use this

Eg – WHERE phone **LIKE ‘\_\_\_\_\_\_\_\_\_9’**  - this is true for a 10 digit phone no whose last digit is 9.

* **REGEXP operator**



Like **LIKE** but allows more flexibility

‘XYZ’ === ‘%XYZ%

Pipes – Regex provides pipes which work like OR

* **IS NULL**

We can’t use where id = NULL as obv NULL can’t be compared

Instead we should use **WHERE** id **IS NULL or WHERE** id **IS NOT NULL**

**CLAUSES**

* **ORDER BY**

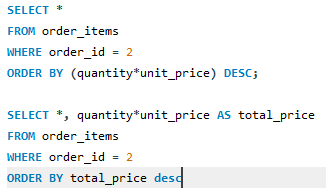
By default data is sorted by primary key. We can change it using ORDER BY colnames

If there are multiple cols specified it first sorts by 1st col then 2nd and so on..

We can also use 1, 2 , 3 .. they map to the columns specified after SELECT – NOT PREFERRED

**DESC**  is used to sort by descending order

Eg.



* **LIMIT Clause**

It is used to control the no of results of a query. [IMP] It is positioned below all the clauses.

LIMIT 3 – Returns only 3 results. If 3 > no of results , all results are returned

**Offset** – If specified it skips that no of results from ans and prints the next specified no of results

**Eg- ORDER 6, 3** – this will return 7th, 8th, 9th results.

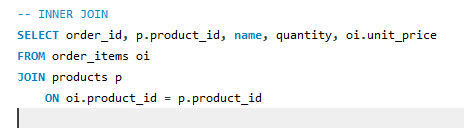
**JOINS**

* **INNER JOIN**

Since instead of changeable values we try to store their ids, now its time to fetch the values by the id.

We can easily do that with inner join

table1.id 🡪 table2.id



We can also use alias for tables, for that we just need to write alias just after table name with space.

**return value-**

Default return value if all cols of t1 then all cols of t2,

But we can specify cols, incase of conflicting names we need to specify the table eg- ‘t1.id’ instead of ‘id’ see above img.