**INTERMEDIATE SQL**

# Relational Languages :

User only needs to specify the answer that they want , not How to compute it.

* High-end have a sophisticated query optimizer .

SQL:standard for structured query language.

1. Data Manipulation Language
2. Data Definition Language
3. Data Control Language

Also include : view definition , Integrity &Referential constrains , Transactions

NOTE:SQL based on bags not sets .

# Aggregates:

Functions that return a single value from a bag of tuples:-

* AVG 🡪 return the average col value
* MIN 🡪 return the minimum col value
* MAX 🡪 return the maximum col value
* SUM🡪 return sum values in col
* COUNT🡪 return # of values for col

Count , sum , avg support distinct

# Group by :-

Project tuples into subsets and calculate aggregates against each subset .

# Having:-

Filters results based on aggregation computation like a where clause for group by.

# Data / Time operation :-

Can be used both output and predicates support / syntax varies widely .

# Output Redirection :-

Store query results in another table :

* Table must not already defined .
* Table will have the same # columns with the same types as the input .

# Output control :-

Ordered by <column\*> [ASI|DESC]

Limit < count > [offset]

# Nested queries :-

Queries containing other queries , they are often difficult to optimize .

ALL🡪 must satisfy expression for all rows in the sub-query .

ANY🡪 must satisfy expression at least one row in the query.

IN🡪 equivalent to ‘= ANY ()’.

EXISTS🡪at least one row is returned.

# Window function :-

Performs a “sliding” calculation across a set of tuples that are related ,like an aggregation but tuples are not grouped into single output tuples .

**The Over KEY specifies how to group together tuples when computing the window function use PARTITION BY to specify group.**

# Common Table Expression:-

Provides away to write auxiliary statements for use in language .