SQL Built-In Methods/Functions

* Count(\*)
  + Returns the no of rows in a table
* SUM(Quantity)
* AVG(Quantity)
* GROUP BY
  + A column needs to be present in both tables to group by that column.

GROUPING RULES

* Every non-aggregate column that appears in the SELECT clause must also appear in the GROUP BY clause.
* You may not use aliases in the HAVING clause. (if count has a alias numemployees then it’s not valid)
* You may use aliases in the ORDER BY clause.
* You may only use calculated fields in the HAVING clause.
* You may use calculated field aliases or actual fields in the ORDER BY clause.

DISTINCT

* To get one row per city.

Joins

* Types
  + Inner Join
  + Left Join
  + Right Join
  + Union
  + Cross Join
  + Equi Join

EQUI JOIN VS INNER JOIN

* Inner join can have equality (=) and other operators (like <, >, <>) in the join condition.
* Equi join only have equality (=) operator in the join condition.
* Equi join can be an inner join, Left Outer join, Right Outer join.

Working with NULL values

* IsNull()

Transaction

* Is a unit of work where you can bundle multiple dml statements.

Joins

|  |  |
| --- | --- |
| Select …. From emp join dept using deptno | Equi join |
| Select …. From emp using dept on (e.deptno > d.deptno) | Non-Equi |
| Select …. From emp left | Left outer join |
|  | Right outer join |
|  | Full outer join |

Joins vs Sub Queries

* Joins are always faster than Sub Queries.
* Sub Queries can not be avoided all together.
* Where condition slows down the query performance.
* In Joins the Query engine uses the optimum path to retrieve data.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Performance Tips   * A single select statement cannot have more than 40 columns. * Not to join more than 4 tables. * Always use joins instead of sub queries.  |  |  |  |  | | --- | --- | --- | --- | |  | Table A | Table B |  | | Join | 4 | 4 | 4 + 4 | | Sub query | 4 | 4 | 4 \* 4 | |

Writing a join with multiple conditions (Always start from bigger to smaller tables)

|  |  |  |
| --- | --- | --- |
| Tables |  |  |
| A(65000) | a |  |
| B(10) |  |  |
| C(80000) |  |  |
| D(5000) |  |  |
| E(20) |  |  |

JOIN ORDER

JOIN from Big Tables to Small

Then

….pic

Scalar

Select \* from gives a single output called scalar

Correlated Subquery

If a query uses a column from an outer query then it is correlated query.

The only place where join cannot be used instead of sub queries is,

* Calling Top records in a sorted table
* Because top cannot be done without sorting the table first

…Pics

ORDER BY clause is invalid in subqueries

* The ORDER BY clause is invalid in views, inline functions, derived tables, subqueries, and common table expressions, unless TOP, OFFSET or FOR XML is also specified.
* Will not work,
  + Select \* from Orders where freight in (select freight from orders order by Freight desc);
* Will work,
  + Select \* from Orders where freight in (select top 3 freight from orders order by Freight desc);

Inline or Derived Table

* Select min(BirthDate) From (select top 3 BirthDate from employees order by BirthDate desc) a;

Questions

* How to find the second / third or the nth max salary receiver from the Employee Table
* How to find the second / third or the nth Oldest person from the Employee Table

|  |
| --- |
| Notes:\*   * Top N Analysis cannot be done by Joins. * It can be done only by using Sub Queries. |

Indexing

* Types of Index
  + Clustered Index
    - Only one per Table
    - Based on Primary Key or Unique Key
  + Non–Clustered Index
    - Max 249 per table.
    - Index on Normal Columns.
    - Most used columns in the where clause.
    - Created for Master tables.
* Indexing is a mechanism to access a value / data with out performing a full table scan.
* When to have an index?
  + Do we need index for a very small table.
  + Do we need an index when we are reading all the records.
* Index should be created only on very large tables, where the requirement is to get a few records from the table with out performing a full table scan.
* Index are not required on small tables, or when the requirement is to fetch the entire records from the table.

SEE PIC

* While searching based on an indexed column at least the first char must be known.
* Useful for transaction table, where records get deleted frequently (de-fragmentation).
* Don’t create too many indexes on the same column.