|  |  |
| --- | --- |
| **DBMS** | RDBMS |
| * DBMS stands for Database Management System. * DBMS is a system software responsible for the creation, retrieval, updation and management of the database. It ensures that our data is consistent, organized and is easily accessible by serving as an interface between the database and its end users or application softwares. | * RDBMS stands for Relational Database Management System. * RDBMS stores data in the form of a collection of tables and relations can be defined between the common fields of these tables. Most modern database management systems like MySQL, Microsoft SQL Server, Oracle, IBM DB2 and Amazon Redshift are based on RDBMS. |

|  |  |
| --- | --- |
| **SQL** | **MySQL** |
| * SQL stands for Structured Query Language. It is the standard language for relational database management systems. * It is especially useful in handling organized data comprised of entities (variables) and relations between different entities of the data. * It is used for retrieving and manipulating structured databases. | * MySQL is a relational database management system, like SQL Server, Oracle or IBM DB2, that is used to manage SQL databases. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DQL**  (Data Query Language ( or) Data Retrieval Language ) | **DML**  (Data Manipulation Language) | **DDL**  (Data Definition Language) | **DCL**  (Data Control Language) | **TCL**  (Transaction Control Language) |
| * SELECT Statement | * INSERT Statement * UPDATE Statement * DELETE Statement | * CREATE Statement * ALTER Statement * DROP Statement * RENAME Statement * TRUNCATE Statement | * GRANT Statement * REVOKE Statement | * COMMIT Statement * ROLLBACK Statement * SAVEPOINT Statement |

## Different subsets of SQL :

## *DDL (Data Definition Language) :*

* It allows you to perform various operations on the database such as CREATE, ALTER and DELETE objects.

***DML******( Data Manipulation Language)***  :

* It allows you to access and manipulate data. It helps you to insert, update, delete and retrieve data from the database.

***DCL ( Data Control Language)*** :

* It allows you to control access to the database. Example – Grant, Revoke access permissions.

|  |  |
| --- | --- |
| Relational *Database Management System* | Non-Relational Database Management System |
| * The data is stored in relations (tables). Example – MySQL. | * There is no concept of relations, tuples and attributes.  Example – Mongo |

### Constraints in SQL :

* Constraints are used to specify the limit on the data type of the table. It can be specified while creating or altering the table statement.

There are two levels :

1. column level constraint
2. table level constraint

* **NOT NULL** - Restricts NULL value from being inserted into a column.
* **CHECK** - Verifies that all values in a field satisfy a condition.
* **DEFAULT** - Automatically assigns a default value if no value has been specified for the field.
* **UNIQUE** - Ensures unique values to be inserted into the field.
* **INDEX** - Indexes a field providing faster retrieval of records.
* **PRIMARY KEY** - Uniquely identifies each record in a table.
* **FOREIGN KEY** - Ensures referential integrity for a record in another table.

**unique constraint :**

* **Unique constraints** ensure that all the values in a column are different.

### Primary key :

* A Primary key is a column (or collection of columns) or a set of columns that uniquely identifies each row in the table.
* Uniquely identifies a single row in the table.
* Null values not allowed

## Foreign key :

* Foreign key maintains referential integrity by enforcing a link between the data in two tables.
* The foreign key in the child table references the primary key in the parent table.
* The foreign key constraint prevents actions that would destroy links between the child and parent tables.

### Difference between primary key and unique key :

* Primary key carries unique value but the field of the primary key cannot be Null on the other hand unique key also carry unique value but it can have a single Null value field.

### Composite primary key :

* The key which is created on multiple columns in a table is generally considered as the Composite primary key. However, it is not always necessary that all of them have the same meaning.

## Unique key :

* Uniquely identifies a single row in the table.
* Multiple values allowed per table.
* Null values allowed.

## Joins in SQL :

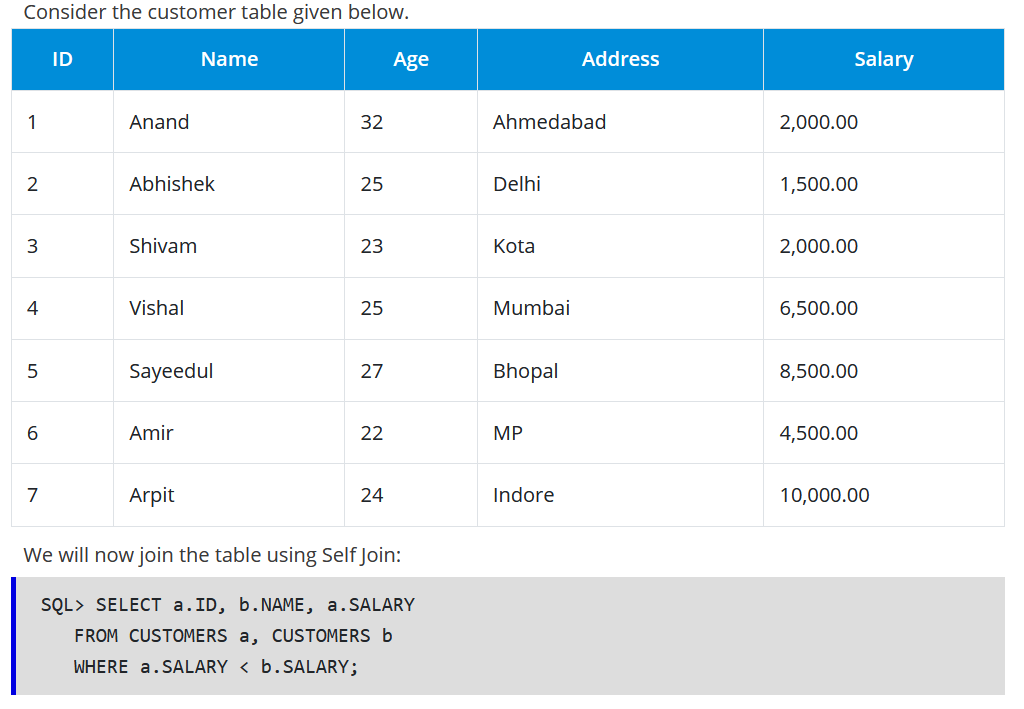
* A JOIN clause is used to combine rows from two or more tables, based on a related column between them. It is used to merge two tables or retrieve data from there.

**Four joins in SQL**:

* **(INNER) JOIN**: Retrieves records that have matching values in both tables involved in the join. This is the widely used join for queries.
* **LEFT (OUTER) JOIN**: Retrieves all the records/rows from the left and the matched records/rows from the right table.
* **RIGHT (OUTER) JOIN**: Retrieves all the records/rows from the right and the matched records/rows from the left table.
* **FULL (OUTER) JOIN**: Retrieves all the records where there is a match in either the left or right table.

### Self-Join :

* A self JOIN is a case of regular join where a table is joined to itself based on some relation between its own column(s). Self-join uses the INNER JOIN or LEFT JOIN clause and a table alias is used to assign different names to the table within the query.



### Cross-Join :

* Cross join can be defined as a cartesian product of the two tables included in the join. The table after join contains the same number of rows as in the cross-product of number of rows in the two tables. If a WHERE clause is used in cross join then the query will work like an INNER JOIN.

### Difference between cross join and natural join :

* The cross join produces the cross product or Cartesian product of two tables.
* The natural join is based on all the columns having the same name and data types in both the tables.

**Super Key :**

* A column or a combination of columns which uniquely identify a record in a table is called a Super Key.

**Difference between CHAR and VARCHAR2 :**

CHAR is used to store fixed-length character strings, and VARCHAR2 is used to store variable-length character strings.

|  |  |
| --- | --- |
| Entities | Relationships |
| * A person, place, or thing in the real world about which data can be stored in a database. Tables store data that represents one type of entity. For example – A bank database has a customer table to store customer information. Customer table stores this information as a set of attributes (columns within the table) for each customer. | * Relation or links between entities that have something to do with each other. For example – The customer name is related to the customer account number and contact information, which might be in the same table. There can also be relationships between separate tables (for example, customer to accounts). |

**There are 4 types of relationships:**

* One to One Relationship
* Many to One Relationship
* Many to Many Relationship
* One to Many Relationship

### Index :

* An index refers to a performance tuning method of allowing faster retrieval of records from the table.
* An index creates an entry for each value and hence it will be faster to retrieve data.

### Types of index :

### ****Unique Index:****

This index does not allow the field to have duplicate values if the column is unique indexed. If a primary key is defined, a unique index can be applied automatically.

### ****Clustered Index:****

This index reorders the physical order of the table and searches based on the basis of key values. Each table can only have one clustered index.

### ****Non-Clustered Index:****

Non-Clustered Index does not alter the physical order of the table and maintains a logical order of the data. Each table can have many nonclustered indexes.

* Unique Index -> Normal index
* Clustered Index -> Composite index
* NonClustered Index -> B-tree index
* Bit-Map index -> function based index

|  |  |
| --- | --- |
| **Clustered Index** | **Non-Clustered Index** |
| * It is used for easy retrieval of data from the database and its faster * Clustered index alters the way records are stored in a database as it sorts out rows by the column * One table can only have one clustered index | * reading from non clustered index is relatively slower. * it does not alter the way it was stored but it creates a separate object within a table which points back to the original table rows after searching. * It can have many non clustered index |

|  |  |
| --- | --- |
| **DELETE** | **TRUNCATE** |
| Delete command is used to delete a row in a table. | Truncate is used to delete all the rows from a table. |
| You can rollback data after using delete statement. | You cannot rollback data. |
| It is a DML command. | It is a DDL command. |
| It is slower than truncate statement. | It is faster. |

|  |  |
| --- | --- |
| **DROP** | **TRUNCATE** |
| DROP removes the entire table from the database and it also cannot be retrieved back. | TRUNCATE removes all rows from the table which cannot be retrieved back |

### ****Normalization :****

* Normalization is the process of organizing data to avoid duplication and redundancy.

### ****Types of Normalization :****

* *First Normal Form (1NF)* – No repeating groups within rows
* *Second Normal Form (2NF)* – Every non-key (supporting) column value is dependent on the whole primary key.
* *Third Normal Form (3NF)* – Dependent solely on the primary key and no other non-key (supporting) column value.

**primary use of Normalization :**

* Normalization is mainly used to add, delete or modify a field that can be made in a single table.
* The primary use of Normalization is to remove redundancy and to remove the insert, delete and update distractions.
* Normalization breaks the table into small partitions and then link them using different relationships so that it will avoid the chances of redundancy.

**Disadvantages of not performing Database Normalization :**

The major disadvantages are:

* The occurrence of redundant terms in the database which causes the waste of the space in the disk.
* Due to redundant terms inconsistency may also occur id any change will be made in the data of one table but not made in the same data of another table then inconsistency will take place, which will lead to the maintenance problem and effects the ACID properties as well.

**Inconsistent dependency :**

* Inconsistent dependency refers to the difficulty of accessing particular data as the path to reach the data may be missing or broken.
* Inconsistent dependency will leads users to search the data in the wrong table which will afterward give the error as an output.

**Denormalization in a Database**

* Denormalization is used to access the data from higher or lower normal form of database. It also processes redundancy into a table by incorporating data from the related tables.
* Denormalization adds required redundant term into the tables so that we can avoid using complex joins and many other complex operations.
* Denormalization doesn?t mean that normalization will not be done, but the denormalization process takes place after the normalization process.

### ****“Trigger” in SQL :****

* Trigger in SQL is are a special type of stored procedures that are defined to execute automatically in place or after data modifications.
* It allows you to execute a batch of code when an insert, update or any other query is executed against a specific table.

### ****Nested Triggers :****

* Triggers may implement data modification logic by using INSERT, UPDATE, and DELETE statements.
* These triggers that contain data modification logic and find other triggers for data modification are called Nested Triggers.

### ****NULL values same as that of zero or a blank space :****

* A NULL value is not at all same as that of zero or a blank space.
* NULL value represents a value which is unavailable, unknown, assigned or not applicable whereas a zero is a number and blank space is a character.

### ****Subquery in SQL :****

* A subquery is a query inside another query where a query is defined to retrieve data or information back from the database.
* In a subquery, the outer query is called as the main query whereas the inner query is called subquery.
* Subqueries are always executed first and the result of the subquery is passed on to the main query.
* It can be nested inside a SELECT, UPDATE or any other query.
* A subquery can also use any comparison operators such as >,< or =.
* There are 3-row comparison operators that are used in subqueries such as IN, ANY and ALL.

### ****Different types of a subquery :****

**Correlated subquery**: These are queries which select the data from a table referenced in the outer query. It is not considered as an independent query as it refers to another table and refers the column in a table.

**Non-Correlated subquery**: This query is an independent query where the output of subquery is substituted in the main query.

|  |  |
| --- | --- |
| **BETWEEN** | **IN** |
| BETWEEN operator is used to display rows based on a range of values in a row | IN condition operator is used to check for values contained in a specific set of values. |

### ****Stored Procedure** :**

* A Stored Procedure is a function which consists of many SQL statements to access the database system.
* Several SQL statements are consolidated into a stored procedure and execute them whenever and wherever required which saves time and avoid writing code again and again.

### ****Advantages****:

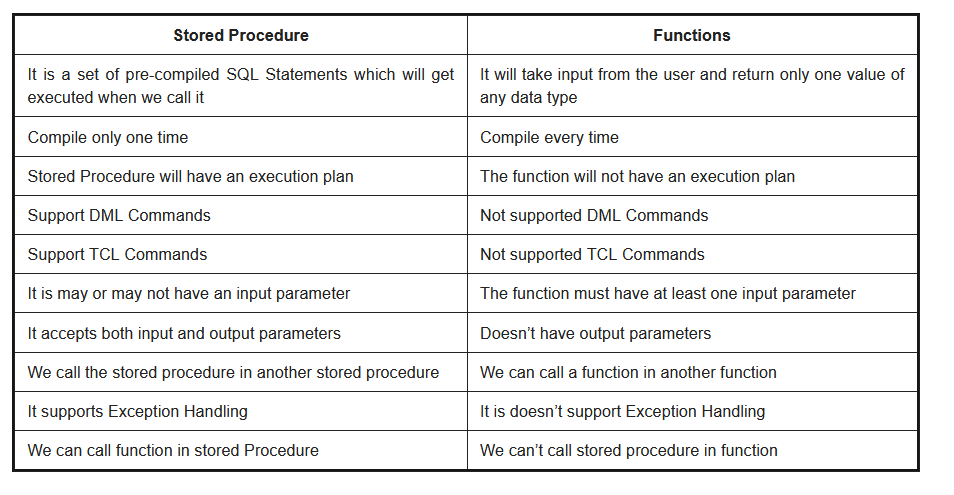
A Stored Procedure can be used as a modular programming which means create once, store and call for several times whenever it is required. This supports faster execution. It also reduces network traffic and provides better security to the data.

### ****Disadvantage****:

The only disadvantage of Stored Procedure is that it can be executed only in the database and utilizes more memory in the database server.

**Recursive Stored Procedure :**

* A stored procedure which calls itself until a boundary condition is reached, is called a recursive stored procedure.
* This recursive function helps the programmers to deploy the same set of code several times as and when required. Some SQL programming languages limit the recursion depth to prevent an infinite loop of procedure calls from causing a stack overflow, which slows down the system and may lead to system crashes.



### ****Local and Global variables** :**

### ****Local variables:****

These variables can be used or exist only inside the function. These variables are not used or referred by any other function.

### ****Global variables:****

These variables are the variables which can be accessed throughout the program. Global variables cannot be created whenever that function is called.

### ****Transactions and their controls :****

* A transaction can be defined as the sequence task that is performed on databases in a logical manner to gain certain results.
* Operations like Creating, updating, deleting records performed in the database come from transactions.

**There are 4 transaction controls such as**

* **COMMIT**: It is used to save all changes made through the transaction.
* **ROLLBACK**: It is used to roll back the transaction. All changes made by the transaction are reverted back and the database remains as before.
* **SET TRANSACTION**: Set the name of the transaction.
* **SAVEPOINT:** It is used to set the point where the transaction is to be rolled back.

### ****Properties of a transaction :****

these properties are referred to as ACID properties. They are:

1. Atomicity
2. Consistency
3. Isolation
4. Durability.

### ****Aggregate functions :****

SQL Aggregate functions determine and calculate values from multiple columns in a table and return a single value.

**There are 7 aggregate functions in SQL:**

* **AVG():** Returns the average value from specified columns.
* **COUNT():** Returns number of table rows.
* **MAX():** Returns the largest value among the records.
* **MIN():** Returns smallest value among the records.
* **SUM():** Returns the sum of specified column values.
* **FIRST():** Returns the first value.
* **LAST():** Returns last value.

### ****Scalar functions :****

Scalar functions are used to return a single value based on the input values.

**Scalar Functions are as follows:**

* **UCASE():** Converts the specified field in the upper case.
* **LCASE():** Converts the specified field in lower case.
* **MID():** Extracts and returns character from the text field.
* **FORMAT():** Specifies the display format.
* **LEN():** Specifies the length of the text field.
* **ROUND():** Rounds up the decimal field value to a number.
* CONCAT: join two or more values together.
* SUBSTR: used to extract the string of specific length.
* LENGTH: return the length of the string in numerical value.
* INSTR: find the exact numeric position of a specified character.
* LPAD: padding of the left-side character value for right-justified value.
* RPAD: padding of right-side character value for left-justified value.
* TRIM: remove all the defined character from the beginning, end or both beginning and end.
* REPLACE: replace a specific sequence of character with other sequences of character.

### ****Working of SQL Privileges :****

SQL GRANT and REVOKE commands are used to implement privileges in SQL multiple user environments. The administrator of the database can grant or revoke privileges to or from users of database objects by using commands like SELECT, INSERT, UPDATE, DELETE, ALL, etc.

**GRANT Command**: This command is used to provide database access to users other than the administrator.

**REVOKE command**: This command is used to provide database deny or remove access to database objects.

### ****SQL Injection :****

* SQL Injection is a type of database attack technique where malicious SQL statements are inserted into an entry field of database in a way that once it is executed, the database is exposed to an attacker for the attack.
* This technique is usually used for attacking data-driven applications to have access to sensitive data and perform administrative tasks on databases.

### ****use of the NVL function :****

* NVL function is used to convert the null value to its actual value.

|  |  |
| --- | --- |
| **NESTED SUBQUEREY** | **CORRELATED SUBQUEREY** |
| * Subquery within another subquery is called Nested Subquery. | * If the output of a subquery depends on column values of the parent query table then the query is called Correlated Subquery. |

### ****Collation :****

* Collation is a set of rules that check how the data is sorted by comparing it. Such as character data is stored using correct character sequence along with case sensitivity, type, and accent.

### ****Database White Box Testing :****

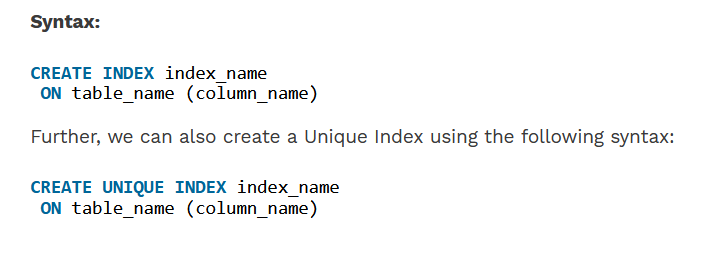
* Database Consistency and ACID properties
* Database triggers and logical views
* Decision Coverage, Condition Coverage, and Statement Coverage
* Database Tables, Data Model, and Database Schema
* Referential integrity rules

### ****Database Black Box Testing :****

* Data Mapping
* Data stored and retrieved
* Use of Black Box testing techniques such as Equivalence Partitioning and Boundary Value Analysis (BVA)

### ****Indexes in SQL :****

* The index can be defined as the way to retrieve the data more quickly. We can define indexes using CREATE statements.



* **MINUS** – returns all distinct rows selected by the first query but not by the second.
* **UNION** – returns all distinct rows selected by either query
* **UNION ALL** – returns all rows selected by either query, including all duplicates.
* **INTERSECT** – returns all distinct rows selected by both queries.

### ****Referential Integrity :****

* Set of rules that restrict the values of one or more columns of the tables based on the values of the primary key or unique key of the referenced table.

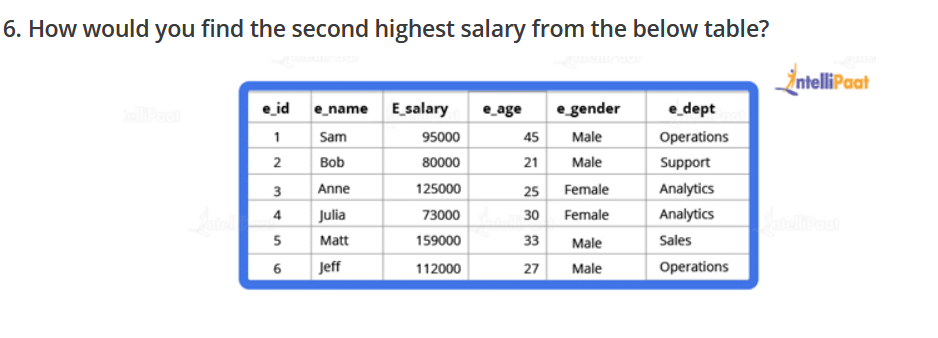
|  |  |
| --- | --- |
| **RENAME** | **ALIAS** |
| * Rename is a permanent name given to a table or column | * Alias is a temporary name given to a table or column. |

|  |  |
| --- | --- |
| **HAVING clause** | **WHERE clause** |
| * specify a search condition * Having clause is used only with the SELECT statement and typically used with GROUP BY clause. * Column operations * Aggreagte functions can have them * Cannot be used without a SELECT statement * Groupby comes before the having clause | * specify a search condition * If GROUP BY clause is not used then Having behaved like WHERE clause only. * Row operations * Aggreagte functions cannot have them * Can be used with SELECT, UPDATE, and DELETE * Groupby Comes after the WHERE clause |

### ****CTE :****

* A CTE or common table expression is an expression that contains temporary result set which is defined in a SQL statement.

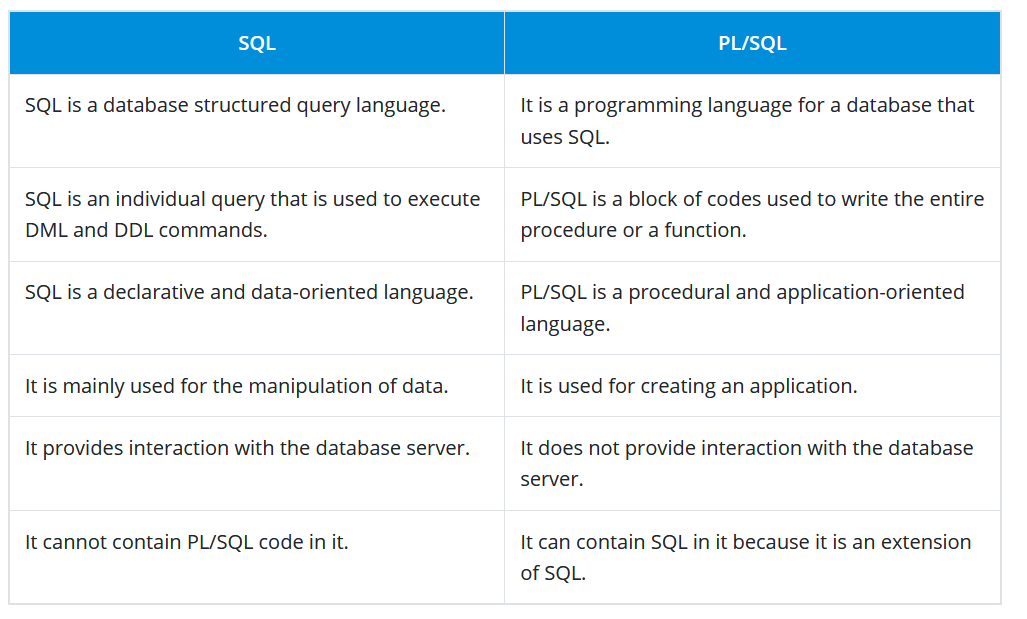
|  |  |
| --- | --- |
| **primary key** | **unique key** |
| * Primary key carries unique value but the field of the primary key cannot be Null | * unique key also carry unique value but it can have a single Null value field. |



Code :

select \* from employee

select max(e\_salary) from employee where e\_salary not in (select max(e\_salary) from employee)



**AUTO\_INCREMENT :**

* AUTO\_INCREMENT is used in SQL to automatically generate a unique number whenever a new record is inserted into a table.
* Since the primary key is unique for each record, we add this primary field as the AUTO\_INCREMENT field so that it is incremented when a new record is inserted.
* The AUTO-INCREMENT value is by default starts from 1 and incremented by 1 whenever a new record is inserted.

**Syntax:**

CREATE TABLE Employee(

Employee\_id int NOT NULL AUTO-INCREMENT,

Employee\_name varchar(255) NOT NULL,

Employee\_designation varchar(255)

Age int,

PRIMARY KEY (Employee\_id)

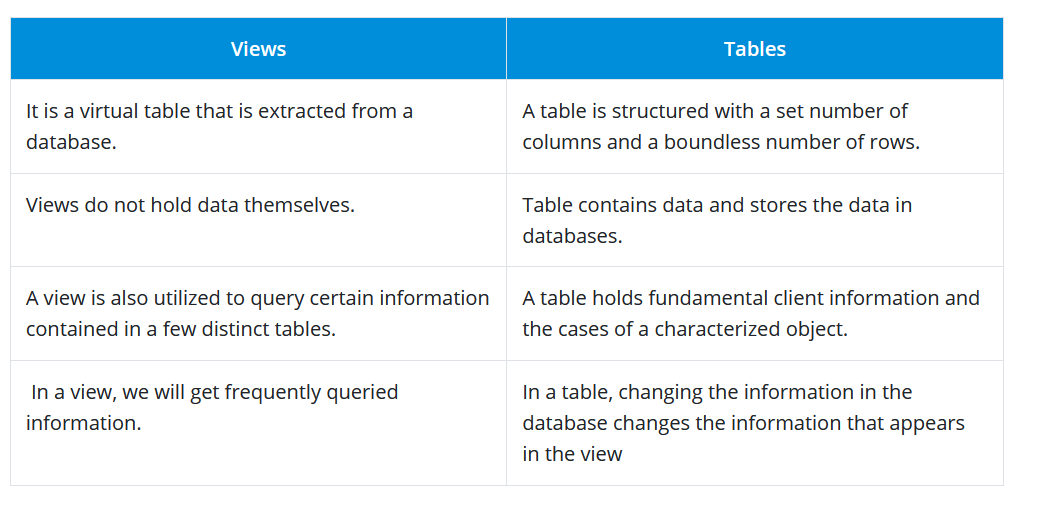
)

**COALESCE function :**

* COALESCE function takes a set of inputs and returns the first non-null value.

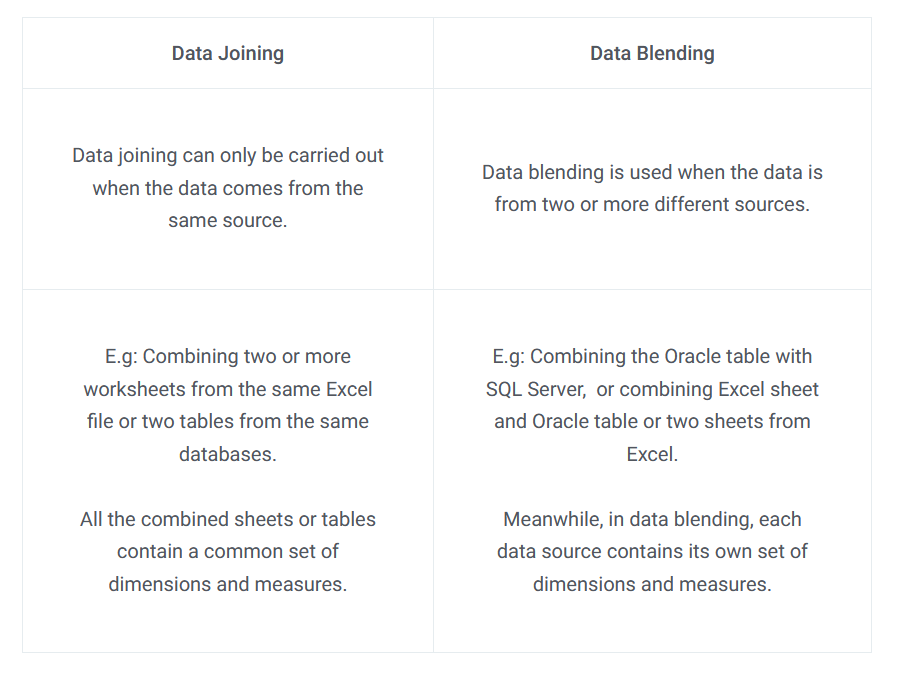
**stuff() function :**

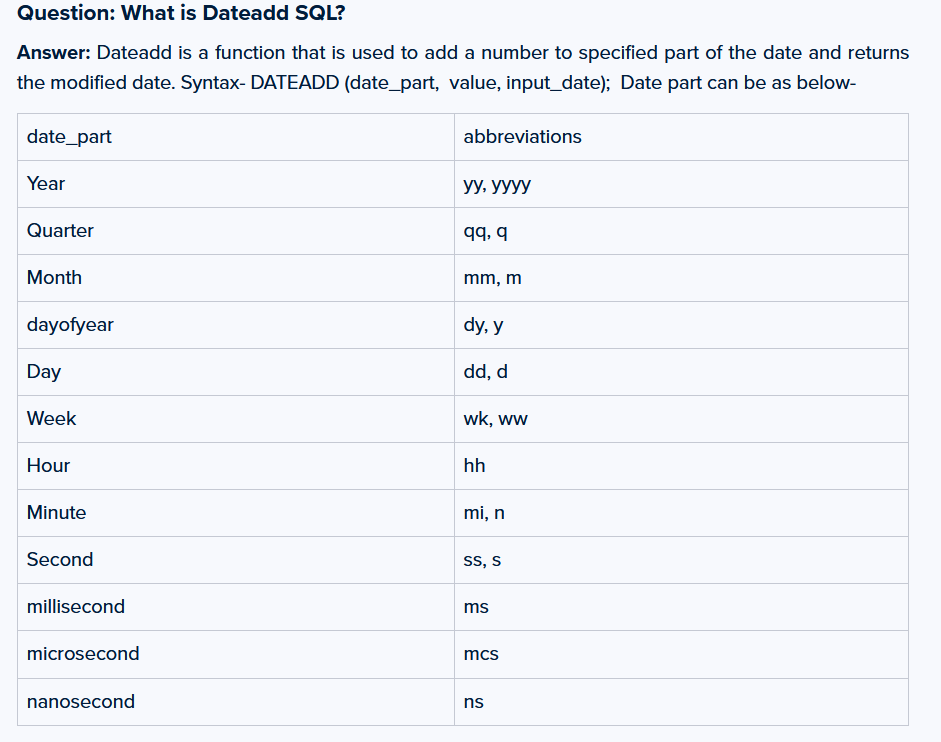
* The stuff function deletes a part of the string and then inserts another part into the string starting at a specified position.



|  |  |
| --- | --- |
| **OLTP** | **OLAP** |
| * **Online Transaction Processing** * The OLTP system often follows decentralized planning to keep away from single points of failure. This system is generally designed for a large audience of end-users to perform short transactions. * Also, queries involved in such databases are generally simple, need fast response time, and in comparison, it returns only a few records. * So, the number of transactions per second acts as an effective measure for those systems. | * **Online Analytical Processing** * **I**t is a category of software programs that are identified by a comparatively lower frequency of online transactions. * For OLAP systems, the efficacy computing depends highly on the response time. * Hence, such systems are generally used for data mining or maintaining aggregated historical data, and they are usually used in multi-dimensional schemas. |

|  |  |
| --- | --- |
| **UNION** | **UNION ALL** |
| * **Union** operator is used to combine the result set of two or more select statements. * if there is a record present in both tables, then we will get only one of them in the final result. | * **Union All** gives all the records from both tables including the duplicates. |





<https://www.softwaretestinghelp.com/50-popular-sql-interview-questions-for-testers/>

<https://www.javatpoint.com/sql-interview-questions>

<https://mindmajix.com/sql-interview-questions>

<https://intellipaat.com/blog/interview-question/sql-interview-questions/>

<https://hackr.io/blog/top-sql-interview-questions>

<https://www.complexsql.com/complex-sql-queries-examples-with-answers/complex-sql-for-interviews/>

<https://mindmajix.com/sql-interview-questions>

<https://mindmajix.com/sql-server-interview-questions-for-2-5-years>

https://www.interviewbit.com/sql-interview-questions/