SQL

DBMS	RDBMS
1. DBMS stores data as file.	RDBMS stores data in **TABULAR** form.
2. No relationship between data.	Data is stored in the form of tables which are **RELATED** to each other. Eg: Foreign key relationship.
3. Normalization is not present.	**NORMALIZATION** is present.
4. It deals with small quantity of data.	It deals with **LARGE** amount of data.
5. Examples: XML	Examples: MySQL, PostgreSQL, SQL Server, Oracle, Microsoft Access etc.

What is a Constraint is SQL? What are its types. `V. IMP.`

❖ SQL constraints are **used to specify rules for the data in a table**.

1. A **PRIMARY KEY** is a field which can uniquely identify each row in a table.

2. **NOT NULL** constraint tells that we cannot store a null value in a column.

3. A **FOREIGN KEY** is a field which can uniquely identify each row in an another table.

4. **CHECK** constraint helps to validate the values of a column to meet a particular condition.

5. **DEFAULT** constraint specifies a default value for the column when no value is specified by the user.

6. **UNIQUE** constraint tells that all the values in the column must be unique.

```
CREATE TABLE Students (
    ID int NOT NULL PRIMARY KEY,
    Name varchar(255) NOT NULL,
    CourseID int FOREIGN KEY REFERENCES Courses(CourseID),
    Age int NOT NULL CHECK (AGE >= 18),
    AdmissionDate date DEFAULT GETDATE(),
    CONSTRAINT UC_Student UNIQUE (ID,Name)
);
```

What is the difference between Primary key and Unique key?

	Primary Key	Unique Key
1	Primary Key Can't Accept Null Values.	Unique Key Can Accept Only One Null Value
2	Creates Clustered Index	Creates Non-Clustered Index
3	Only One Primary key in a Table	More than One Unique Key in a Table.

SQL Se

- Triggers are stored programs, which are **AUTOMATICALLY** executed or fired when some events (insert, delete and update) occur.

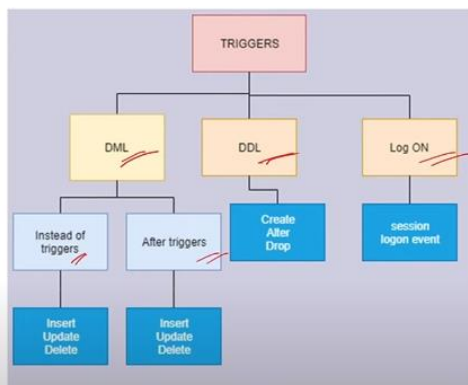- Locations   ->   LocationHist

- Example of After(DML) Trigger

```
CREATE TRIGGER TR_UPD_Locations ON (Locations)    → Table Name
FOR UPDATE                              → Trigger Name
NOT FOR REPLICATION      → DML Event
AS

BEGIN                                   → T-SQL block that runs
    INSERT INTO LocationHist              against specified DML
    SELECT LocationID                     Event
         ,getdate()
    FROM inserted
END
```

- In after trigger, update on the table executed first and then trigger will run.

TRIGGERS

DML   DDL   Log ON

Instead of   After triggers   Create Alter Drop   session logon event
triggers

Insert Update Delete   Insert Update Delete

SUBSCR

- Example of Instead of(DML) Trigger

```
1
2  CREATE TRIGGER [dbo].[TRG_VM_EMPDETAILS]          Trigger Name
3    ON [dbo].[VW_EMPDETAILS]
4    INSTEAD OF INSERT                                View Name
5    AS
6  BEGIN
7    -- LOGIC HERE                                    INSTEAD OF Trigger
8    |
9  END
10
11
12    This trigger is only for INSERT
13
```

- In INSTEAD OF trigger, you can skip an INSERT, DELETE or UPDATE statement to a table and execute other statements defined in the trigger instead.

SQL Ser

- A view is a VIRTUAL table which consists of a subset of data contained in a table or more than one table.

CREATE VIEW [India-Customers] AS

SELECT CustomerName, ContactName

FROM Customers

WHERE Country = 'India';

Table A

Table B

Rows

Query

View

Columns
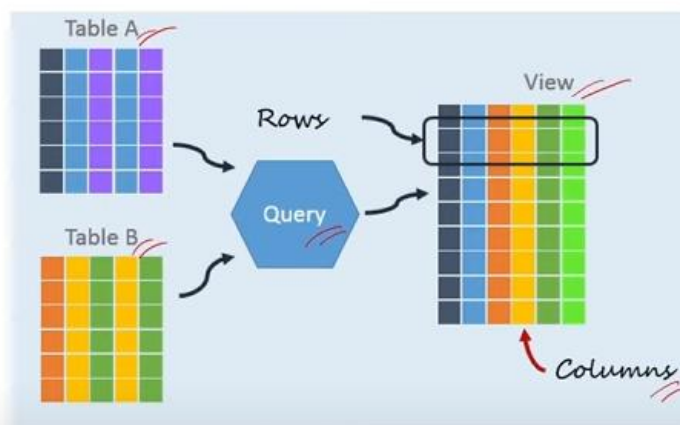
- Views are not stored in memory like tables then why to use views.

1. Indexed Views to improve the performance.

2. Extra security – DBA can hide the actual table names and expose views for Read operations.

- Remember, in a view query is stored but the data is never stored like a table.

SUBSCRIBE

---

1. WHERE Clause is used before GROUP BY Clause.

HAVING Clause is used after GROUP BY Clause.

```
SELECT COUNT(CustomerID), Country
FROM Customers
WHERE Country = "India"
GROUP BY Country
HAVING COUNT(CustomerID) > 5;
```

2. WHERE Clause cannot contain AGGREGATE function.

HAVING Clause can contain aggregate function.

```
SELECT EmpName FROM Employee
GROUP BY EmpName
HAVING SUM(EmpSalary) <30000
```

What is Sub query or Nested query or Inner query in SQL?

❖ A Subquery or Inner query or a Nested query is a query within another SQL query and **embedded** within the **WHERE** clause.

```
SELECT  LastName, FirstName          ┌─────────────┐
                                     │ Outer Query │
FROM    Employees                    └─────────────┘

WHERE   OfficeCode IN (SELECT OfficeCode
                       FROM   Offices        ┌─────────────┐
                       WHERE Country = 'USA');│ Inner Query │
                                             └─────────────┘
```

What is Auto Increment/ Identity column in SQL Server?

❖ Auto-increment allows a unique number to be **generated automatically** when a new record is inserted into a table.

❖ Mostly it is the primary key only.

```
CREATE TABLE Persons (
                                1,2,3 ...
    Personid int IDENTITY(1,1) PRIMARY KEY,

    LastName varchar(255) NOT NULL,

    FirstName varchar(255),

    Age int

);
```

What is the difference between Delete, Truncate and Drop commands? V. IMP.

DELETE	TRUNCATE	DROP
1. It is a DML.	1. It is a DDL.	1. It is a DDL.
2. It is used to delete one or all rows from the table based on where condition, but it will not delete schema.	2. It is used to delete all rows from the table, but it will not delete schema.	2. It is used to delete all rows from the table with structure/schema.
3. It can be rollback.	3. It can not be rollback.	3. It can not be rollback.

```
DELETE FROM Employees     TRUNCATE TABLE Employees;     DROP TABLE Employees;
WHERE Emp_Id = 7;
```
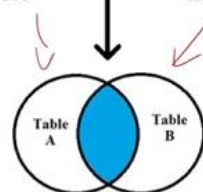
What are Joins in SQL?

❖ A join clause is used to COMBINE rows from two or more tables, based on a related column between them.

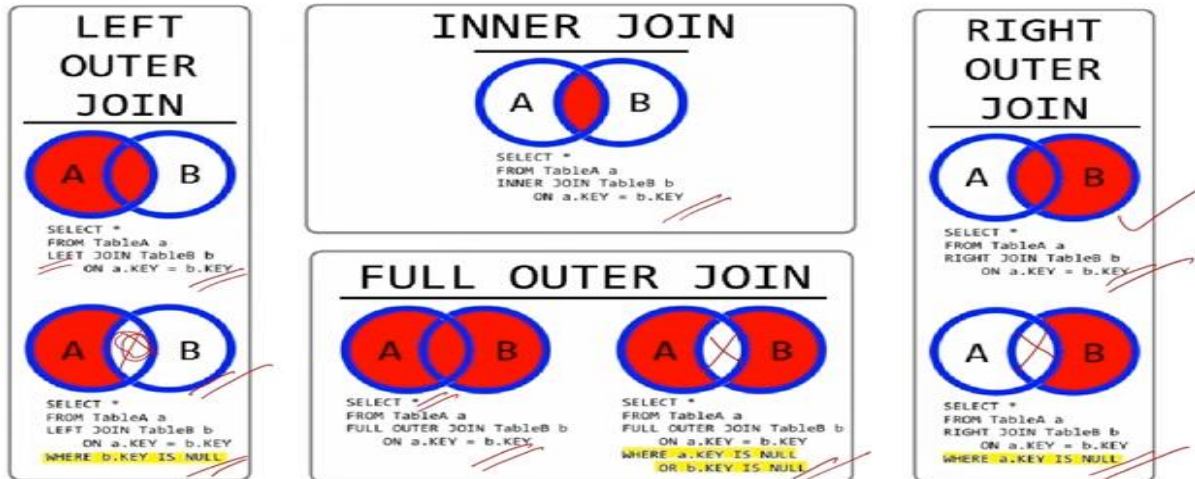| Student ID	Name
1001	A
1002	B
1003	C
1004	D

Table A

Student ID	Department
1004	Mathematics
1005	Mathematics
1006	History
1007	Physics
1008	Computer Science

Table B

Student ID	Name	Department
1004	D	Mathematics

LEFT OUTER JOIN

A ● B

```
SELECT *
FROM TableA a
LEFT JOIN TableB b
  ON a.KEY = b.KEY
```

A ● B

```
SELECT *
FROM TableA a
LEFT JOIN TableB b
  ON a.KEY = b.KEY
WHERE b.KEY IS NULL
```

INNER JOIN

A ● B

```
SELECT *
FROM TableA a
INNER JOIN TableB b
  ON a.KEY = b.KEY
```

FULL OUTER JOIN

A ● B

```
SELECT *
FROM TableA a
FULL OUTER JOIN TableB b
  ON a.KEY = b.KEY
```

A ● B

```
SELECT *
FROM TableA a
FULL OUTER JOIN TableB b
  ON a.KEY = b.KEY
WHERE a.KEY IS NULL
  OR b.KEY IS NULL
```

RIGHT OUTER JOIN

A ● B

```
SELECT *
FROM TableA a
RIGHT JOIN TableB b
  ON a.KEY = b.KEY
```

A ● B

```
SELECT *
FROM TableA a
RIGHT JOIN TableB b
  ON a.KEY = b.KEY
WHERE a.KEY IS NULL
```

What is Self-Join?

❖ A self join is **a join of a table to itself**.

❖ When to use Self Join??

employees

* employee_id
first_name
email
phone_number
hire_date
job_id
salary
manager_id
department_id

→ This manager id is employee id of the manager of an employee

❖ Now your task is to get the employees name with their manager names??

```
SELECT
    e.first_name AS employee,
    m.first_name AS manager
FROM
    employees e LEFT JOIN
    employees m
    ON m.employee_id = e.manager_id
ORDER BY manager;
```

employee	manager
Steven King	NULL
Bruce Ernst	Alexander Hunold
David Austin	Alexander Hunold
Valli Pataballa	Alexander Hunold
Diana Lorentz	Alexander Hunold
Alexander Khoo	Den Raphaely

What are Indexes in SQL Server?

❖ SQL Indexes are used in relational databases to retrieve data **VERY FAST**.

Index 't1'.'fld1'		Table 't1'		
idx		fld1	...	fldN
3	→	12	...	...
6		3		
12		27	...	...
27	→	12	...	...
		6	...	...
		3	...	...
		3	...	...
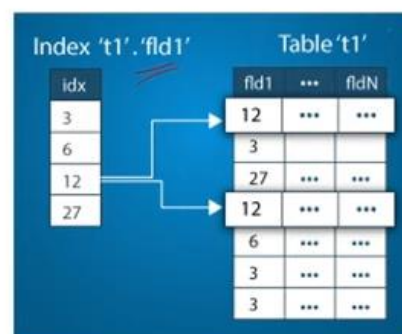
❖ They are similar to indexes at the start of the BOOKS, which purpose is to find a topic quickly.

Table of Contents

What is Clustered index?

❖ A clustered index defines the order in which data is **physically** stored in a table.

❖ Table data can be sorted in only way, therefore, there can be only one clustered index per table.

❖ In SQL Server, if you set a primary key on a column then it will automatically create a clustered index on that particular column.

Dictionary

What is Non-Clustered index?

❖ A non-clustered index is stored at one place and table data is stored in another place.

❖ A table can have multiple non-clustered index in a table.

Book Index

What is the difference between Clustered and Non-Clustered index? **V. IMP.**

1. A clustered index defines the order in which data is **physically** stored in a table. For example, Dictionary.

A non-clustered index is stored at one place and table data is stored in another place. For example, Book Index.

2. A table can have only one clustered index.

A table can have multiple non-clustered index.

3. Clustered index is faster.

Non-clustered index is slower.

How to create Clustered and Non-Clustered index in a table?

❖ CLUSTERED INDEX

```
CREATE CLUSTERED INDEX <index_name>
ON <table_name>(<column_name> ASC/DESC)
```

❖ NON-CLUSTERED INDEX

```
CREATE NONCLUSTERED INDEX <index_name>
ON <table_name>(<column_name> ASC/DESC)
```

❖ When you create a PRIMARY KEY constraint, a clustered index on the column or columns is automatically created.

In which column you will apply the indexing to optimize this query. "select id, class from student where name="happy""?

select id, class from student where name="happy"

❖ The column **after WHERE** condition, which is "NAME" here.

Q What is the difference between Stored Procedure and Functions (at least 3)? **V. IMP.** SQL Se

Stored Procedure	Function
1. SP may or may not **return** a value	Function must return a value
2. Can have input/**output** parameters	Only has input parameters
3. We can **call** function inside SP	Cannot call SP insied a function
4. We cannot use SP in **SQL statements** like SELECT, INSERT, UPDATE, DELETE, MERGE, etc.	We can use them with function. SELECT *, dbo.fnCountry(city.long) FROM city;
5. We can use try-catch exception handling in SP	We can not use try-catch in functions
6. We can use transactions inside SP.	We can not use transactions inside functions.

```
CREATE PROCEDURE proc_name
(@Ename varchar(50),
 @EId int output)
AS
BEGIN
    ---- function
    INSERT INTO Employee (EmpName)
    VALUES (@Ename)
    SELECT @EId= SCOPE_IDENTITY()
END
```

```
CREATE FUNCTION function_name
 (parameters) --only input parameter
RETURNS data_type AS
BEGIN            SP X
    -- SQL statements
    RETURN value
END;
```

Q What is a Cursor? Why to avoid them?

❖ A database Cursor is a control which enables traversal/ iteration over the rows or records in the table.

❖ 5 step process:

1. Declare
2. Open
3. Fetch using while loop
4. Close
5. Deallocate

❖ LIMITATION

A cursor is a MEMORY resident set of pointers - meaning it occupies lots of memory from your system which is not good for performance.

```
DECLARE
    @product_name VARCHAR(MAX),
    @list_price   DECIMAL;

DECLARE cursor_product CURSOR
FOR SELECT
        product_name,
        list_price
    FROM
        production.products;

OPEN cursor_product;

FETCH NEXT FROM cursor_product INTO
    @product_name,
    @list_price;

WHILE @@FETCH_STATUS = 0
    BEGIN
        PRINT @product_name + CAST(@list_price AS varchar);
        FETCH NEXT FROM cursor_product INTO
            @product_name,
            @list_price;
    END;

CLOSE cursor_product;

DEALLOCATE cursor_product;
```

What is CTE in SQL Server?

❖ A Common Table Expression, is a TEMPORARY named result set, that you can reference within a SELECT, INSERT, UPDATE, or DELETE statement.

```
WITH                      CTE name
with engineers as (
  select *
  from employees              CTE Body
  where dept='Engineering'
)


select *
from engineers          CTE Usage
where ...
```

drop auto.

How to optimize a Stored Procedure or SQL Query?

1. Use SET NOCOUNT ON

2. Specify column names instead of using * .
   SELECT col1, col2 FROM table1

3. Use schema name before objects or tablenames.
   SELECT EmpID, Name FROM dbo.Employee

4. Do not use DYNAMIC QUERIES. They are vulnerable to SQL Injections.

5. Use EXISTS () instead of COUNT ().
   Incorrect: SELECT Count(1) FROM dbo.Employee
   Correct: IF(EXISTS (SELECT 1 FROM db.Employees))

6. Use TRANSACTION when required only

```
SQLQuery2.sql - D...         ved (51))*   ⇌ ×

  Update  CustTransaction
  SET col1 =  Col2,
      Col2 =  col1,
      Col3 = Col4,
      Col4 = Col3,
      COL5 = COL6,
      COL6 = COL5
  WHERE id = @LoopStartPosition

100 %   ▾

  Messages

  (1 row affected)
  (1 row affected)
  (1 row affected)
  (1 row affected)
  (1 row affected)
100 %   ▾

  ✓ Query executed successfully.
```

How to get the Nth highest salary of an employee?   V. IMP.

SQL Serv

1. The logic is first select TOP 3 salaries in descending order.

2. Put the result in "Result" and then do order by asc

3. Select top 1 salary from result set

```
SELECT SALARY
FROM (
```

```
SELECT TOP 1 SALARY
FROM (
```

```
SELECT DISTINCT TOP N SALARY
FROM tbl_Employees
ORDER BY SALARY DESC
```

```
SELECT DISTINCT TOP N SALARY
FROM tbl_Employees
ORDER BY SALARY DESC
```

```
SELECT DISTINCT TOP N SALARY
FROM tbl_Employees
ORDER BY SALARY DESC
```

```
) RESULT
ORDER BY SALARY
```

```
) RESULT
ORDER BY SALARY
```

Salary
5000
10000
6000
4000
2000
7000

10000
7000
6000

Result
6000
7000
10000

Result
6000

What are ACID properties?

❖ ACID properties are used when you are handling **transactions** in SQL.

❖ For example, multiple inserts are coming at same point of time.

Atomicity
Each transaction is "all or nothing"

Consistency
Data should be valid according to all defined rules

Isolation
Transactions do not affect each other

Durability
Committed data would not be lost, even after power failure.