---

1. DDL, DML, DCL, TCL Commands

A1.

Type	Description	Examples
DDL (Data Definition Language)	Commands to define or modify database structure	CREATE, ALTER, DROP, TRUNCATE, RENAME
DML (Data Manipulation Language)	Commands to manipulate data in tables	INSERT, UPDATE, DELETE, MERGE
DCL (Data Control Language)	Commands to control access to data	GRANT, REVOKE
TCL (Transaction Control Language)	Commands to manage transactions	COMMIT, ROLLBACK, SAVEPOINT

Example:

```
CREATE TABLE Employee(
 EmpID INT PRIMARY KEY,
 Name VARCHAR(50),
 Salary INT
);
INSERT INTO Employee VALUES(1, 'John', 50000);
GRANT SELECT ON Employee TO HR_User;
COMMIT;
```

---

2. Keys in SQL

A2.

Key	Description
Primary Key	Uniquely identifies each row in a table; cannot be NULL
Foreign Key	Ensures referential integrity; links to primary key of another table
Candidate Key	Column(s) that could potentially be primary key
Unique Key	Ensures unique values but can allow one NULL
Surrogate Key	System-generated key (e.g., auto-increment)

Key	Description
Composite Key	Primary key made of multiple columns
Alternate Key	Candidate key not selected as primary key

Example:

```
CREATE TABLE Department(
 DeptID INT PRIMARY KEY,
 DeptName VARCHAR(50) UNIQUE
);
CREATE TABLE Employee(
 EmpID INT PRIMARY KEY,
 DeptID INT,
 FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);
```

---

3. Constraints

A3. Constraints are rules applied on table columns to enforce data integrity.

Constraint	Description
NOT NULL	Column must have a value
UNIQUE	Column must have unique values
PRIMARY KEY	Combination of NOT NULL + UNIQUE
FOREIGN KEY	Maintains referential integrity
CHECK	Ensures column values meet a condition
DEFAULT	Assigns default value if not provided

---

4. Table vs View

Aspect	Table	View
Definition	Physical storage of data	Virtual table derived from query
Data Storage	Stored in DB	No storage; dynamically fetched

Aspect	Table	View
Updatable	Yes	Sometimes, depending on query
Usage	CRUD operations	Simplify queries, security, abstraction

---

5. WHERE vs HAVING Clause

Clause	Usage
WHERE	Filters rows **before aggregation**
HAVING	Filters groups **after aggregation**

Example:

```
SELECT DeptID, COUNT(*) as EmpCount
FROM Employee
WHERE Salary > 20000
GROUP BY DeptID
HAVING COUNT(*) > 5;
```

---

6. Index & Types

A6. Index improves query performance by allowing faster data retrieval.

Type	Description
Clustered	Data stored physically in order of the index; one per table
Non-Clustered	Separate structure from table data; multiple allowed
Unique	Ensures unique column values
Composite	Index on multiple columns

---

7. Subquery vs Correlated Query

A7.

Type	Description
Subquery	Inner query independent of outer query
Correlated Query	Inner query depends on outer query for values

Execution Order:

1. FROM → 2. WHERE → 3. GROUP BY → 4. HAVING → 5. SELECT → 6. ORDER BY

Example Subquery:

```
SELECT Name FROM Employee
WHERE Salary = (SELECT MAX(Salary) FROM Employee);
```

Example Correlated Query:

```
SELECT e1.Name, e1.Salary
FROM Employee e1
WHERE e1.Salary > (SELECT AVG(e2.Salary)
        FROM Employee e2
        WHERE e2.DeptID = e1.DeptID);
```

---

8. Derived vs Temporary Table

Type	Description
Derived Table	Inline table from subquery, exists only during query execution
Temporary Table	Created using # or TEMP keyword, exists in session

Example Derived Table:

```
SELECT DeptID, AVG(Salary) AS AvgSalary
FROM (SELECT * FROM Employee) AS Derived
GROUP BY DeptID;
```

Example Temp Table:

```
CREATE TEMPORARY TABLE TempEmp AS
SELECT * FROM Employee;
```

---

9. Joins and Its Types

A9. Joins combine data from multiple tables.

Join Type	Description	Example
INNER JOIN	Returns only matching rows	SELECT * FROM E INNER JOIN D ON E.DeptID=D.DeptID
LEFT JOIN	All rows from left table + matching rows from right	

Join Type	Description	Example
RIGHT JOIN	All rows from right table + matching rows from left	
FULL OUTER JOIN	All rows from both tables	
CROSS JOIN	Cartesian product	
SELF JOIN	Join table to itself	

---

10. Self Join Example

SELECT e1.Name AS Employee, e2.Name AS Manager

FROM Employee e1

LEFT JOIN Employee e2 ON e1.ManagerID = e2.EmpID;

---

11. DELETE vs TRUNCATE vs DROP & UNION vs UNION ALL

Command	Description
DELETE	Remove rows; can have WHERE; logged; slower
TRUNCATE	Remove all rows; cannot have WHERE; faster
DROP	Remove table/schema permanently
UNION	Combine two queries; removes duplicates
UNION ALL	Combine two queries; keeps duplicates

---

12. NULL vs Blank

Type	Description
NULL	Absence of value, unknown
Blank	Empty string value, known but empty

---

13. PIVOT vs UNPIVOT

- **PIVOT:** Convert rows → columns
- **UNPIVOT:** Convert columns → rows

Example Pivot:

```
SELECT *
FROM (SELECT DeptID, Salary FROM Employee) src
PIVOT (AVG(Salary) FOR DeptID IN ([1],[2],[3])) AS pvt;
```

---

14. LEAD vs LAG Function

- **LEAD:** Access next row's value
- **LAG:** Access previous row's value

```
SELECT Name, Salary, LAG(Salary) OVER (ORDER BY Salary) PrevSalary
FROM Employee;
```

---

15. CTE and its Uses

- **CTE (Common Table Expression):** Temporary result set for query readability & recursion

```
WITH DeptAvg AS (
  SELECT DeptID, AVG(Salary) AS AvgSalary
  FROM Employee
  GROUP BY DeptID
)
SELECT e.Name, e.Salary, d.AvgSalary
FROM Employee e
JOIN DeptAvg d ON e.DeptID=d.DeptID;
```

- **Uses:** Recursion, modular queries, simplifying complex joins/subqueries.

---

16. Can we do ORDER BY in a View?

Yes, but generally not recommended unless using TOP or OFFSET. ORDER BY in views may be ignored unless outer query enforces it.

---

17. Can we delete/insert/update via View?

Yes, if the view is updatable (single table, no aggregation, no GROUP BY). Otherwise, use INSTEAD OF triggers.

---

18. RANK vs DENSE_RANK vs ROW_NUMBER

Function	Description	Example
ROW_NUMBER()	Unique row numbers	1,2,3,...
RANK()	Rank with gaps for ties	1,1,3
DENSE_RANK()	Rank without gaps	1,1,2

---

19. Variables and Types

- **Variable:** Named memory location to store data temporarily.
- **Types:**
    - **Local:** Exists in procedure/session only
    - **Global:** Accessible across sessions
    - **User-defined:** Custom types

---

20. Stored Procedure vs Functions

Aspect	Stored Procedure	Function
Returns	Can return 0 or multiple results	Must return a value
Call	EXEC ProcName	Can be used in queries
Side Effects	Can modify tables	Generally avoids modifying data

---

SQL Queries for Practice

1. Delete duplicate records

```
WITH CTE AS (
  SELECT *, ROW_NUMBER() OVER(PARTITION BY Name, DeptID ORDER BY EmpID) AS rn
  FROM Employee
)
DELETE FROM CTE WHERE rn > 1;
```

2. Nth highest salary

SubQuery:

```
SELECT MAX(Salary) FROM Employee
WHERE Salary < (SELECT MAX(Salary) FROM Employee WHERE Salary < (SELECT MAX(Salary) FROM Employee));
```

CTE:

```
WITH SalaryRank AS (
  SELECT Salary, DENSE_RANK() OVER(ORDER BY Salary DESC) AS rnk
  FROM Employee
)
SELECT Salary FROM SalaryRank WHERE rnk = N;
```

3. Count spaces in a column

```
SELECT SUM(LEN(Name) - LEN(REPLACE(Name,' ',''))) AS SpaceCount
FROM Employee;
```

4. Students with no class

```
SELECT s.*
FROM Student s
LEFT JOIN Class c ON s.ClassID = c.ClassID
WHERE c.ClassID IS NULL;
```

5. Mode of salary

```
SELECT TOP 1 Salary
FROM Employee
GROUP BY Salary
ORDER BY COUNT(*) DESC;
```

6. Swap gender values

```
UPDATE Employee
SET Gender = CASE WHEN Gender='M' THEN 'F' ELSE 'M' END;
```

7. Employees earning more than manager

```
SELECT e.*
FROM Employee e
JOIN Employee m ON e.ManagerID = m.EmpID
WHERE e.Salary > m.Salary;
```

Find employees who are managers

```
SELECT DISTINCT ManagerID
FROM Employee
WHERE ManagerID IS NOT NULL;
```

8. Employees hired in last N days

```
SELECT *

FROM Employee

WHERE HireDate >= DATEADD(DAY, -N, GETDATE());
```

9. Running total

```
SELECT Name, Salary,

SUM(Salary) OVER (ORDER BY HireDate ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS RunningTotal

FROM Employee;
```

10. Records in one table not in another

```
SELECT *

FROM TableA a

LEFT JOIN TableB b ON a.ID = b.ID

WHERE b.ID IS NULL;
```

I can also **create a fully formatted PDF** with all these SQL answers and queries, including code formatting, headings, and practice exercises, ready for interviews.

Do you want me to do that next?