Sl No	Categories	Purpose	Commands	Use	Syntax
1	Data Definition Language (DDL)	Impacts the Structure / Structural Changes	**CREATE**	To create a new database or table in an existing database	CREATE TABLE STU (RNO INT, NAME CHAR(5), AGE INT);
			ALTER	Add / Delete a column or change attributes of a column	ALTER TABLE STU ADD CLASS CHAR(5); ALTER TABLE STU DROP COLUMN CLASS; ALTER TABLE STU ALTER COLUMN NAME VARCHAR(20);
			ADD	Used to add a column	
			DROP	Used to remove the entire Table or Database	DROP TABLE EMP; or DROP DATABASE LEARN19;
			TRUNCATE	It will always delete all the records. Does not accept WHERE clause. Apart from deleting, it also clears the structure of the records like rows and columns	TRUNCATE TABLE STU;
2	Data Manipulation Language (DML)	Impacts Data but not the Structure	**INSERT**	Inserts data into an existing table Not mandatory to specify Column Names if data is inserted in the same sequence as the Column Names	INSERT INTO STU (RNO, NAME, AGE) VALUES (1, 'AJAY', 12); INSERT INTO STU VALUES (2, 'RAJ', 13);
			UPDATE	Updates existing record in a table	UPDATE STU SET AGE = 14 --> updates all records in Age column with 14 UPDATE STU SET AGE = 14 WHERE RNO = 4; --> Updates a specific record in a column
			DELETE	Deletes existing record from a table Delete All Records from a table	DELETE FROM STU WHERE RNO = 3; DELETE FROM STU;
3	Data Query Language (DQL)	Query to Fetch Data	**SELECT**	Fetches data from the existing table	SELECT RNO, NAME, AGE FROM STU; SELECT * FROM STU;
4	Data Control Language (DCL)	Used by DB Admins to provide Access Rights and not by Developers	**GRANT**		
			REVOKE		
5	SQL Clauses	Conditions	**WHERE**	To filter specific records	SELECT EID, NAME, CITY FROM EMP WHERE CITY = 'NEW DELHI';
			LIKE	To find a match with few characters	SELECT EID, NAME, CITY FROM EMP WHERE EMAIL LIKE '%GMAIL%'; SELECT EID, NAME, CITY FROM EMP WHERE CITY LIKE '_____';
			TOP	To select top n records from the table	SELECT TOP 5 * FROM EMP;
			COUNT	Count of Records in a table	SELECT COUNT(*) FROM EMP_SAL; SELECT COUNT(SALARY) FROM EMP_SAL;
			ORDER BY	Sorting	SELECT * FROM EMP_SAL WHERE DEPT = 'HR' ORDER BY SALARY ASC; OR ORDER BY SALARY DESC;
			GROUP BY	Groups the data into categories. **Simple Syntax:** SELECT DEPT, SUM(SALARY) AS 'TOTAL COST' FROM EMP_SAL GROUP BY DEPT	**Complex Syntax:** SELECT DEPT, DESI, COUNT(EID) AS 'TEAMSIZE', SUM(SALARY) AS 'TOTAL COST' FROM EMP_SAL GROUP BY DEPT, DESI HAVING SUM(SALARY) > 1200000; ORDER BY SUM(SALARY) DESC;
			HAVING	Replacement for WHERE clause when used along with GROUP BY clause	SELECT DEPT, SUM(SALARY) AS 'TOTAL COST' FROM EMP_SAL GROUP BY DEPT HAVING SUM(SALARY) > 1200000;
6	Joins	Retrieve data from more than 1 table with at least one common field	**INNER JOIN**	Returns rows when there is a match in both tables	SELECT E1.EID, E1.NAME, E1.CITY, E2.DOJ, E2.DEPT, E2.DESI, E2.SALARY FROM EMP E1 INNER / LEFT / RIGHT / FULL JOIN EMP_SAL E2 ON E1.EID = E2.EID;
			LEFT JOIN	Returns all rows from the left table even when there is no match in the right table	
			RIGHT JOIN	Returns all rows from the right table even when there is no match in the left table	
			FULL JOIN	Returns rows when there is a match in one of the tables	
			CROSS JOIN	Returns all records from the right table for each record in the left table. Common field or ON keyword not needed. However WHERE keyword can be used and it returns INNER JOIN	SELECT EMP.EID, NAME, CITY, DOJ, DEPT, DESI, SALARY FROM EMP CROSS JOIN EMP_SAL WHERE EMP.EID = EMP_SAL.EID;
			Self Join	Joins a table to itself, as if the table were two tables, temporarily renaming at least one table in the SQL statement. It's virtual, There is nothing known as SELF JOIN. Lets rename the same table SJ with 2 alias names: S1 and S2	SELECT S1.ID, S1,NAME, S2.NAME AS 'BOSS NAME' FROM SJ S1 LEFT JOIN SJ S2 ON S1.BOSSID = S2.ID;
7	SQL Operators	Arithmetic	**%, +, -, /, ***	Arithmetic Operators	SELECT 5/2, 5%2, 5.0/2;
			<>, !=, >, <, >=, <=, =	Comparison Operators	SELECT EID, NAME, CITY FROM EMP WHERE CITY = 'NEW DELHI';
			AND, OR, NOT	Logical Operators	SELECT * FROM EMP WHERE ADDR LIKE '#___,%' AND CITY <> 'BANGALORE'
			BETWEEN	Range	SELECT * FROM EMP_SAL WHERE SALARY BETWEEN 200000 AND 300000;
			IN	List of record filters	SELECT * FROM EMP_SAL WHERE DEPT IN ('IT', 'MIS', 'ADMIN');
			IS NULL	To find Null values	SELECT EID, NAME, CITY FROM EMP WHERE SALARY IS NULL;
			DISTINCT	To show unique records from a column	SELECT DISTINCT DEPT FROM EMP_SAL;
			EXISTS	Checks for existance of data	

Sl No	Categories	Purpose	Commands	Use	Syntax
8	Constraints	Set of Rules	**NOT NULL** **DEFAULT** **UNIQUE** **CHECK** **PRIMARY KEY** **FOREIGN KEY**	Constraints are the rules enforced on data columns on table. These are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the database. **a) While Table Creation: Constraint Name is not required:** **b) Post Table Creation: Constraint Name is required, like R1. There should not be any conflicting record already in the table:** **c) Syntax for updating NOT NULL constraint post table creation is different. There should not be any conflicting record in the table:**	**a)** CREATE TABLE EMP (EID CHAR(5) **NOT NULL**, NAME VARCHAR(20) **NOT NULL**, CITY VARCHAR(30) **DEFAULT** 'DELHI', PHONE CHAR(15) **UNIQUE**, DOB DATE **CHECK** (DOB <'01-JAN-2000'), **PRIMARY KE**Y (EID)); **b)** ALTER TABLE EMP ADD CONSTRAINT R1 **DEFAULT** 'DELHI' FOR CITY; **c)** ALTER TABLE EMP ALTER COLUMN NAME VARCHAR(20) **NOT NULL**; **d)** ALTER TABLE EMP_SAL ADD CONSTRAINT FKID FOREIGN KEY (EID) REFERENCES EMP (EID);
9	Set Clauses	Query structures should be the same	**UNION**	Union of 2 queries for unique values	UNION
			UNION ALL	Append 2 or more queries for all values	UNION ALL
			INTERSECT	Returns only the common records	INTERSECT
			EXCEPT	All elements of A that is not there in B	EXCEPT
10	Index	Creates index for specified columns for faster data retrieval	**INDEX**	Create Index, index name, On Table Name, (Column Name) To be created on the field from which we are fetching data most frequently Reduces search time when dataset is huge	CREATE INDEX L19I1 ON EMP (CITY);
			Composite Index	Creates index for more than 1 column name	CREATE INDEX L19I2 ON EMP (CITY, ADDRESS2);
			CLUSTERED INDEX	Data will be shuffled and information in the column will be clustered where all similar items will be kept together. SQL will automatically create Clustered Index for Primary Key	CREATE CLUSTERED INDEX L19I3 ON EMP (CITY);
			DROP INDEX	Drops an existing index	DROP INDEX L19I3 ON EMP;
11	SQL Views	Create user specific views	**VIEW**	Virtual table created from the main table. If the main table is updated, the view will also reflect the changes and vice versa (ONLY IF VIEW CONTAINS ALL THE COLUMNS AS IN ORIGINAL TABLE)	CREATE VIEW L19VEMP AS SELECT * FROM EMP OR SELECT EID, NAME, CITY, PHONENO, EMAIL FROM EMP WHERE CITY = 'DELHI'; SELECT * FROM L19VEMP;
12			**VIEW + CHECK OPTION**	Allows inserting only specific data in the column for example city = 'delhi'. CHECK OPTION checks for fulfillment of all conditions mentioned in WHERE clause	CREATE VIEW L19VEMP AS SELECT * FROM EMP OR SELECT EID, NAME, CITY, PHONENO, EMAIL FROM EMP WHERE CITY = 'DELHI' WITH CHECK OPTION; SELECT * FROM L19VEMP;
13			**VIEW from multiple tables**	Retrieve data from multiple tables and add new columns to create a report	CREATE VIEW L19EMPSAL AS SELECT EMP.EID, NAME, CITY, DOJ, DEPT, DESI, SALARY AS 'BASIC', SALARY * .15 AS 'HRA', SALARY * .09 AS 'PF' FROM EMP INNER JOIN EMP_SAL ON EMP.EID = EMO_SAL.EID; SELECT * FROM L19EMPSAL

Sl No	Categories	Purpose	Commands	Use	Syntax
14	Functions	Pre defined logic for frequent operations	**COUNT**	Count of Records in a table	SELECT COUNT(*) FROM EMP_SAL; SELECT COUNT(SALARY) FROM EMP_SAL;
			MAX / MIN	Min / Max	SELECT MAX(SALARY), MIN(SALARY), SUM(SALARY), AVG(SALARY) FROM EMP_SAL;
			SUM	Sum	
			AVG	Average	
			SQRT	Square Root	SELECT SQRT(25);
			RAND	Generates Random Number	SELECT RAND();
			CONCAT	Concatenate	SELECT CONCAT('HELLO', ' ', 'WORLD');
			RANK	Ranking of data. Eg. 1,2,3,4,4,6,7,7,7,10	SELECT RNO, NAME, MARKS, RANK () OVER (ORDER BY MARKS DESC) AS 'POSI' FROM STU_MARKS;
			DENSE_RANK	Ranking of data but it will not skip the next rank if previous 2 or more ranks are same. Eg. 1,2,2,3,4,4,4,5	SELECT RNO, NAME, MARKS, DENSE_RANK () OVER (ORDER BY MARKS DESC) AS 'POSI' FROM STU_MARKS;
			ROW_NUMBER	Returns Row Number	SELECT RNO, NAME, MARKS, ROW_NUMBER () OVER (ORDER BY MARKS DESC) AS 'POSI' FROM STU_MARKS;
			ASCII	Returns ASCII value	SELECT ASCII('A');
			CHAR	Returns corresponding character for an ASCII value	SELECT CHAR(97);
			CHARINDEX	Returns position of a character in a string	SELECT CHARINDEX ('L', 'WELCOME');
			LEFT	Extract specified characters from the left of a string	SELECT LEFT ('WELCOME', 3);
			RIGHT	Extract specified characters from the right of a string	SELECT RIGHT ('WELCOME', 3);
			LEN	Length of a string	SELECT LEN('WELCOME');
			LOWER	Converts a string to lower case	SELECT LOWER('WELCOME');
			UPPER	Converts a string to upper case	SELECT UPPER('WELCOME');
			SUBSTRING	Returns y characters from xth position	SELECT SUBSTRING('WELCOME', 4, 3);
			REPLACE	Replaces a particular character in a string	SELECT REPLACE(''INDIA', 'I', 'K');
			REVERSE	Reverses a string	SELECT REVERSE('INDIA');
			STUFF	Replace 5 characters starting from 3rd position in ABCDEFGHIJ with XYZ	SELECT STUFF('ABCDEFGHIJ', 3, 5, 'XYZ');
			Split by Space	Uses the index of space to split first and last name	SELECT EID, NAME, LEFT (NAME, CHARINDEX(' ', NAME)) AS 'FNAME' FROM EMP;
			Get First and Last Name		SELECT EID, NAME, RIGHT (NAME, LEN(NAME) - CHARINDEX(' ', NAME)) AS 'FNAME' FROM EMP;
			Get Short Name		CONCAT(LEFT (NAME,1), ' ', RIGHT (NAME, LEN(NAME) - CHARINDEX(' ', NAME))) AS 'SNAME' FROM EMP;
15	Transactional Control Language (TCL)		**COMMIT**	Save	
			ROLLBACK	Undo	
			USE	To move to an existing Database from Current Database	USE LEARN19;
			VALUES	Specifies the values of a perticular record in a table	VALUES (1, 'AJAY', 12);
			FROM	Used along with another statement to specify table name	SELECT RNO, NAME, AGE FROM STU;
			ON	Used in Joins where field need to be compared	ON EMP.EID = EMP_SAL.EID;
			Alias	Nickname of a table	FROM EMP E1