

The **SQL**

6 Weeks of Knowledge in 1 PDF

What's SQL?

Database Architect!

From **"What is SQL?"** to
Database Architect:
The Complete Guide

Q1: What does SQL stand for?

A: SQL stands for Structured Query Language.

Q2: What is the primary function of SQL?

A: SQL is the standard language used to communicate with Relational Database Management Systems (RDBMS). It allows users to store, retrieve, manage, and manipulate data within a database.

Q3: Is SQL a general-purpose programming language like Python or Java?

A: No, SQL is a domain-specific language. While general-purpose languages are used to build applications, SQL is specifically designed for managing and querying data held in a relational database.

Q4: Can SQL be used with non-relational databases?

A: Generally, no. SQL is designed for relational databases (like MySQL, PostgreSQL, SQL Server). However, some non-relational (NoSQL) databases have adopted SQL-like query languages (like CQL for Cassandra) to make it easier for users to adapt.

Q5: What is "Data" in the context of computing?

A: Data refers to raw, unorganized facts and figures (e.g., "105", "Smith", "25.5"). By itself, data often lacks context. When data is processed, organized, and structured to have meaning, it becomes Information.

Q6: What is a Database?

A: A database is an organized collection of structured information or data, typically stored electronically in a computer system. Think of it as a digital filing cabinet where files are organized so that you can easily find what you need.

Q7: What is a DBMS?

A: DBMS (Database Management System) is the software that interacts with end-users, applications, and the database itself to capture and analyze the data. It serves as an interface between the user and the database.

Examples: Microsoft SQL Server, Oracle Database, MySQL.

Q8: What is the relationship between a Database and a DBMS?

A: The Database is the passive container (the books in the library).

The DBMS is the active software that manages it (the librarian). You cannot access the database directly; you must go through the DBMS.

Q9: What are the two main categories of modern databases?

A: Relational Databases (SQL): Organize data into tables with rows and columns. They enforce strict schemas.

Non-Relational Databases (NoSQL): Store data in flexible formats like documents, key-values, or graphs. They are scalable and flexible.

Q10: What defines a Relational Database (RDBMS)?

A: An RDBMS stores data in tables (relations). These tables are linked to each other using keys (Primary and Foreign keys). This structure minimizes redundancy and ensures data integrity.

Q11: When should you use a NoSQL database over an RDBMS?

A: You might choose NoSQL when:

You are dealing with massive amounts of unstructured data (Big Data).

You need rapid development with a flexible schema (data structure changes frequently).

You need extremely high read/write speeds for real-time applications (e.g., social media feeds).

Q12: Give examples of popular databases for each type.

A: Relational: PostgreSQL, MySQL, Oracle, SQLite.

NoSQL: MongoDB (Document), Redis (Key-Value), Cassandra (Wide-column).

Q13: What is the purpose of DDL?

A: DDL commands define, modify, and remove the structure of database objects (like tables, indexes, and views). They do not touch the data inside the tables, just the tables themselves.

Q14: What are the common DDL commands?

A: CREATE: Creates a new table or database.

ALTER: Modifies an existing table structure (e.g., adding a column).

DROP: Deletes the entire table (structure + data).

TRUNCATE: Removes all records from a table but keeps the structure.

Q15: What is the difference between DROP and TRUNCATE?

A: DROP: Removes the table definition and all its data. The table ceases to exist.

TRUNCATE: Deletes all data inside the table but leaves the table structure (columns/datatypes) intact for future use.

Q16: What is the purpose of DML?

A: DML commands allow you to handle the data present in the database. This is what you use for day-to-day operations like adding new records or updating old ones.

Q17: What are the common DML commands?

A: INSERT: Adds new rows of data to a table.

UPDATE: Modifies existing data in a table.

DELETE: Removes specific rows of data from a table.

(Note: SELECT is technically Data Query Language (DQL), but is often grouped loosely with DML in conversation.)

Q18: Summary Comparison Table: DDL vs. DML

Feature DDL	(Data Definition Language)	DML (Data Manipulation Language)
Focus	Database Structure (Schema)	Data (Rows/Records)
Key Commands	CREATE, ALTER, DROP, TRUNCATE	INSERT, UPDATE, DELETE
Effect	Changes how data is stored.	Changes what data is stored.
Reversibility	Generally Auto-committed (Hard/Impossible to undo). (if transactions are used).	Can be rolled back

Q19: What is the primary difference between Relational and Special operators in SQL?

A: Relational Operators (e.g., =, >, <) are standard mathematical comparisons used to compare exact values or relative magnitudes.

Special Operators (e.g., LIKE, IN, BETWEEN) are designed for more complex filtering scenarios, such as pattern matching, checking against a list, or validating specific ranges.

Q20: Why must we use IS NULL instead of = NULL when filtering for missing data?

A: In SQL, NULL represents an unknown or missing value. It is not zero or a blank space. Since you cannot verify if one "unknown" is equal to another "unknown," logical comparisons like = NULL will always fail (return false/null). You must use the special operator IS NULL to check for the state of being missing.

Q21: How do the wildcards % and _ differ when using the LIKE operator?

A: % (Percent): Represents zero, one, or multiple characters. (e.g., 'A%' finds "A", "Apple", and "Algorithm").

_ (Underscore): Represents exactly one single character. (e.g., 'A_' finds "At" or "An", but not "Apple").

Q22: What is the equivalent of the BETWEEN operator using standard relational operators?

A: The query WHERE salary BETWEEN 1000 AND 2000 is logically equivalent to: WHERE salary >= 1000 AND salary <= 2000. (Note: BETWEEN is inclusive, meaning it includes the boundary values.)

Q23: What defines a "Single-Row Function"?

A: A single-row function processes one row at a time and returns one result per row. For example, if you apply UPPER() to a column in a table with 100 rows, you get 100 uppercase results.

Q24: How can UPPER() and LOWER() help in data retrieval?

A: They are often used to perform case-insensitive searches. If a user searches for "Smith" but the database has "SMITH", the query WHERE LAST_NAME = 'Smith' might fail. Using WHERE UPPER(LAST_NAME) = 'SMITH' ensures the match is found regardless of how the data was typed.

Q25: What does SUBSTR('Database', 1, 4) return?

A: It returns 'Data'.

The function extracts a substring starting at position 1 and takes the next 4 characters.

Q26: What is the critical difference between ROUND() and TRUNC()?

A: ROUND(): Rounds the number to the nearest specified decimal. (e.g., ROUND(45.926, 2) becomes 45.93).

TRUNC(): Simply cuts off (truncates) the number at the specified decimal without rounding. (e.g., TRUNC(45.926, 2) becomes 45.92).

Q27: What is the result of MOD(10, 3) and when is it useful?

A: The result is 1.

MOD returns the remainder of a division (10 divided by 3 is 3 with a remainder of 1). It is commonly used to determine if a number is odd or even (e.g., MOD(num, 2) = 0 means even).

Q28: What data type does TO_CHAR(SYSDATE, 'YYYY-MM-DD') return?

A: It returns a String (Character) data type, not a Date. This function is used to convert a raw date value into a specific readable text format.

Q29: If today is January 31st, what will ADD_MONTHS(SYSDATE, 1) return?

A: It will return February 28th (or 29th in a leap year). SQL date functions are smart enough to handle month-end adjustments automatically.

Q30: What is the DUAL table?

A: DUAL is a special dummy table present in Oracle (and some other databases) that contains exactly one column and one row.

Q31: Why do we need the DUAL table?

A: In standard SQL, a SELECT statement usually requires a FROM clause pointing to a real table. If you want to calculate 2 + 2 or get the current system date (SYSDATE) without referencing real data, you select from DUAL.

Example: SELECT SYSDATE FROM DUAL;

Q32: What happens if you try to DROP or INSERT into the DUAL table?

A: You should never do this! DUAL is a system object. Modifying it can cause the database or system functions to fail. It is strictly for "scratchpad" calculations (reading, not writing).

Q33: What is the fundamental difference between Single-Row Functions and Aggregate Functions?

A: Single-Row Functions (like UPPER, ROUND) operate on one row at a time and return one result per row.

Aggregate Functions (like SUM, COUNT) operate on a set of multiple rows and return a single summary value for the entire set.

Q34: How do Aggregate Functions handle NULL values?

A: With the exception of COUNT(*), all aggregate functions ignore NULL values.

Example: If you AVG() a column with values {10, NULL, 20}, the calculation is $(10+20)/2 = 15$. It does not treat the NULL as a zero (which would have made the average 10).

Q35: What is the difference between COUNT(*) and COUNT(column_name)?

A: COUNT(*): Counts all rows in the table, including duplicates and rows containing NULL values.

COUNT(column_name): Counts only the rows where the specific column is NOT NULL.

Q36: Can MIN() and MAX() be used on non-numeric data types?

A: Yes.

Dates: MIN returns the earliest date; MAX returns the latest date.

Strings (Varchar): MIN returns the first value alphabetically (A-Z); MAX returns the last value.

Q37: What is the primary purpose of the GROUP BY clause?

A: GROUP BY allows you to arrange identical data into groups. It is mandatory when you want to select a non-aggregated column (like Department_ID) alongside an aggregated metric (like SUM(Salary)).

Q38: What is the "Golden Rule" of the SELECT list when using GROUP BY?

A: Any column in your SELECT list that is not part of an aggregate function must be listed in the GROUP BY clause. If you miss this, the database will throw an error because it doesn't know which individual row value to display for the group.

Q39: The Ultimate Interview Question: What is the difference between WHERE and HAVING?

A: | Feature | WHERE Clause | HAVING Clause | | :--- | :--- | :--- | | Timing | Filters rows before grouping. | Filters groups after grouping. | | Data Scope | Filters individual records. | Filters summarized group results. | | Aggregates | Cannot use aggregates (e.g., SUM). | Can use aggregates (e.g., SUM, COUNT). |

Example: "Show me employees with salary > 5000" uses WHERE. "Show me departments where the average salary > 5000" uses HAVING.

Q40: Can you use both WHERE and HAVING in the same query?

A: Yes. The execution order is:

WHERE filters the raw rows.

GROUP BY groups the remaining rows.

HAVING filters the final groups.

Q41: Is SQLPlus the same thing as SQL?

A: No.

SQL is the language used to talk to the database (SELECT, INSERT, etc.).

SQLPlus is a command-line environment/tool provided by Oracle to run SQL queries and format the output. Commands like SET LINESIZE are instructions for the tool, not the database.

Q42: What does SET LINESIZE do, and why is it important for reporting?

A: SET LINESIZE controls the maximum number of characters printed on a single horizontal line. If this is set too low (default is usually 80), wide tables will wrap around to the next line, making the report unreadable (the "spaghetti code" of output).

Q43: How do you save your query results to a file for an audit trail?

A: You use the SPOOL command.

SPOOL filename.txt (Starts recording). Run your queries.

SPOOL OFF (Stops recording and saves the file).

Q44: What is the purpose of SET TIMING ON?

A: It displays the elapsed computer time required to execute your SQL statement. This is critical for performance tuning—it helps you prove if your new query is faster than the old one.

Q45: How do TTITLE and BTITLE enhance a report?

A:

TTITLE (Top Title): Places a header on every page of the report output (e.g., "Monthly Sales Report").

BTITLE (Bottom Title): Places a footer on every page (e.g., "Confidential - Internal Use Only").

Q46: What is the main difference between an Inner Query and an Outer Query?

A: The Inner Query is the nested query that executes first to produce a result. The Outer Query (or Main Query) uses that result to filter or manipulate its own data set.

Q47: When is a Sub-query preferred over a simple JOIN?

A: A Sub-query is often preferred when you need to compare a column against an aggregate value (like an Average or Max) that isn't known until the query runs. You cannot use a WHERE clause to filter by AVG(Salary) directly; you must use a sub-query to calculate the average first.

Q48: What happens if a Sub-query returns multiple values but you use a single-row operator like =?

A: The database will throw an error (e.g., "single-row subquery returns more than one row"). If a sub-query returns multiple values, you must use multi-row operators like IN, ANY, or ALL.

Q49: A VIEW is often called a "Virtual Table." Does it store its own data?

A: No. A View does not store data on the disk (with the exception of Materialized Views). It stores only the SQL query definition. Every time you select from a View, the database runs the underlying query in the background.

Q50: How does an INDEX speed up performance, and is there a downside?

A: * Benefit: It provides a pointer to the physical location of data, allowing the database to skip scanning every row (Full Table Scan).

Downside: It takes up extra disk space and can slow down DML operations (INSERT, UPDATE, DELETE) because the index must be updated every time the data changes.

Q51: What is the purpose of CACHE 20 in a SEQUENCE?

A: It tells the database to pre-generate 20 numbers and keep them in memory. This improves performance because the database doesn't have to access the disk for every new ID requested.

Q52: Why would a developer use a SYNONYM?

A: 1. Simplicity: To provide a short name for a long, complex table name. 2. Abstraction: If the underlying table name changes or moves to a different schema, you only update the synonym, and the application code remains the same.

Q53: What is the difference between an Authentication and a Privilege in DCL?

A: * Authentication: Handled by CREATE USER. It proves who you are (Username/Password).

Privilege: Handled by GRANT. It defines what you can do (e.g., Select from the Customers table).

Q54: If you GRANT SELECT ON Customers TO User_A, can User_A delete records?

A: No. Privileges are specific. SELECT only allows reading. To delete, you would need to explicitly GRANT DELETE.

Q55: What does the QUOTA 10M ON USERS mean in a user creation script?

A: It limits the user to using only 10 Megabytes of physical space in the "USERS" tablespace. This prevents a single user from accidentally or intentionally filling up the entire server's storage.

Q56: What is a "Pseudo Column"?

A: It is a column that acts like a table column but is not actually stored in the table. The values are generated by the database engine during query execution.

Q57: How does ROWNUM differ from a Primary Key ID?

A: * Primary Key: A permanent, unique identifier stored in the table.

ROWNUM: A temporary number assigned to a row after it passes the query filters. The first row returned is always 1, the second is 2, and so on.

Q58: What is the difference between NEXTVAL and CURRVAL?

A:

NEXTVAL: Increments the sequence and returns the new value.

CURRVAL: Returns the current value of the sequence for the current session without incrementing it.

Q59: What is a "Transaction" in a database?

A: A transaction is a single unit of work that consists of one or more DML statements (like INSERT, UPDATE, DELETE). TCL ensures that either the entire unit is saved (Commit) or the entire unit is cancelled (Rollback), maintaining data consistency.

Q60: When does a transaction officially begin and end?

A: * Begins: With the first executable SQL statement.

Ends: When you issue a COMMIT or ROLLBACK, or when you perform a DDL operation (like CREATE), which issues an automatic commit.

Q61: Can you ROLLBACK after you have executed a COMMIT?

A: No. Once a COMMIT is executed, the changes are written permanently to the disk. They cannot be undone using TCL. You would have to manually delete or update the data to "fix" it.

Q62: What is the specific benefit of using a SAVEPOINT?

A: A SAVEPOINT allows for a partial rollback. Without it, a ROLLBACK would undo every change made since the last COMMIT. With Savepoints, you can undo only the most recent (or specific) parts of a complex transaction while keeping the earlier parts intact.

Q63: What happens to your uncommitted changes if the database crashes or the power goes out?

A: The database will automatically perform a ROLLBACK. Since the changes were never "Committed," the database ensures data integrity by reverting to the last known "safe" state.

Q64: What are the three main files involved in a SQL Loader operation?

A: 1. Data File: The source file (usually .csv or .txt) containing the raw data. 2. Control File: The "instruction manual" (usually .ctl) that tells SQL Loader which table to use and how to map the data columns. 3. Log File: A file created by the tool to show the results of the load and any errors that occurred.

Q65: Why use SQL Loader instead of writing a Python script with 10,000 INSERT statements?

A: Speed. Standard INSERT statements are processed one by one, which is slow for large datasets. SQL Loader uses "Direct Path Load" (it can bypass much of the database processing overhead) to write data directly to the data blocks, making it exponentially faster.

Q66: What does the instruction FIELDS TERMINATED BY ',' do in a Control File?

A: It defines the delimiter. It tells SQL Loader that every time it sees a comma, it should treat the following text as a new column or field.

Q67: What happens if some rows in your CSV file are corrupted or have the wrong data type?

A: SQL Loader will skip those specific rows and write them into a Bad File (.bad). The rest of the valid rows will still be loaded into the table. You can then check the Bad File to fix the errors.

Q68: Is SQL Loader a part of the SQL Language?

A: No. It is a Client-Side Utility or tool provided by Oracle. You run it from the command prompt (cmd/terminal), not from inside a SQL worksheet like SELECT statements.

Q69: What is the "Common Column" (Join Key) and why is it necessary?

A: The common column is a field that exists in both tables (usually a Primary Key in one and a Foreign Key in the other). It acts as the "bridge" that allows the database to match a row in Table A with its corresponding row in Table B.

Q70: If an employee belongs to a department that doesn't exist in the Departments table, will that employee show up in an INNER JOIN?

A: No. An INNER JOIN only returns rows where there is a match in both tables. If the department ID is missing from the right table, the employee record is excluded from the results.

Q71: When would you use a LEFT JOIN instead of an INNER JOIN?

A: Use a LEFT JOIN when you want to see all records from your main table, even those that don't have a match.

Example: "Show me all customers and their orders." Using a LEFT JOIN ensures you see customers who haven't placed an order yet (they will show NULL for order details), whereas an INNER JOIN would hide them completely.

Q72: What is the result of a FULL OUTER JOIN?

A: It combines the results of both a Left and a Right Join. It returns all records from both tables. Where there is a match, the data is joined; where there is no match, the missing side will contain NULL values.

Q73: What is a "Self-Join"?

A: A self-join is when you join a table to itself. This is useful when a table has a hierarchical relationship within it.

Example: An Employees table where one column is Employee_ID and another is Manager_ID (who is also an employee in the same table).

Q74: Can a table have more than one PRIMARY KEY?

A: No. A table can have only one Primary Key. However, that Primary Key can consist of multiple columns (called a Composite Key), but as a constraint, there is only one per table.

Q75: What is the difference between a PRIMARY KEY and a UNIQUE constraint?

A:

Similarity: Both ensure that all values in a column are different.

Difference: A PRIMARY KEY strictly forbids NULL values. A UNIQUE constraint allows NULL values (and in most databases, it allows multiple NULLs because one unknown is not considered "equal" to another).

Q76: What is "Referential Integrity" in the context of Foreign Keys?

A: It is a rule that ensures the relationship between tables remains consistent. It prevents you from:
Adding a record to the child table with a Foreign Key that doesn't exist in the parent table.
Deleting a record from the parent table that is still being "pointed to" by records in the child table.

Q77: How does a CHECK constraint improve data quality?

A: It acts as a gatekeeper for business logic at the database level.

Example: CHECK (Age >= 18) ensures that no one accidentally enters a minor into a system meant for adults. This prevents "bad data" from ever entering the system, regardless of the application used.

Q78: What happens if you try to insert a record without providing a value for a column that has a DEFAULT constraint?

A: The database will not throw an error. Instead, it will automatically "plug in" the value defined in the DEFAULT constraint. If you do provide a value, the database will use your value instead of the default.

Every end has
a new
beginning.