Structured Topic Overview: SQL (Structured Query Language)

Level 1: Foundational Concepts

1.1 Core SQL Fundamentals

- **Definition and Purpose**: SQL is a standard language for accessing, manipulating, and managing relational databases[1] [2]
- **ANSI/ISO Standards**: SQL became an ANSI standard in 1986 and ISO standard in 1987[2]
- **Basic Database Operations**: Create, Read, Update, Delete (CRUD operations)[3]
- **Relational Database Management Systems (RDBMS)**: MS SQL Server, MySQL, PostgreSQL, Oracle[1]

1.2 Database Architecture

- **Tables**: Basic elements consisting of rows and columns[1]
- **Primary Keys**: Unique identifiers for table records[4]
- **Foreign Keys**: Constraints that maintain referential integrity between tables[5] [6]
- **Relationships**: One-to-one, one-to-many, many-to-many associations[7]
- **Schema**: Logical structure defining database organization[1]

Level 2: Core SQL Operations

2.1 Data Query Language (DQL)

- **SELECT Statements**: Retrieving data from databases[8]
  - Column selection and filtering
  - WHERE clause for conditional filtering
  - ORDER BY for result sorting
- **Basic Functions**: COUNT(), SUM(), AVG(), MIN(), MAX()[9] [10]
- **DISTINCT**: Retrieving unique values
- **LIMIT/TOP**: Controlling result set size

2.2 Data Manipulation Language (DML)

- **INSERT**: Adding new records to tables[3]
- **UPDATE**: Modifying existing data[3]
- **DELETE**: Removing records from tables[3]
- **Transaction Control**: COMMIT, ROLLBACK operations

2.3 Data Definition Language (DDL)

- **CREATE**: Database and table creation[11]
- **ALTER**: Modifying database structure[11]
- **DROP**: Removing database objects[11]
- **Constraints**: NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK[12]

Level 3: Intermediate Concepts

3.1 Join Operations

- **INNER JOIN**: Returns matching records from both tables[13] [7]
- **LEFT JOIN**: All records from left table plus matching from right[7] [13]
- **RIGHT JOIN**: All records from right table plus matching from left[13] [7]
- **FULL OUTER JOIN**: All records from both tables[7] [13]
- **Self Joins**: Joining a table with itself
- **Cross Joins**: Cartesian product of two tables

3.2 Subqueries and CTEs

- **Subqueries**: Nested queries within main queries[14]
- **Common Table Expressions (CTEs)**: Temporary named result sets[15] [14]
- **Recursive CTEs**: For hierarchical data processing[16] [17]
- **Correlated vs Non-correlated Subqueries**

3.3 Data Types and Functions

- **String Functions**: LEN, TRIM, SUBSTRING, UPPER, LOWER[18] [19]
- **Mathematical Functions**: ROUND, ABS, CEIL, FLOOR[19] [18]
- **Date/Time Functions**: NOW(), DATE(), YEAR(), MONTH()[18] [19]
- **Aggregate Functions**: Advanced usage with GROUP BY and HAVING[10] [9]

Level 4: Advanced Query Techniques

4.1 Window Functions

- **Ranking Functions**: ROW_NUMBER(), RANK(), DENSE_RANK()[20] [21] [22]
- **Aggregate Window Functions**: SUM(), AVG(), COUNT() OVER()[21] [22] [20]
- **Navigation Functions**: LEAD(), LAG(), FIRST_VALUE(), LAST_VALUE()[23]
- **PARTITION BY**: Dividing result sets into groups[20] [21]
- **Window Frames**: ROWS BETWEEN, RANGE BETWEEN[20]

4.2 Advanced Data Manipulation

- **Pivoting**: Converting rows to columns[17] [24] [16]
- **Unpivoting**: Converting columns to rows[24]
- **CASE Statements**: Conditional logic in queries[17]
- **Complex Joins**: Multiple table joins with complex conditions
- **Set Operations**: UNION, INTERSECT, EXCEPT

Level 5: Database Design and Normalization

5.1 Normalization

- **First Normal Form (1NF)**: Atomic values and primary keys[25] [26] [4]
- **Second Normal Form (2NF)**: Eliminating partial dependencies[26] [25] [4]
- **Third Normal Form (3NF)**: Removing transitive dependencies[25] [4] [26]
- **Boyce-Codd Normal Form (BCNF)**: Advanced normalization[4] [25]
- **Fourth Normal Form (4NF)**: Eliminating multi-valued dependencies[25] [4]

5.2 Database Design Patterns

- **Single Table Inheritance (STI)**[27]
- **Class Table Inheritance (CTI)**[27]
- **Entity-Attribute-Value (EAV)**[27]
- **Star Schema**: Data warehouse design pattern[28]
- **Snowflake Schema**: Normalized dimension tables[28]

Level 6: Performance Optimization

6.1 Indexing Strategies

- **Clustered Indexes**: Physical data ordering[29] [30] [31]
- **Non-clustered Indexes**: Separate index structures[30] [31] [29]
- **Composite Indexes**: Multi-column indexes[29] [30]
- **Index Fragmentation**: Maintenance and optimization[32]
- **Query Execution Plans**: Understanding database optimization[30] [29]

6.2 Query Performance Tuning

- **Avoiding SELECT ***: Selecting only needed columns[33] [29]
- **Sargable Queries**: Search argument able conditions[34] [29]
- **Join Optimization**: Efficient join strategies[29] [30]
- **Statistics Management**: Keeping database statistics updated[30] [29]
- **Partitioning**: Dividing large tables for better performance[29] [30]

Level 7: Advanced Database Objects

7.1 Stored Procedures and Functions

- **Stored Procedures**: Precompiled SQL code for reuse[35] [36] [37]
- **User-Defined Functions**: Custom functions for data processing[36] [35]
- **Parameters**: Input and output parameters[35] [36]
- **Error Handling**: TRY-CATCH blocks and error management[36]

7.2 Views and Triggers

- **Views**: Virtual tables for data abstraction[38] [35] [36]
- **Materialized Views**: Physically stored view results[27]
- **Triggers**: Automated code execution on data changes[39] [35] [36]
- **DML Triggers**: INSERT, UPDATE, DELETE triggers[39] [35]
- **DDL Triggers**: Schema change triggers[39]

Level 8: Transaction Management and Concurrency

8.1 ACID Properties

- **Atomicity**: All-or-nothing transaction execution[40] [41] [42]
- **Consistency**: Maintaining database integrity[41] [42] [40]
- **Isolation**: Transaction independence[42] [40] [41]
- **Durability**: Permanent transaction effects[40] [41] [42]

8.2 Concurrency Control

- **Isolation Levels**: Read Uncommitted, Read Committed, Repeatable Read, Serializable[43] [40]
- **Locking Mechanisms**: Shared locks, exclusive locks[44]
- **Deadlock Prevention and Resolution**[45]
- **Optimistic vs Pessimistic Concurrency Control**[45]

Level 9: Security and Access Control

9.1 Authentication and Authorization

- **Logins and Users**: Server-level and database-level principals[46] [47]
- **Server Roles**: sysadmin, dbcreator, securityadmin[47] [48] [46]
- **Database Roles**: db_owner, db_datareader, db_datawriter[46] [47]
- **Object-level Permissions**: GRANT, REVOKE, DENY[49] [47]

9.2 Security Best Practices

- **Principle of Least Privilege**: Minimal necessary permissions[50]
- **Role-based Access Control**: Managing permissions through roles[51] [46]
- **SQL Injection Prevention**: Parameterized queries and input validation
- **Data Encryption**: Protecting sensitive data at rest and in transit

Level 10: Advanced Topics and Specialized Areas

10.1 Complex Query Patterns

- **Recursive Queries**: Hierarchical data processing[52] [16] [17]
- **Graph Traversal**: Working with connected data structures[16]
- **Analytical Functions**: Advanced statistical operations[53] [23]
- **Time Series Analysis**: Working with temporal data patterns

10.2 Modern SQL Extensions

- **JSON Support**: Working with semi-structured data
- **XML Processing**: Handling XML data types and operations
- **Full-text Search**: Advanced text searching capabilities
- **Spatial Data**: Geographic and geometric data processing

Level 11: Database Architecture and Design Patterns

11.1 Database Design Patterns

- **Normalized Form Pattern**: Reducing redundancy and anomalies[54]
- **Denormalization**: Trading storage for query performance[54]
- **Audit Log Pattern**: Tracking data changes over time[27]
- **Versioning Pattern**: Managing data versions[27]
- **Composite Key Pattern**: Multi-column primary keys[27]

11.2 Enterprise Patterns

- **Data Warehouse Design**: OLAP vs OLTP considerations[55] [54]
- **Master Data Management**: Consistent enterprise data[54]
- **Data Lake Architecture**: Handling diverse data types[55]
- **Microservices Data Patterns**: Database per service[56]

Level 12: SQL vs NoSQL Considerations

12.1 Comparative Analysis

- **Relational vs Non-relational**: Structured vs flexible schemas[57] [58] [59]
- **ACID vs BASE**: Consistency vs availability trade-offs[59] [57]
- **Vertical vs Horizontal Scaling**: Growth strategies[58] [57]
- **Use Case Considerations**: When to choose SQL vs NoSQL[60] [58]

12.2 Database Technology Evolution

- **NewSQL Databases**: Combining SQL benefits with NoSQL scalability
- **Multi-model Databases**: Supporting multiple data models
- **Cloud-native Databases**: Serverless and managed database services
- **Distributed SQL**: Handling global-scale applications

Level 13: Best Practices and Standards

13.1 SQL Coding Standards

- **Naming Conventions**: Consistent table and column naming[61] [62] [33]
- **Code Formatting**: Readable query structure[33] [61] [34]
- **Documentation**: Comments and query explanations[62] [61]
- **Version Control**: Managing SQL code changes[61] [33]

13.2 Development Best Practices

- **Code Reusability**: Creating maintainable SQL code[63] [33]
- **Error Handling**: Robust error management strategies
- **Testing Strategies**: Unit testing and integration testing for SQL
- **Performance Monitoring**: Continuous performance assessment[30] [29]

This comprehensive topic overview covers SQL from fundamental concepts through advanced enterprise-level patterns and considerations. Each level builds upon previous knowledge, providing a structured learning path for mastering SQL across all proficiency levels, from basic database operations to complex distributed systems and modern database architectures.

⁂

Structured Topic Overview: SQL (Structured Query Language)

Level 1: Foundational Concepts

1.1 Core SQL Fundamentals

- **Definition and Purpose**: SQL is a standard language for accessing, manipulating, and managing relational databases
- **ANSI/ISO Standards**: SQL became an ANSI standard in 1986 and ISO standard in 1987
- **Basic Database Operations**: Create, Read, Update, Delete (CRUD operations)
- **Relational Database Management Systems (RDBMS)**: MS SQL Server, MySQL, PostgreSQL, Oracle

1.2 Database Architecture

- **Tables**: Basic elements consisting of rows and columns
- **Primary Keys**: Unique identifiers for table records
- **Foreign Keys**: Constraints that maintain referential integrity between tables
- **Relationships**: One-to-one, one-to-many, many-to-many associations
- **Schema**: Logical structure defining database organization

Level 2: Core SQL Operations

2.1 Data Query Language (DQL)

- **SELECT Statements**: Retrieving data from databases
  - Column selection and filtering
  - WHERE clause for conditional filtering
  - ORDER BY for result sorting
- **Basic Functions**: COUNT(), SUM(), AVG(), MIN(), MAX()
- **DISTINCT**: Retrieving unique values
- **LIMIT/TOP**: Controlling result set size

2.2 Data Manipulation Language (DML)

- **INSERT**: Adding new records to tables
- **UPDATE**: Modifying existing data
- **DELETE**: Removing records from tables
- **Transaction Control**: COMMIT, ROLLBACK operations

2.3 Data Definition Language (DDL)

- **CREATE**: Database and table creation
- **ALTER**: Modifying database structure
- **DROP**: Removing database objects
- **Constraints**: NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK

Level 3: Intermediate Concepts

3.1 Join Operations

- **INNER JOIN**: Returns matching records from both tables
- **LEFT JOIN**: All records from left table plus matching from right
- **RIGHT JOIN**: All records from right table plus matching from left
- **FULL OUTER JOIN**: All records from both tables
- **Self Joins**: Joining a table with itself
- **Cross Joins**: Cartesian product of two tables

3.2 Subqueries and CTEs

- **Subqueries**: Nested queries within main queries
- **Common Table Expressions (CTEs)**: Temporary named result sets
- **Recursive CTEs**: For hierarchical data processing
- **Correlated vs Non-correlated Subqueries**

3.3 Data Types and Functions

- **String Functions**: LEN, TRIM, SUBSTRING, UPPER, LOWER
- **Mathematical Functions**: ROUND, ABS, CEIL, FLOOR
- **Date/Time Functions**: NOW(), DATE(), YEAR(), MONTH()
- **Aggregate Functions**: Advanced usage with GROUP BY and HAVING

Level 4: Advanced Query Techniques

4.1 Window Functions

- **Ranking Functions**: ROW_NUMBER(), RANK(), DENSE_RANK()
- **Aggregate Window Functions**: SUM(), AVG(), COUNT() OVER()
- **Navigation Functions**: LEAD(), LAG(), FIRST_VALUE(), LAST_VALUE()
- **PARTITION BY**: Dividing result sets into groups
- **Window Frames**: ROWS BETWEEN, RANGE BETWEEN

4.2 Advanced Data Manipulation

- **Pivoting**: Converting rows to columns
- **Unpivoting**: Converting columns to rows
- **CASE Statements**: Conditional logic in queries
- **Complex Joins**: Multiple table joins with complex conditions
- **Set Operations**: UNION, INTERSECT, EXCEPT

Level 5: Database Design and Normalization

5.1 Normalization

- **First Normal Form (1NF)**: Atomic values and primary keys
- **Second Normal Form (2NF)**: Eliminating partial dependencies
- **Third Normal Form (3NF)**: Removing transitive dependencies
- **Boyce-Codd Normal Form (BCNF)**: Advanced normalization
- **Fourth Normal Form (4NF)**: Eliminating multi-valued dependencies

5.2 Database Design Patterns

- **Single Table Inheritance (STI)**
- **Class Table Inheritance (CTI)**
- **Entity-Attribute-Value (EAV)**
- **Star Schema**: Data warehouse design pattern

- **Snowflake Schema**: Normalized dimension tables

Level 6: Performance Optimization

6.1 Indexing Strategies

- **Clustered Indexes**: Physical data ordering
- **Non-clustered Indexes**: Separate index structures
- **Composite Indexes**: Multi-column indexes
- **Index Fragmentation**: Maintenance and optimization
- **Query Execution Plans**: Understanding database optimization

6.2 Query Performance Tuning

- **Avoiding SELECT ***: Selecting only needed columns
- **Sargable Queries**: Search argument able conditions
- **Join Optimization**: Efficient join strategies
- **Statistics Management**: Keeping database statistics updated
- **Partitioning**: Dividing large tables for better performance

Level 7: Advanced Database Objects

7.1 Stored Procedures and Functions

- **Stored Procedures**: Precompiled SQL code for reuse
- **User-Defined Functions**: Custom functions for data processing
- **Parameters**: Input and output parameters
- **Error Handling**: TRY-CATCH blocks and error management

7.2 Views and Triggers

- **Views**: Virtual tables for data abstraction
- **Materialized Views**: Physically stored view results
- **Triggers**: Automated code execution on data changes
- **DML Triggers**: INSERT, UPDATE, DELETE triggers
- **DDL Triggers**: Schema change triggers

Level 8: Transaction Management and Concurrency

8.1 ACID Properties

- **Atomicity**: All-or-nothing transaction execution
- **Consistency**: Maintaining database integrity
- **Isolation**: Transaction independence
- **Durability**: Permanent transaction effects

8.2 Concurrency Control

- **Isolation Levels**: Read Uncommitted, Read Committed, Repeatable Read, Serializable
- **Locking Mechanisms**: Shared locks, exclusive locks
- **Deadlock Prevention and Resolution**
- **Optimistic vs Pessimistic Concurrency Control**

Level 9: Security and Access Control

9.1 Authentication and Authorization

- **Logins and Users**: Server-level and database-level principals
- **Server Roles**: sysadmin, dbcreator, securityadmin
- **Database Roles**: db_owner, db_datareader, db_datawriter
- **Object-level Permissions**: GRANT, REVOKE, DENY

9.2 Security Best Practices

- **Principle of Least Privilege**: Minimal necessary permissions
- **Role-based Access Control**: Managing permissions through roles
- **SQL Injection Prevention**: Parameterized queries and input validation
- **Data Encryption**: Protecting sensitive data at rest and in transit

Level 10: Advanced Topics and Specialized Areas

10.1 Complex Query Patterns

- **Recursive Queries**: Hierarchical data processing
- **Graph Traversal**: Working with connected data structures
- **Analytical Functions**: Advanced statistical operations
- **Time Series Analysis**: Working with temporal data patterns

10.2 Modern SQL Extensions

- **JSON Support**: Working with semi-structured data
- **XML Processing**: Handling XML data types and operations
- **Full-text Search**: Advanced text searching capabilities
- **Spatial Data**: Geographic and geometric data processing

Level 11: Database Architecture and Design Patterns

11.1 Database Design Patterns

- **Normalized Form Pattern**: Reducing redundancy and anomalies
- **Denormalization**: Trading storage for query performance
- **Audit Log Pattern**: Tracking data changes over time
- **Versioning Pattern**: Managing data versions
- **Composite Key Pattern**: Multi-column primary keys

11.2 Enterprise Patterns

- **Data Warehouse Design**: OLAP vs OLTP considerations
- **Master Data Management**: Consistent enterprise data
- **Data Lake Architecture**: Handling diverse data types
- **Microservices Data Patterns**: Database per service

Level 12: SQL vs NoSQL Considerations

12.1 Comparative Analysis

- **Relational vs Non-relational**: Structured vs flexible schemas
- **ACID vs BASE**: Consistency vs availability trade-offs
- **Vertical vs Horizontal Scaling**: Growth strategies
- **Use Case Considerations**: When to choose SQL vs NoSQL

12.2 Database Technology Evolution

- **NewSQL Databases**: Combining SQL benefits with NoSQL scalability
- **Multi-model Databases**: Supporting multiple data models
- **Cloud-native Databases**: Serverless and managed database services
- **Distributed SQL**: Handling global-scale applications

Level 13: Best Practices and Standards

13.1 SQL Coding Standards

- **Naming Conventions**: Consistent table and column naming
- **Code Formatting**: Readable query structure
- **Documentation**: Comments and query explanations
- **Version Control**: Managing SQL code changes

13.2 Development Best Practices

- **Code Reusability**: Creating maintainable SQL code
- **Error Handling**: Robust error management strategies
- **Testing Strategies**: Unit testing and integration testing for SQL
- **Performance Monitoring**: Continuous performance assessment

This comprehensive topic overview covers SQL from fundamental concepts through advanced enterprise-level patterns and considerations. Each level builds upon previous knowledge, providing a structured learning path for mastering SQL across all proficiency levels, from basic database operations to complex distributed systems and modern database architectures.

1. https://aws.amazon.com/what-is/sql/
2. https://www.w3schools.com/sql/sql_intro.asp
3. https://www.youtube.com/watch?v=3s0IFtUrhSQ
4. https://popsql.com/blog/normalization-in-sql
5. https://www.geeksforgeeks.org/sql/foreign-key-constraint-in-sql/
6. https://hightouch.com/sql-dictionary/sql-foreign-key
7. https://www.geeksforgeeks.org/sql/sql-join-set-1-inner-left-right-and-full-joins/
8. https://support.microsoft.com/en-us/office/access-sql-basic-concepts-vocabulary-and-syntax-444d0303-cde1-424e-9a74-e8dc3e460671
9. https://www.w3schools.com/sql/sql_aggregate_functions.asp
10. https://www.geeksforgeeks.org/sql/aggregate-functions-in-sql/
11. https://www.geeksforgeeks.org/sql/sql-concepts-and-queries/
12. https://www.w3schools.com/sql/sql_constraints.asp
13. https://www.w3schools.com/sql/sql_join.asp
14. https://datalemur.com/sql-tutorial/sql-cte-subquery
15. https://learn.microsoft.com/en-us/sql/t-sql/queries/with-common-table-expression-transact-sql?view=sql-server-ver17
16. https://airbyte.com/data-engineering-resources/advanced-sql-concepts
17. https://builtin.com/data-science/advanced-sql
18. https://www.geeksforgeeks.org/mysql/various-string-numeric-and-date-time-functions-in-mysql/
19. https://www.c-sharpcorner.com/article/sql-function-explained/

20. https://decodingdatascience.com/mastering-sql-window-functions-advanced-techniques-for-data-analysis/
21. https://dev.to/gervaisamoah/advanced-sql-part-1-window-functions-explained-with-precision-146o
22. https://www.geeksforgeeks.org/sql/window-functions-in-sql/
23. https://mode.com/sql-tutorial/sql-window-functions/
24. https://www.dataforgelabs.com/advanced-sql-concepts/complex-sql-queries
25. https://www.simplilearn.com/tutorials/sql-tutorial/what-is-normalization-in-sql
26. https://en.wikipedia.org/wiki/Database_normalization
27. https://www.geeksforgeeks.org/system-design/design-patterns-for-relational-databases/
28. https://www.integrate.io/blog/complete-guide-to-database-schema-design-guide/
29. https://www.dataforgelabs.com/advanced-sql-concepts/sql-performance-tuning
30. https://www.geeksforgeeks.org/sql/sql-performance-tuning/
31. https://digma.ai/how-indexing-enhances-query-performance/
32. https://learn.microsoft.com/en-us/sql/relational-databases/indexes/reorganize-and-rebuild-indexes?view=sql-server-ver17
33. https://datalemur.com/sql-tutorial/best-practices-for-writing-sql-queries
34. https://docs.ed-fi.org/community/sdlc/code-contribution-guidelines/coding-standards/sql-coding-standards/
35. https://www.sirkel.net/triggers-views-stored-procedures-functions/
36. https://www.halvorsen.blog/documents/technology/database/resources/Views-Stored Proecedures-Triggers in SQL Server.pdf
37. https://www.w3schools.com/sql/sql_stored_procedures.asp
38. https://eecs.csuohio.edu/~sschung/CIS408/ViewSPTriggerMySQL.pdf
39. https://learn.microsoft.com/en-us/sql/t-sql/statements/create-trigger-transact-sql?view=sql-server-ver17
40. https://memgraph.com/blog/acid-transactions-meaning-of-isolation-levels
41. https://www.databricks.com/glossary/acid-transactions
42. https://www.geeksforgeeks.org/dbms/acid-properties-in-dbms/
43. https://mariadb.com/docs/general-resources/database-theory/acid-concurrency-control-with-transactions
44. https://learn.microsoft.com/en-us/sql/relational-databases/sql-server-transaction-locking-and-row-versioning-guide?view=sql-server-ver17
45. https://blog.bytebytego.com/p/a-guide-to-database-transactions
46. https://satoricyber.com/sql-server-security/sql-server-roles/
47. https://www.gethynellis.com/2024/06/sql-server-security-model.html
48. https://learn.microsoft.com/en-us/sql/relational-databases/security/authentication-access/server-level-roles?view=sql-server-ver17
49. https://learn.microsoft.com/en-us/sql/relational-databases/security/permissions-database-engine?view=sql-server-ver17
50. https://calcomsoftware.com/default-permissions-in-sql-server-public-role/

51. https://www.mssqltips.com/sqlservertip/8045/sql-server-database-and-server-roles-for-security-and-permissions/
52. https://www.geeksforgeeks.org/sql/cte-in-sql/
53. https://towardsdatascience.com/two-advanced-sql-techniques-that-can-drastically-improve-your-queries-81a97c92ddd0/
54. https://vertabelo.com/blog/database-design-patterns/
55. https://vertabelo.com/blog/database-design-trends/
56. https://microservices.io/patterns/data/database-per-service.html
57. https://www.geeksforgeeks.org/sql/difference-between-sql-and-nosql/
58. https://www.scylladb.com/learn/nosql/nosql-vs-sql/
59. https://www.talend.com/resources/sql-vs-nosql/
60. https://www.ibm.com/think/topics/sql-vs-nosql
61. https://www.dbvis.com/thetable/best-practices-for-sql-coding-and-development/
62. https://docs.telemetry.mozilla.org/concepts/sql_style
63. https://www.metabase.com/learn/sql/working-with-sql/sql-best-practices
64. https://www.reddit.com/r/webdev/comments/xb4bbi/sql_vs_nosql_which_one_to_choose/
65. https://www.youtube.com/watch?v=efrR9eP2hUo
66. https://www.geeksforgeeks.org/dbms/introduction-of-database-normalization/
67. https://learn.microsoft.com/en-us/troubleshoot/microsoft-365-apps/access/database-normalization-description
68. https://www.simplilearn.com/tutorials/sql-tutorial/what-is-sql
69. https://www.freecodecamp.org/news/database-normalization-1nf-2nf-3nf-table-examples/
70. https://www.khanacademy.org/computing/computer-programming/sql
71. https://www.reddit.com/r/SQL/comments/1g37oki/what_are_considered_as_advanced_sql_skills/
72. https://pragmaticworks.com/blog/inner-vs.-outer-in-sql
73. https://dev.mysql.com/doc/refman/8.4/en/create-table-foreign-keys.html
74. https://learn.microsoft.com/en-us/sql/relational-databases/tables/primary-and-foreign-key-constraints?view=sql-server-ver17
75. https://www.reddit.com/r/SQL/comments/v5gmxx/can_someone_explain_me_in_simple_terms_about/
76. https://stackoverflow.com/questions/706972/difference-between-cte-and-subquery
77. https://stackoverflow.com/questions/5706437/whats-the-difference-between-inner-join-left-join-right-join-and-full-join
78. https://learnsql.com/blog/cte-vs-subquery/
79. https://docs.snowflake.com/en/sql-reference/constraints-overview
80. https://www.youtube.com/watch?v=2HVMiPPuPIM
81. https://www.reddit.com/r/dataengineering/comments/1bpgbpu/subqueries_vs_cte_whats_your_preference/
82. https://www.cockroachlabs.com/docs/stable/foreign-key
83. https://learn.microsoft.com/en-us/sql/relational-databases/performance/joins?view=sql-server-ver17
84. https://www.youtube.com/watch?v=sqSwZKkcURo

85. https://www.emergentsoftware.net/blog/sql-performance-tuning-best-practices/
86. https://www.sqltutorial.org/sql-functions/
87. https://stackoverflow.com/questions/30856674/speeding-up-a-sql-query-with-indexes
88. https://www.w3schools.com/sql/sql_ref_sqlserver.asp
89. https://www.reddit.com/r/dataengineering/comments/1axd7cy/what_are_your_top_sql_query_optimization_tips/
90. https://www.youtube.com/watch?v=Hf9O8oNnHg8
91. https://www.reddit.com/r/SQL/comments/iq3fjm/what_would_you_consider_beginner_intermediate_and/
92. https://stackoverflow.com/questions/2054130/what-is-advanced-sql
93. https://www.mongodb.com/resources/basics/databases/acid-transactions
94. https://dev.to/bilelsalemdev/advanced-sql-mastering-query-optimization-and-complex-joins-4gph
95. https://www.netwrix.com/how-to-check-user-roles-in-sql-server.html
96. https://stackoverflow.com/questions/5951245/are-there-any-published-coding-style-guidelines-for-sql
97. https://www.integrate.io/blog/the-sql-vs-nosql-difference/
98. https://www.freecodecamp.org/news/how-to-design-structured-database-systems-using-sql-full-book/
99. https://www.mongodb.com/resources/basics/databases/nosql-explained/nosql-vs-sql
100. https://learn.microsoft.com/en-us/azure/architecture/patterns/