Learning SQL (Structured Query Language) involves understanding several key concepts. Here's a list of fundamental concepts you should learn when starting with SQL:

1. \*\*Database Basics\*\*:

- Understanding what a database is.

- Different types of databases (relational, NoSQL, etc.).

- Basic terminology like tables, rows, columns, and indexes.

2. \*\*SQL Syntax\*\*:

- Learning SQL commands such as SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, etc.

- Understanding the structure of SQL queries and statements.

- Learning how to terminate SQL statements (usually with a semicolon).

3. \*\*Data Types\*\*:

- Different data types in SQL (e.g., integer, float, string, date, etc.).

- Understanding the appropriate use of each data type.

4. \*\*Database Design\*\*:

- Understanding database normalization (1NF, 2NF, 3NF, BCNF).

- Relational database concepts (foreign keys, primary keys, relationships).

- Entity-Relationship (ER) modeling.

5. \*\*Querying Data\*\*:

- SELECT statement for retrieving data from tables.

- WHERE clause for filtering data.

- ORDER BY clause for sorting data.

- Aggregate functions (COUNT, SUM, AVG, MAX, MIN) for data summarization.

6. \*\*Data Manipulation\*\*:

- INSERT statement for adding new rows into tables.

- UPDATE statement for modifying existing data.

- DELETE statement for removing data from tables.

7. \*\*Joins\*\*:

- INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL OUTER JOIN.

- Understanding how to join tables based on common columns.

8. \*\*Subqueries\*\*:

- Understanding nested queries and their applications.

9. \*\*Constraints\*\*:

- PRIMARY KEY constraint.

- FOREIGN KEY constraint.

- UNIQUE constraint.

- CHECK constraint.

- NOT NULL constraint.

10. \*\*Indexing\*\*:

- Understanding indexes and their impact on query performance.

- Creating and managing indexes.

11. \*\*Transactions\*\*:

- Understanding ACID properties (Atomicity, Consistency, Isolation, Durability).

- Transaction control statements (BEGIN TRANSACTION, COMMIT, ROLLBACK).

12. \*\*Views and Stored Procedures\*\*:

- Creating and managing views.

- Creating and managing stored procedures.

13. \*\*Security\*\*:

- Understanding SQL injection and how to prevent it.

- User authentication and authorization.

- Granting and revoking privileges.

14. \*\*Normalization and Optimization\*\*:

- Optimizing SQL queries for performance.

- Understanding denormalization when needed for performance tuning.

15. \*\*Advanced Topics\*\* (optional):

- Window functions.

- Common Table Expressions (CTEs).

- Recursive queries.

- Analytic functions.

These concepts should give you a solid foundation in SQL. Remember that practice is essential in mastering SQL, so be sure to work on plenty of exercises and real-world projects to reinforce your learning.

That sounds like a fantastic project! Working with the Health Insurance Marketplace database will give you a comprehensive practice experience. Here’s a structured plan to create 100 queries that cover a wide range of SQL skills:

1. **Basic SELECT Queries (1-20):**
   * Start with simple queries to select different columns from a single table.
   * Use WHERE clauses to filter data.
2. **Aggregate Functions (21-40):**
   * Practice using COUNT, SUM, AVG, MAX, and MIN to summarize data.
   * Group results with GROUP BY and filter with HAVING.
3. **JOIN Operations (41-60):**
   * Use INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL OUTER JOIN to combine tables.
   * Explore different join conditions and multi-table joins.
4. **Subqueries and Nested SELECTs (61-70):**
   * Write queries that use subqueries in the SELECT, FROM, and WHERE clauses.
   * Practice correlated subqueries.
5. **Data Manipulation (71-80):**
   * Insert new records, update existing ones, and delete records.
   * Use transactions to ensure data integrity.
6. **Advanced Filtering (81-90):**
   * Utilize LIKE, IN, and BETWEEN operators.
   * Work with NULL values and COALESCE function.
7. **Window Functions and Analytic Queries (91-95):**
   * Use OVER() with PARTITION BY, ORDER BY, and frame specifications.
   * Calculate running totals, rankings, and moving averages.
8. **Views, Indexes, and Stored Procedures (96-100):**
   * Create and query views.
   * Understand the impact of indexes on query performance.
   * Write basic stored procedures.

As you progress, make sure to test and validate each query to ensure it’s returning the expected results. This will not only help you learn SQL but also create a robust portfolio project showcasing your skills. Good luck with your SQL journey! 🚀