LinkedIn Learning

Oracle Database 12c: Basic SQL

Course completed by Siddharth Bahekar

Feb 17, 2024 at 02:14AM UTC • 3 hours 12 minutes

Top skills covered

SQL

Oracle Database

Head of Content Strategy, Learning

COURSE COMPLETION

Certificate ID: 8f5ff7d99182e629b464916d8d1c5ce65a01f4e3026a6bfa73e17afb3e6cd361

Learning outcomes from the course -

Basic SELECT queries and functions -

In this lesson, we learned the basics of writing SQL queries to retrieve data from database tables. Here are the key points:

- Basic Select Query: Use the SELECT keyword to retrieve data, specify columns to retrieve after SELECT. Use FROM to indicate the table from which to retrieve data.
- Selecting Specific Columns: Use column names instead of * to specify individual columns. Separate column names with commas.
- Filtering Rows with WHERE Clause: Use the WHERE clause to filter rows based on conditions. Conditions can involve comparisons, such as greater than (>), or equality (=).
- Filtering with Multiple Conditions: Combine conditions using AND or OR. Use parentheses to control the order of evaluation.
- Filtering with String Values: Use the LIKE keyword for string pattern matching. % is a wildcard representing any sequence of characters.
- Filtering with Range of Values: Use BETWEEN for numeric ranges. Combine multiple conditions for more complex filtering.
- Filtering with IN and NOT IN: Use IN for multiple value matches. Use NOT IN to exclude specific values.
- Filtering with Dates: Use date functions like TO_DATE to filter based on dates.
- Filtering with Exact String Matches: Use = for exact string matches. Consider case sensitivity when filtering.
- Using DISTINCT: Use DISTINCT to filter out duplicate rows.
- Performing Calculations: Perform arithmetic operations in the SELECT clause.
- Column Aliases: Use AS to assign meaningful names to columns in the output.
- Viewing Table Structure: Use DESC or DESCRIBE command to view table structure.
- Sorting Results with ORDER BY: Use ORDER BY to sort results. Specify column names and sort order (ascending or descending).
- Sorting by Multiple Columns: Order results by multiple columns to achieve a combined sort.

String, Number and date functions -

In the Oracle SQL course, we covered various functions to manipulate strings, numbers, and dates.

String Functions:

- SUBSTR: Used to extract a portion of a string. For example, you can get characters starting from a specific position.
- Length: Determines the length (number of characters) of a string.
- CONCAT: Combines two or more strings into a single string.
- LOWER and UPPER: Convert letters in a string to lowercase or uppercase, respectively.
- INSTR: Finds the position of a substring within a string.
- RPAD and LPAD: Right and left pads a string with a specified character to reach a specified length.
- Number Functions:
- TO_CHAR: Converts numbers to characters. Useful for formatting numbers in a specific way.
- ROUND: Rounds a number to a specified number of decimal places.
- TRUNC: Truncates a number to a specified number of decimal places.
- Date Functions:
- TO_CHAR: Converts dates to characters with a specified format.
- Oracle Dual Table: A special one-row, one-column table often used to perform functions without referencing an actual table.
- SYSDATE: Returns the current date and time.
- SYSTIMESTAMP: Returns the current timestamp, including fractional seconds.
- MONTHS_BETWEEN: Calculates the number of months between two dates.
- ADD_MONTHS: Adds a specified number of months to a date.

Joining table data -

I learned about different types of joins, each serving a distinct purpose in combining information from multiple tables. An inner join, which is the default join type, merges rows that have matching values in both tables, filtering out unmatched rows. A left join includes all rows from the left table and matching rows from the right table, filling in with NULLs for non-matching entries on the right. Conversely, a right join includes all rows from the right table, and matching rows from the left, again employing NULLs for non-matching entries on the left. An outer join, which encompasses both left and right joins, retains all rows from both tables, populating with NULLs when matches are absent. Employing table aliases in joins enhances query readability and simplifies syntax, especially when working with complex queries involving multiple tables. Understanding these join types equips me with the ability to seamlessly merge data from various tables, offering a comprehensive view of the information stored in an Oracle database.

Grouping the data -

I learned about manipulating data using Data Definition Language (DDL) and Data Manipulation Language (DML) commands. The course introduced the essential DDL commands, focusing on creating and altering the structure of database objects. I learned how to create a simple table, defining its columns and data types, a fundamental step in database design. The course also covered Data Manipulation Language (DML) commands, including inserting new rows into a table, a basic yet crucial operation in populating a database. Understanding the concepts of Commit and Rollback added a layer of control, ensuring the integrity of my changes, and introducing me to the concept of transactions. I got to know the functionality of truncating all data from a table. Additionally, I learned how to add and drop columns from a table, giving me the flexibility to adapt the database structure as per requirements.

Advanced Features -

The concept of a primary key was covered, showing its important role in uniquely identifying records within a table. The significance of the Not Null constraint was focused, highlighting its role in ensuring that essential data fields are never left empty. I learned check constraints, understand how they help set rules for data entry, keeping the data reliable. Another key topic was foreign key constraints, which I learned are essential for building connections between tables, ensuring data consistency across the database. Subqueries in SQL are a way to nest multiple commands inside one query. For example, if we want to find employees in the Marketing department, instead of directly specifying the department ID, we can use a subquery. It works like a mini query within the main one. Subqueries provide a powerful tool to manage and extract data more efficiently.