Lecture 3: Question Bank

Topics: "SQL: SELECT Statement, Restricting Data, Sorting Data, Single-Row Functions"

1	Explain the practice of placing clauses on separate lines in SQL statements. How does this contribute to the maintainability of code.
2	Discuss the importance of not abbreviating or splitting keywords across lines in SQL statements. How does adherence to this guideline impact the clarity of SQL code?
3	Explain the significance of SQL statements not being case-sensitive. How does this impact the writing and execution of SQL queries?
4	Why is it recommended to write SQL statements on one or more lines? How does this practice contribute to the readability of SQL code?
5	Describe the role of indents in SQL statements and how they contribute to code readability. Provide an example illustrating the use of indents.
6	What is the basic syntax for a SELECT statement in SQL?
7	What integrity rules exist in the DBMS?
8	How do you select specific columns from a table in SQL?
9	Explain the concept of operator precedence in SQL and provide an example.
10	How can parentheses be used to control the order of evaluation in SQL expressions?
11	How does SQL handle arithmetic expressions containing NULL values, and why is it important to be aware of this when working with databases?
12	Explain the difference between NULL values, zero, and blank spaces in SQL, and how this distinction is crucial in database management.
13	Can you provide a use case scenario where NULL values in a SQL database would be appropriate, and how would you handle them in your queries?
14	What is a column alias in SQL, and how does it contribute to enhancing the readability of query results?
15	What is a column alias in SQL, and how does it contribute to enhancing the readability of query results?
16	How do you restrict the rows returned in a SQL query and provide an example using the WHERE clause?
17	Explain the significance of the ORDER BY clause in SQL, and provide an example of sorting in descending order.
18	How do you sort the result set of a SQL query in ascending order, and where in the SELECT statement does the ORDER BY clause come?
19	Difference between single row function and multiple row function.
20	Explain Entity, Entity Type, and Entity Set in DBMS?
21	What are super, primary, candidate, and foreign keys?
22	Display first three characters from first name.
23	Display last two character from last name.
24	Display all the first name and position of a in that name (first occurrence of a).
25	Display all the first name and position of a in that name (second occurrence of a).
26	Display all the name which contain two or more number of a 's in the first name.
27	Difference between SUBSTR and INSTR function.
28	Difference between REPLACE and TRANSLATE function.
29	Difference between LPAD and RPAD.
30	Difference between LTRIM and RTRIM.
31	Display all the first name and its length.
32	List out all the number functions in Oracle.
33	List out all the Date functions in Oracle.

34	Display all the first name and their total year of experience. Rename first name column name as name and second column name as Year of Exp.
35	How to display months between two given date.
36	Write a query to display today's date.
37	Write a query to display the date after 3 months from today.
38	Display last date of the current month.
39	Display the upcoming Wednesday date.

ANSWERS

Q1. Explain the practice of placing clauses on separate lines in SQL statements. How does this contribute to the maintainability of code.
Ans: Placing clauses on separate lines in SQL statements enhances code maintainability. Each clause serves a specific purpose, and separating them onto different lines allows developers to easily identify and modify specific parts of the query without affecting the entire statement. This practice promotes clean and organized code, making it simpler to troubleshoot and update as needed.

Q2. Discuss the importance of not abbreviating or splitting keywords across lines in SQL statements. How does adherence to this guideline impact the clarity of SQL code?
Ans: Abbreviating or splitting keywords across lines in SQL statements can lead to confusion and errors. Complete keywords ensure that the SQL parser correctly identifies the intended command. Furthermore, keeping keywords intact contributes to the overall clarity of the code, making it more comprehensible to developers and reducing the likelihood of syntax errors.

Q3. Explain the significance of SQL statements not being case-sensitive. How does this impact the writing and execution of SQL queries?
Ans: SQL statements being case-insensitive means that the keywords and identifiers used in queries are not sensitive to the case in which they are written. For example, 'SELECT' is equivalent to 'select'. This characteristic simplifies query writing and execution, as developers do not have to be concerned about the case of SQL keywords, making the code more flexible and easier to read.

Q4. Why is it recommended to write SQL statements on one or more lines? How does this practice contribute to the readability of SQL code?
Ans: Writing SQL statements on one or more lines enhances code readability. It allows developers to structure their queries in a more organized manner, making it easier to understand and maintain. Breaking down statements into multiple lines also facilitates the identification of individual clauses, which can be particularly useful when dealing with complex queries.

Q5. Describe the role of indents in SQL statements and how they contribute to code readability. Provide an example illustrating the use of indents.
Ans: Indents are used in SQL statements to enhance readability by visually indicating the structure of the query. They help identify nested clauses and make the code more visually appealing. For example:

```sql
SELECT column1, column2
FROM table1
WHERE condition1 AND condition2;
```

In this example, indents are used to align the columns, FROM, and WHERE clauses, making it easier to understand the logical structure of the query.

Q6. What is the basic syntax for a SELECT statement in SQL?
Ans: The basic syntax for a SELECT statement in SQL is:

```
SELECT * | {[DISTINCT] column | expression [alias], ...}
FROM table;
```

Q7. What integrity rules exist in the DBMS?
Ans: There are two major integrity rules that exist in the DBMS.

- *Entity Integrity:* This states a very important rule that the value of a Primary key can never have a NULL value.
- *Referential Integrity:* This rule is related to the Foreign key which states that either the value of a Foreign key is a NULL value or it should be the primary key of any other relation.

Q8. How do you select specific columns from a table in SQL?
Ans: To select specific columns from a table, you can use the SELECT statement with the column names, as shown in the following example:
SELECT department_id, location_id FROM departments;

Q9. Explain the concept of operator precedence in SQL and provide an example.
Ans: Operator precedence in SQL determines the order in which different operators are evaluated in an expression. In SQL, multiplication and division take priority over addition and subtraction.

Q10. How can parentheses be used to control the order of evaluation in SQL expressions?
Ans: Parentheses in SQL are used to force prioritized evaluation and to clarify statements. They allow you to control the order in which operators are applied.

Q11. How does SQL handle arithmetic expressions containing NULL values, and why is it important to be aware of this when working with databases?
Ans: In SQL, arithmetic expressions involving NULL values evaluate to NULL. For example, if you perform any mathematical operation with a NULL value, the result will be NULL. It's important to be aware of this behavior to prevent unexpected results in calculations. When working with such expressions, it's advisable to use functions like COALESCE or NULLIF to handle NULL values explicitly and provide meaningful results in your queries.

Q12. Explain the difference between NULL values, zero, and blank spaces in SQL, and how this distinction is crucial in database management.
Ans: In SQL, a NULL value represents the absence of a value or an unknown value, whereas zero is a numerical value and a blank space is a character value. The distinction is crucial because NULL indicates a lack of information or an undefined state, while zero and blank spaces represent actual known values. Understanding and handling NULL values appropriately is essential for accurate data representation and analysis in SQL, preventing misinterpretations or errors in query results.

Q13. Can you provide a use case scenario where NULL values in a SQL database would be appropriate, and how would you handle them in your queries?
Ans: Certainly. Consider a scenario where you have an employee database, and some employees may not have assigned a project yet. In this case, the "project_id" column for those employees would be appropriate to store NULL values. This indicates that the employee is not currently assigned to any project. When querying, you would need to consider these NULL values and use appropriate handling techniques to get accurate results, such as using the IS NULL or IS NOT NULL conditions.

Q14. What is a column alias in SQL, and how does it contribute to enhancing the readability of query results?
Ans: In SQL, a column alias serves the purpose of renaming a column heading. It is particularly valuable when dealing with calculations or when a more descriptive name is desired for a column. A column alias immediately follows the column name, and you can use the optional "AS" keyword between the column name and alias. It becomes necessary to enclose the alias in double quotation marks if it contains spaces, special characters, or is case-sensitive.

Q15. What is a column alias in SQL, and how does it contribute to enhancing the readability of query results?

Ans: In SQL, a column alias serves the purpose of renaming a column heading. It is particularly valuable when dealing with calculations or when a more descriptive name is desired for a column. A column alias immediately follows the column name, and you can use the optional "AS" keyword between the column name and alias. It becomes necessary to enclose the alias in double quotation marks if it contains spaces, special characters, or is case-sensitive.

Q16. How do you restrict the rows returned in a SQL query and provide an example using the WHERE clause?

Ans: You can restrict the rows returned in a SQL query using the WHERE clause.

For instance:

```
SELECT employee_id, last_name, job_id, department_id
FROM employees
WHERE department_id = 90;
```

Q17. Explain the significance of the ORDER BY clause in SQL, and provide an example of sorting in descending order.

Ans: The ORDER BY clause in SQL is used to sort rows in the result set. It defaults to ascending order (ASC), but DESC can be used for descending order.

```
SELECT last_name, job_id, department_id, hire_date
FROM employees
ORDER BY hire_date DESC;
```

Q18. How do you sort the result set of a SQL query in ascending order, and where in the SELECT statement does the ORDER BY clause come?

Ans: To sort in ascending order, you use the ASC keyword with the ORDER BY clause, and it comes last in the SELECT statement. For instance:

```
SELECT last_name, job_id, department_id, hire_date FROM employees
ORDER BY hire_date ASC;
```

Q19. Difference between single row function and multiple row function.

Ans:

Single Row Function
Input Single Row
Output Single Row

Multiple Row Function
Input Multiple Rows
Output Single Rows

Q20. Explain Entity, Entity Type, and Entity Set in DBMS?

Ans: The entity is an object, place, or thing which has its independent existence in the real world and about which data can be stored in a database. For Example, any person, book, etc.

Entity Type is a collection of entities that have the same attributes. For Example, the STUDENT table contains rows in which each row is an entity holding the attributes like name, age, and id of the students, hence STUDENT is an Entity Type that holds the entities having the same attributes.

Entity Set is a collection of entities of the same type. For Example, A collection of the employees of a firm.

Q21. What are super, primary, candidate, and foreign keys?
Ans: A super key is a set of attributes of a relation schema upon which all attributes of the schema are functionally dependent. No two rows can have the same value of super key attributes.
A Candidate key is a minimal superkey, i.e., no proper subset of Candidate key attributes can be a superkey.
A Primary Key is one of the candidate keys. One of the candidate keys is selected as most important and becomes the primary key. There cannot be more than one primary key in a table..
A Foreign key is a field (or collection of fields) in one table that uniquely identifies a row of another table.

Q22. Display first three characters from first name.
Ans: SELECT substr(First_name,1,3)"First_Name" FROM employees;

Q23. Display last two character from last name.
Ans: SELECT substr(last_name,-2,2)"LAST_NAME" FROM Employees;

Q24. Display all the first name and position of a in that name (first occurrence of a).
Ans: SELECT First_name, instr(first_name,'a',1) FROM employees;

Q25. Display all the first name and position of a in that name (second occurrence of a).
Ans: SELECT First_Name,instr(First_name,'a',1,2) FROM employees;

Q26. Display all the name which contain two or more number of a 's in the first name.
Ans:
SELECT First_Name,instr(First_name,'a',1,2) from employees where instr(First_name,'a',1,2)<>0

Q27. Difference between SUBSTR and INSTR function.
Ans:
- *SUBSTR:* It returns a specified portion of a string
- *INSTR:* It returns a character position(ie occurances of the character).

Q28. Difference between REPLACE and TRANSLATE function.
Ans:
- *Replace:* The Replace Function replaces single character with multiple characters.
- *Translate:* Translate Function replaces single character with single character only.

Q29. Difference between LPAD and RPAD.
Ans:
- LPAD : To adjust the left hand side padding of charecters,numbers.
- RPAD : To adjust the right hand side padding of charecters, numbers.

Q30. Difference between LTRIM and RTRIM.
Ans:
- LTRIM: Trim the charecters and numbers in left hand side.
- RTRIM: Trim the charecters and numbers in Right hand side.

Q31. Display all the first name and its length.
Ans: SELECT First_name, length(first_name) from employees;

Q32. List out all the number functions in Oracle.
Ans:
ROUND();
MOD();
TRUNC();
POWER();

Q33. List out all the Date functions in Oracle.
Ans:
ADD_MONTHS();
MONTHS_BETWEEN();
NEXT_DAY();
LAST_DAY();

Q34. Display all the first name and their total year of experience. rename first name column name as name and second column name as Year of Exep.
Ans:
SELECT First_Name "Name" ,round(months_between(sysdate,hire_date)/12)"Year of Experiance"
FROM employees;

Q35. How to display months between two given date.
Ans:
SELECT months_between('12-jan-2016','12-dec-2016')
FROM dual;

Q36. Write a query to display today's date.
Ans: Select sysdate from dual;

Q37. Write a query to display the date after 3 months from today.
Ans: Select add_months(sysdate, 3) from dual;

Q38. Display last date of the current month.
Ans: Select last_day(sysdate)from dual;

Q39. Display the up coming Wednesday date.
Ans: Select next_day(sysdate,'wednesday') from dual;