Arundeep Singh

SQL Assignment

Date: 10/11/2015

1. What are different types of joins and explain them?

SQL JOIN is a method to retrieve data from two or more database tables.

There are total of five JOINs:

1. Inner Join
2. Outer Join
3. Natural Join
4. Cross Join
5. Self Join
6. Inner Join: In this join all the records that match the condition in both the tables and records in both the tables that do not match are not reported. ONLY the matching entries in BOTH the tables SHOULD be listed.
7. Outer Join: The matched rows from one table and all rows in the other tables. There are 3 kinds of Outer Join:
8. Left Join: This join returns all the rows from the left table in conjunction with the matching rows from the right table. If there are no columns matching in the right table, it returns NULL values.
9. Right Join: This JOIN returns all the rows from the right table in conjunction with the matching rows from the left table. If there are no columns matching in the left table, it returns NULL values.
10. Full Join: This JOIN combines LEFT JOIN and RIGHT JOIN. It returns row from either table when the conditions are met and returns NULL value when there is no match.
11. Natural Join: It is based on two conditions:
12. The JOIN is made on all the columns with the same name for equality.
13. Removes duplicate columns from the result.
14. Cross Join: It is the Cartesian product of the two tables involved. We don’t use it that much though.
15. Self Join: It is not a different from of JOIN, rather it is a JOIN (INNER, OUTER, etc.) of a table to itself.
16. What is subquery? Show an example.

A **subquery** is a query within another query, also known as a nested query. A **subquery** is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved. **Subqueries** are used with the SELECT, INSERT, UPDATE, and DELETE statements.

Example from given assignment, where we have to find out first name, last name and employee id of top 10 employees based on their salary:

SELECT \* FROM (SELECT FIRST\_NAME, LAST\_NAME, EMPLOYEE\_ID, SALARY FROM HR.EMPLOYEES ORDER BY SALARY DESC) WHERE ROWNUM <=10;

1. What is group by?

The GROUP BY statement is used in conjunction with the aggregate functions to group the result-set by one or more columns.

1. What is the difference between order by and group by?

The ORDER BY clause’s purpose is to sort the query result by specific columns whereas the GROUP BY clause’s purpose is to summarize unique combinations of columns values.

1. What is the difference between left outer join and right outer join?

LEFT JOIN returns all the rows from the left table in conjunction with the matching rows from the right table. If there are no columns matching in the right table, it returns NULL values whereas RIGHT JOIN returns all the rows from the right table in conjunction with the matching rows from the left table. If there are no columns matching in the left table, it returns NULL values.

1. What is self join? Explain with an example?

A self-join is a query in which a table is joined (compared) to itself.  Self-joins are used to compare values in a column with other values in the *same column in the same table*.  One practical use for self-joins:  obtaining running counts and running totals in an SQL query.

To write the query, select from the same table listed twice with different aliases, set up the comparison, and eliminate cases where a particular value would be equal to itself.

Example is:

SELECT DISTINCT c1.ContactName, c1.Address, c1.City, c1.Region FROM Customers AS c1, Customers AS c2 WHERE c1.Region = c2.Region AND c1.ContactName <> c2.ContactName ORDER BY c1.Region, c1.ContactName;

1. What is stored procedure?

A stored procedure is nothing more than prepared SQL code that you save so you can reuse the code over and over again.  So if you think about a query that you write over and over again, instead of having to write that query each time you would save it as a stored procedure and then just call the stored procedure to execute the SQL code that you saved as part of the stored procedure.

1. What is stored function?

A stored function is a special kind of stored program function that returns a single value. You use stored functions to encapsulate common formulas or business rules that may be reusable among SQL statements or stored programs. Different from a stored procedure, you can use a stored function in SQL statements wherever an expression is used. This helps improve the readability and maintainability of the procedural code.

1. What is the difference between stored procedure and function?

Difference between stored procedure and function is this that you can use a stored function in SQL statements whenever an expression is used. This helps improve readability and maintainability of the procedural code. Some other difference are:

1. Function must return a value but it is optional in procedure.
2. Functions can have only input parameters for it whereas procedures can have input/output parameters.
3. Procedure allows SELECT as well as DML(INSERT/UPDATE/DELETE) statement in it whereas Function allows only SELECT statement in it.
4. Procedures cannot be utilized in a SELECT statement whereas Function can be embedded in a SELECT statement.
5. Stored Procedures cannot be used in the SQL statements anywhere in the WHERE/HAVING/SELECT section whereas Function can be.
6. Exception can be handled by try-catch block in a Procedure whereas try-catch block cannot be used in a Function.
7. We can go for Transaction Management in Procedure whereas we can't go in Function.
8. Functions that return tables can be treated as another rowset. This can be used in JOINs with other tables.
9. The most important feature of stored procedures over function is to retention and reuse the execution plan while in case of function it will be compiled every time.
10. What is view?

In SQL, a view is a virtual table based on the result-set of an SQL statement.

A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.

You can add SQL functions, WHERE, and JOIN statements to a view and present the data as if the data were coming from one single table.

1. What are indexes?

The CREATE INDEX statement is used to create indexes in tables.

Indexes allow the database application to find data fast; without reading the whole table.

1. What are different types of constraints? Explain them.

Constraints are the rules enforced on data columns on table. These are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the database.

Following are the commonly used constraints available in SQL:

1. NOT NULL: Ensures that a column cannot have NULL value.
2. DEFAULT: Provides a default value for a column when none is specified.
3. UNIQUE: Ensures that all values in a column are different.
4. PRIMARY: Uniquely identified each rows/records in a database table.
5. FOREIGN: Uniquely identified a rows/records in any another database table.
6. CHECK: Ensures that all values in any another column satisfy certain conditions.
7. INDEX: Use to create and retrieve data form the database very quickly.
8. Create SQL Query for the following with the HR Schema in your Oracle Instance.
9. Display all the employee id, employee name, department name, the country where this department is located?

SELECT E.EMPLOYEE\_ID, E.FIRST\_NAME, E.LAST\_NAME, D.DEPARTMENT\_NAME, L.COUNTRY\_ID FROM HR.EMPLOYEES E inner join HR.DEPARTMENTS D on E.DEPARTMENT\_ID = D.DEPARTMENT\_ID inner join HR.LOCATIONS L on D.LOCATION\_ID = L.LOCATION\_ID;

1. Find out all the employees who joined before their manager?

SELECT E1.FIRST\_NAME FROM HR.EMPLOYEES E1 JOIN HR.EMPLOYEES E2 ON (E1.MANAGER\_ID = E2.EMPLOYEE\_ID) WHERE E1.HIRE\_DATE< E2.HIRE\_DATE;

1. Find out all the employees who do not belong to any department?

SELECT \* FROM HR.EMPLOYEES WHERE DEPARTMENT\_ID is NULL;

1. Find out all the departments in which there are no employees?

SELECT DEPARTMENT\_ID, DEPARTMENT\_NAME FROM HR.DEPARTMENTS WHERE MANAGER\_ID IS NULL;

1. Find out first name, last name and employee id of top 10 employees based on their salary.

SELECT \* FROM( SELECT FIRST\_NAME, LAST\_NAME, EMPLOYEE\_ID, SALARY FROM HR.EMPLOYEES ORDER BY SALARY DESC) WHERE ROWNUM <=10;

1. Find out the bottom 10 employees based on their salary.

SELECT \* FROM( SELECT FIRST\_NAME, LAST\_NAME, EMPLOYEE\_ID, SALARY FROM HR.EMPLOYEES ORDER BY SALARY ASC) WHERE ROWNUM <=10;

1. Find out the location with most number of employees.
2. Display details of jobs in the descending order of the title.

SELECT \* FROM HR.JOBS ORDER BY JOB\_TITLE DESC;

1. Display details of the employees where commission percentage is null and salary in the range 5000 to 10000 and department is 30.

SELECT \* FROM HR.EMPLOYEES WHERE COMMISSION\_PCT IS NULL AND DEPARTMENT\_ID = 30 AND SALARY BETWEEN 5000 AND 10000;

1. Display first name of employees who joined after 2001.

SELECT FIRST\_NAME FROM HR.EMPLOYEES WHERE HIRE\_DATE>'31-DEC-01';

1. Display first name and last name after converting the first letter of each name to upper case and the rest to lower case.

SELECT INITCAP(LOWER(FIRST\_NAME)), INITCAP(LOWER(LAST\_NAME)) FROM HR.EMPLOYEES;

1. Display first name in upper case and email address in lower case for employees where the first name and email address are same irrespective of the case.

SELECT UPPER(FIRST\_NAME), LOWER(EMAIL) FROM HR.EMPLOYEES;

1. Display number of employees joined after 15th of the month.

SELECT \* FROM HR.EMPLOYEES WHERE EXTRACT(DAY FROM HIRE\_DATE)>15;

1. Display the country ID and number of cities we have in the country.

SELECT C.COUNTRY\_NAME, COUNT(L.CITY) FROM HR.COUNTRIES C LEFT JOIN HR.LOCATIONS L ON C.COUNTRY\_ID = L.COUNTRY\_ID GROUP BY (C.COUNTRY\_NAME);

1. Display years in which more than 10 employees joined.

SELECT EXTRACT(YEAR FROM HIRE\_DATE), COUNT((EXTRACT(YEAR FROM HIRE\_DATE)))

FROM HR.EMPLOYEES HAVING COUNT(EXTRACT(YEAR FROM HIRE\_DATE))>10 GROUP BY EXTRACT(YEAR FROM HIRE\_DATE);

1. Change salary of employee 115 to 8000 if the existing salary is less than 6000.

UPDATE HR.EMPLOYEES SET SALARY = 8000 WHERE EMPLOYEE\_ID=115 AND SALARY <6000;

1. Display department name and manager first name.

SELECT D.DEPARTMENT\_NAME,E.FIRST\_NAME FROM HR.EMPLOYEES E INNER JOIN HR.DEPARTMENTS D ON E.DEPARTMENT\_ID = D.DEPARTMENT\_ID AND D.MANAGER\_ID = E.EMPLOYEE\_ID;

1. Display details of manager who manages more than 5 employees.

SELECT DISTINCT DEPARTMENT\_ID FROM HR.EMPLOYEES GROUP BY DEPARTMENT\_ID, MANAGER\_ID HAVING COUNT(EMPLOYEE\_ID)>5;

1. Display third highest salary of all employees

SELECT SALARY FROM HR.EMPLOYEES E WHERE 2 = (SELECT COUNT(DISTINCT SALARY) FROM HR.EMPLOYEES WHERE SALARY > E.SALARY);