**DDL (Data Definition Language)**

* [**CREATE**](https://www.geeksforgeeks.org/sql-create/) – is used to create the database or its objects (like table, index, function, views, store procedure and triggers).
* [**DROP**](https://www.geeksforgeeks.org/sql-drop-truncate/) – is used to delete objects from the database.
* [**ALTER**](https://www.geeksforgeeks.org/sql-alter-add-drop-modify/)-is used to alter the structure of the database.
* [**TRUNCATE**](https://www.geeksforgeeks.org/sql-drop-truncate/)–is used to remove all records from a table, including all spaces allocated for the records are removed.
* [**COMMENT**](https://www.geeksforgeeks.org/sql-comments/) –is used to add comments to the data dictionary.
* [**RENAME**](https://www.geeksforgeeks.org/sql-alter-rename/)–is used to rename an object existing in the database.

**DML (Data Manipulation Language)**

* [**SELECT**](https://www.geeksforgeeks.org/sql-select-clause/) – is used to retrieve data from a database.
* [**INSERT**](https://www.geeksforgeeks.org/sql-insert-statement/) – is used to insert data into a table.
* [**UPDATE**](https://www.geeksforgeeks.org/sql-update-statement/) – is used to update existing data within a table.
* [**DELETE**](https://www.geeksforgeeks.org/sql-delete-statement/) – is used to delete records from a database table.

**DCL(Data Control Language)**

* **GRANT**-gives user’s access privileges to database.
* **REVOKE**-withdraw user’s access privileges given by using the GRANT command.

**TCL(transaction Control Language)**

* **COMMIT**– commits a Transaction.
* [**ROLLBACK**](https://www.geeksforgeeks.org/sql-transactions/)– rollbacks a transaction in case of any error occurs.
* **SAVEPOINT**–sets a savepoint within a transaction.
* **SET TRANSACTION**–specify characteristics for the transaction.

|  |  |  |
| --- | --- | --- |
| **PL Sql vs Sql** | | |
| Comparison | SQL | PL/SQL |
| Execution | Single command at a time | Block of code |
| Application | Source of data to be displayed | Application created by data aquired by SQL |
| Structures include | DDL and DML based queries and commands | Includes procedures, functions, etc |
| Recommended while | Performing CRUD operations on data | Creating applications to display data obtained using sql |
| Compatibility with each other | SQL can be embedded into PL/SQL | PL/SQL cant be embedded in SQL |

**What is a Literal? Give an example where it can be used?**

 Select last\_name||’is a’||job\_id As “emp details” from employee; (Here “is a” is a literal).

**Single-Row functions:** These functions operate on a single row to give one result per row.

Character, Number, Date, Conversion, General

**Multiple-Row functions:** These functions operate on groups of rows to give one result per group of rows.

AVG, COUNT, MAX, MIN, SUM, STDDEV, VARIANCE.

**What is a Dual Table?**

Dual table is owned by the user SYS and can be accessed by all users. It contains one column **Dummy** and one row with the value **X.** The Dual Table is useful when you want to return a value only once. The value can be a constant, pseudo column or expression that is not derived from a table with user data.

Conversion Function:

**TO\_CHAR** function is used to convert NUMBER or DATE data type to CHARACTER format.

Select TO\_CHAR (hiredate, ‘MM/YY’) from employee.

**TO\_DATE** function is used to convert Character string to date format.

Select TO\_DATE (‘may 24 2007’,’mon dd rr’) from dual;

TO\_NUMBER function converts a string to a number.

select TO\_NUMBER ('1210.73') from dual;

**EMP Table:**



**NVL:** Converts a null value to an actual value. NVL (exp1, exp2). If exp1 is null, then NVL function return value of exp2.

**NVL2:** If exp1 is not null, nvl2 returns exp2, if exp1 is null, nvl2 returns exp3. The argument exp1 can have any data type. NVL2 (exp1, exp2, exp3)

SELECT NVL (NAME,'NA') AS NAME FROM TEST1;



SELECT NVL2 (NAME, 'Hi ' || NAME, 'Hello') AS NAME FROM TEST1;



**NULLIF:** Compares two expressions and returns null if they are equal or the first expression if they are not equal. NULLIF (exp1, exp2)

SELECT NULLIF('A','A') AS COMPARE FROM DUAL;

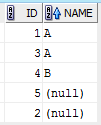


SELECT NULLIF('A','B') AS COMPARE FROM DUAL;



**COUNT (\*), COUNT (expression), COUNT (distinct expression)**

**EMP**



SELECT COUNT (\*) AS ALLROW, COUNT (NAME) ALLNONNULLROW, COUNT (DISTINCT NAME) UNIQUENONNULLROW FROM TEST1;



**COUNT (\*):** Returns number of rows in a table including duplicates rows and rows containing null values in any of the columns.

**COUNT (EXP):** Returns the number of non-null values in the column identified by expression.

**COUNT (DISTINCT EXP):** Returns the number of unique, non-null values in the column identified by expression.

**Types of sub queries:**

**Single-Row Sub query:**Queries that return only one row from the Inner select statement. Single-row comparison operators are: =, >, >=, <, <=, <>

**Multiple-Row Sub query:**Queries that return more than one row from the inner Select statement. There are also multiple-column sub queries that return more than one column from the inner select statement. Operators includes: IN, ANY, ALL.

Difference between ALL and ANY

ALL

The ALL comparison condition is used to compare a value to a list or subquery. It must be preceded by =, !=, >, <, <=, >= and followed by a list or subquery.

SELECT empno, sal FROM emp WHERE sal > ALL (2000, 3000, 4000);

Equivalent to

SELECT empno, sal FROM emp WHERE sal > 2000 AND sal > 3000 AND sal > 4000;

ANY

The ANY comparison condition is used to compare a value to a list or subquery. It must be preceded by =, !=, >, <, <=, >= and followed by a list or subquery.

SELECT empno, sal FROM emp WHERE sal > ANY (2000, 3000, 4000);

Equivalent to

SELECT empno, sal FROM emp WHERE sal > 2000 OR sal > 3000 OR sal > 4000;

**Describe few Data Types used in SQL?**

VARCHAR2(size): Minimum size is ‘1’ and Maximum size is ‘4000’

CHAR(size): Minimum size is ‘1’and Maximum size is ‘2000’

NUMBER(P,S): " Precision" can range from 1 to 38 and the “Scale” can range from -84 to 127.

DATE

LONG: 2GB

CLOB: 4GB

RAW (size): Maximum size is 2000

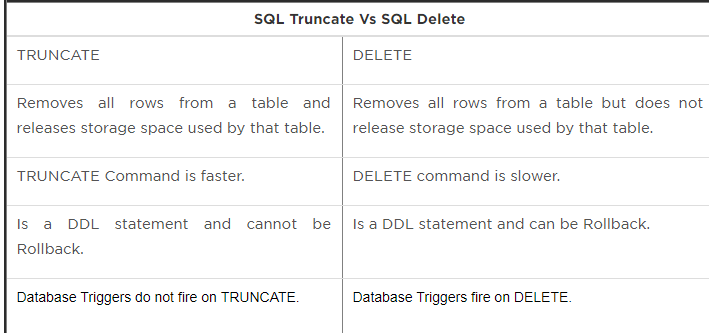
LONG RAW: 2GB

BLOB: 4GB

BFILE: 4GB

ROWID: A 64 base number system representing the unique address of a row in the table.

**What is a difference between Truncate and Delete?**



**Constraints**

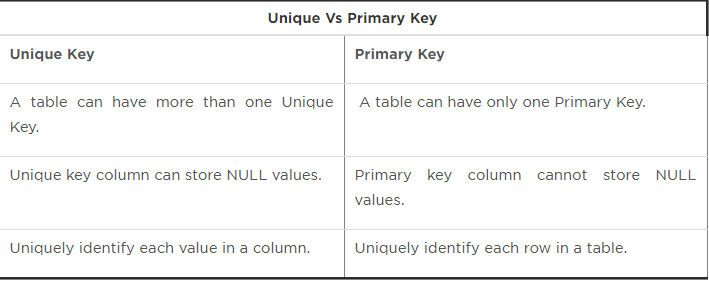
Not Null Constraint

Unique Key Constraint

Primary Key Constraint

Foreign Key Constraint

Check Key Constraint.



**What is a difference between ON DELETE CASCADE and ON DELETE SET NULL?**

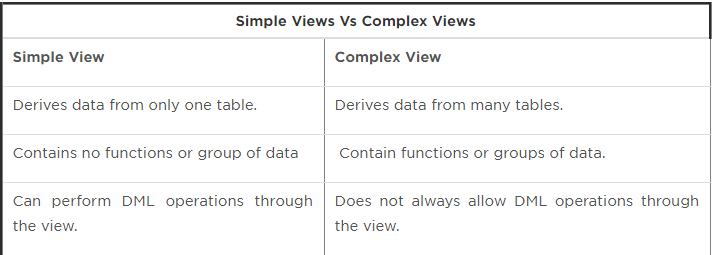
ON DELETE CASCADE Indicates that when the row in the parent table is deleted, the dependent rows in the child table will also be deleted. ON DELETE SET NULL Coverts foreign key values to null when the parent value is removed. Without the ON DELETE CASCADE or the ON DELETE SET NULL options, the row in the parent table cannot be deleted if it is referenced in the child table.

**What are Views and why they are used?**

A View logically represents subsets of data from one or more table. A View is a logical table based on a table or another view. A View contains no data of its own but is like a window through which data from tables can be viewed or changed. The tables on which a view is based are called Base Tables. The View is stored as a SELECT statement in the data dictionary. View definitions can be retrieved from the data dictionary table: USER\_VIEWS.

Views are used:

* To restrict data access
* To make complex queries easy
* To provide data Independence
* Views provide groups of users to access data according to their requirement.



**What is PL/SQL?**

PL/SQL is a procedural language extension with SQL Language.

1. It is a combination of SQL and Procedural Statements and used for creating applications.
2. Basically PL/SQL is a block structure programming language whenever we are submitting PL/SQL
3. Blocks then all SQL statements are executing separately by using sql engine and all procedure statements are executed separately.

**What are the different functionalities of a Trigger?**

Trigger is also same as stored procedure & also it will be automatically invoked whenever DML operation performed against table or view.

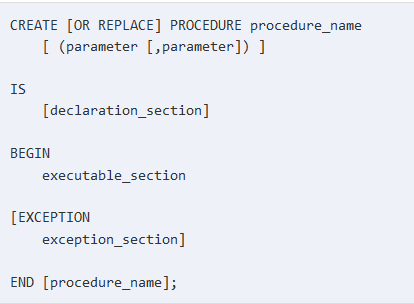
There are two types of triggers supported by PL/SQL

1. Statement Level Trigger.
2. Row Level Trigger

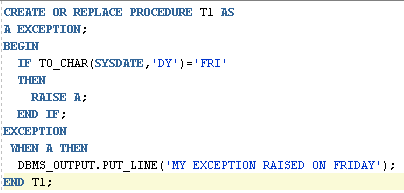
**Statement Level Trigger:**In statement level trigger, trigger body is executed only once for DML statement.

**Row Level Trigger:**In row level trigger, trigger body is executed for each row DML statements. It is the reason, we are employing each row clause and internally stored DML transaction in trigger specification.

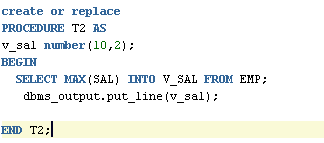
Sample Procedure syntax:



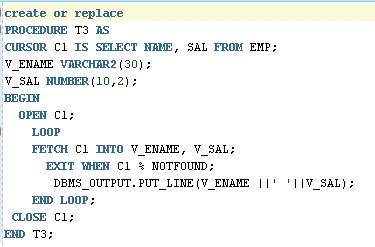
**Write a PL/SQL Program which raise a user defined exception**



**Write a PL/SQL program to retrieve emp table and then display the salary?**



**Write a PL/SQL cursor program to display all employee names and their salary from emp table by using % not found attributes?**



* 1. **Query to find 2nd Highest Salary**

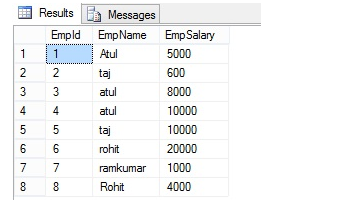
SELECT \* FROM emp e1 WHERE (N-1) = (SELECT count (DISTINCT sal) FROM emp e2 WHERE e1.sal<e2.sal);

OR

SELECT MIN(sal) FROM (SELECT DISTINCT sal FROM emp ORDER BY sal DESC) WHERE ROWNUM<=N;

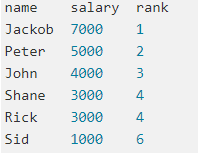
* 1. **Difference Between Row\_Number() Rank() And Dense\_Rank()**

Employee table:

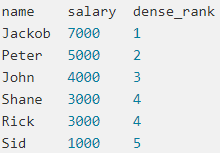


**ROW\_NUMBER () Function without Partition By clause**

Row\_Number function can help to perform more complex ordering of row in the report format than allow the over clause

**ROW\_NUMBER() Example**  
It always generates a unique value for each row, even if they are same and ORDER BY clause cannot distinguish between them.   
  
select e.\*, row\_number() over (order by salary desc) row\_number from #Employee e result:  
  
 **RANK() Example**  
The rank() function will assign the same rank to same values i.e. which are not distinguishable by ORDER BY. Also, the next different rank will not start from immediately next number but there will be gap i.e. if 4th and 5th employee have the same salary then they will have same rank 4, and 6th employee which has different salary will have new rank 6.  
  
 select e.\*, rank() over (order by salary desc) rank from #Employee e result: name salary rank Jackob 7000 1 Peter 5000 2 John 4000 3 Shane 3000 4 Rick 3000 4 Sid 1000 6  
  
 

**DENSE\_RANK() Example**  
The dense\_rank function is similar to rank() window function i.e. same values will be assigned the same rank, but the next different value will have rank which is just one more than the previous rank, i.e. if 4th and 5th employee has the same salary then they will have same rank but 6th employee, which has different salary will have rank 5, unlike rank 6 as is the case with rank() function. There will be no gap on ranking in case of dense\_rank() as shown in the following example:

select e.\*, dense\_rank() over (order by salary desc) dense\_rank from #Employee e name salary dense\_rank  
  
   
  
 

Query to find duplicate row?

**Select \* from Employee a where rowid <>( select max(rowid) from Employee b where a.Employee\_num=b.Employee\_num);**

**What is the Query to fetch first record from Employee table**

*Select \* from Employee where [Rownum](http://www.complexsql.com/rowid-rownum/" \t "_blank)=1;*

**What is the Query to fetch last record from the table?**

*Select \* from Employee where Rowid= select max(Rowid) from Employee;*

**What is Query to display first 5 Records from Employee table?**

*Select \* from Employee where Rownum <= 5;*

**What is Query to display last 5 Records from Employee table?**

*select \* from (Select \* from Employee e order by rowid desc) where rownum <=5;*

**What is Query to display Nth Record from Employee table?**

*select \* from (select a.\*, rownum rnum from ( YOUR\_QUERY\_GOES\_HERE — including the order by ) a where rownum <= N\_ROWS ) where rnum >= N\_ROWS*

**How to Display**[**Odd[http://cdncache-a.akamaihd.net/items/it/img/arrow-10x10.png](http://www.complexsql.com/complex-sql-queries-examples-with-answers/#56367082)**](http://www.complexsql.com/complex-sql-queries-examples-with-answers/#56367082)**rows in Employee table?(90% asked Complex SQL Queries Examples)**

*Select \* from(Select rownum as rno,E.\* from Employee E) where Mod(rno,2)=1;*

**Display first 50% records from Employee table?**

*select rownum, e.\* from emp e where rownum<=(select count(\*)/2 from emp);*

**Display last 50% records from Employee table?**

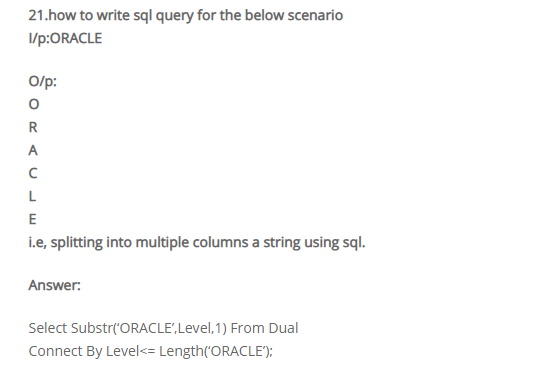
*Select rownum,E.\* from Employee E*

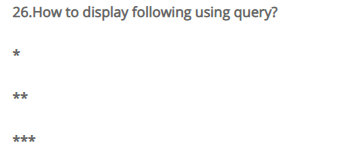
*minus*

*Select rownum,E.\* from Employee E where rownum<=(Select count(\*)/2) from Employee);*

**How Can i create table with same structure with data of Employee table?**

*Create table Employee1 as select \* from Employee;*





*SELECT lpad (‘\*’, ROWNUM,’\*’) FROM Student WHERE ROWNUM <4;*

**How to display 1 to 100 Numbers with query?**

*Select level from dual connect by level <=100;*

**How to remove duplicate rows from table?**

*Delete FROM Student WHERE ROWID <>*

*(Select max (rowid) from Student b where rollno=b.rollno);*

**How to find count of duplicate rows?**

*Select rollno, count (rollno) from Student*

*Group by rollno*

*Having count (rollno)>1*

*Order by count (rollno) desc;*

