# LATERAL VIEW

**Lateral View Syntax**

|  |
| --- |
| lateralView: LATERAL VIEW udtf(expression) tableAlias AS columnAlias (',' columnAlias)\*  fromClause: FROM baseTable (lateralView)\* |

**Description**

Lateral view is used in conjunction with user-defined table generating functions (UDTF) such as explode().a UDTF generates zero or more output rows for each input row. A lateral view first applies the UDTF to each row of base table and then joins resulting output rows to the input rows to form a virtual table having the supplied table alias

You can refer link below to get a list of built in UDTF in hive

<https://cwiki.apache.org/confluence/display/Hive/LanguageManual+UDF#LanguageManualUDF-Built-inTable-GeneratingFunctions(UDTF)>

**Example**

|  |
| --- |
| CREATE TABLE pageAds (  pageid STRING,  adid\_list ARRAY<INT>  )  ROW FORMAT DELIMITED  FIELDS TERMINATED BY ','  COLLECTION ITEMS TERMINATED BY '#' ; |

Pageadds.txt

|  |
| --- |
| front\_page,1#2#3  contact\_page,3#4#5 |

|  |
| --- |
| load data local inpath '/home/shalaj/pageadds.txt' into table pageAds; |

|  |
| --- |
|  |

|  |
| --- |
| SELECT pageid, adid FROM pageAds **LATERAL VIEW** explode(adid\_list) adTable AS adid; |

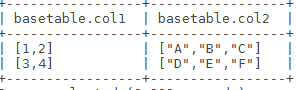
In order to count the number of times a particular ad appears, count/group by can be used:

|  |
| --- |
| SELECT adid, count(adid) as cnt\_adid FROM pageAds **LATERAL VIEW explode**(adid\_list) adTable AS adid GROUP BY adid; |

|  |
| --- |
| create table baseTable (  col1 ARRAY<INT>,  col2 ARRAY<STRING>  )  ROW FORMAT DELIMITED  FIELDS TERMINATED BY ','  COLLECTION ITEMS TERMINATED BY '#' ; |

|  |
| --- |
| 1#2,A#B#C  3#4,D#E#F |

|  |
| --- |
| load data local inpath '/home/shalaj/basetable.txt' into table basetable; |



|  |
| --- |
| SELECT myCol1, col2 FROM baseTable **LATERAL VIEW** explode(col1) myTable1 AS myCol1; |

|  |
| --- |
| SELECT myCol1, myCol2 FROM baseTable  **LATERAL VIEW** explode(col1) myTable1 AS myCol1  **LATERAL VIEW** explode(col2) myTable2 AS myCol2; |

# Outer Lateral view

The user can specify the optional OUTER keyword to generate rows even when a LATERAL VIEW usually would not generate a row

|  |
| --- |
| select deptno,empno,ename,sal from emp; |

|  |
| --- |
| select deptno,empno,ename,sal,a FROM emp **LATERAL VIEW** explode(array()) C AS a |

|  |
| --- |
| select deptno,empno,ename,sal,a FROM emp **LATERAL VIEW** **OUTER** explode(array()) C AS a |

|  |
| --- |
| select deptno,empno,ename,sal,a FROM emp **LATERAL VIEW** explode(array(1,2)) C AS a |

# PosExplode

posexplode() is similar to explode but instead of just returning the elements of the array it returns the element as well as its position in the original array. **Tuple of (pos, value**).

|  |
| --- |
| select posexplode(adid\_list) from pageAds ; |

|  |
| --- |
| select deptno,empno,ename,sal,position,value FROM emp **LATERAL VIEW** **posexplode**(array('x','y','z')) C AS position,value |