ORACLE DATABASE PL/SQL

ORACLE PL/SQL PIVOT/UNPIVOT

How Pivot Table works in ORACLE? Explain with Different Examples? How will you convert UNPIVOT Data into PIVOT Data?

PIVOTING:

Pivoting helps in converting Columns values into Attributes **(Transpose rows into columns)**. Following are the steps to perform Pivoting:

- Separate the Rows
- Aggregate Required Data
- Convert Aggregated Data into Columns

Parameters or Arguments

AGGREGATE FUNCTION
It can be a function such as SUM, COUNT, MIN, MAX, or AVG functions.

IN (Expr1, Expr2, ... Expr n)
A list of values for *column2* to pivot into headings in the cross-tabulation query results.

SUBQUERY
It can be used instead of a list of values. In this case, the results of the subquery would be used to determine the values for *column* to pivot into headings in the cross-tabulation query results.

Let's check this with the below Examples:

EXAMPLE 1: TABLE(EMP_TEST): EMPLOYEE Data As: EMP_NO, ENAME, DEPTNO, HIREDATE, WORK_LOCATION
So, the Table Data looks like below:

	EMP_NO	ENAME	DEPTNO	HIREDATE	WORK_LOCATION
1	1001	RAVI	10	19-08-16	BANGALORE
2	1002	SURYA	20	26-12-16	KOCHI
3	1003	ANKIT	30	12-12-16	Pune
4	1004	NIKHIL	40	12-12-10	DELHI
5	1005	PRITESH	50	19-08-16	DELHI
6	1006	RAJAN	20	16-08-10	DELHI
7	1007	MANU	20	16-08-10	KOLKATA
8	1008	KARAN	20	16-08-10	KOLKATA
9	1009	GAURAV	50	19-03-17	Pune
10	1010	SHAHRUKH	40	11-03-17	KOCHI
11	1011	KHAN	30	11-03-16	BANGALORE

Purpose: We need to know how many people were placed for different Departments at different Locations.

```
SELECT * From
(
SELECT WORK_LOCATION,DEPTNO
From EMP_TEST
)
PIVOT
(
Count(DEPTNO)
For DEPTNO IN (10,20,30,40,50)
);
```

When you run the above Query, it will show the below Output:

	WORK_LOCATION	10	20	30	40	50
1	KOCHI	0	1	0	1	0
2	DELHI	0	1	0	1	1
3	KOLKATA	0	2	0	0	0
4	BANGALORE	1	0	1	0	0
5	Pune	0	0	1	0	1

Let's make some more changes on above Query- Instead of DEPTNO we will display DEPARTMENT NAME which is available in Department Table.

TABLE(DEPT_TEST): DEPTNO, DEPTNAME

So, the Table Data looks like below:

	DEPTNO	DEPTNAME
1	10	Accounts
2	20	Retail
3	30	Insurance
4	40	Banking
5	50	Cloud

Let's make changes on above Query to populate Department Name instead of Department Number.

With Simple SELECT:

```
SELECT * From
(
SELECT WORK_LOCATION,DEPTNAME
From EMP_TEST E, DEPT_TEST D
Where E.DEPTNO=D.DEPTNO
)
PIVOT
(
Count(DEPTNAME)
For DEPTNAME IN ('Accounts','Retail','Insurance','Banking','Cloud')
);
```

Using WITH Clause:

We can also Query using WITH Clause to improve performance.

```
WITH T AS
(
SELECT WORK_LOCATION,DEPTNAME
From EMP_TEST E, DEPT_TEST D
Where E.DEPTNO=D.DEPTNO
)
SELECT * From T
PIVOT
(
Count(DEPTNAME)
For DEPTNAME IN ('Accounts','Retail','Insurance','Banking','Cloud')
);
```

When you run any of the above Query, it will show the below Output:

	WORK_LOCATION	'Accounts'	'Retail'	'Insurance'	'Banking'	'Cloud'
1	KOCHI	0	1	0	1	0
2	DELHI	0	1	0	1	1
3	KOLKATA	0	2	0	0	0
4	BANGALORE	1	0	1	0	0
5	Pune	0	0	1	0	1

EXAMPLE 2: In this Example, we will use the CUSTOMER Table.

```
CREATE TABLE CUSTOMER
(
  ID            NUMBER,
  CUST_ID       NUMBER,
  PRODUCT_CD    VARCHAR2(5),
  Quantity      NUMBER
);
```

I inserted some data into above Table as below:

ID	CUST_ID	PRODUCT_CD	QUANTITY
1	1	A	10
2	1	B	20
3	1	C	30
4	2	A	40
5	2	C	50
6	3	A	60
7	3	B	70
8	3	C	80
9	3	D	90
10	4	A	100

Purpose: We need to Get Customer with the Total Product Quantity ordered by them for different Products As
CUSTOMER_ID, TOTAL_QUANTITY, A_SUM_QUANTITY, B_SUM_QUANTITY, C_SUM_QUANTITY, D_SUM_QUANTITY.

Solution: This can be done with Pivot Table. We will calculate on TOTAL_QUANTITY separately. Let's first try to get sum for individual products.

```
SELECT * FROM
(
SELECT CUST_ID,PRODUCT_CD,QUANTITY
FROM   CUSTOMER
)
PIVOT
(SUM(QUANTITY) AS SUM_QUANTITY FOR PRODUCT_CD
IN ('A' AS A, 'B' AS B, 'C' AS C, 'D' AS D))
ORDER BY CUST_ID;
```

When you run above Query, it will show the below Output:

CUST_ID	A_SUM_QUANTITY	B_SUM_QUANTITY	C_SUM_QUANTITY	D_SUM_QUANTITY
1	10	20	30	(null)
2	40	(null)	50	(null)
3	60	70	80	90
4	100	(null)	(null)	(null)

So, we have got what we wanted. Now, we need to add Total Quantity of all the Products. In this example, we will first Calculate the Total Quantity for all the Customers using WITH clause and then will use it in above Query. So, our Query will look like below:

```
SELECT * FROM
(
WITH T AS
(
SELECT CUST_ID, SUM(QUANTITY) TOTAL_QUANTITY
From CUSTOMER
GROUP BY CUST_ID
)
SELECT C.CUST_ID,C.PRODUCT_CD,C.QUANTITY,T.TOTAL_QUANTITY
From T,CUSTOMER C
Where T.CUST_ID=C.CUST_ID
)
PIVOT
(SUM(QUANTITY) AS SUM_QUANTITY FOR PRODUCT_CD
IN ('A' AS A, 'B' AS B, 'C' AS C, 'D' AS D))
ORDER BY CUST_ID;
```

When you run above Query, it will show the below Output:

CUST_ID	TOTAL_QUANTITY	A_SUM_QUANTITY	B_SUM_QUANTITY	C_SUM_QUANTITY	D_SUM_QUANTITY
1	60	10	20	30	(null)
2	90	40	(null)	50	(null)
3	300	60	70	80	90
4	100	100	(null)	(null)	(null)

The above Query without PIVOT will look like below but would be more expensive than above if you compare the Execution Plan:

```
SELECT CUST_ID,
       SUM(QUANTITY) AS TOTAL_QUANTITY,
       SUM(DECODE(PRODUCT_CD, 'A', QUANTITY, 0)) AS A_SUM_QUANTITY,
       SUM(DECODE(PRODUCT_CD, 'B', QUANTITY, 0)) AS B_SUM_QUANTITY,
       SUM(DECODE(PRODUCT_CD, 'C', QUANTITY, 0)) AS C_SUM_QUANTITY,
       SUM(DECODE(PRODUCT_CD, 'D', QUANTITY, 0)) AS D_SUM_QUANTITY
FROM   CUSTOMER
GROUP BY CUST_ID
ORDER BY CUST_ID;
```

When you run above Query, it will show the below Output:

Popular Posts

CUST_ID	TOTAL_QUANTITY	A_SUM_QUANTITY	B_SUM_QUANTITY	C_SUM_QUANTITY	D_SUM_QUANTITY
1	60	10	20	30	0
2	90	40	0	50	0
3	300	60	70	80	90
4	100	100	0	0	0

UNPIVOTING:

The UNPIVOT operator converts Column-Based data into individual rows.

Let's create a Table with any of the above Query and then will see how UNPIVOTING works.

Following Query will create a Table **CUSTOMER_PRODUCT_QUANTITY** with the same Data as above. So, the idea is to convert Data from table same as **CUSTOMER** Table which we have used in PIVOTING.

```
CREATE TABLE CUSTOMER_PRODUCT_QUANTITY
AS
(
SELECT CUST_ID,
       SUM(QUANTITY) AS TOTAL_QUANTITY,
       SUM(DECODE(PRODUCT_CD, 'A', QUANTITY, 0)) AS A_SUM_QUANTITY,
       SUM(DECODE(PRODUCT_CD, 'B', QUANTITY, 0)) AS B_SUM_QUANTITY,
       SUM(DECODE(PRODUCT_CD, 'C', QUANTITY, 0)) AS C_SUM_QUANTITY,
       SUM(DECODE(PRODUCT_CD, 'D', QUANTITY, 0)) AS D_SUM_QUANTITY
FROM   CUSTOMER
GROUP BY CUST_ID
);
```

Let's see Data from new Table CUSTOMER_PRODUCT_QUANTITY

CUST_ID	TOTAL_QUANTITY	A_SUM_QUANTITY	B_SUM_QUANTITY	C_SUM_QUANTITY	D_SUM_QUANTITY
1	60	10	20	30	0
2	90	40	0	50	0
4	100	100	0	0	0
3	300	60	70	80	90

Following Query will be used to change the Column Data into individual rows:

```
SELECT ROWNUM AS ID, T.CUST_ID,T.PRODUCT_CD, T.QUANTITY
From
(
SELECT *
FROM   CUSTOMER_PRODUCT_QUANTITY
UNPIVOT (QUANTITY FOR PRODUCT_CD IN (A_SUM_QUANTITY AS 'A', B_SUM_QUANTITY AS 'B', C_SUM_QUANTITY AS 'C', D_SUM_QUANTITY A
ORDER BY CUST_ID
) T
Where T.QUANTITY<>0;
```

When you run any the above Query, it will show the below Output:

ID	CUST_ID	PRODUCT_CD	QUANTITY
1	1	A	10
2	1	B	20
3	1	C	30
4	2	A	40
5	2	C	50
6	3	A	60
7	3	B	70
8	3	C	80
9	3	D	90
10	4	A	100

NOTE: Where T.QUANTITY<>0 is just used to Exclude NULLS

Get involved and leave your Comments in the Box Below. The more people get involved, the more we all benefit. **So, leave your thoughts before you leave the page.**

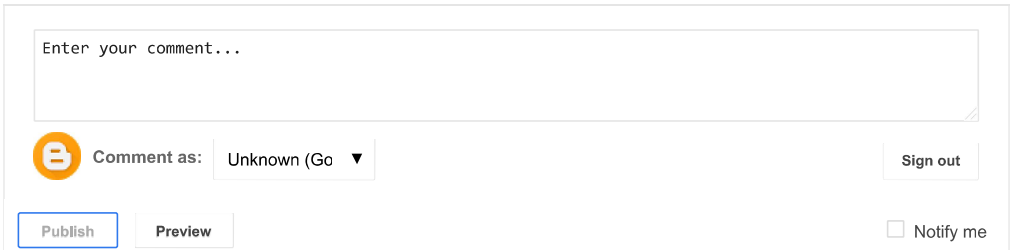
1 comment:

Mathavan J March 8, 2018 at 10:03 PM

going to use this for my studies Oracle Training in Chennai
. not going to refer any other sites. Please provide your email id or mobile number

Reply

- ORACLE COLLECTIONS
- ORACLE SQL* Loader
- EXPORT TABLE Data to Flat Files
- SYS_REFCURSOR Vs. REF CURSOR
- ORACLE TABLE PARTITIONING
- ORACLE PL/SQL Tuning
- UTL_FILE Import Data into ORACLE TA
- ORACLE UTL_FILE
- Using BULK Collect



Links to this post

Create a Link

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)