# **ROLLUP and CUBE with GROUP BY Sql clause**

## Setup

## The examples in this document require the following table.

DROP TABLE myemp;

CREATE TABLE myemp (

myempno NUMBER(4) CONSTRAINT pk\_myemp PRIMARY KEY,

ename VARCHAR2(10),

job VARCHAR2(9),

mgr NUMBER(4),

hiredate DATE,

sal NUMBER(7,2),

comm NUMBER(7,2),

deptno NUMBER(2)

);

INSERT INTO myemp VALUES (7369,'SMITH','CLERK',7902,to\_date('17-12-1980','dd-mm-yyyy'),800,NULL,20);

INSERT INTO myemp VALUES (7499,'ALLEN','SALESMAN',7698,to\_date('20-2-1981','dd-mm-yyyy'),1600,300,30);

INSERT INTO myemp VALUES (7521,'WARD','SALESMAN',7698,to\_date('22-2-1981','dd-mm-yyyy'),1250,500,30);

INSERT INTO myemp VALUES (7566,'JONES','MANAGER',7839,to\_date('2-4-1981','dd-mm-yyyy'),2975,NULL,20);

INSERT INTO myemp VALUES (7654,'MARTIN','SALESMAN',7698,to\_date('28-9-1981','dd-mm-yyyy'),1250,1400,30);

INSERT INTO myemp VALUES (7698,'BLAKE','MANAGER',7839,to\_date('1-5-1981','dd-mm-yyyy'),2850,NULL,30);

INSERT INTO myemp VALUES (7782,'CLARK','MANAGER',7839,to\_date('9-6-1981','dd-mm-yyyy'),2450,NULL,10);

INSERT INTO myemp VALUES (7788,'SCOTT','ANALYST',7566,to\_date('13-JUL-87','dd-mm-rr')-85,3000,NULL,20);

INSERT INTO myemp VALUES (7839,'KING','PRESIDENT',NULL,to\_date('17-11-1981','dd-mm-yyyy'),5000,NULL,10);

INSERT INTO myemp VALUES (7844,'TURNER','SALESMAN',7698,to\_date('8-9-1981','dd-mm-yyyy'),1500,0,30);

INSERT INTO myemp VALUES (7876,'ADAMS','CLERK',7788,to\_date('13-JUL-87', 'dd-mm-rr')-51,1100,NULL,20);

INSERT INTO myemp VALUES (7900,'JAMES','CLERK',7698,to\_date('3-12-1981','dd-mm-yyyy'),950,NULL,30);

INSERT INTO myemp VALUES (7902,'FORD','ANALYST',7566,to\_date('3-12-1981','dd-mm-yyyy'),3000,NULL,20);

INSERT INTO myemp VALUES (7934,'MILLER','CLERK',7782,to\_date('23-1-1982','dd-mm-yyyy'),1300,NULL,10);

COMMIT;

**Simple GROUP BY clause for pairwise aggregate function**.

SELECT

deptno,

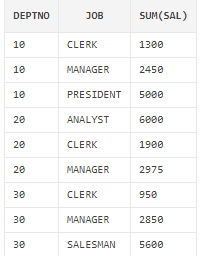
job,

sum(sal)

FROM myemp

GROUP BY (deptno,job)

ORDER BY deptno,job;



**ROLLUP with GROUP BY clause:**

The **ROLLUP** clause extends GROUP BY to return a row containing a subtotal for each group of rows, plus a row containing a total for all the groups.

SELECT

deptno,

job,

sum(sal)

FROM myemp

GROUP BY ROLLUP(deptno,job)

ORDER BY deptno,job;



**CUBE with GROUP BY clause :**

The **CUBE** clause extends GROUP BY to return rows containing a subtotal for all combinations of columns, plus a row containing the grand total

SELECT

deptno,

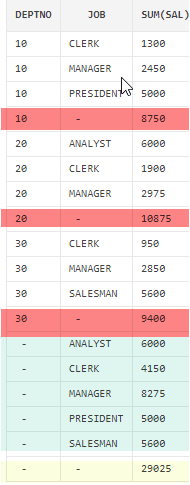
job,

sum(sal)

FROM myemp

GROUP BY CUBE(deptno,job)

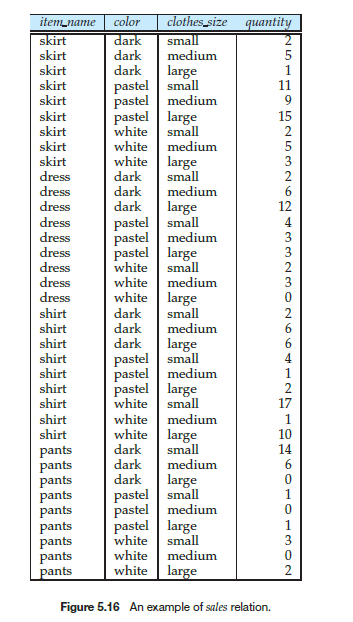
ORDER BY deptno,job;



**OLAP\*\***

An online analytical processing (OLAP) system is an interactive system that permits an analyst to view different summaries of multidimensional data.

CUBE, ROLLUP, Pivot are examples of OLAP.



**Suppose that,**

***Item\_name =>* can take on the values (skirt, dress, shirt, pants),**

***Color =>* can take on the values (dark, pastel, white)**

***Clothes\_size =>* can take on values (small, medium, large), and**

***Quantity =>* is an integer value representing the total number of**

**items of a given *{item name*, *color*, *clothes size }*.**

**Data of Sales relation is shown in Figure 5.16.**

Several RDBMS, such as Microsoft SQL Server, and Oracle, support

a **pivot** clause in SQL, which allows creation of cross-tabs. Given the *sales* relation

from Figure 5.16, the query:

**select** \*

**from** *sales*

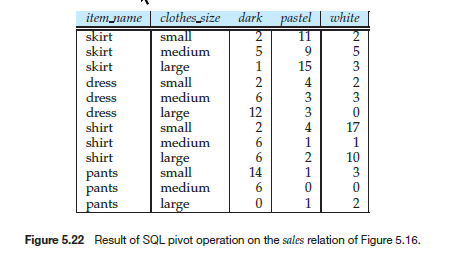
**pivot** ( **sum**(*quantity*) **for** *color* **in** (’dark’,’pastel’,’white’)

)

**order by** *item name*;

Above OLAP query returns the cross-tab shown in Figure 5.22. Note that the **for** clause within the

**Pivot** clause specifies what values from the attribute *color* should appears as attribute names in the pivot result.



SELECT \* FROM (select deptno,job,sal from myemp)

PIVOT

( SUM(sal)FOR job IN ('SALESMAN','CLERK','PRESIDENT','MANAGER','ANALYST')

)

ORDER BY deptno;