Structured Query Language (SQL) - Part 2

Lecture 7

Learning Outcomes

- In this chapter, students will learn:
  - How to use SQL for data administration (to drop tables and create views)
  - How to use SQL for data manipulation (to delete and retrieve data)
  - How to use SQL to query a database for useful information
  - How to use SQL functions to manipulate dates, strings, and other data

Advanced Data Definition Commands

- All changes in table structure are made by using **ALTER** command
- Three options:
  - ***ALTER COLUMN*** changes column characteristics
    - *ADD* adds a column
    - *DROP* deletes a column
    - *ALTER* column set data type
- Can also be used to:
  - Add table constraints (e.g., foreign key)
  - Remove table constraints

Changing a Column's Data Characteristics

- Use ALTER to change column's data characteristics
- Syntax:
  - **ALTER TABLE** <table name>
    ALTER COLUMN <column name> **SET DATA TYPE** <new column data type characteristic>
- Changes in column's characteristics are permitted if changes do not alter the existing data type
- Example:
  - **ALTER TABLE** PRODUCT **ALTER COLUMN** V_CODE
    SET DATA TYPE CHAR(5)
  - **ALTER TABLE** PRODUCT **ALTER COLUMN** P_PRICE
    SET DATA TYPE NUMBER(9,2)

Adding a Column

- ADD column
  - Do not include the NOT NULL clause for new column

- Example:
  - ALTER TABLE PRODUCT ADD SALECODE CHAR(10)
  - ALTER TABLE STUDENT ADD GENDER CHAR(1) DEFAULT 'F'

Dropping a Column

- Use ALTER to drop column
  - Some RDBMSs impose restrictions on the deletion of an attribute

- Syntax:
  - **ALTER TABLE** <tablename>
    DROP <columnname>

- Example:
  - **ALTER TABLE** VENDOR **DROP COLUMN** V_ORDER

Ordering a Listing

- ORDER BY clause is useful when listing order is important
- Syntax:
  - **SELECT** columnlist
    FROM tablelist
    [**WHERE** conditionlist]
    [**ORDER BY** columnlist [ASC | DESC]]
- Ascending order by default
- Example:
  - **SELECT** P_CODE, P_DESCRIPT, P_INDATE, P_PRICE
    FROM PRODUCT
    ORDER BY P_PRICE

Order Results in Ascending Order

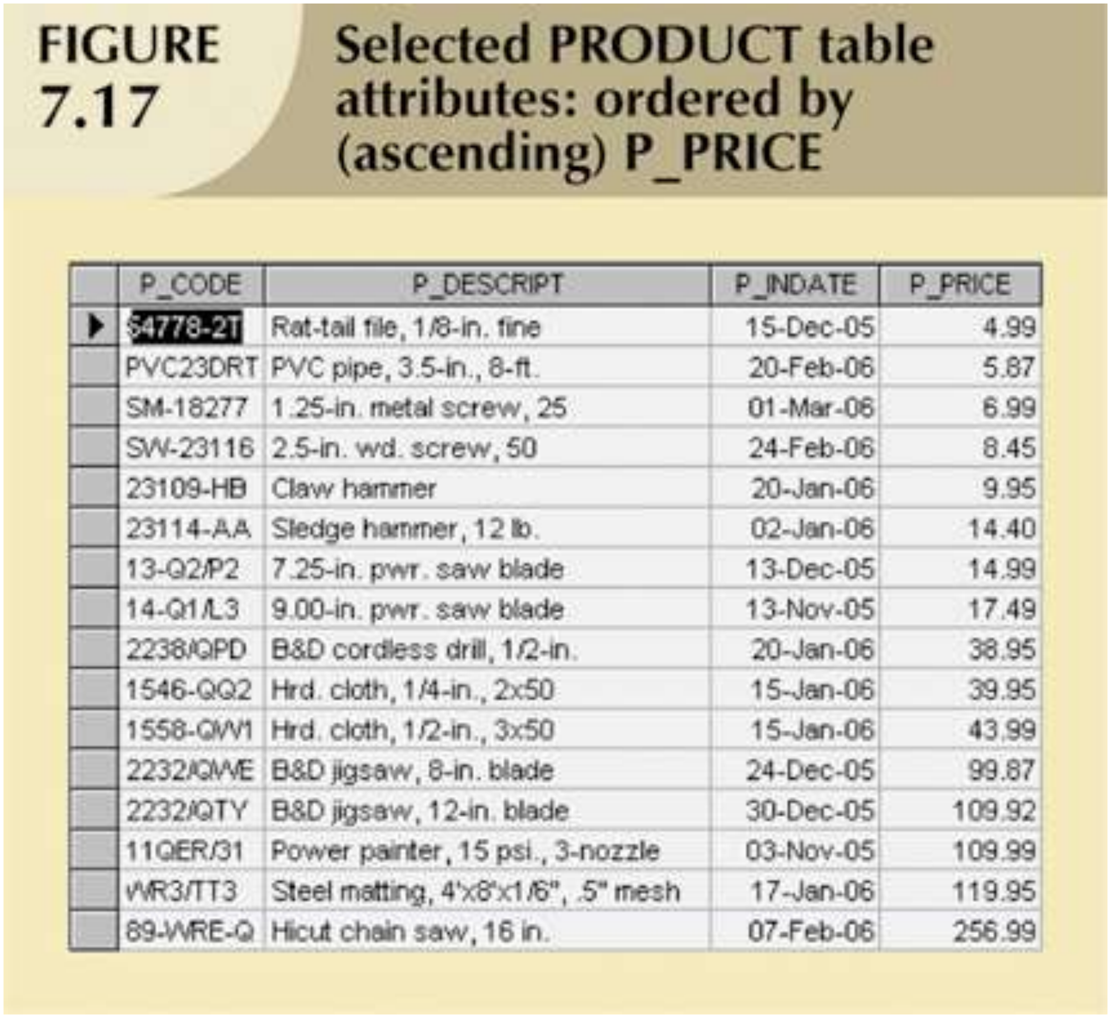


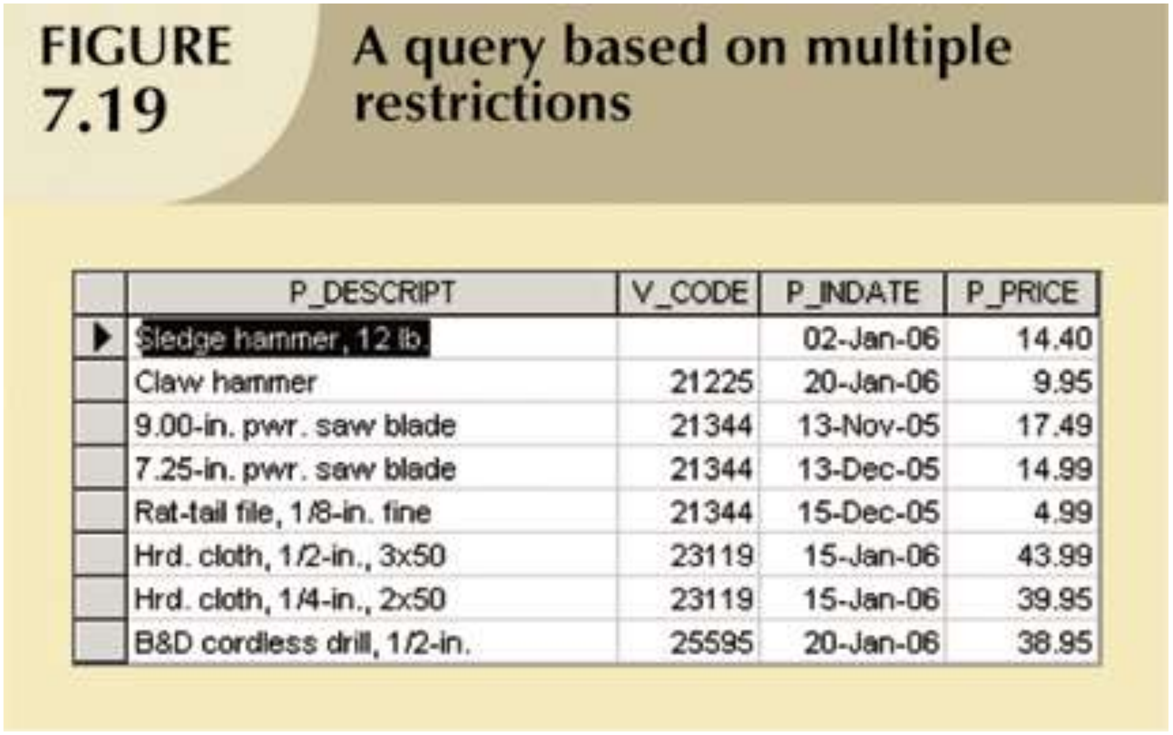
FIGURE 7.17 Selected PRODUCT table attributes: ordered by (ascending) P_PRICE

P_CODE	P_DESCRIPT	P_INDATE	P_PRICE
54778-2T	Rat-tail file, 1/8-in. fine	15-Dec-05	4.99
PVC23DRT	PVC pipe, 3.5-in., 8-ft.	20-Feb-06	5.87
SM-18277	1.25-in. metal screw, 25	01-Mar-06	6.99
SW-23116	2.5-in. wd. screw, 50	24-Feb-06	8.45
23109-HB	Claw hammer	20-Jan-06	9.95
23114-AA	Sledge hammer, 12 lb.	02-Jan-06	14.40
13-Q2/P2	7.25-in. pwr. saw blade	13-Dec-05	14.99
14-Q1/L3	9.00-in. pwr. saw blade	13-Nov-05	17.49
2238/QPD	B&D cordless drill, 1/2-in.	20-Jan-06	38.95
1546-QQ2	Hrd. cloth, 1/4-in., 2x50	15-Jan-06	39.95
1558-QW1	Hrd. cloth, 1/2-in., 3x50	15-Jan-06	43.99
2232/QWE	B&D jigsaw, 8-in. blade	24-Dec-05	99.87
2232/QTY	B&D jigsaw, 12-in. blade	30-Dec-05	109.92
11QER/31	Power painter, 15 psi., 3-nozzle	03-Nov-05	109.99
WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	17-Jan-06	119.95
89-WRE-Q	Hicut chain saw, 16 in.	07-Feb-06	256.99

A Query based on Multiple Restrictions

```
SELECT P_DESCRIPT, V_CODE, P_INDATE, P_PRICE
FROM PRODUCT
WHERE P_INDATE > '1999-08-20' AND P_PRICE <= 50.00
ORDER BY V_CODE, P_INDATE
```

FIGURE 7.19 A query based on multiple restrictions



P_DESCRIPT	V_CODE	P_INDATE	P_PRICE
Sledge hammer, 12 lb.		02-Jan-06	14.40
Claw hammer	21225	20-Jan-06	9.95
9.00-in. pwr. saw blade	21344	13-Nov-05	17.49
7.25-in. pwr. saw blade	21344	13-Dec-05	14.99
Rat-tail file, 1/8-in. fine	21344	15-Dec-05	4.99
Hrd. cloth, 1/2-in., 3x50	23119	15-Jan-06	43.99
Hrd. cloth, 1/4-in., 2x50	23119	15-Jan-06	39.95
B&D cordless drill, 1/2-in.	25595	20-Jan-06	38.95

Listing Unique Values

- **DISTINCT** clause produces list of only values that are different from one another
- Example:
  - **SELECT DISTINCT** V_CODE
    FROM PRODUCT

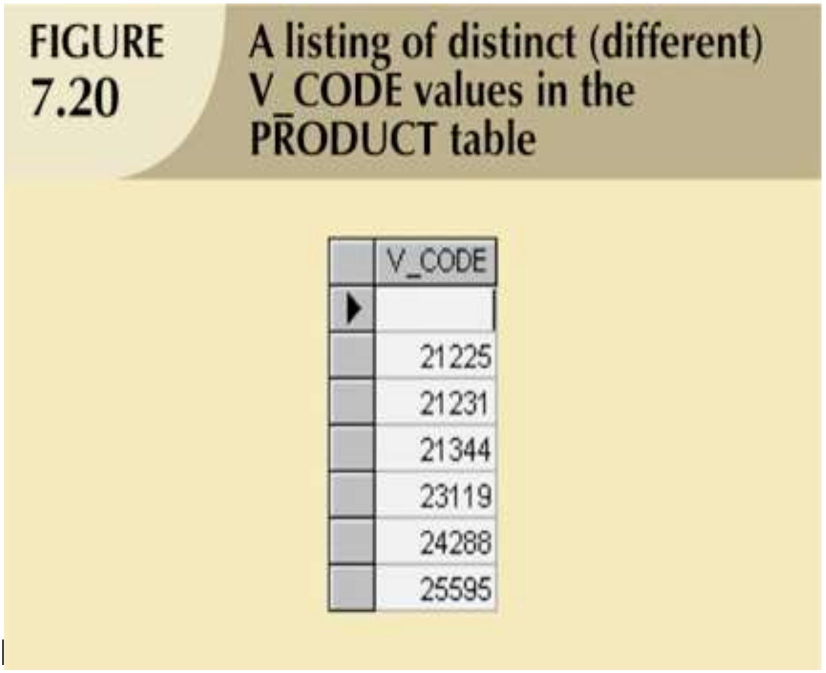


FIGURE 7.20 A listing of distinct (different) V_CODE values in the PRODUCT table

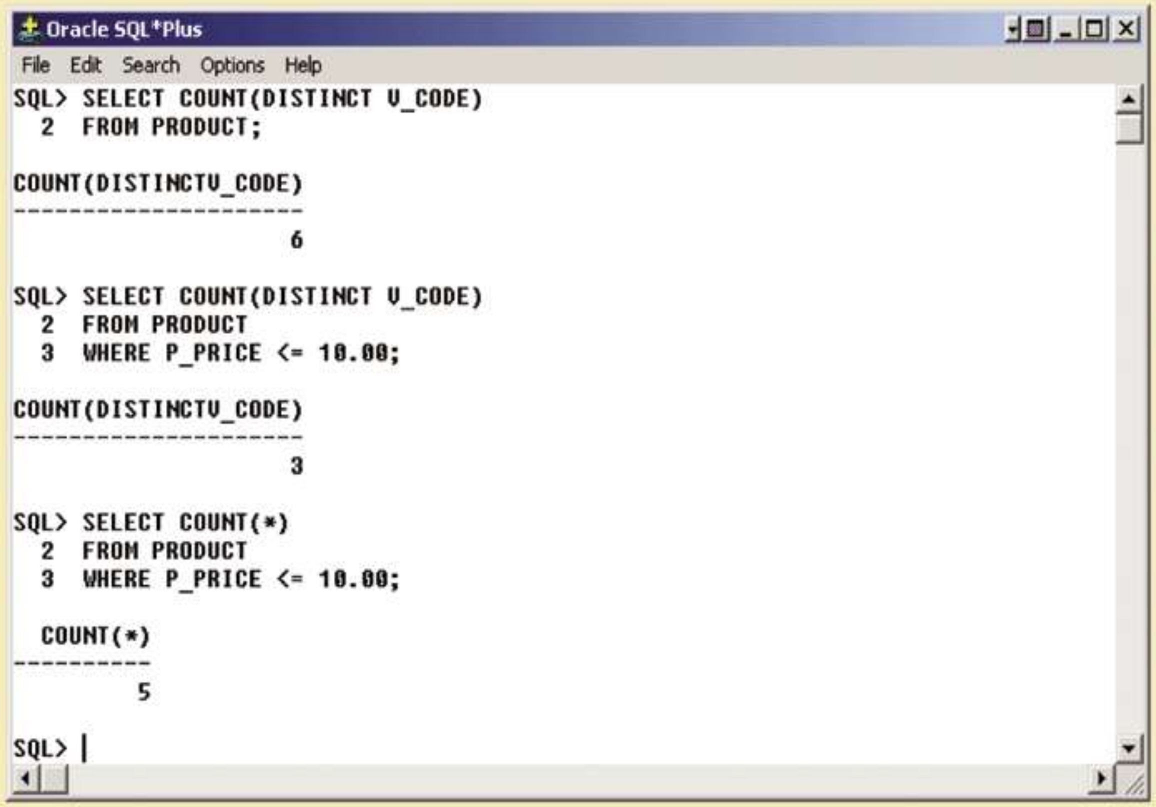
V_CODE
21225
21231
21344
23119
24288
25595

Aggregate Functions

- ***COUNT*** function tallies number of non-null values of an attribute
  - Takes one parameter: usually a column name
  - Can be used with DISTINCT clause
- ***MAX*** and ***MIN*** find highest (lowest) value in a table
- ***SUM*** computes total sum for any specified attribute
- ***AVG*** function format is similar to MIN and MAX

Example Output of COUNT Function

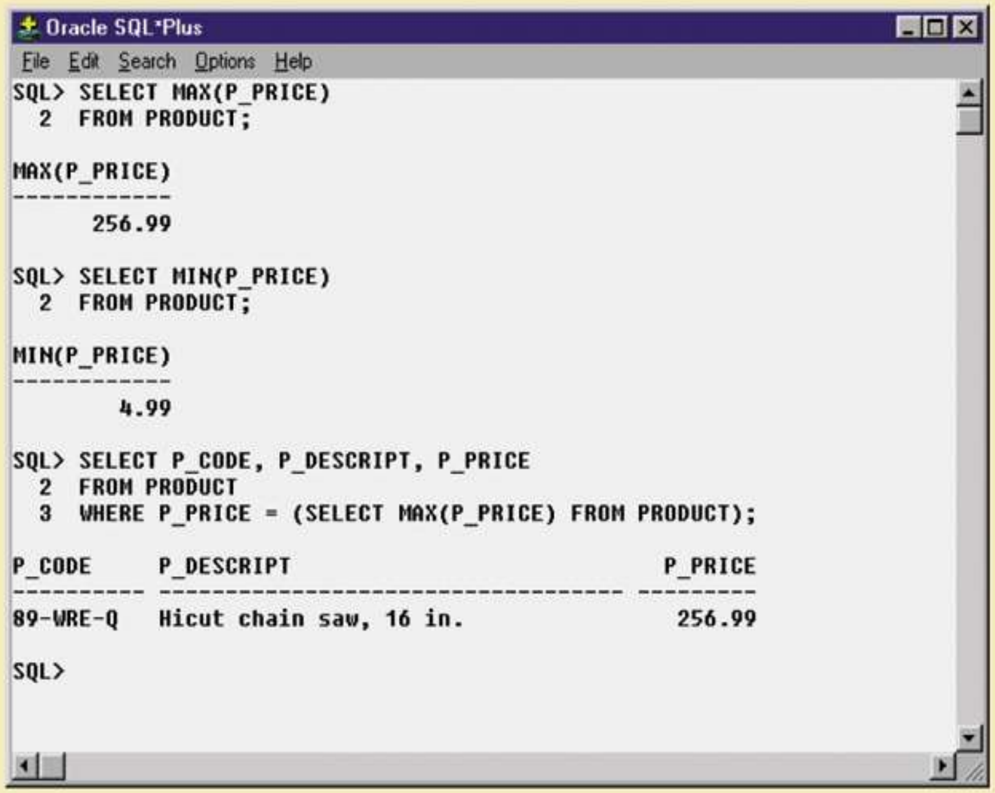
FIGURE 7.21 COUNT function output examples



```
SQL> SELECT COUNT(DISTINCT V_CODE)
  2  FROM PRODUCT;

COUNT(DISTINCTV_CODE)
---------------------
                    6

SQL> SELECT COUNT(DISTINCT V_CODE)
  2  FROM PRODUCT
  3  WHERE P_PRICE <= 10.00;

COUNT(DISTINCTV_CODE)
---------------------
                    3

SQL> SELECT COUNT(*)
  2  FROM PRODUCT
  3  WHERE P_PRICE <= 10.00;

  COUNT(*)
----------
         5

SQL> |
```

Example Output of MAX and MIN Functions

FIGURE 7.22 MAX and MIN function output examples



```
SQL> SELECT MAX(P_PRICE)
  2  FROM PRODUCT;

MAX(P_PRICE)
------------
      256.99

SQL> SELECT MIN(P_PRICE)
  2  FROM PRODUCT;

MIN(P_PRICE)
------------
        4.99

SQL> SELECT P_CODE, P_DESCRIPT, P_PRICE
  2  FROM PRODUCT
  3  WHERE P_PRICE = (SELECT MAX(P_PRICE) FROM PRODUCT);

P_CODE     P_DESCRIPT                            P_PRICE
---------- ----------------------------------- ---------
89-WRE-Q   Hicut chain saw, 16 in.                256.99

SQL>
```

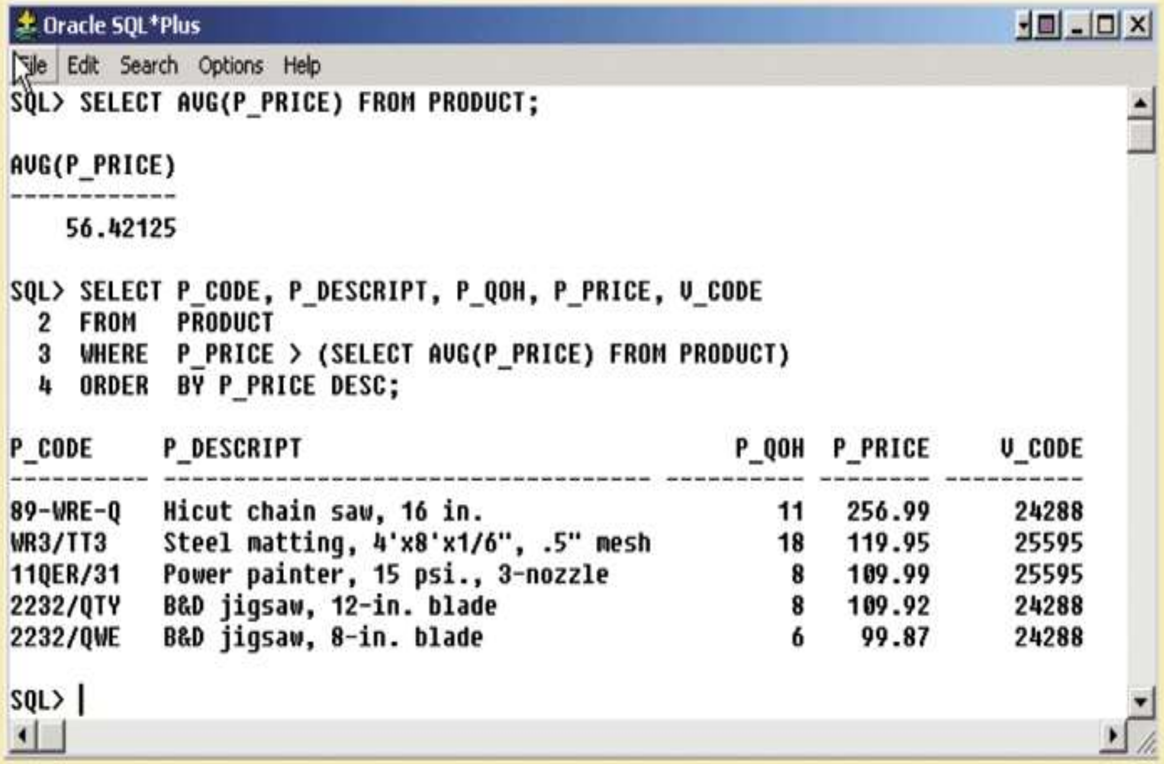
Example of SUM and AVG Functions

- SUM
  - **SELECT SUM** (P_ONHAND * P_PRICE)
    FROM PRODUCT

- AVG
  - **SELECT AVG** (P_PRICE)
    FROM PRODUCT

Example Output of AVG Function

FIGURE 7.24 AVG function output examples



```
SQL> SELECT AVG(P_PRICE) FROM PRODUCT;

AVG(P_PRICE)
------------
    56.42125

SQL> SELECT P_CODE, P_DESCRIPT, P_QOH, P_PRICE, V_CODE
  2  FROM   PRODUCT
  3  WHERE  P_PRICE > (SELECT AVG(P_PRICE) FROM PRODUCT)
  4  ORDER  BY P_PRICE DESC;

P_CODE     P_DESCRIPT                               P_QOH  P_PRICE     V_CODE
---------- ----------------------------------- ---------- -------- ----------
89-WRE-Q   Hicut chain saw, 16 in.                     11   256.99      24288
WR3/TT3    Steel matting, 4'x8'x1/6", .5" mesh         18   119.95      25595
11QER/31   Power painter, 15 psi., 3-nozzle             8   109.99      25595
2232/QTY   B&D jigsaw, 12-in. blade                     8   109.92      24288
2232/QWE   B&D jigsaw, 8-in. blade                      6    99.87      24288

SQL> |
```

Grouping Data

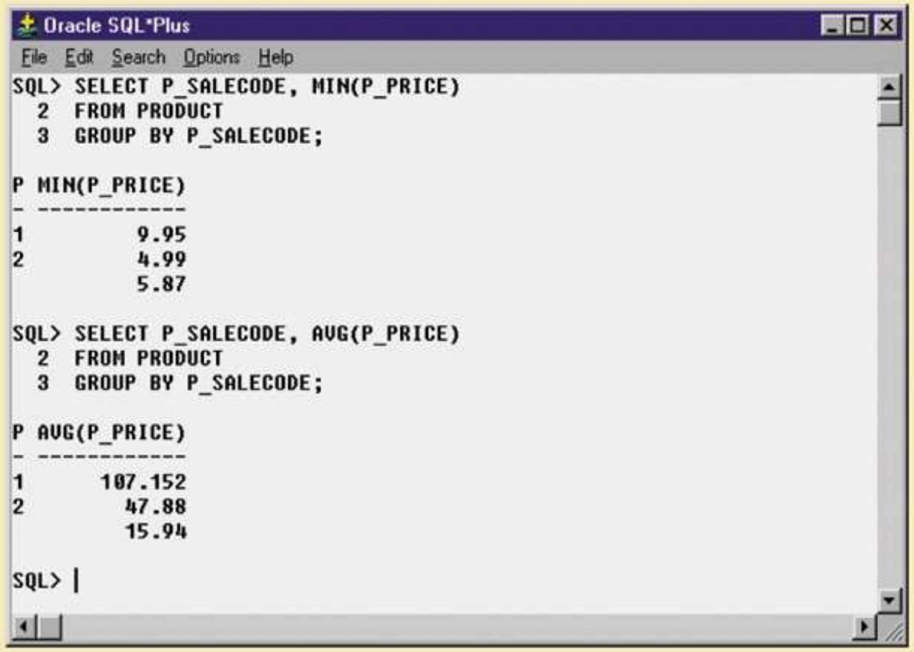
- Frequency distributions created by **GROUP BY** clause within SELECT statement
- Syntax:

  -
    ```
    SELECT      columnlist
    FROM        tablelist
    [WHERE      conditionlist]
    [GROUP BY   columnlist]
    [HAVING     conditionlist]
    [ORDER BY   columnlist [ASC | DESC] ]
    ```
- Only valid when used in conjunction with one of the SQL aggregate function (COUNT, MIN, MAX,AVG,SUM)

Example output of GROUP BY clause

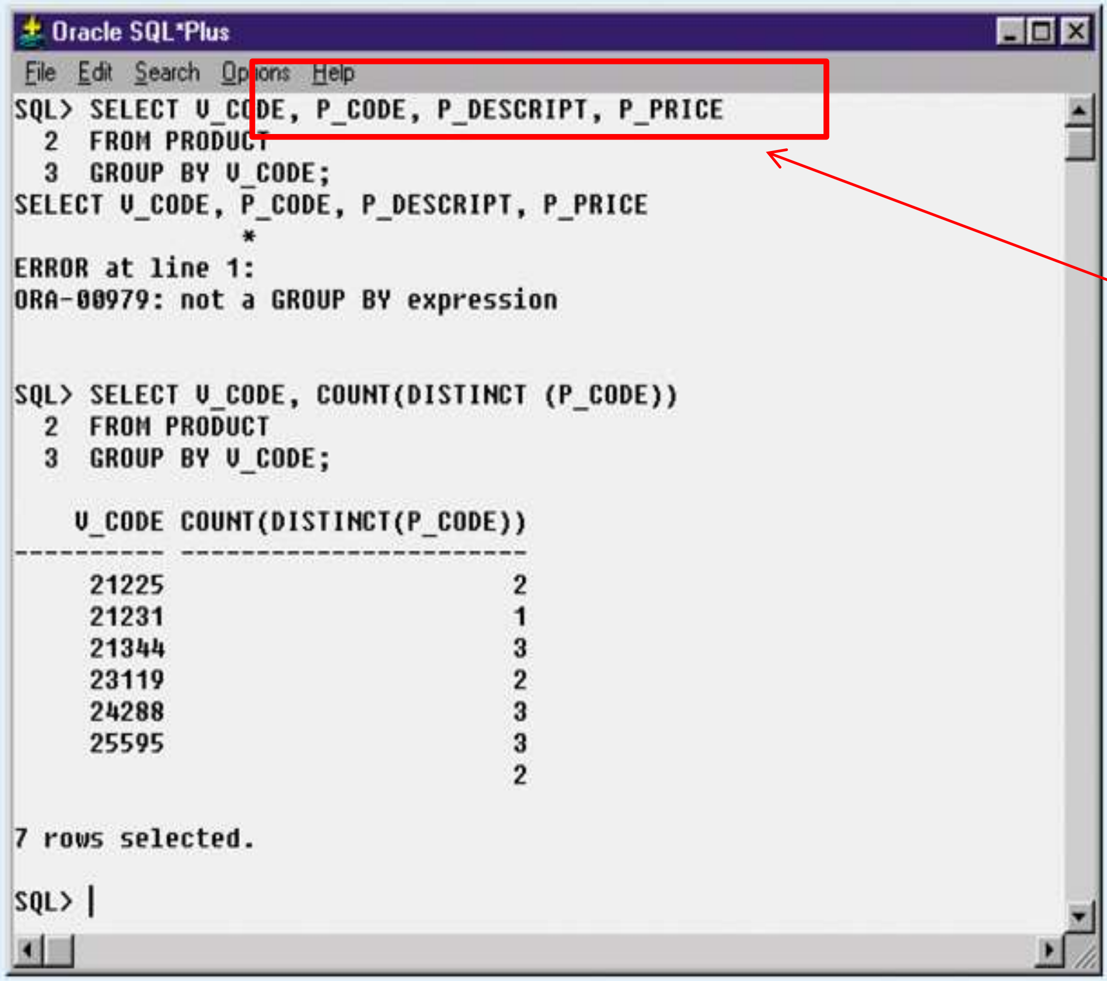
SELECT P_SALECODE, **MIN**(P_PRICE)
FROM PRODUCT
GROUP BY P_SALECODE

FIGURE 7.25 GROUP BY clause output examples



```
SQL> SELECT P_SALECODE, MIN(P_PRICE)
  2  FROM PRODUCT
  3  GROUP BY P_SALECODE;

P MIN(P_PRICE)
- ------------
1         9.95
2         4.99
          5.87

SQL> SELECT P_SALECODE, AVG(P_PRICE)
  2  FROM PRODUCT
  3  GROUP BY P_SALECODE;

P AVG(P_PRICE)
- ------------
1      107.152
2        47.88
         15.94

SQL> |
```

FIGURE 7.26 Incorrect and correct use of the GROUP BY clause



```
SQL> SELECT V_CODE, P_CODE, P_DESCRIPT, P_PRICE
  2  FROM PRODUCT
  3  GROUP BY V_CODE;
SELECT V_CODE, P_CODE, P_DESCRIPT, P_PRICE
               *
ERROR at line 1:
ORA-00979: not a GROUP BY expression


SQL> SELECT V_CODE, COUNT(DISTINCT (P_CODE))
  2  FROM PRODUCT
  3  GROUP BY V_CODE;

    V_CODE COUNT(DISTINCT(P_CODE))
---------- -----------------------
     21225                       2
     21231                       1
     21344                       3
     23119                       2
     24288                       3
     25595                       3
                                 2

7 rows selected.

SQL> |
```

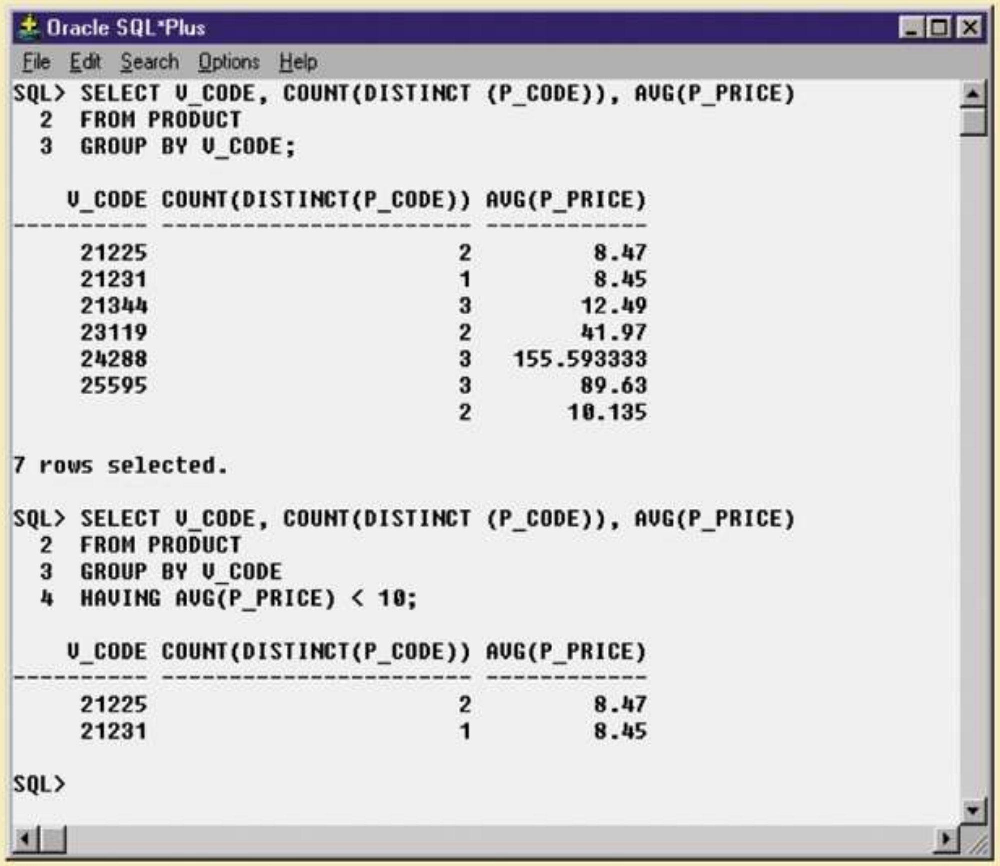
NO aggregate function (COUNT, MIN, MAX,AVG,SUM)

Grouping Data with HAVING Clause

- **HAVING** clause
  - Operates very much like WHERE clause
  - **WHERE** clause – applies to columns and expressions for individual rows
  - **HAVING** clause – applies to the output of a **GROUP BY** operation

Grouping Data with HAVING Clause

FIGURE 7.27 An application of the HAVING clause



```
SQL> SELECT V_CODE, COUNT(DISTINCT (P_CODE)), AVG(P_PRICE)
  2  FROM PRODUCT
  3  GROUP BY V_CODE;

    V_CODE COUNT(DISTINCT(P_CODE)) AVG(P_PRICE)
---------- ----------------------- ------------
     21225                       2         8.47
     21231                       1         8.45
     21344                       3        12.49
     23119                       2        41.97
     24288                       3   155.593333
     25595                       3        89.63
                                 2       10.135

7 rows selected.

SQL> SELECT V_CODE, COUNT(DISTINCT (P_CODE)), AVG(P_PRICE)
  2  FROM PRODUCT
  3  GROUP BY V_CODE
  4  HAVING AVG(P_PRICE) < 10;

    V_CODE COUNT(DISTINCT(P_CODE)) AVG(P_PRICE)
---------- ----------------------- ------------
     21225                       2         8.47
     21231                       1         8.45

SQL>
```

Subqueries/Nested Queries

- A subquery is a query inside a query
  - Normally expressed inside parentheses
- The first query in the SQL statement is known as the **outer** query
- The query inside the SQL statement known as the **inner** query
  - Inner query executed first
  - Output of an inner query is used as the input for the output query
- The entire SQL statement is sometimes referred to as **nested queries**

Subqueries/Nested Queries *(Used with Equal Operator or IN Operator)*

- Example:
  - **UPDATE** Product
    SET P_Price =
    (SELECT AVG (P_Price) FROM Product)
    WHERE V_Code **IN**
    (SELECT V_CODE FROM Vendor WHERE
    V_AREACODE = '615')

Outer query

Inner query

Inner query

Subqueries/Nested Queries *(Used with Equal Operator or IN Operator)*

- Example:

Outer query

```
DELETE FROM Product
WHERE V_Code IN
        (SELECT V_CODE FROM Vendor WHERE
                        V_AREACODE  = '615')
```

Inner query

Copying Parts of Tables

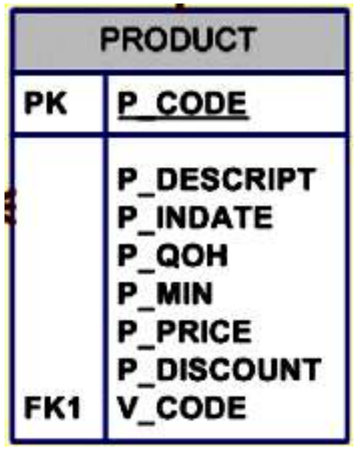
- SQL permits copying contents of selected table columns
  - Data need not re-entered manually into newly created table(s)
  - Can use **SUBQUERIES**

- 1st Step: Create new table structure

- 2nd Step: Add rows to new table using table rows from another table

Copying Parts of Tables

```
CREATE TABLE PART (
 PART_CODE       CHAR(8) NOT NULL,
 PART_DESCRIPT CHAR(35),
 PART_PRICE      DECIMAL(8,2),
V_CODE              VARCHAR2(5),
 PRIMARY KEY(PART_CODE) )
```



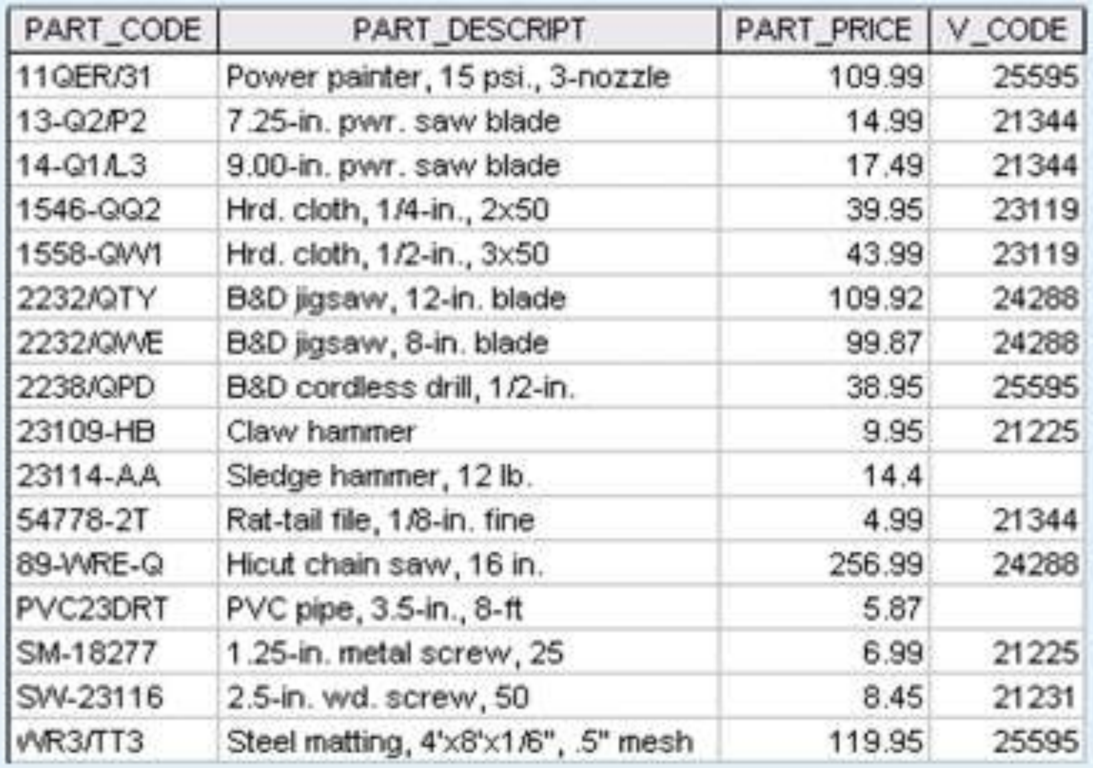
New table

Copy from Product table

```
INSERT INTO PART (PART_CODE, PART_DESCRIPT,
                  PART_PRICE,V_CODE)
   SELECT P_CODE, P_DESCRIPT, P_PRICE,V_CODE FROM
          PRODUCT
```

Copying Parts of Tables

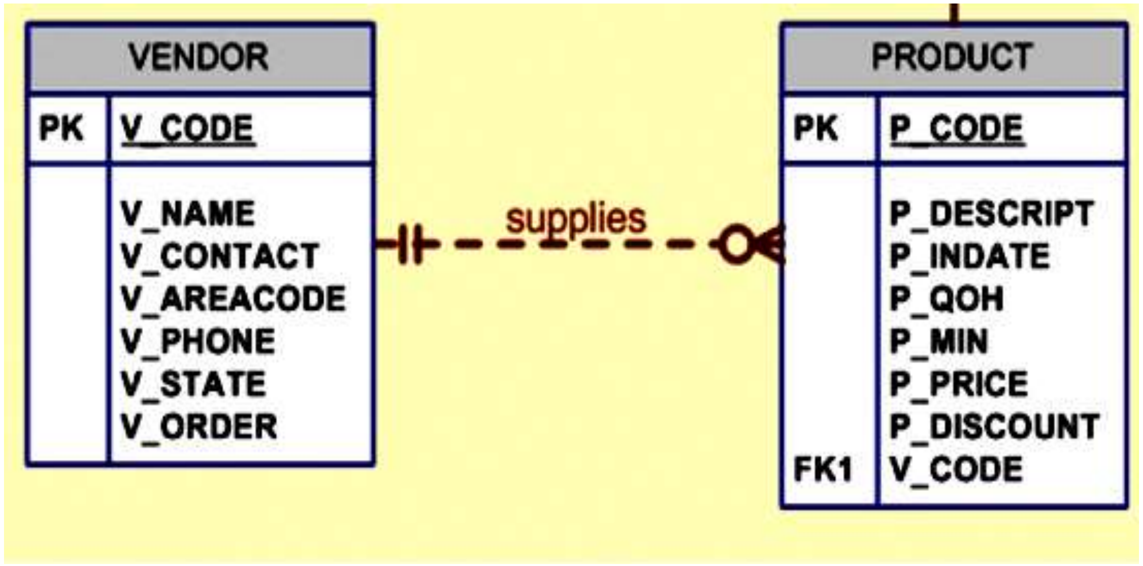
FIGURE 7.16 PART table attributes copied from the PRODUCT table



PART_CODE	PART_DESCRIPT	PART_PRICE	V_CODE
11QER/31	Power painter, 15 psi., 3-nozzle	109.99	25595
13-Q2/P2	7.25-in. pwr. saw blade	14.99	21344
14-Q1/L3	9.00-in. pwr. saw blade	17.49	21344
1546-QQ2	Hrd. cloth, 1/4-in., 2x50	39.95	23119
1558-QW1	Hrd. cloth, 1/2-in., 3x50	43.99	23119
2232/QTY	B&D jigsaw, 12-in. blade	109.92	24288
2232/QWE	B&D jigsaw, 8-in. blade	99.87	24288
2238/QPD	B&D cordless drill, 1/2-in.	38.95	25595
23109-HB	Claw hammer	9.95	21225
23114-AA	Sledge hammer, 12 lb.	14.4	
54778-2T	Rat-tail file, 1/8-in. fine	4.99	21344
89-WRE-Q	Hicut chain saw, 16 in.	256.99	24288
PVC23DRT	PVC pipe, 3.5-in., 8-ft	5.87	
SM-18277	1.25-in. metal screw, 25	6.99	21225
SW-23116	2.5-in. wd. screw, 50	8.45	21231
WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	119.95	25595

Joining Database Tables

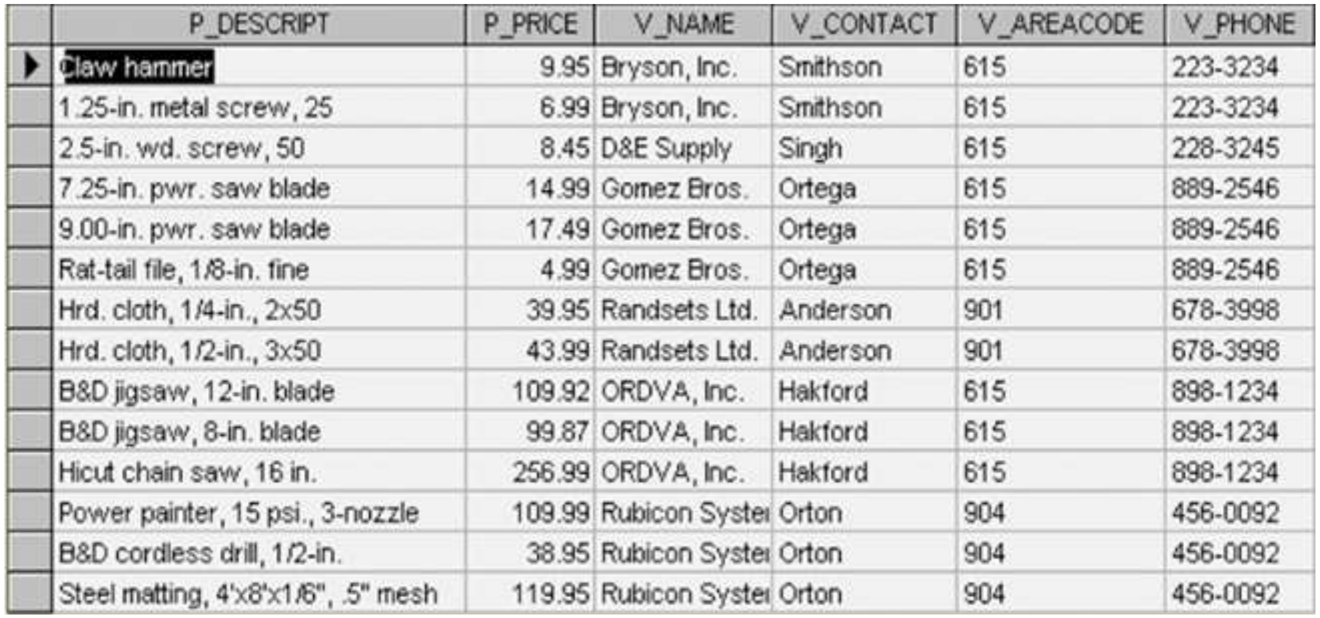
- Join is performed when data are retrieved from more than one table at a time
  - Use equality comparison (=) to join foreign key and primary key of related tables



Joining Database Tables: INNER JOIN

- **SELECT** P_DESCRIPT, P_PRICE, V_NAME, V_CONTACT, V_AREACODE, V_PHONE

  FROM PRODUCT **INNER JOIN** VENDOR
- **ON PRODUCT.V_CODE = VENDOR.V_CODE**



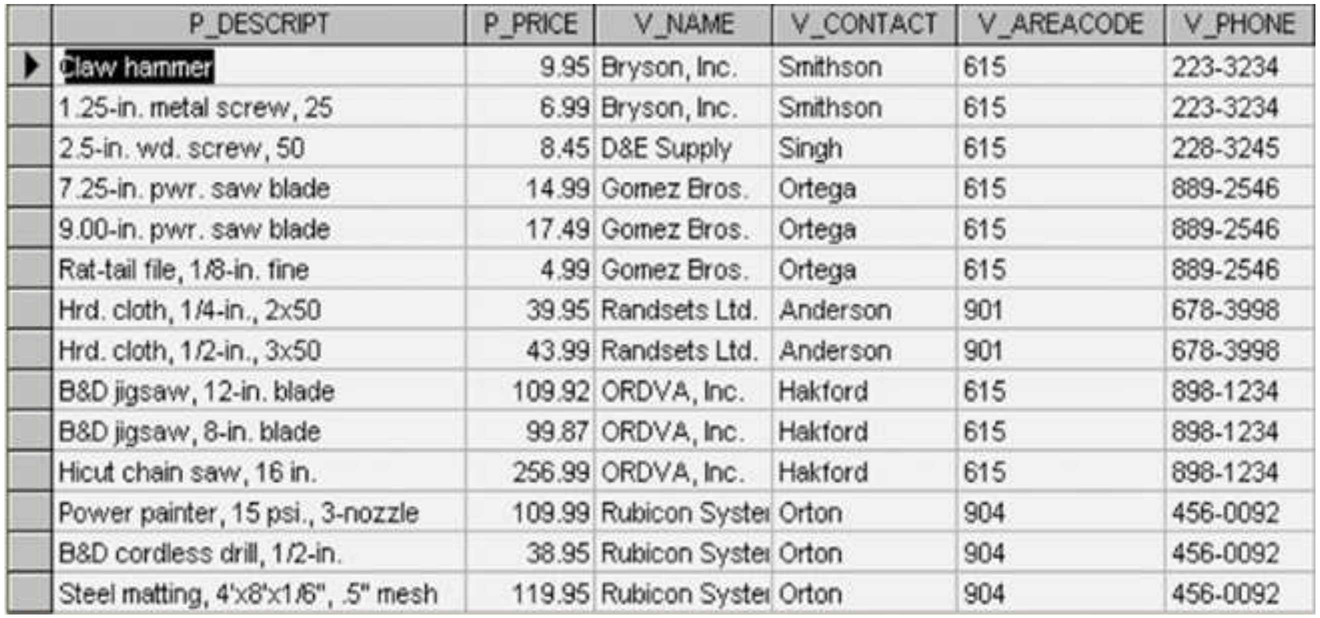
P_DESCRIPT	P_PRICE	V_NAME	V_CONTACT	V_AREACODE	V_PHONE
Claw hammer	9.95	Bryson, Inc.	Smithson	615	223-3234
1.25-in. metal screw, 25	6.99	Bryson, Inc.	Smithson	615	223-3234
2.5-in. wd. screw, 50	8.45	D&E Supply	Singh	615	228-3245
7.25-in. pwr. saw blade	14.99	Gomez Bros.	Ortega	615	889-2546
9.00-in. pwr. saw blade	17.49	Gomez Bros.	Ortega	615	889-2546
Rat-tail file, 1/8-in. fine	4.99	Gomez Bros.	Ortega	615	889-2546
Hrd. cloth, 1/4-in., 2x50	39.95	Randsets Ltd.	Anderson	901	678-3998
Hrd. cloth, 1/2-in., 3x50	43.99	Randsets Ltd.	Anderson	901	678-3998
B&D jigsaw, 12-in. blade	109.92	ORDVA, Inc.	Hakford	615	898-1234
B&D jigsaw, 8-in. blade	99.87	ORDVA, Inc.	Hakford	615	898-1234
Hicut chain saw, 16 in.	256.99	ORDVA, Inc.	Hakford	615	898-1234
Power painter, 15 psi., 3-nozzle	109.99	Rubicon Systei	Orton	904	456-0092
B&D cordless drill, 1/2-in.	38.95	Rubicon Systei	Orton	904	456-0092
Steel matting, 4'x8'x1/6", .5" mesh	119.95	Rubicon Systei	Orton	904	456-0092

Joining Database Tables: Equal Operator

- SELECT P_DESCRIPT, P_PRICE, V_NAME, V_CONTACT, V_AREACODE, V_PHONE

  FROM PRODUCT, VENDOR

  WHERE PRODUCT.V_CODE = VENDOR.V_CODE

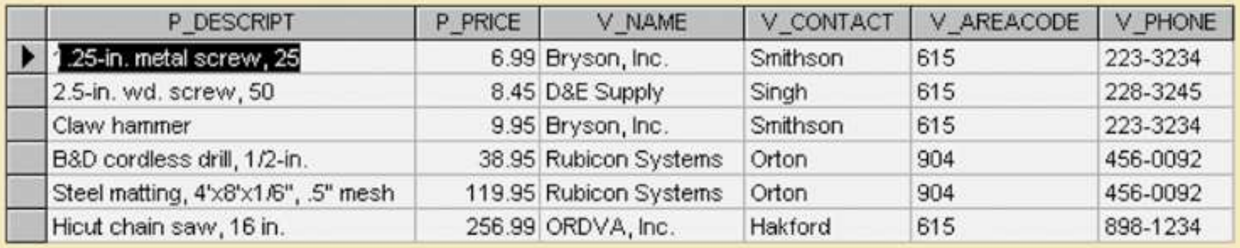


P_DESCRIPT	P_PRICE	V_NAME	V_CONTACT	V_AREACODE	V_PHONE
Claw hammer	9.95	Bryson, Inc.	Smithson	615	223-3234
1.25-in. metal screw, 25	6.99	Bryson, Inc.	Smithson	615	223-3234
2.5-in. wd. screw, 50	8.45	D&E Supply	Singh	615	228-3245
7.25-in. pwr. saw blade	14.99	Gomez Bros.	Ortega	615	889-2546
9.00-in. pwr. saw blade	17.49	Gomez Bros.	Ortega	615	889-2546
Rat-tail file, 1/8-in. fine	4.99	Gomez Bros.	Ortega	615	889-2546
Hrd. cloth, 1/4-in., 2x50	39.95	Randsets Ltd.	Anderson	901	678-3998
Hrd. cloth, 1/2-in., 3x50	43.99	Randsets Ltd.	Anderson	901	678-3998
B&D jigsaw, 12-in. blade	109.92	ORDVA, Inc.	Hakford	615	898-1234
B&D jigsaw, 8-in. blade	99.87	ORDVA, Inc.	Hakford	615	898-1234
Hicut chain saw, 16 in.	256.99	ORDVA, Inc.	Hakford	615	898-1234
Power painter, 15 psi., 3-nozzle	109.99	Rubicon Systei	Orton	904	456-0092
B&D cordless drill, 1/2-in.	38.95	Rubicon Systei	Orton	904	456-0092
Steel matting, 4'x8'x1/6", .5" mesh	119.95	Rubicon Systei	Orton	904	456-0092

Joining Database Tables

- SELECT P_DESCRIPT, P_PRICE, V_NAME, V_CONTACT,
  V_AREACODE, V_PHONE
  FROM PRODUCT, VENDOR
  WHERE PRODUCT.V_CODE = VENDOR.V_CODE
  AND P_INDATE > '1999-08-15'

FIGURE 7.30 An ordered and limited listing after a join



P_DESCRIPT	P_PRICE	V_NAME	V_CONTACT	V_AREACODE	V_PHONE
1.25-in. metal screw, 25	6.99	Bryson, Inc.	Smithson	615	223-3234
2.5-in. wd. screw, 50	8.45	D&E Supply	Singh	615	228-3245
Claw hammer	9.95	Bryson, Inc.	Smithson	615	223-3234
B&D cordless drill, 1/2-in.	38.95	Rubicon Systems	Orton	904	456-0092
Steel matting, 4'x8'x1/6", .5" mesh	119.95	Rubicon Systems	Orton	904	456-0092
Hicut chain saw, 16 in.	256.99	ORDVA, Inc.	Hakford	615	898-1234

Joining Tables with an Alias

- Alias can be used to identify source table
- Any legal table name can be used as alias
- Add alias after table name in FROM clause
  - FROM <*tablename*> <*alias*>

- Example:

```
SELECT P_DESCRIPT,  P_PRICE, V_NAME,  V_CONTACT,
                    V_AREACODE, V_PHONE
FROM PRODUCT P,  VENDOR V
WHERE P.V_CODE = V.V_CODE
ORDER BY P_PRICE
```

Outer Joins

- Outer join (recap from lecture 2!)
  - Returns not only the rows matching the join condition but also the rows with unmatched values
- Two types of outer join
  - *Left outer join*
  - *Right outer join*

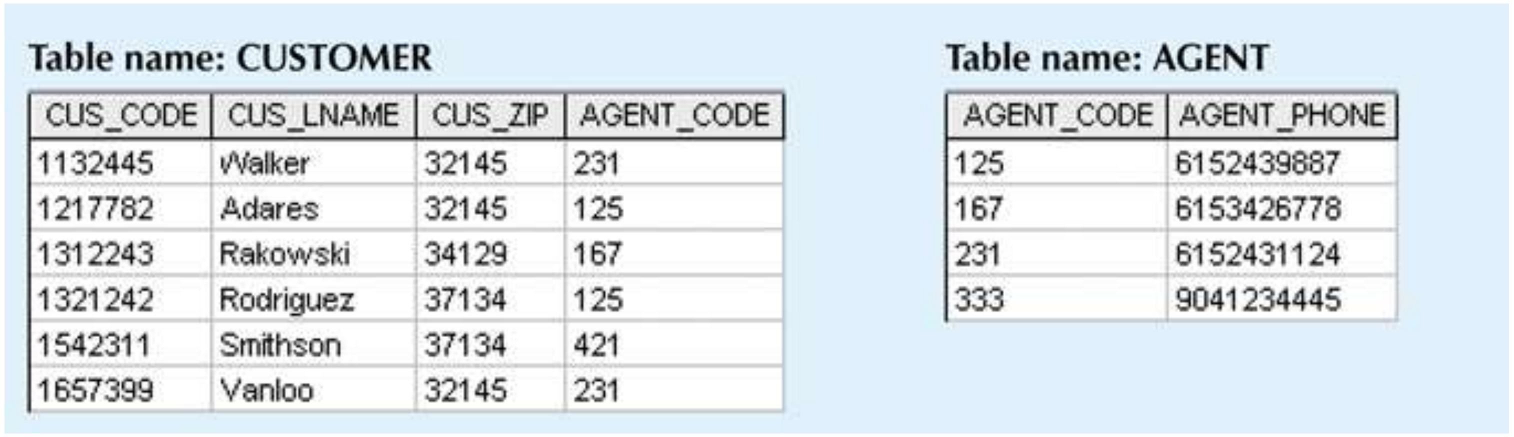


Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CODE
1132445	Walker	32145	231
1217782	Adares	32145	125
1312243	Rakowski	34129	167
1321242	Rodriguez	37134	125
1542311	Smithson	37134	421
1657399	Vanloo	32145	231

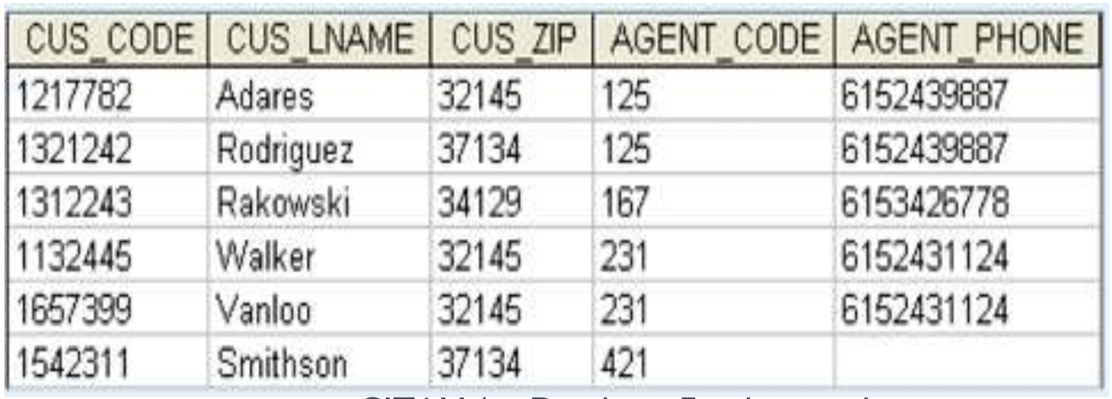
Table name: AGENT

AGENT_CODE	AGENT_PHONE
125	6152439887
167	6153426778
231	6152431124
333	9041234445

Outer Joins

- **LEFT join**
  - Returns rows in the left side table with unmatched values in the right side table
  - Example:

```
SELECT CUS_CODE, CUS_LNAME, CUS_ZIP,
        AGENT_CODE,  AGENT_PHONE
FROM CUSTOMER LEFT JOIN AGENT ON
      CUSTOMER.AGENT_CODE = AGENT.AGENT_CODE
```

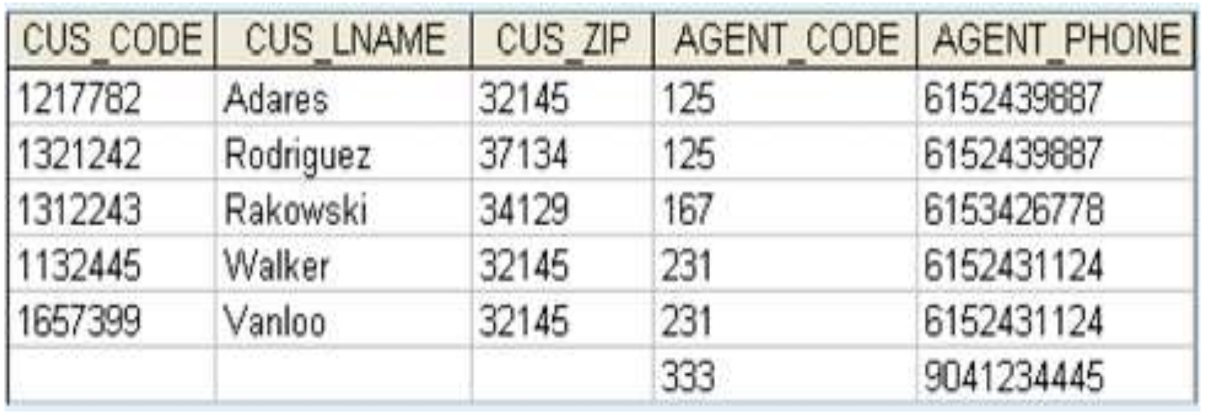


CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CODE	AGENT_PHONE
1217782	Adares	32145	125	6152439887
1321242	Rodriguez	37134	125	6152439887
1312243	Rakowski	34129	167	6153426778
1132445	Walker	32145	231	6152431124
1657399	Vanloo	32145	231	6152431124
1542311	Smithson	37134	421	

Outer Joins

- **RIGHT join**
  - Returns rows in the right side table with unmatched values in the left side table
  - Example:

```
SELECT CUS_CODE, CUS_LNAME, CUS_ZIP,
        AGENT_CODE,  AGENT_PHONE
FROM CUSTOMER RIGHT JOIN AGENT ON
      CUSTOMER.AGENT_CODE = AGENT.AGENT_CODE
```



CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CODE	AGENT_PHONE
1217782	Adares	32145	125	6152439887
1321242	Rodriguez	37134	125	6152439887
1312243	Rakowski	34129	167	6153426778
1132445	Walker	32145	231	6152431124
1657399	Vanloo	32145	231	6152431124
			333	9041234445

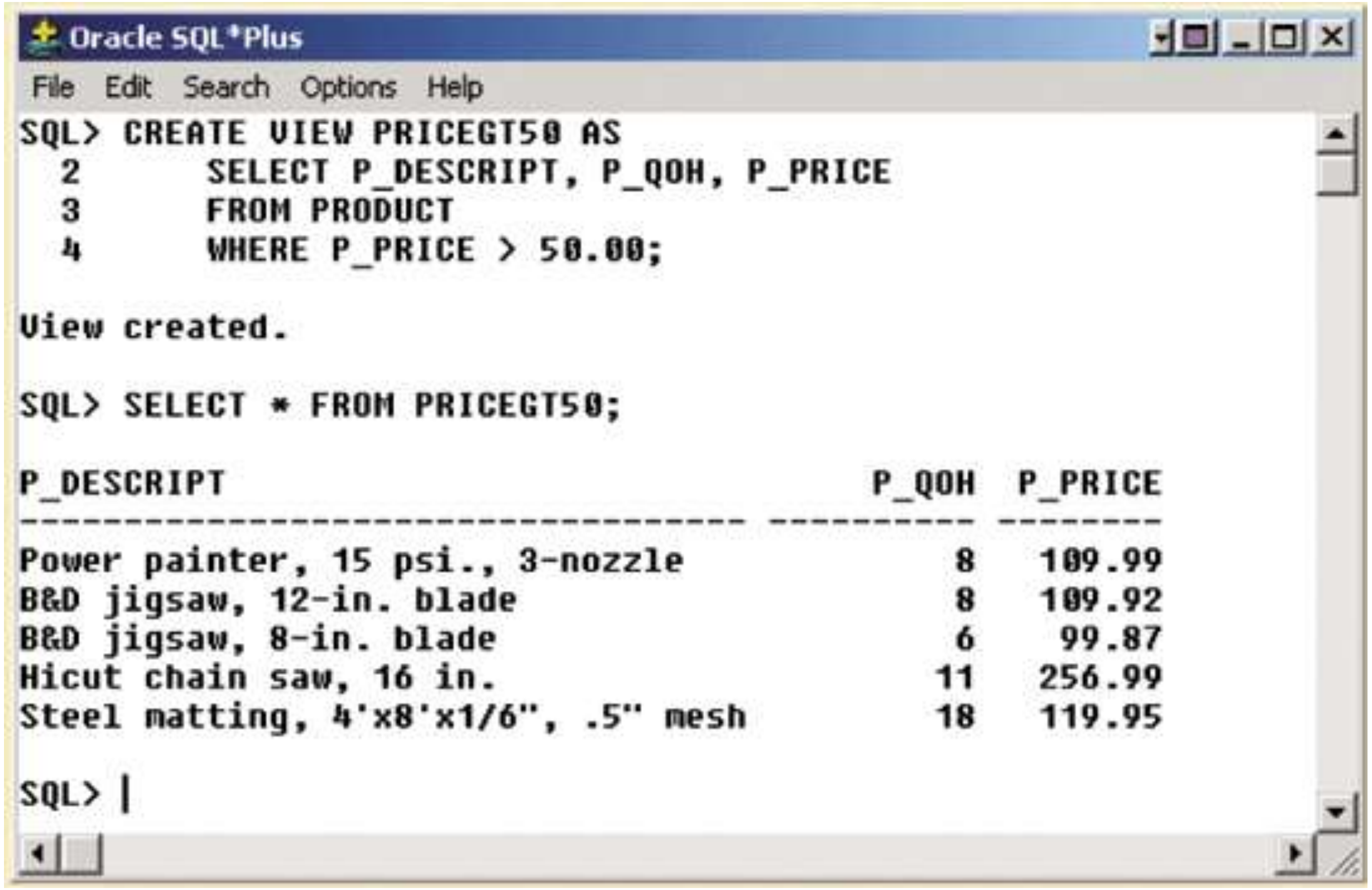
Virtual Tables: Creating a View

- View is a virtual table based on SELECT query
- Create view by using **CREATE VIEW** command
- Syntax:
  - **CREATE VIEW** *<viewname>* **AS**
    SELECT *statement*

- Example:
  - **CREATE VIEW** ProductView **AS**
    SELECT * from Product

  - To display the contents of the virtual table:
    SELECT * FROM ProductView

Virtual Tables: Creating a View

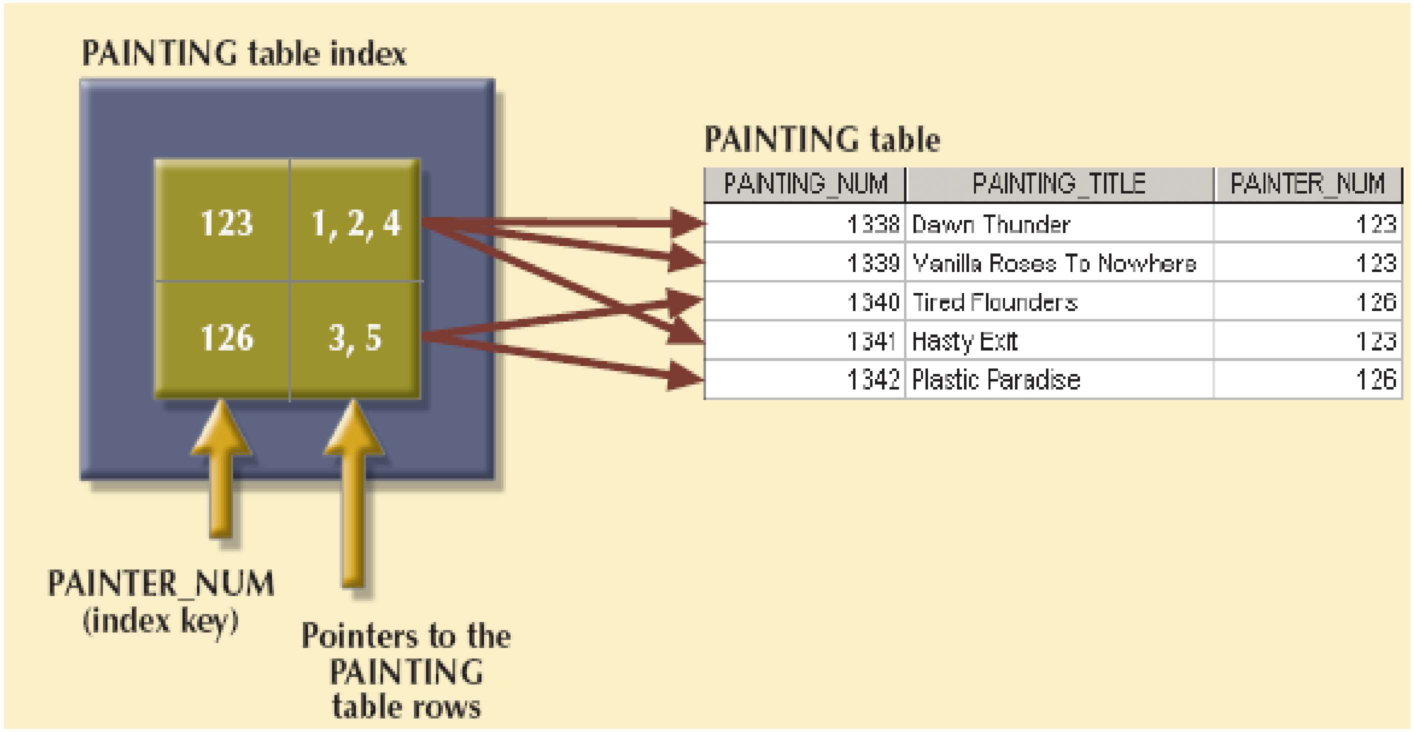


```
SQL> CREATE VIEW PRICEGT50 AS
  2       SELECT P_DESCRIPT, P_QOH, P_PRICE
  3       FROM PRODUCT
  4       WHERE P_PRICE > 50.00;

View created.

SQL> SELECT * FROM PRICEGT50;

P_DESCRIPT                                P_QOH   P_PRICE
----------------------------------- ---------- --------
Power painter, 15 psi., 3-nozzle              8   109.99
B&D jigsaw, 12-in. blade                      8   109.92
B&D jigsaw, 8-in. blade                       6    99.87
Hicut chain saw, 16 in.                      11   256.99
Steel matting, 4'x8'x1/6", .5" mesh          18   119.95

SQL> |
```

SQL Indexes

- When primary key is declared, DBMS automatically creates unique index
- Often need additional indexes to improve search
- Syntax:
  - **CREATE INDEX** <indexname> **ON** <tablename (column)>
- Example:
  - **CREATE INDEX** P_INDATEX **ON** PRODUCT (P_INDATE)
  - DROP INDEX P_INDATEX

SQL Indexes



PAINTING_NUM	PAINTING_TITLE	PAINTER_NUM
1338	Dawn Thunder	123
1339	Vanilla Roses To Nowhere	123
1340	Tired Flounders	126
1341	Hasty Exit	123
1342	Plastic Paradise	126