Chapter 7
Introduction to Structured Query Language (SQL)

Learning Objectives

- In this chapter, you will learn:
  - The basic commands and functions of SQL
  - How to use SQL for data administration (to create tables and indexes)
  - How to use SQL for data manipulation (to add, modify, delete, and retrieve data)
  - How to use SQL to query a database for useful information

Introduction to SQL

- Categories of SQL functions:
  - Data definition language (DDL)
  - Data manipulation language (DML)
- Nonprocedural language with basic command vocabulary set of less than 100 words
- Differences in SQL dialects are minor

Table 7.1 - SQL Data Definition Command

TABLE 7.1

SQL DATA DEFINITION COMMANDS

COMMAND OR OPTION	DESCRIPTION
CREATE SCHEMA AUTHORIZATION	Creates a database schema
CREATE TABLE	Creates a new table in the user's database schema
NOT NULL	Ensures that a column will not have null values
UNIQUE	Ensures that a column will not have duplicate values
PRIMARY KEY	Defines a primary key for a table
FOREIGN KEY	Defines a foreign key for a table
DEFAULT	Defines a default value for a column (when no value is given)
CHECK	Validates data in an attribute
CREATE INDEX	Creates an index for a table
CREATE VIEW	Creates a dynamic subset of rows and columns from one or more tables (see Chapter 8, Advanced SQL)
ALTER TABLE	Modifies a table's definition (adds, modifies, or deletes attributes or constraints)
CREATE TABLE AS	Creates a new table based on a query in the user's database schema
DROP TABLE	Permanently deletes a table (and its data)
DROP INDEX	Permanently deletes an index
DROP VIEW	Permanently deletes a view

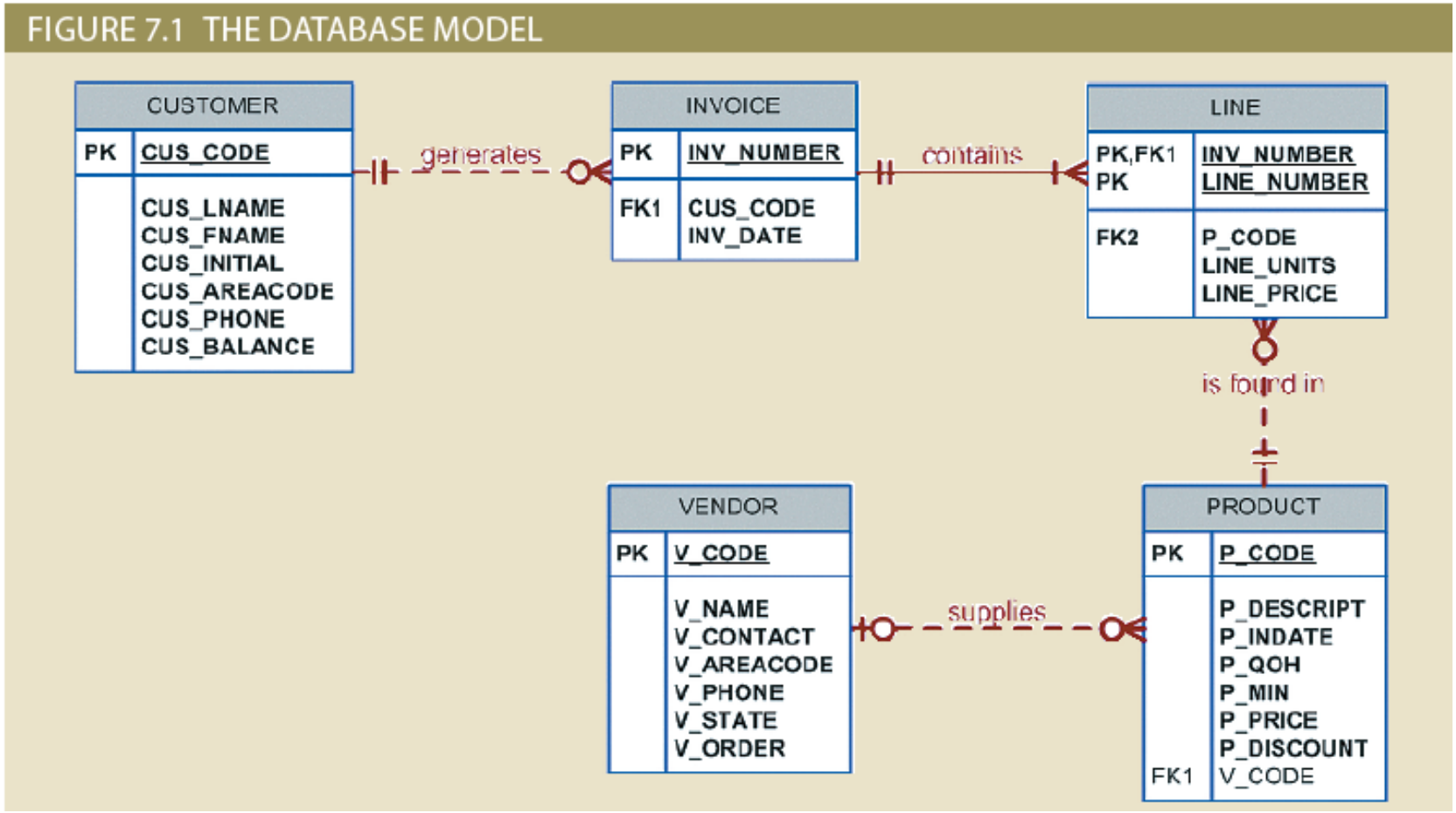
Table 7.2 - SQL Data Manipulation Commands

TABLE 7.2

SQL DATA MANIPULATION COMMANDS

COMMAND OR OPTION	DESCRIPTION
INSERT	Inserts row(s) into a table
SELECT	Selects attributes from rows in one or more tables or views
WHERE	Restricts the selection of rows based on a conditional expression
GROUP BY	Groups the selected rows based on one or more attributes
HAVING	Restricts the selection of grouped rows based on a condition
ORDER BY	Orders the selected rows based on one or more attributes
UPDATE	Modifies an attribute's values in one or more table's rows
DELETE	Deletes one or more rows from a table
COMMIT	Permanently saves data changes
ROLLBACK	Restores data to its original values
Comparison operators	
=, <, >, <=, >=, <>, !=	Used in conditional expressions
Logical operators	
AND/OR/NOT	Used in conditional expressions
Special operators	Used in conditional expressions
BETWEEN	Checks whether an attribute value is within a range
IS NULL	Checks whether an attribute value is null
LIKE	Checks whether an attribute value matches a given string pattern
IN	Checks whether an attribute value matches any value within a value list
EXISTS	Checks whether a subquery returns any rows
DISTINCT	Limits values to unique values
Aggregate functions	Used with SELECT to return mathematical summaries on columns
COUNT	Returns the number of rows with non-null values for a given column
MIN	Returns the minimum attribute value found in a given column
MAX	Returns the maximum attribute value found in a given column
SUM	Returns the sum of all values for a given column
AVG	Returns the average of all values for a given column

Figure 7.1 - The Database Model



Creating the Database

- Create database structure
  - RDBMS creates physical files that will hold database
  - Differs from one RDBMS to another
- **Authentication** is the process DBMS uses to verify that only registered users access the database
  - Required for the creation tables
  - User should log on to RDBMS using user ID and password created by database administrator

The Database Schema

- Logical group of database objects – such as tables and indexes - related to each other
- Command:
  - CREATE SCHEMA AUTHORIZATION {creator};
  - Seldom used directly as command is usually optional

Table 7.4 - Common SQL Data Types

TABLE 7.4

SOME COMMON SQL DATA TYPES

DATA TYPE	FORMAT	COMMENTS
Numeric	NUMBER(L,D) or NUMERIC(L,D)	The declaration NUMBER(7,2) or NUMERIC(7,2) indicates that numbers will be stored with two decimal places and may be up to seven digits long, including the sign and the decimal place (for example, 12.32 or –134.99).
	INTEGER	May be abbreviated as INT. Integers are (whole) counting numbers, so they cannot be used if you want to store numbers that require decimal places.
	SMALLINT	Like INTEGER but limited to integer values up to six digits. If your integer values are relatively small, use SMALLINT instead of INT.
	DECIMAL(L,D)	Like the NUMBER specification, but the storage length is a *minimum* specification. That is, greater lengths are acceptable, but smaller ones are not. DECIMAL(9,2), DECIMAL(9), and DECIMAL are all acceptable.
Character	CHAR(L)	Fixed-length character data for up to 255 characters. If you store strings that are not as long as the CHAR parameter value, the remaining spaces are left unused. Therefore, if you specify CHAR(25), strings such as *Smith* and *Katzenjammer* are each stored as 25 characters. However, a U.S. area code is always three digits long, so CHAR(3) would be appropriate if you wanted to store such codes.
	VARCHAR(L) or VARCHAR2(L)	Variable-length character data. The designation VARCHAR2(25) or VARCHAR(25) will let you store characters up to 25 characters long. However, unlike CHAR, VARCHAR will not leave unused spaces. Oracle automatically converts VARCHAR to VARCHAR2.
Date	DATE	Stores dates in the Julian date format.

Creating Table Structures

- Use one line per column (attribute) definition
- Use spaces to line up attribute characteristics and constraints
- Table and attribute names are fully capitalized
- Features of table creating command sequence:
  - NOT NULL specification ensures data entry
  - UNIQUE specification avoids duplicated values
- Table definition enclosed in parentheses
- RDBMS automatically enforces referential integrity for foreign keys.

SQL Constraints

NOT NULL

- Ensures that column does not accept nulls

UNIQUE

- Ensures that all values in column are unique

DEFAULT

- Assigns value to attribute when a new row is added to table

CHECK

- Validates data when attribute value is entered

```
CREATE TABLE tablename (
    column1     data type     [constraint] [,
    column2     data type     [constraint] ] [,
    PRIMARY KEY (column1      [, column2]) ] [,
    FOREIGN KEY (column1      [, column2]) REFERENCES
                              tablename] [,
    CONSTRAINT  constraint ] );
```

```
CREATE TABLE VENDOR (
V_CODE       INTEGER      NOT NULL  UNIQUE,
V_NAME       VARCHAR(35)  NOT NULL,
V_CONTACT    VARCHAR(25)  NOT NULL,
V_AREACODE   CHAR(3)      NOT NULL,
V_PHONE      CHAR(8)      NOT NULL,
V_STATE      CHAR(2)      NOT NULL,
V_ORDER      CHAR(1)      NOT NULL,
PRIMARY KEY (V_CODE));
```

```
CREATE TABLE PRODUCT (
P_CODE        VARCHAR(10)      NOT NULL      UNIQUE,
P_DESCRIPT    VARCHAR(35)      NOT NULL,
P_INDATE      DATE             NOT NULL,
P_QOH         SMALLINT         NOT NULL,
P_MIN         SMALLINT         NOT NULL,
P_PRICE       NUMBER(8,2)      NOT NULL,
P_DISCOUNT    NUMBER(5,2)      NOT NULL,
V_CODE        INTEGER,
PRIMARY KEY (P_CODE),
FOREIGN KEY (V_CODE) REFERENCES VENDOR ON UPDATE
CASCADE);
```

```
CREATE TABLE CUSTOMER (
CUS_CODE        NUMBER          PRIMARY KEY,
CUS_LNAME       VARCHAR(15)     NOT NULL,
CUS_FNAME       VARCHAR(15)     NOT NULL,
CUS_INITIAL     CHAR(1),
CUS_AREACODE    CHAR(3)         DEFAULT '615'    NOT NULL
                                CHECK(CUS_AREACODE IN
                                ('615','713','931')),
CUS_PHONE       CHAR(8)         NOT NULL,
CUS_BALANCE     NUMBER(9,2)     DEFAULT 0.00,
CONSTRAINT CUS_UI1 UNIQUE (CUS_LNAME, CUS_FNAME));
```

SQL Indexes

- When primary key is declared, DBMS automatically creates unique index
- The **CREATE INDEX** command can be used to create indexes on the basis of any selected attribute
- **UNIQUE** qualifier prevents a value that has been used before
- Composite indexes prevent data duplication
- To delete an index use the **DROP INDEX** command

```
CREATE INDEX P_INDATEX ON
PRODUCT(P_INDATE);
```

EMP_NUM	TEST_NUM	TEST_CODE	TEST_DATE	TEST_SCORE
110	1	WEA	15-Jan-2014	93
110	2	WEA	12-Jan-2014	87
111	1	HAZ	14-Dec-2013	91
111	2	WEA	18-Feb-2014	95
111	3	WEA	18-Feb-2014	95
112	1	CHEM	17-Aug-2013	91

```
CREATE UNIQUE INDEX EMP_TESTDEX ON TEST(EMP_NUM,
TEST_CODE, TEST_DATE);
```

Data Manipulation Commands

INSERT: Command to insert data into table

- Syntax - INSERT INTO tablename VALUES();
- Used to add table rows with NULL and NOT NULL attributes

COMMIT: Command to save changes

- Syntax - COMMIT [WORK];
- Ensures database update integrity

```
INSERT INTO VENDOR    VALUES (21225,'Bryson, Inc.','Smithson','615','223-3234','TN','Y');
INSERT INTO VENDOR    VALUES (21226,'Superloo, Inc.','Flushing','904','215-8995','FL','N');
```

```
INSERT INTO PRODUCT   VALUES ('11QER/31','Power painter, 15 psi., 3-nozzle','03-Nov-13',8,5,109.99,0.00,25595);
INSERT INTO PRODUCT   VALUES ('13-Q2/P2','7.25-in. pwr. saw blade','13-Dec-13',32,15,14.99, 0.05, 21344);
```

```
INSERT INTO PRODUCT   VALUES ('BRT-345','Titanium drill bit','18-Oct-13', 75, 10, 4.50, 0.06, NULL);
```

```
INSERT INTO PRODUCT(P_CODE, P_DESCRIPT) VALUES ('BRT-345','Titanium drill bit');
```

Data Manipulation Commands

SELECT: Command to list the contents

- Syntax - SELECT *columnlist* FROM *tablename*;
- **Wildcard character**(*): Substitute for other characters/command

UPDATE: Command to modify data

- Syntax - UPDATE *tablename* SET *columnname* = expression [, *columnname* = *expression*] [WHERE *conditionlist*];

```
SELECT * FROM
PRODUCT;

SELECT    P_CODE, P_DESCRIPT, P_INDATE, P_QOH, P_MIN, P_PRICE, P_DISCOUNT,
          V_CODE
FROM      PRODUCT;

  UPDATE    PRODUCT
  SET       P_INDATE =
            '18-JAN-2014'
  WHERE     P_CODE = '13-Q2/P2';

  UPDATE    PRODUCT
  SET       P_INDATE = '18-JAN-2014', P_PRICE = 17.99, P_MIN
            = 10
  WHERE     P_CODE = '13-Q2/P2';
```

Data Manipulation Commands

WHERE condition

- Specifies the rows to be selected

ROLLBACK: Command to restore the database

- Syntax - ROLLBACK;
- Undoes the changes since last COMMIT command

DELETE: Command to delete

- Syntax - DELETE FROM *tablename*
- [WHERE *conditionlist*];

```
DELETE FROM     PRODUCT
WHERE           P_CODE = 'BRT-345';
```

Inserting Table Rows with a SELECT Subquery

- Syntax
  - INSERT INTO *tablename* SELECT *columnlist* FROM *tablename*
- Used to add multiple rows using another table as source
- SELECT command - Acts as a subquery and is executed first
  - **Subquery**: Query embedded/nested inside another query

INSERT INTO *tablename*

SELECT *columnlist*

FROM *tablename*;

Selecting Rows Using Conditional Restrictions

- Can select partial table contents by placing restrictions on rows to be included
- Syntax enables to specify which rows to select:
  - SELECT *columnlist*
  - FROM *tablelist*
  - [WHERE *conditionlist*];
- WHERE clause adds conditional restrictions to the SELECT statement

```
SELECT    columnlist
FROM      tablelist
[WHERE    conditionlist
];
```

Table 7.6 - Comparison Operators

- Adds conditional restrictions on selected character attributes and dates

TABLE 7.6

COMPARISON OPERATORS

SYMBOL	MEANING
=	Equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
<> or !=	Not equal to

Arithmetic Operators

- **The Rule of Precedence**: Establish the order in which computations are completed
- Performed in this order:
  - Operations within parentheses
  - Power operations
  - Multiplications and divisions
  - Additions and subtractions

Table 7.7 - The Arithmetic Operators

TABLE 7.7

THE ARITHMETIC OPERATORS

OPERATOR	DESCRIPTION
+	Add
-	Subtract
*	Multiply
/	Divide
^	Raise to the power of (some applications use ** instead of ^)

Logical Operators: AND, OR and NOT

- **OR** and **AND**: Used to link multiple conditional expressions in a WHERE or HAVING clause
  - **OR** requires only one of the conditional expressions to be true
  - **AND** requires all of the conditional expressions to be true
- **NOT** is used to negate the result of a conditional expression
- **Boolean algebra** is dedicated to the use to logical operations

Comparison Operators: Computed Columns and Column Aliases

- SQL accepts any valid expressions/formulas in the computed columns
- **Alias**: Alternate name given to a column or table in any SQL statement to improve the readability
- Computed column, an alias, and date arithmetic can be used in a single query

Figure 7.12 - Selected PRODUCT Table Attributes: The Logical OR

```
SELECT P_DESCRIPT, P_INDATE, P_PRICE, V_CODE
FROM PRODUCT
WHERE V_CODE = 21344 OR V_CODE = 24288;
```

FIGURE 7.12 SELECTED PRODUCT TABLE ATTRIBUTES: THE LOGICAL OR

P_DESCRIPT	P_INDATE	P_PRICE	V_CODE
7.25-in. pwr. saw blade	13-Dec-15	14.99	21344
9.00-in. pwr. saw blade	13-Nov-15	17.49	21344
B&D jigsaw, 12-in. blade	30-Dec-15	109.92	24288
B&D jigsaw, 8-in. blade	24-Dec-15	99.87	24288
Rat-tail file, 1/8-in. fine	15-Dec-15	4.99	21344
Hicut chain saw, 16 in.	07-Feb-16	256.99	24288

Figure 7.13 - Selected PRODUCT Table Attributes: The Logical AND

```
SELECT P_DESCRIPT, P_INDATE, P_PRICE, V_CODE
FROM PRODUCT
WHERE P_PRICE < 50 AND P_INDATE > '15-JAN-2016';
```

FIGURE 7.13 SELECTED PRODUCT TABLE ATTRIBUTES: THE LOGICAL AND

P_DESCRIPT	P_INDATE	P_PRICE	V_CODE
B&D cordless drill, 1/2-in.	20-Jan-16	38.95	25595
Claw hammer	20-Jan-16	9.95	21225
PVC pipe, 3.5-in., 8-ft	20-Feb-16	5.87	
1.25-in. metal screw, 25	01-Mar-16	6.99	21225
2.5-in. wd. screw, 50	24-Feb-16	8.45	21231

Figure 7.14 - Selected PRODUCT Table Attributes: The Logical AND and OR

```
SELECT P_DESCRIPT, P_INDATE, P_PRICE, V_CODE
FROM PRODUCT
WHERE (P_PRICE < 50 AND P_INDATE > '15-JAN-2016') OR V_CODE = 24288;
```

FIGURE 7.14 SELECTED PRODUCT TABLE ATTRIBUTES: THE LOGICAL AND AND OR

P_DESCRIPT	P_INDATE	P_PRICE	V_CODE
B&D jigsaw, 12-in. blade	30-Dec-15	109.92	24288
B&D jigsaw, 8-in. blade	24-Dec-15	99.87	24288
B&D cordless drill, 1/2-in.	20-Jan-16	38.95	25595
Claw hammer	20-Jan-16	9.95	21225
Hicut chain saw, 16 in.	07-Feb-16	256.99	24288
PVC pipe, 3.5-in., 8-ft	20-Feb-16	5.87	
1.25-in. metal screw, 25	01-Mar-16	6.99	21225
2.5-in. wd. screw, 50	24-Feb-16	8.45	21231

Special Operators

BETWEEN

- Checks whether attribute value is within a range

IS NULL

- Checks whether attribute value is null

LIKE

- Checks whether attribute value matches given string pattern

IN

- Checks whether attribute value matches any value within a value list

EXISTS

- Checks if subquery returns any rows

```
SELECT      *
FROM        PRODUCT
WHERE       P_PRICE BETWEEN 50.00 AND 100.00;
```

```
SELECT      P_CODE, P_DESCRIPT, V_CODE
FROM        PRODUCT
WHERE       V_CODE IS NULL;
```

- % means any and all *following* or *preceding* characters are eligible. For example:

  'J%' includes Johnson, Jones, Jernigan, July, and J-231Q.

  'Jo%' includes Johnson and Jones.

  '%n' includes Johnson and Jernigan.

- _means any *one* character may be substituted for the underscore. For example:

  '_23-456-6789' includes 123-456-6789, 223-456-6789, and 323-456-6789.

  '_23-_56-678_' includes 123-156-6781, 123-256-6782, and 823-956-6788.

  '_o_es' includes Jones, Cones, Cokes, totes, and roles.

```
SELECT    V_NAME, V_CONTACT, V_AREACODE, V_PHONE
FROM      VENDOR
WHERE     V_CONTACT LIKE 'Smith%';
```

```
SELECT      *
FROM        PRODUCT
WHERE       V_CODE = 21344
OR          V_CODE = 24288;
```

```
SELECT      *
FROM        PRODUCT
WHERE       V_CODE IN (21344, 24288);
```

```
SELECT      V_CODE, V_NAME
FROM        VENDOR
WHERE       V_CODE IN (SELECT V_CODE FROM PRODUCT);
```

```
SELECT *
FROM suppliers
WHERE EXISTS (SELECT *
              FROM orders
              WHERE suppliers.supplier_id = orders.supplier_id);
```

This SQL EXISTS condition example will return all records from the suppliers table where there is at least one record in the orders table with the same supplier_id.

```
SELECT *
FROM suppliers
WHERE NOT EXISTS (SELECT *
                  FROM orders
                  WHERE suppliers.supplier_id = orders.supplier_id);
```

This SQL EXISTS example will return all records from the suppliers table where there are **no** records in the *orders* table for the given supplier_id.

Additional Data Definition Commands

- **ALTER TABLE** command: To make changes in the table structure
- Keywords use with the command
  - ADD - Adds a column
  - MODIFY - Changes column characteristics
  - DROP - Deletes a column
- Used to:
  - Add table constraints
  - Remove table constraints

Changing a Column's Data Type and Data Characteristics

- ALTER used to change data type and characteristics
  - Some RDBMSs do not permit changes to data types unless column is empty
  - Changes in characteristics are permitted if they do not alter the existing data type
- Syntax:
  - Data Type: ALTER TABLE *tablename* MODIFY *(columnname(datatype))*;
  - Data Characteristic: ALTER TABLE *tablename* MODIFY *(columnname(characteristic))*;

Adding and Dropping Columns

- Adding a column
  - Use ALTER and ADD
  - Do not include the NOT NULL clause for new column
- Dropping a column
  - Use ALTER and DROP
  - Some RDBMSs impose restrictions on the deletion of an attribute

Advanced Data Updates

- UPDATE command updates only data in existing rows
- If a relationship is established between entries and existing columns, the relationship can assign values to appropriate slots
- Arithmetic operators are useful in data updates
- In Oracle, ROLLBACK command undoes changes made by last two UPDATE statements

```
UPDATE      PRODUCT
SET         P_SALECODE = '2'
WHERE       P_CODE = '1546-QQ2';

UPDATE      PRODUCT
SET         P_SALECODE = '1'
WHERE       P_CODE IN ('2232/QWE', '2232/QTY');

UPDATE      PRODUCT
SET         P_SALECODE = '1'
WHERE       P_INDATE >= '16-Jan-2014' AND P_INDATE <='10-Feb-2014';

UPDATE      PRODUCT
SET         P_PRICE = P_PRICE * 1.10
WHERE       P_PRICE < 50.00;
```

Copying Parts of Tables

- SQL permits copying contents of selected table columns
  - Data need not be reentered manually into newly created table(s)
- Table structure is created
- Rows are added to new table using rows from another table

```
CREATE TABLE PART(
PART_CODE                    CHAR(8),
PART_DESCRIPT                CHAR(35),
PART_PRICE                   DECIMAL(8,2),
V_CODE                       INTEGER,
PRIMARY KEY (PART_CODE));

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

```
CREATE TABLE PART AS
SELECT      P_CODE AS PART_CODE, P_DESCRIPT AS
            PART_DESCRIPT, P_PRICE AS PART_ PRICE,
            V_CODE
FROM        PRODUCT;
```

Adding Primary and Foreign Key Designations

- A created new table based on another table does not include old table's integrity rule (no primary key)
- Can re-establish integrity rules using **ALTER** command
- Use **ALTER TABLE** command to ADD primary and foreign keys
  - Composite primary keys and multiple foreign keys can be designated in a single SQL command

```
ALTER TABLE    PART
    ADD        PRIMARY KEY (PART_CODE);
```

```
ALTER TABLE    PART
    ADD        FOREIGN KEY (V_CODE) REFERENCES VENDOR;
```

Deleting a Table from the Database

- **DROP TABLE**: Deletes table from database
  - Syntax - DROP TABLE tablename;
- Can drop a table only if it is not the one side of any relationship
  - RDBMS generates a foreign key integrity violation error message if the table is dropped

Additional SELECT Query Keywords

- Logical operators work well in the query environment
- SQL provides useful functions that:
  - Count
  - Find minimum and maximum values
  - Calculate averages
- SQL allows user to limit queries to entries:
  - Having no duplicates
  - Whose duplicates can be grouped

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]];
- **Cascading order sequence**: Multilevel ordered sequence
  - Created by listing several attributes after the ORDER BY clause

Listing Unique Values

- **DISTINCT** clause: Produces list of values that are unique
- Syntax - SELECT DISTINCT *columnlist*

  FROM *tablelist*;
- Placement of nulls does not affect list contents
  - In Oracle can place nulls at top of list

```
SELECT    DISTINCT V_CODE
FROM      PRODUCT;
```

V_CODE
21225
21231
21344
23119
24288
25595

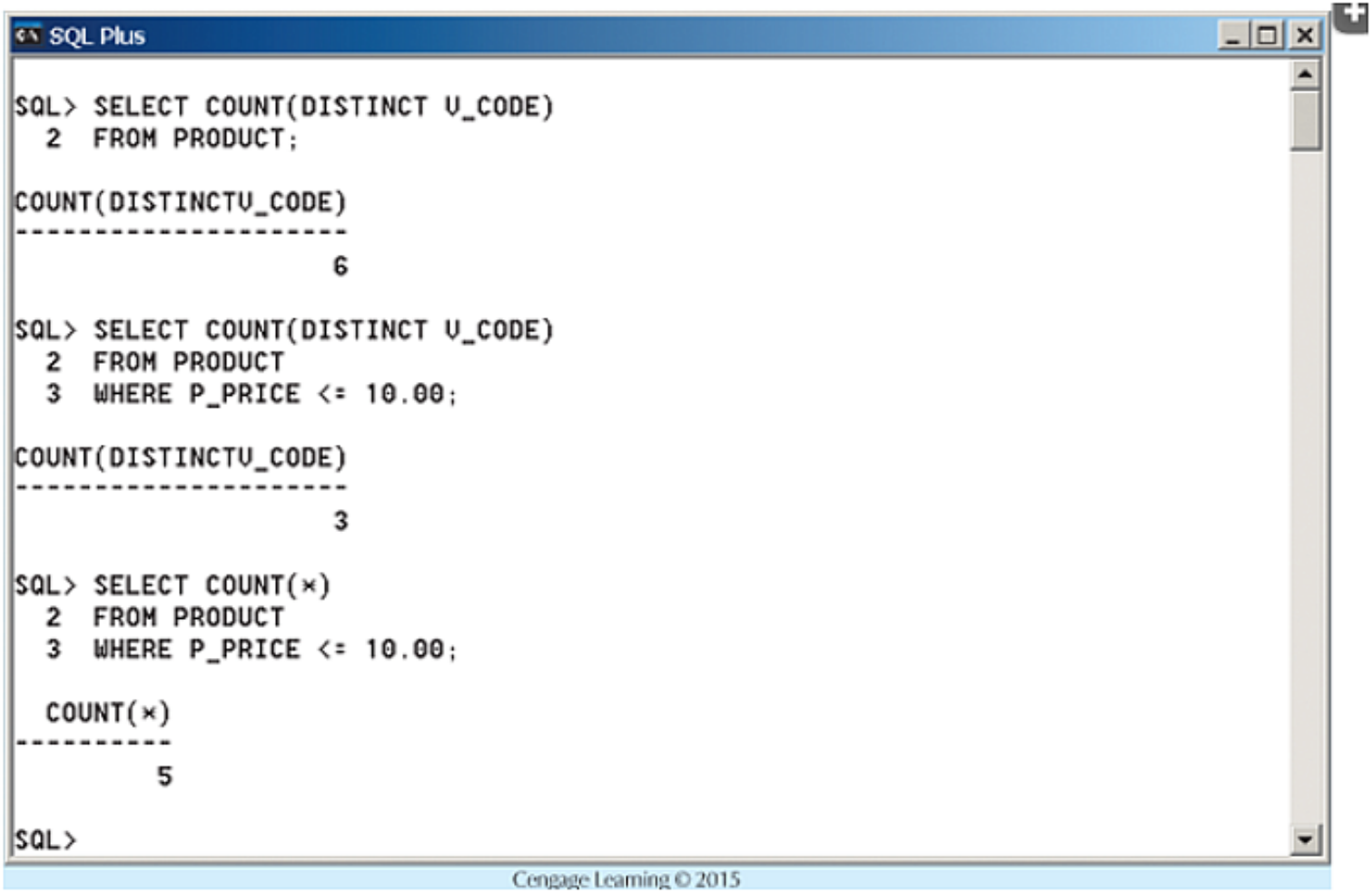
Cengage Learning © 2015

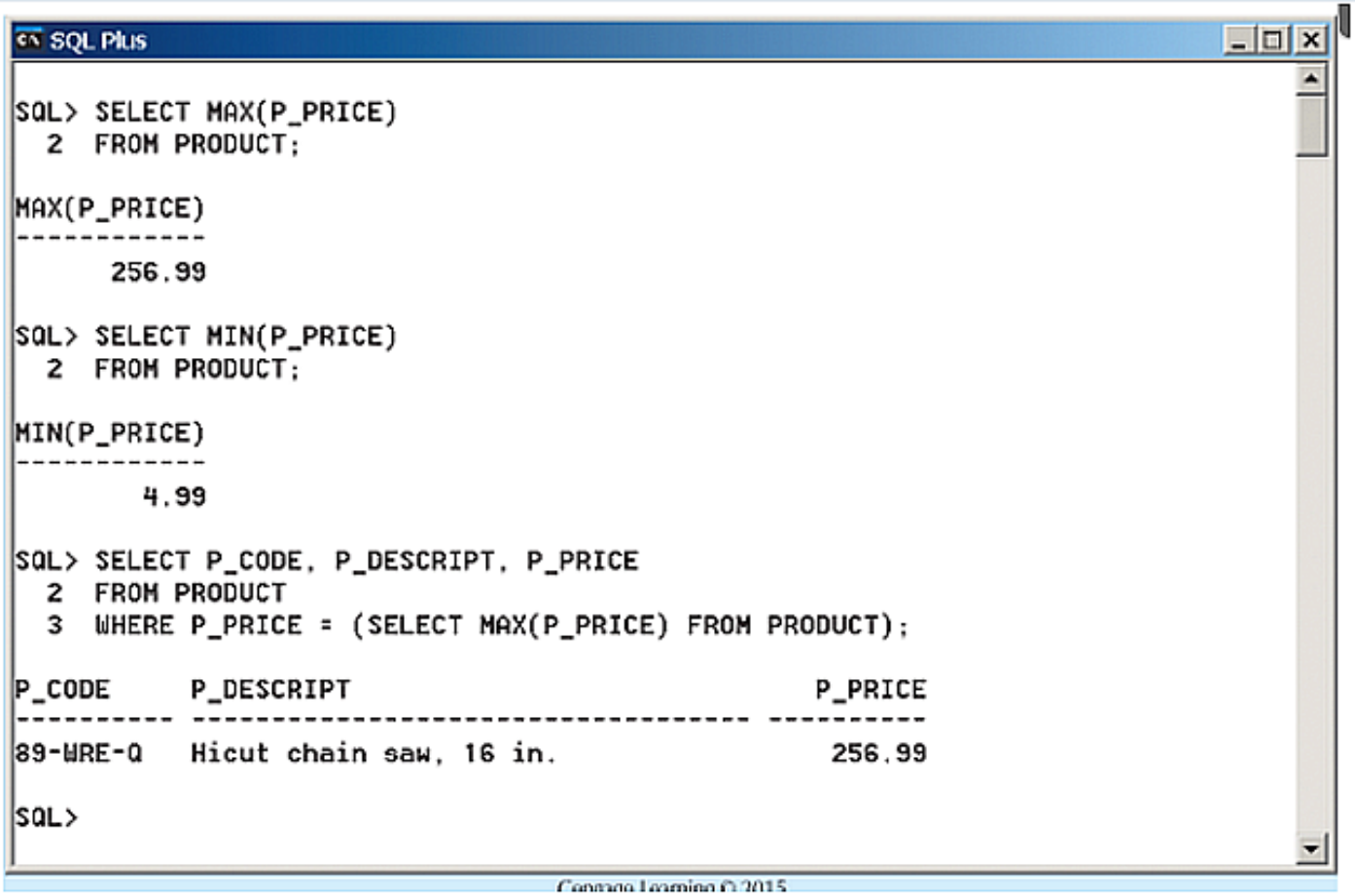
Table 7.8 - Some Basic SQL Aggregate Functions

TABLE 7.8

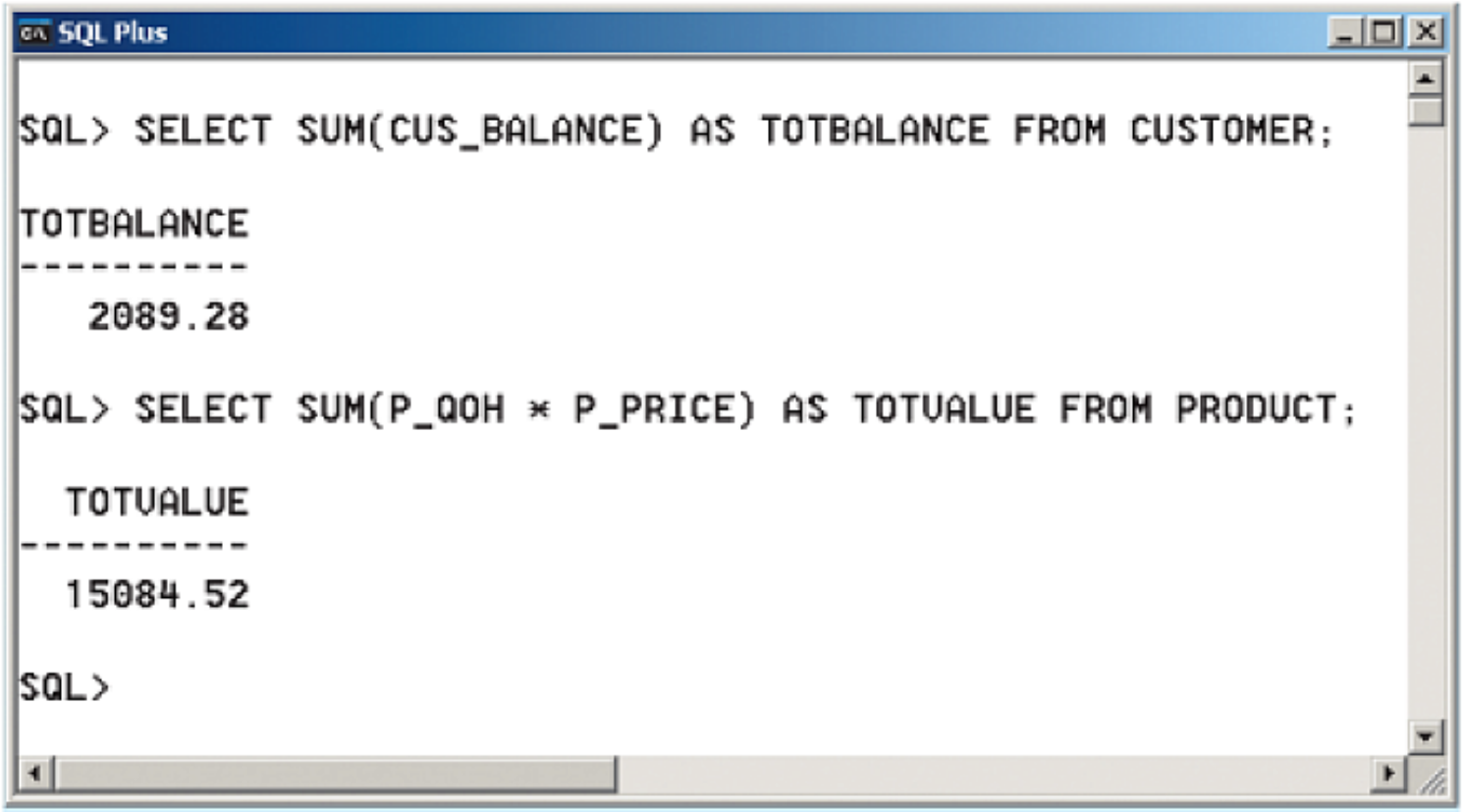
SOME BASIC SQL AGGREGATE FUNCTIONS

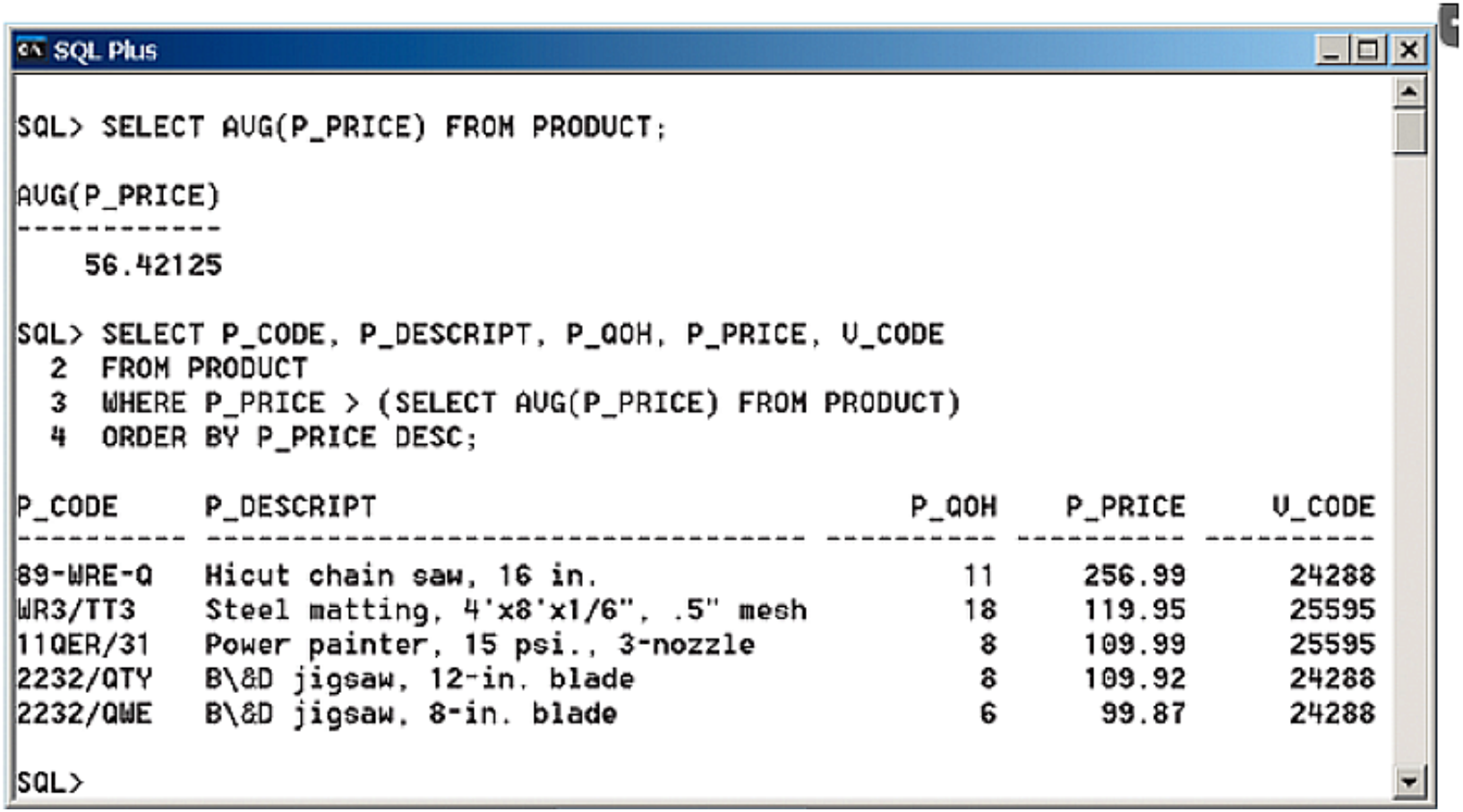
FUNCTION	OUTPUT
COUNT	The number of rows containing non-null values
MIN	The minimum attribute value encountered in a given column
MAX	The maximum attribute value encountered in a given column
SUM	The sum of all values for a given column
AVG	The arithmetic mean (average) for a specified column






```
SELECT *
FROM PRODUCT
WHERE P_QOH*P_PRICE = (SELECT MAX(P_QOH*P_PRICE) FROM PRODUCT);
```





Grouping Data

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

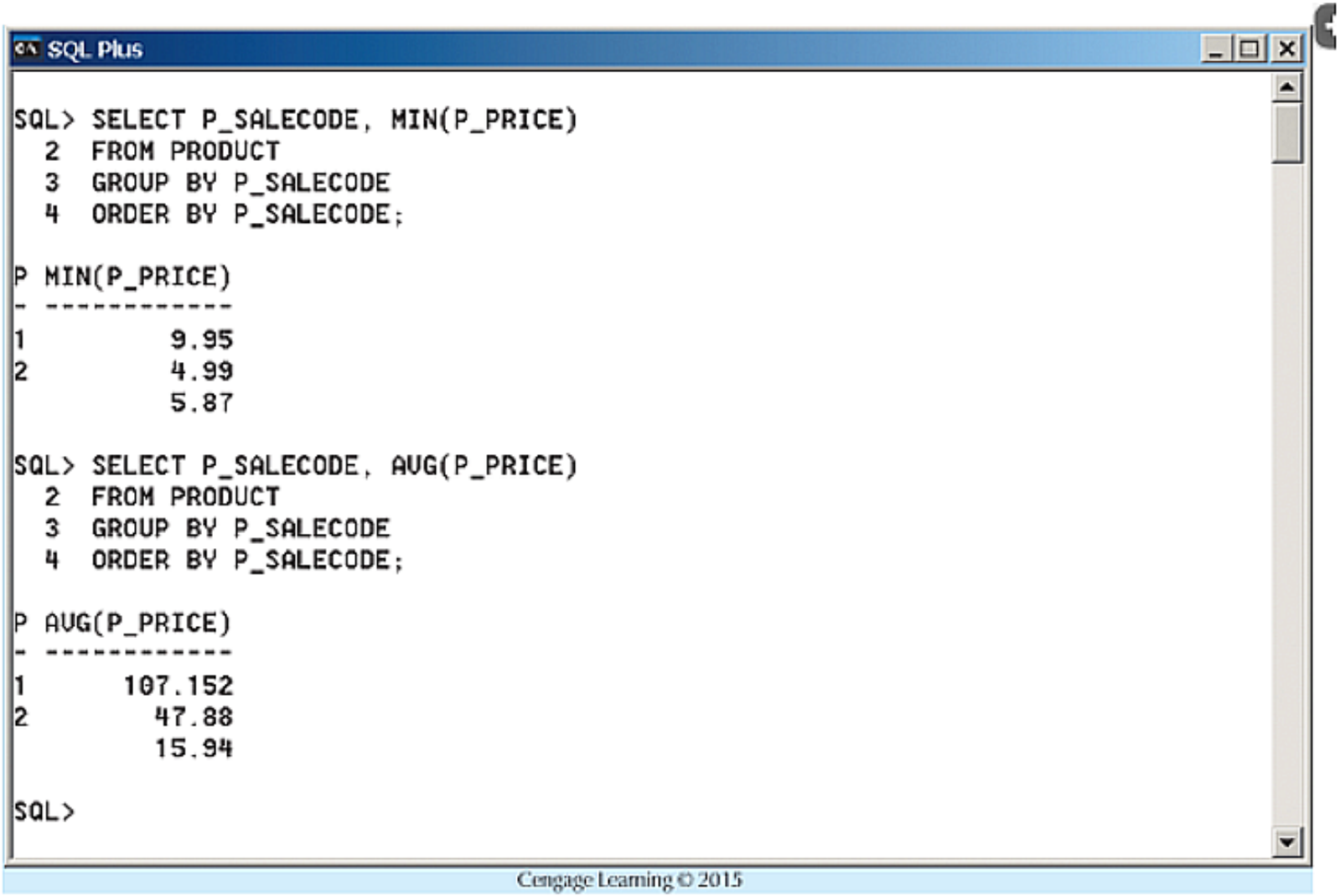
  FROM *tablelist*

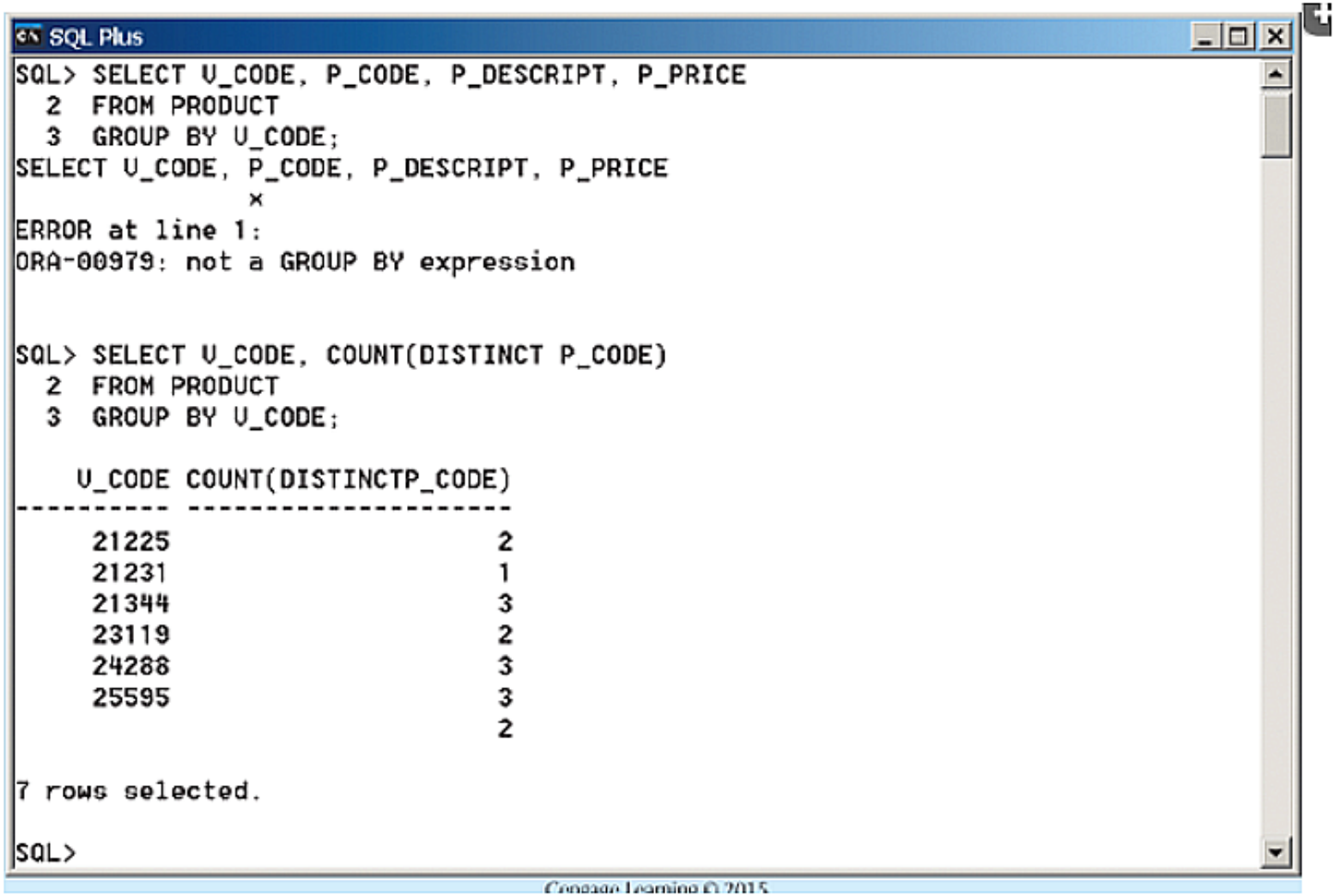
  [WHERE *conditionlist*]

  [GROUP BY *columnlist*]

  [HAVING *conditionlist*]

  [ORDER BY *columnlist* [ASC | DESC]];





SK
SUPER KING
MARKETS
SUPER KING

Join "Royal Rewards" today
and Start Earning Points!

2716 N. San Fernando Rd.
Los Angeles, CA. 90065
(323) 225-0044
Store Hours 7 am - 9 pm
www.SuperKingMarkets.com

Purchase $ 8.74
PIN Used
Debit Card #SXXXXXXXXXXXX4428
Auth # 568771 Payment from primary
Lane # 06 Cashier # 766
01/29/16 09:34 Ref/Seq # 065808
Mrch=912871 Term=001 IC=DC
EPS Sequence # 065808

DELI
SHAMROCK 2% STRAWBER 1.89 *
GROCERY
DORITOS SPICY NACHO
1 @ 2 FOR 5.00 2.50 *
MIXED NUTS
FRIED RFD SKIN PEANU
0.66 lb @ 1.99/ lb 1.31 *
PRODUCE
BLACKBERRIES BERRY L V 1.99 *
CACAHUATE QUEMADO
0.53 lb @ 1.99/ lb 1.05 *

BALANCE DUE 8.74
Debit Card 8.74
Auth Code = 568771
CHANGE 0.00
Total number of items sold = 5

CASHIER NAME: JULIA P.
STORE:00002 REGISTER:006 CASHIER:0766
TICKET#:4826 29JAN2016 9:34:32
*ALL RETURNS AND EXCHANGES CAN ONLY BE ACCEPTED WITHIN 2 DAYS FROM THE DATE OF PURCHASE AND MUST BE WITH RECEIPT.
*RETURNS OR EXCHANGES OF NON-FOOD ITEMS SHALL BE IN ORIGINAL PACKAGE *NO CASH REFUND ON ITEMS PAID BY CREDIT CARD
*NO RETURNS ON BABYFOOD AND FORMULA
ALL SALES ARE FINAL ON ALL ALCOHOLIC BEVERAGES & TOBACCO
LA County State Tax Rate of 9.00%

HAVING Clause

- Extension of GROUP BY feature
- Applied to output of GROUP BY operation
- Used in conjunction with GROUP BY clause in second SQL command set
- Similar to WHERE clause in SELECT statement

Joining Database Tables

- Performed when data are retrieved from more than one table at a time
  - Equality comparison between foreign key and primary key of related tables
- Tables are joined by listing tables in FROM clause of SELECT statement
  - DBMS creates Cartesian product of every table in the FROM clause

Joining Tables With an Alias and Recursive Joins

- Alias identifies source table from which data are taken
  - Any legal table name can be used as alias
  - Add alias after table name in **FROM** clause
- **Recursive query**: Table is joined to itself using alias
  - Use aliases to differentiate the table from itself

```
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 Systems	Orton	904	456-0092
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


```
ORDER BY    PRODUCT.P_PRICE;
```

```
SELECT      CUS_LNAME, INVOICE.INV_NUMBER, INV_DATE, P_DESCRIPT
FROM        CUSTOMER, INVOICE, LINE, PRODUCT
WHERE       CUSTOMER.CUS_CODE = INVOICE.CUS_CODE
AND         INVOICE.INV_NUMBER = LINE.INV_NUMBER
AND         LINE.P_CODE = PRODUCT.P_CODE
AND         CUSTOMER.CUS_CODE = 10014
ORDER BY    INV_NUMBER;
```

Joining n tables will need (n-1) join conditions.

Recursive Joins

- **Recursive query**: Table is joined to itself using alias
- Use aliases to differentiate the table from itself

EMP_NUM	EMP_TITLE	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_DOB	EMP_HIRE_DATE	EMP_AREACODE	EMP_PHONE	EMP_MGR
100	Mr.	Kolmycz	George	D	15-Jun-42	15-Mar-85	615	324-5456	
101	Ms.	Lewis	Rhonda	G	19-Mar-65	25-Apr-86	615	324-4472	100
102	Mr.	Vandam	Rhett		14-Nov-58	20-Dec-90	901	675-8993	100
103	Ms.	Jones	Anne	M	16-Oct-74	28-Aug-94	615	898-3456	100
104	Mr.	Lange	John	P	08-Nov-71	20-Oct-94	901	504-4430	105
105	Mr.	Williams	Robert	D	14-Mar-75	08-Nov-98	615	890-3220	
106	Mrs.	Smith	Jeanine	K	12-Feb-68	05-Jan-89	615	324-7883	105
107	Mr.	Diante	Jorge	D	21-Aug-74	02-Jul-94	615	890-4567	105
108	Mr.	Wiesenbach	Paul	R	14-Feb-66	18-Nov-92	615	897-4358	
109	Mr.	Smith	George	K	18-Jun-61	14-Apr-89	901	504-3339	108
110	Mrs.	Genkazi	Leighla	W	19-May-70	01-Dec-90	901	569-0093	108
111	Mr.	Washington	Rupert	E	03-Jan-66	21-Jun-93	615	890-4925	105
112	Mr.	Johnson	Edward	E	14-May-61	01-Dec-83	615	896-4387	100
113	Ms.	Smythe	Melanie	P	15-Sep-70	11-May-99	615	324-9006	105
114	Ms.	Brandon	Marie	G	02-Nov-55	15-Nov-79	901	882-0845	108
115	Mrs.	Saranda	Hermine	R	25-Jul-72	23-Apr-93	615	324-5505	105
116	Mr.	Smith	George	A	08-Nov-65	10-Dec-88	615	890-2984	108

```
SELECT      E.EMP_NUM, E.EMP_LNAME, E.EMP_MGR, M.EMP_LNAME
FROM        EMP E, EMP M
WHERE       E.EMP_MGR=M.EMP_NUM
ORDER BY    E.EMP_MGR;
```

EMP_NUM	E.EMP_LNAME	EMP_MGR	M.EMP_LNAME
112	Johnson	100	Kolmycz
103	Jones	100	Kolmycz
102	Vandam	100	Kolmycz
101	Lewis	100	Kolmycz
115	Saranda	105	Williams
113	Smythe	105	Williams
111	Washington	105	Williams
107	Diante	105	Williams
106	Smith	105	Williams
104	Lange	105	Williams
116	Smith	108	Wiesenbach
114	Brandon	108	Wiesenbach
110	Genkazi	108	Wiesenbach
109	Smith	108	Wiesenbach

Cengage Learning © 2015