Unit 7:
Introduction to Structured Query Language (SQL)

RECAP

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 Commands (1 of 2)

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)

Table 7.1 – SQL Data Definition Commands (2 of 2)

COMMAND OR OPTION	DESCRIPTION
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 (1 of 2)

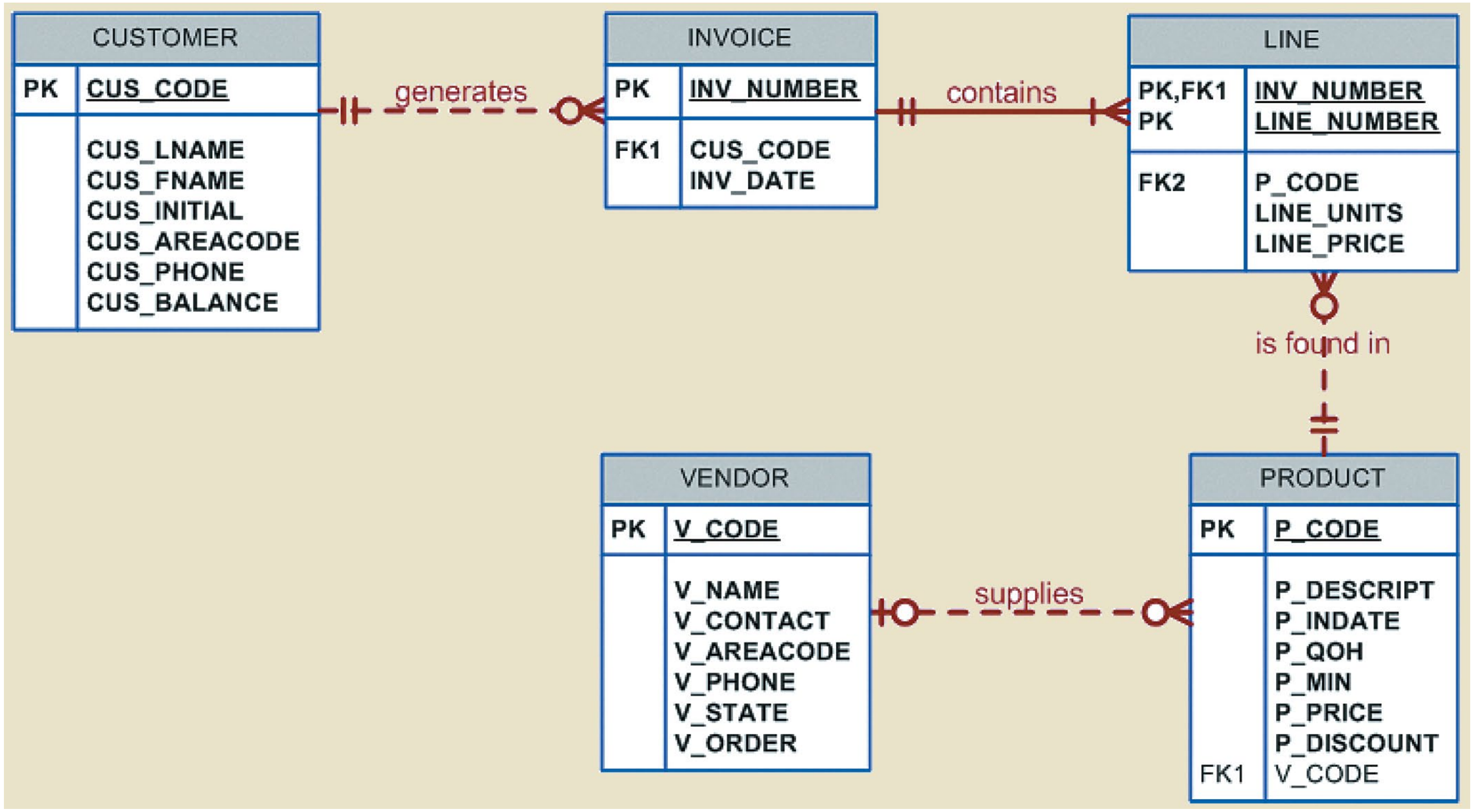
COMMAND OR OPTION	**DESCRIPTION**
INSERT	Inserts row(s) into a table
SELECT	Selects attributes from rows in one or more tables or views
WHERE	Restricts the selection of rows based on a conditional expression
GROUP BY	Groups the selected rows based on one or more attributes
HAVING	Restricts the selection 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

CENGAGE

Table 7.2 – SQL Data Manipulation Commands (2 of 2)

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 (1 of 2)

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.

Table 7.4 - Common SQL Data Types (2 of 2)

DATA TYPE	FORMAT	COMMENTS
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.

Full list of MySQL data types - https://dev.mysql.com/doc/refman/8.0/en/data-types.html

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.

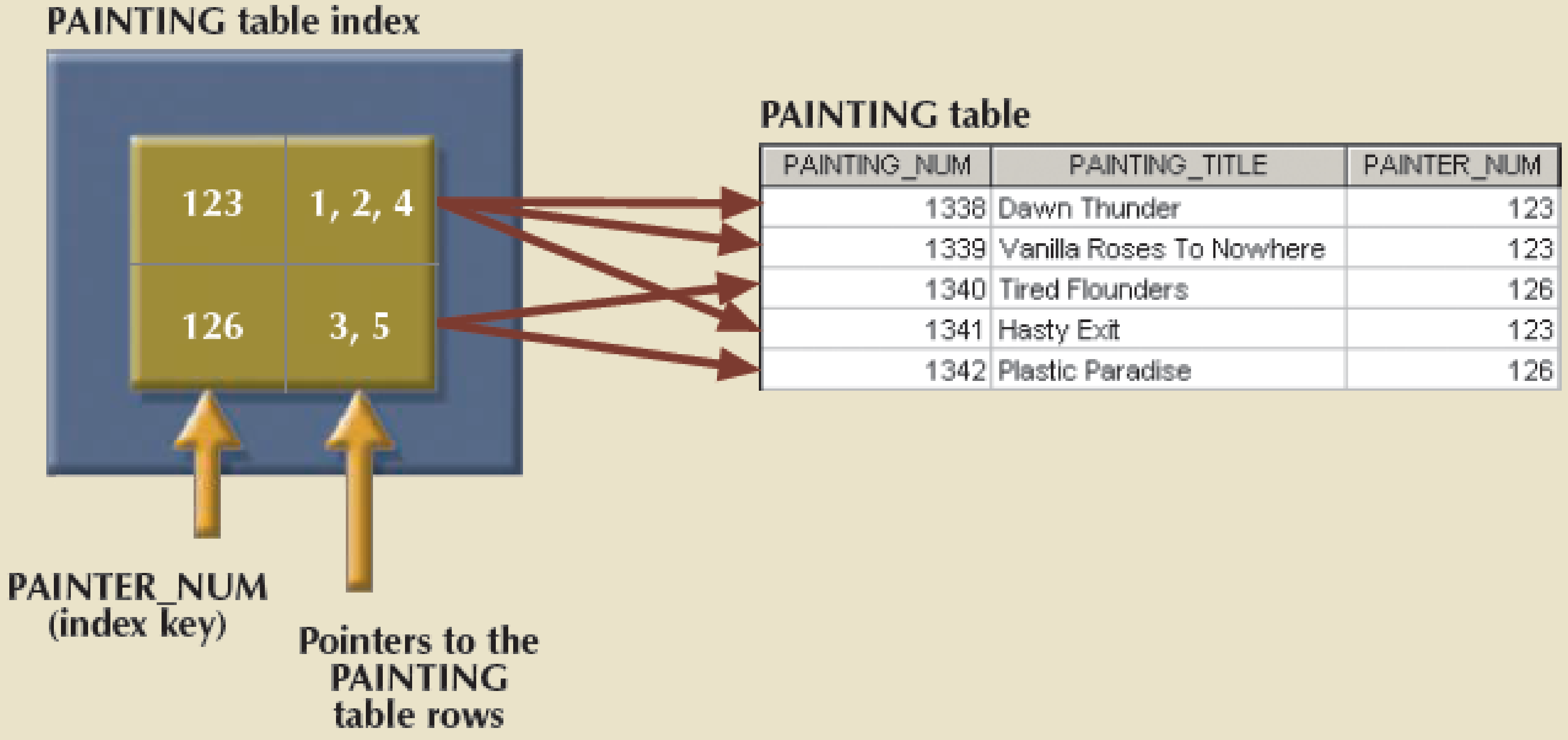
SQL Constraints

- **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

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

FIGURE 3.31 COMPONENTS OF AN INDEX



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

CENGAGE

Data Manipulation Commands (1 of 3)

- 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

Data Manipulation Commands (2 of 3)

- 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*];

Data Manipulation Commands (3 of 3)

- 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*];

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

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

Table 7.6 - Comparison Operators

- Adds conditional restrictions on selected character attributes and dates

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

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

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

CENGAGE

Table 7.7 - The Arithmetic Operators

OPERATOR	DESCRIPTION
+	Add
−	Subtract
*	Multiply
/	Divide
^	Raise to the power of (some application 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

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

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

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

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

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

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

Grouping Data

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

```
SELECT columnlist
FROM tablelist
[WHERE conditionlist]
[GROUP BY columnlist]
[HAVING conditionlist]
[ORDER BY  columnlist [ASC | DESC]];
```

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