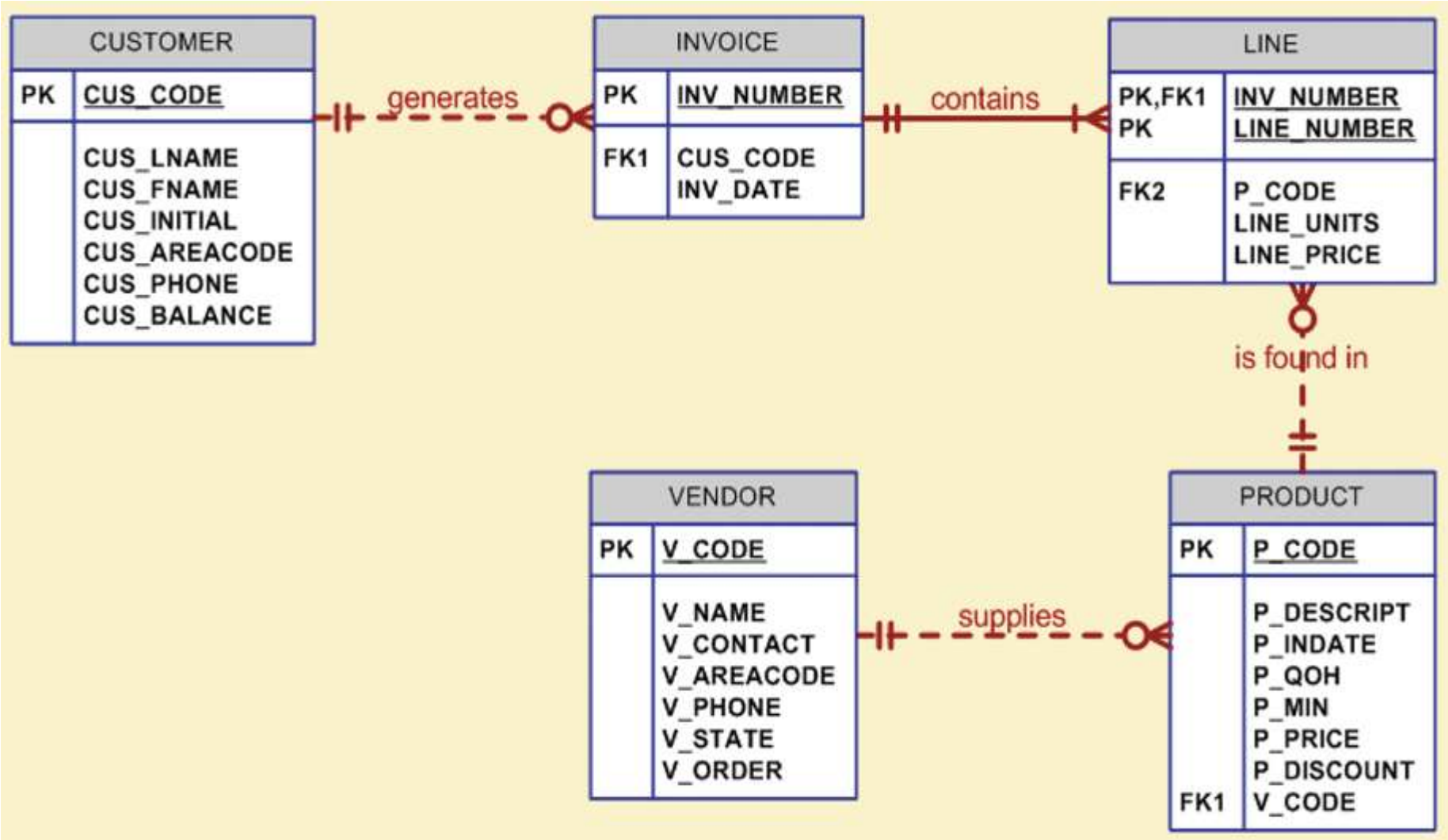
Structured Query Language (SQL) - Part 1

Lecture 6

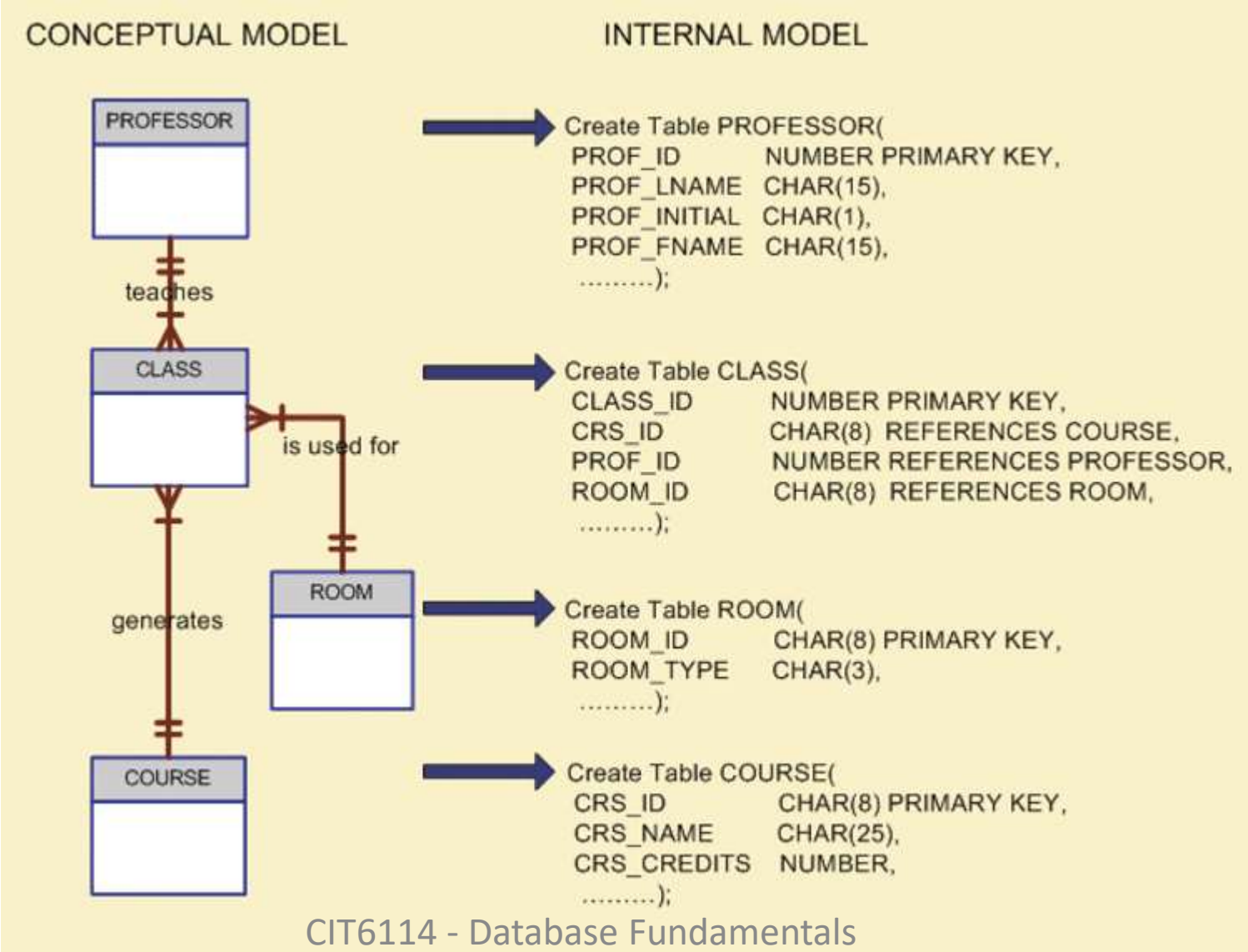
Learning Outcomes

- In this chapter, students will learn:
  - The basic commands and functions of SQL
  - How to use SQL for data administration (to create tables)
  - 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

The Database Conceptual Model (Recap!)



Maps Conceptual Model to the DBMS



Introduction to SQL

- Structured Query Language (SQL) is a non-procedural language (command **what** to do and *not* how)
- SQL provides statements that help work with the database
- SQL functions fit into two broad categories:
  - **Data definition language (DDL):** ***create*** database, tables, indexes, views, trigger, procedure
  - **Data manipulation language (DML):** ***insert***, ***update***, ***delete***, ***retrieve*** data
- American National Standards Institute (ANSI) prescribes a standard SQL

Creating the Database

- Syntax:
  - **CREATE DATABASE** <database_name>;

- Example:
  - **CREATE DATABASE** Tiny_College;

Creating Tables

- A database consists of tables (entities)
- Syntax:
  - **CREATE TABLE** <table name>
    (<attribute1 name> <attribute1 characteristics>,
    <attribute2 name> <attribute2 characteristics>,
    primary key <attribute name>,
    foreign key <attribute name>);

Table name

```
CREATE TABLE Faculty
(
    fac_id char(3)  primary key not null,
    fac_name char(50)
);
```

Data type

Attribute name

- Table names and column names must:
  - Begin with a letter
  - Be 1–30 characters long
  - Contain only A–Z, a–z, 0–9, _, $, and #
  - Not duplicate the name of another object owned by the same user
  - Not be a reserved word

Data Types

Three groups of Data Types:

- **Character - store character (alphanumeric) data in strings**
- **Numeric**
- **Date**

Data Type	Format
Numeric	INTEGER BIGINT SMALLINT DECIMAL(L,D)
Character	CHAR(L) VARCHAR(L)
Date	DATE

Data Types

- **Numeric**
  - E.g., **INTEGER**

-9878149

150581

    - Integers are (whole) counting numbers
    - CANNOT use to store decimal places
    - Size: $[-2^{31}, 2^{31}-1]$ = [-2147483648 – 2147483647]

Data Types

- **Numeric**
  - E.g., **BIGINT**

    123456987034

    -126992389155

    - Up to <u>nineteen</u> digits long
    - Size: $[-2^{63}, 2^{63}-1]$
  - E.g., **SMALLINT**

    3456

    -19531

    - Like INTEGER, but limited to integer values up to <u>five</u> digits
    - Size: [-32768, 65535]

Data Types

- **Numeric**
  - E.g., **DECIMAL(5,2)**

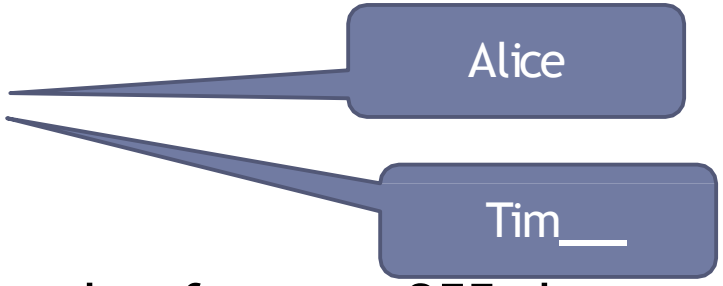
1623.99 X

88.45

    - Storage length is a minimum specification (i.e., smaller lengths are acceptable but greater ones are <u>not</u>)

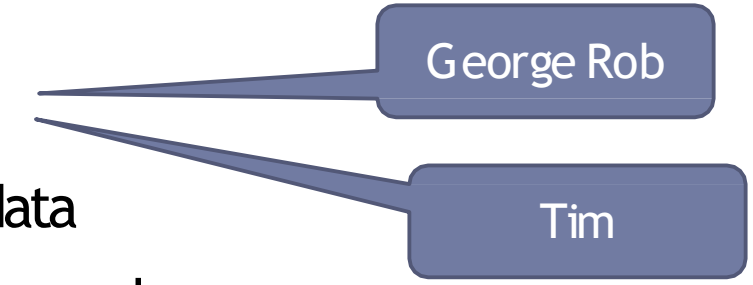
Data Types

- **Character**
  - E.g., **CHAR(5)**



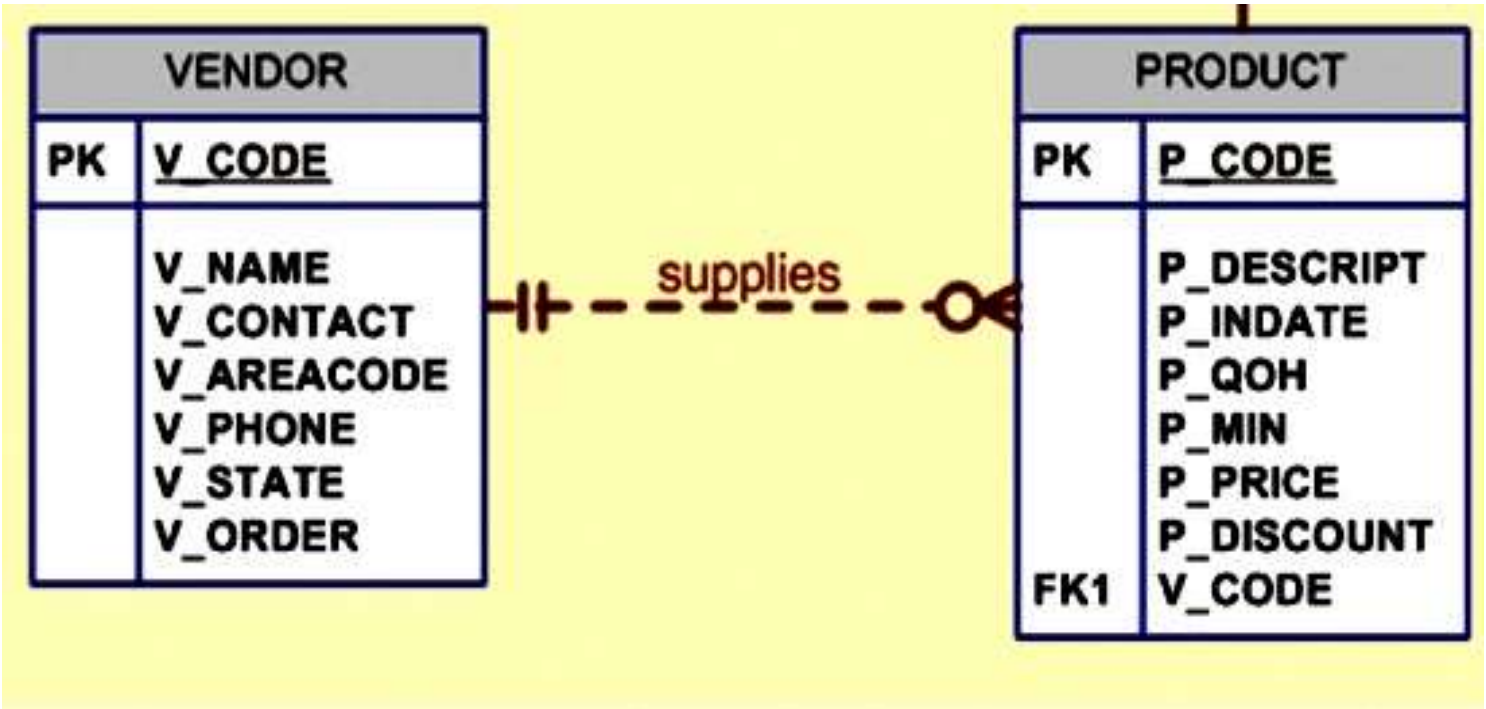
    - Fixed-length character data for up to 255 characters.
    - If you store strings that are not as long as the CHAR parameter value, the remaining spaces are left unused

  - E.g., **VARCHAR(10)**

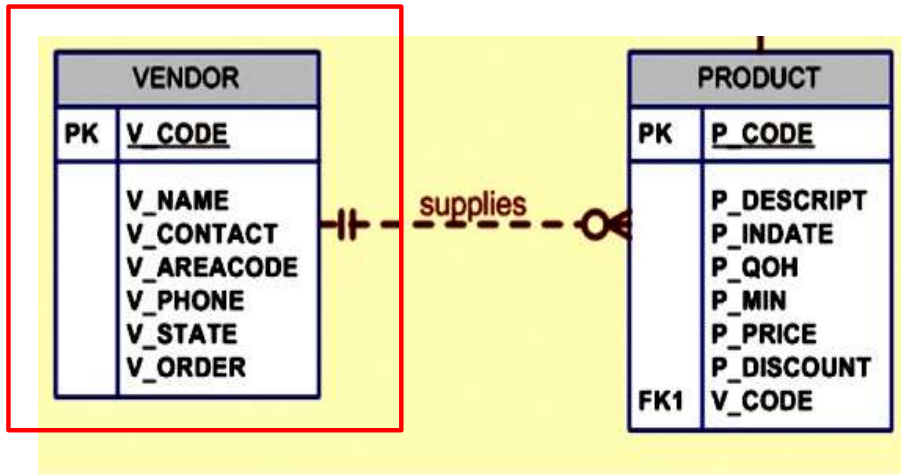


    - Variable-length character data
    - VARCHAR will not leave unused spaces
    - Oracle can use VARCHAR or VARCHAR2

Creating Table Structure: Example



Creating Table Structure: Example

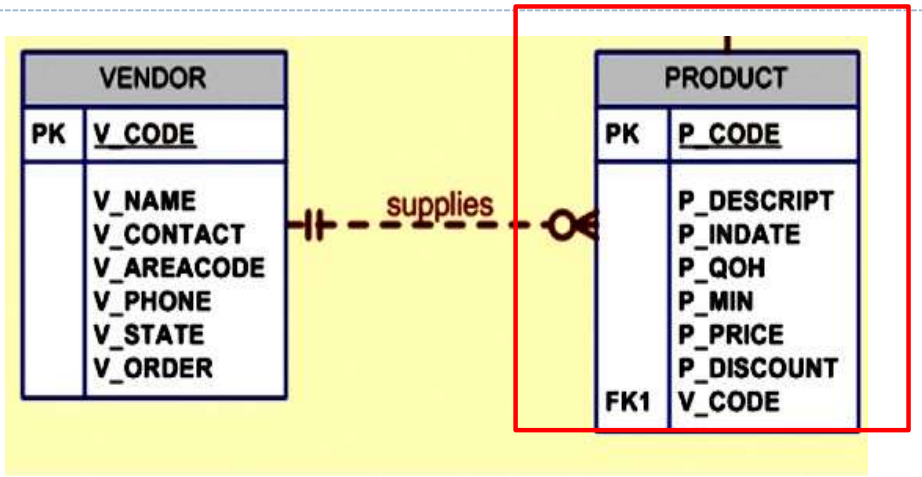


- **CREATE TABLE** VENDOR (

```
    v_code        varchar(5)  PRIMARY KEY NOT NULL,
    v_name        varchar(20),
    v_contact     varchar(15),
    v_areacode    integer,
    v_phone       varchar(10),
    v_state       char(2),
    v_order       char(1) );
```

Entity integrity (recap!):
- Primary key cannot contain NULL value

Creating Table Structure: Example



- **CREATE TABLE** PRODUCT (

```
p_code        varchar(10)  PRIMARY KEY NOT NULL,
p_descript    varchar(50),
p_indate      date,
p_onhand      integer,
p_min         integer,
p_price       decimal(5,2),
p_discount    decimal (3,2),
v_code        varchar(5),
FOREIGN KEY (v_code) REFERENCES VENDOR ON DELETE RESTRICT);
```

Referential integrity rules – the value must match the value in the parent table or it contains null

Referential Integrity Rules

- **DELETE rules**
  - **RESTRICT** – cannot delete a record if records are found in both parent and dependent table
  - **NO ACTION** – cannot delete a record if records are found in both parent and dependent table.The row in the dependent table MUST be deleted first before deleting the row from the parent table
  - **CASCADE** – deleting row in parent table automatically deletes any related rows in dependent tables
  - **SET NULL** – foreign key is set to null, other columns left unchanged

Referential Integrity - Example

custID	custName
100	Ali
101	Bob

orderNum	orderDate	custID
1	06-11-2014	100
2	06-12-2014	100

```
… FOREIGN KEY (custID) REFERENCES customer ON DELETE RESTRICT;

… FOREIGN KEY (custID) REFERENCES customer ON DELETE CASCADE;

… FOREIGN KEY (custID) REFERENCES customer ON DELETE SET NULL;
```

SQL Constraints

- **NOT NULL** constraint
  - Ensures that column does not accept nulls
- **UNIQUE** constraint
  - Ensures that all values in column are unique
- **DEFAULT** constraint
  - Assigns value to attribute when a new row is added to table
- **CHECK** constraint
  - Validates data when attribute value is entered

SQL Constraints

- **Example:**

```
CREATE TABLE STUDENT (
STU_ID VARCHAR(8) PRIMARY KEY NOT NULL,
STU_Name VARCHAR(20) NOT NULL UNIQUE,
STU_Gender CHAR(1) CHECK (STU_Gender IN ('F','M')),
STU_Country CHAR(8) DEFAULT 'MALAYSIA' );
```

Data Manipulation Commands

- INSERT
- SELECT
- UPDATE
- DELETE

Adding Table Rows

```
CREATE TABLE STUDENT (
STU_ID VARCHAR(8) PRIMARY KEY NOT NULL,
STU_Name VARCHAR(20) NOT NULL UNIQUE,
STU_Gender CHAR(1) CHECK (STU_Gender IN ('F', 'M')),
STU_Country CHAR(8) DEFAULT 'MALAYSIA' );
```

- **INSERT :**
  - Used to enter data into table
- Syntax:
  - **INSERT INTO** <columnname1,columnname2,...,columnnameN>
    VALUES (value1, value2, ... , valueN);
- Example1:
  - **INSERT INTO** STUDENT **VALUES** ('12340', 'John', 'M', 'UK');
  - **INSERT INTO** STUDENT **VALUES** ('56780', 'Anne', 'F', default);

STUDENT:

	STU_ID	STU_NAME	STU_GENDER	STU_COUNTRY
1	56 780	Anne	F	MALAYSIA
2	12 340	John	M	UK

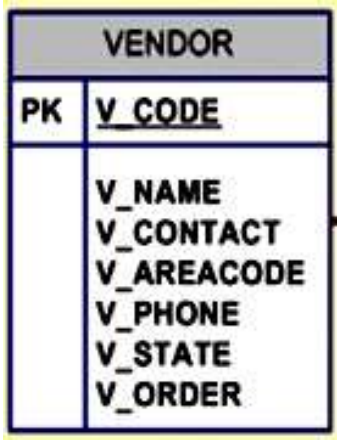
Adding Table Rows (cont'd.)

```
CREATE TABLE VENDOR (
    v_code       varchar(5)  PRIMARY KEY NOT NULL,
    v_name       varchar(20),
    v_contact    varchar(15),
    v_areacode   integer,
    v_phone      varchar(10),
    v_state      char(2),
    v_order      char(1) );
```

- When entering values, notice that:
  - Character and date values are entered between apostrophes (')
  - Numerical entries are **not** enclosed in apostrophes
  - Attribute entries are separated by commas (,)
  - A value is required for each column

```
INSERT INTO VENDOR VALUES
('21226', 'Bryson Inc.', 'Smithson', 615, '223234', 'TN', 'Y');
```

V_Code → '21226'
V_NAME → 'Bryson Inc.'
V_CONTACT → 'Smithson'
V_AREACODE → 615
V_PHONE → '223234'
V_STATE → 'TN'
V_Order → 'Y'



Listing Table Rows

- SELECT
  - Used to list contents of table
- Syntax:
  - **SELECT** <columnlist> **FROM** <tablename>;
- Columnlist represents one or more attributes, separated by commas
  - E.g., **SELECT** p_code, p_descript, p_indate **FROM** product;
  - E.g., **SELECT * FROM** product;

Use * to list all attributes

SELECT STU_Name, STU_ID **FROM** STUDENT;

	STU_NAME	STU_ID
1	Anne	56,780
2	John	12,340

Updating Table Rows

- UPDATE
  - Modify data in a table
- Syntax:
  - **UPDATE** tablename
    SET <columnname = expression> [, columnname = expression]
    [**WHERE** conditionlist];

- Example:
  - **UPDATE** product
    SET p_indate = '2021-07-19'
    WHERE p_code = '13-Q2/P2';

STUDENT

	STU_ID	STU_NAME	STU_GENDER	STU_COUNTRY
1	56 780	Anne	F	MALAYSIA
2	12 340	John	M	UK

```
UPDATE STUDENT
SET STU_Name = 'Hidayah '
WHERE STU_ID = '56780';
```

STUDENT

	STU_ID	STU_NAME	STU_GENDER	STU_COUNTRY
1	56 780	Hidayah	F	MALAYSIA
2	12 340	John	M	UK

Updating Table Rows

- If more than one attribute is to be updated in row, separate corrections with *commas*
- Example:

```
UPDATE product
SET P_INDATE = '12/11/96', P_PRICE = 15.99, P_MIN=10
WHERE P_CODE = '13-Q2/P2';
```

Deleting Table Rows

- DELETE : Deletes a table row
- Syntax:
  - **DELETE FROM** tablename **WHERE** [conditionlist];
- Example:
  - **DELETE FROM** product **WHERE** p_code = '2238/QPD';

DELETE FROM Student **WHERE** STU_ID = '12340';

OR

DELETE FROM Student **WHERE** STU_NAME = 'John';

STUDENT

	STU_ID	STU_NAME	STU_GENDER	STU_COUNTRY
1	56 780	Anne	F	MALAYSIA
2	12 340	John	M	UK

STUDENT

	STU_ID	STU_NAME	STU_GENDER	STU_COUNTRY
1	56 780	Hidayah	F	MALAYSIA

Deleting Table Rows

- WHERE condition is *optional*
- If WHERE condition is not specified, all rows from specified table will be deleted
- Example:
  - **DELETE FROM** product;

SELECT Queries

- Fine-tune SELECT command by adding restrictions to search criteria using:
  - Conditional restrictions/Comparison operator (=, <=, >=, etc.)
  - Arithmetic operators (+, -, /, *)
  - Logical operators (AND, OR, NOT)
  - Special operators (BETWEEN, EXISTS, etc.)

Comparison Operators

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

Selecting Rows with Conditional Restrictions

- Select partial table contents by placing restrictions on rows to be included in output
  - Add conditional restrictions to SELECT statement, using WHERE clause

- Syntax:

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

Selected PRODUCT Table Attributes for VENDOR Code 21344

```
SELECT P_DESCRIPT, P_INDATE, P_PRICE,V_CODE
FROM PRODUCT
WHEREV_CODE = 21344;
```

P_DESCRIPT	P_INDATE	P_PRICE	V_CODE
7.25-in. pwr. saw blade	13-Dec-05	14.99	21344
9.00-in. pwr. saw blade	13-Nov-05	17.49	21344
Rat-tail file, 1/8-in. fine	15-Dec-05	4.99	21344

Selected PRODUCT Table Attributes for VENDOR Codes Other than 21344

```
SELECT P_DESCRIPT,P_INDATE,P_PRICE,V_CODE
FROM PRODUCT
WHEREV_CODE <> 21344;
```

P_DESCRIPT	P_INDATE	P_PRICE	V_CODE
Power painter, 15 psi., 3-nozzle	03-Nov-05	109.99	25595
Hrd. cloth, 1/4-in., 2x50	15-Jan-06	39.95	23119
Hrd. cloth, 1/2-in., 3x50	15-Jan-06	43.99	23119
B&D jigsaw, 12-in. blade	30-Dec-05	109.92	24288
B&D jigsaw, 8-in. blade	24-Dec-05	99.87	24288
B&D cordless drill, 1/2-in.	20-Jan-06	38.95	25595
Claw hammer	20-Jan-06	9.95	21225
Hicut chain saw, 16 in.	07-Feb-06	256.99	24288
1.25-in. metal screw, 25	01-Mar-06	6.99	21225
2.5-in. wd. screw, 50	24-Feb-06	8.45	21231
Steel matting, 4'x8'x1/6", .5" mesh	17-Jan-06	119.95	25595

Selected PRODUCT Table Attributes with a P_PRICE Restriction

```
SELECT P_DESCRIPT, P_QOH, P_MIN, P_PRICE
FROM PRODUCT
WHERE P_PRICE <= 10;
```

P_DESCRIPT	P_QOH	P_MIN	P_PRICE
Claw hammer	23	10	9.95
Rat-tail file, 1/8-in. fine	43	20	4.99
PVC pipe, 3.5-in., 8-ft	188	75	5.87
1.25-in. metal screw, 25	172	75	6.99
2.5-in. wd. screw, 50	237	100	8.45

Selected PRODUCT Table Attributes: The ASCII Code Effect

```
SELECT P_CODE, P_DESCRIPT, P_QOH, P_MIN, P_PRICE
FROM PRODUCT
WHERE P_CODE < '1558-QW1';
```

P_CODE	P_DESCRIPT	P_QOH	P_MIN	P_PRICE
11QER/31	Power painter, 15 psi., 3-nozzle	8	5	109.99
13-Q2/P2	7.25-in. pwr. saw blade	32	15	14.99
14-Q1/L3	9.00-in. pwr. saw blade	18	12	17.49
1546-QQ2	Hrd. cloth, 1/4-in., 2x50	15	8	39.95

Selected PRODUCT Table Attributes: Date Restriction

```
SELECT P_DESCRIPT, P_QOH, P_MIN, P_PRICE, P_INDATE
FROM PRODUCT
WHERE P_INDATE >= '2004-01-20';
```

P_DESCRIPT	P_QOH	P_MIN	P_PRICE	P_INDATE
38D cordless drill, 1/2-in	12	5	38.95	20-Jan-06
Claw hammer	23	10	9.95	20-Jan-06
Hicut chain saw, 16 in.	11	5	256.99	07-Feb-06
PVC pipe, 3.5-in., 8-ft.	188	75	5.87	20-Feb-06
1.25-in. metal screw, 25	172	75	6.99	01-Mar-06
2.5-in. wd. screw, 50	237	100	8.45	24-Feb-06

SELECT Statement with a Computed Column

```
SELECT P_DESCRIPT,P_QOH,P_PRICE, P_ONHAND * P_PRICE
AS P_SUM
FROM PRODUCT;
```

P_DESCRIPT	P_QOH	P_PRICE	Expr1
Power painter, 15 psi., 3-nozzle	8	109.99	879.92
7.25-in. pwr. saw blade	32	14.99	479.68
9.00-in. pwr. saw blade	18	17.49	314.82
Hrd. cloth, 1/4-in., 2x50	15	39.95	599.25
Hrd. cloth, 1/2-in., 3x50	23	43.99	1011.77
B&D jigsaw, 12-in. blade	8	109.92	879.36
B&D jigsaw, 8-in. blade	6	99.87	599.22
B&D cordless drill, 1/2-in.	12	38.95	467.40
Claw hammer	23	9.95	228.85
Sledge hammer, 12 lb.	8	14.40	115.20
Rat-tail file, 1/8-in. fine	43	4.99	214.57
Hicut chain saw, 16 in.	11	256.99	2826.89
PVC pipe, 3.5-in., 8-ft.	188	5.87	1103.56
1.25-in. metal screw, 25	172	6.99	1202.28
2.5-in. wd. screw, 50	237	8.45	2002.65
Steel matting, 4'x8'x1/6", .5" mesh	18	119.95	2159.10

SELECT Statement with a Computed Column and an Alias

```
SELECT P_DESCRIPT,P_QOH,P_PRICE,
       P_ONHAND * P_PRICE AS TOTVALUE
FROM PRODUCT;
```

P_DESCRIPT	P_QOH	P_PRICE	TOTVALUE
Power painter, 15 psi., 3-nozzle	8	109.99	879.92
7.25-in. pwr. saw blade	32	14.99	479.68
9.00-in. pwr. saw blade	18	17.49	314.82
Hrd. cloth, 1/4-in., 2x50	15	39.95	599.25
Hrd. cloth, 1/2-in., 3x50	23	43.99	1011.77
B&D jigsaw, 12-in. blade	8	109.92	879.36
B&D jigsaw, 8-in. blade	6	99.87	599.22
B&D cordless drill, 1/2-in.	12	38.95	467.40
Claw hammer	23	9.95	228.85
Sledge hammer, 12 lb.	8	14.40	115.20
Rat-tail file, 1/8-in. fine	43	4.99	214.57
Hicut chain saw, 16 in.	11	256.99	2826.89
PVC pipe, 3.5-in., 8-ft.	188	5.87	1103.56
1.25-in. metal screw, 25	172	6.99	1202.28
2.5-in. wd. screw, 50	237	8.45	2002.65
Steel matting, 4'x8'x1/6", .5" mesh	18	119.95	2159.10

Arithmetic Operators: The Rule of Precedence

- Perform operations within parentheses
- Perform power operations
- Perform multiplications and divisions
- Perform additions and subtractions

TABLE 7.7 The Arithmetic Operators

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

Logical Operators: AND, OR, and NOT

- Searching data involves multiple conditions
- Logical operators: **AND**, **OR**, and **NOT**
- Can be combined
  - Parentheses enforce precedence order
    - Conditions in parentheses are always executed first
- NOT negates result of conditional expression

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;
```

P_DESCRIPT	P_INDATE	P_PRICE	V_CODE
7.25-in. pwr. saw blade	13-Dec-05	14.99	21344
9.00-in. pwr. saw blade	13-Nov-05	17.49	21344
B&D jigsaw, 12-in. blade	30-Dec-05	109.92	24288
B&D jigsaw, 8-in. blade	24-Dec-05	99.87	24288
Rat-tail file, 1/8-in. fine	15-Dec-05	4.99	21344
Hicut chain saw, 16 in.	07-Feb-06	256.99	24288

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 > '2004-01-15';
```

P_DESCRIPT	P_INDATE	P_PRICE	V_CODE
B&D cordless drill, 1/2-in	20-Jan-06	38.95	25595
Claw hammer	20-Jan-06	9.95	21225
PVC pipe, 3.5-in., 8-ft.	20-Feb-06	5.87	
1.25-in. metal screw, 25	01-Mar-06	6.99	21225
2.5-in. wd. screw, 50	24-Feb-06	8.45	21231

Selected PRODUCT Table Attributes: The Logical AND and OR

```
SELECT P_DESCRIPT, P_INDATE, P_PRICE,V_CODE
FROM PRODUCT
WHERE (P_PRICE < 50AND P_DATE > '2004-01-15')
        OR  V_CODE = 24288;
```

P_DESCRIPT	P_INDATE	P_PRICE	V_CODE
B&D jigsaw, 12-in. blade	30-Dec-05	109.92	24288
B&D jigsaw, 8-in. blade	24-Dec-05	99.87	24288
B&D cordless drill, 1/2-in.	20-Jan-06	38.95	25595
Claw hammer	20-Jan-06	9.95	21225
Hicut chain saw, 16 in.	07-Feb-06	256.99	24288
PVC pipe, 3.5-in., 8-ft.	20-Feb-06	5.87	
1.25-in. metal screw, 25	01-Mar-06	6.99	21225
2.5-in. wd. screw, 50	24-Feb-06	8.45	21231

Selected PRODUCT Table Attributes: The Logical AND and OR

```
SELECT P_DESCRIPT,P_INDATE,P_PRICE,V_CODE
FROM PRODUCT
WHERE NOT (V_CODE=21344);
```

P_DESCRIPT	P_INDATE	P_PRICE	V_CODE
Power painter, 15 psi, 3-nozzle	03-Nov-05	109.99	25595
Hrd. cloth, 1/4-in., 2x50	15-Jan-06	39.95	23119
Hrd. cloth, 1/2-in., 3x50	15-Jan-06	43.99	23119
B&D jigsaw, 12-in. blade	30-Dec-05	109.92	24288
B&D jigsaw, 8-in. blade	24-Dec-05	99.87	24288
B&D cordless drill, 1/2-in.	20-Jan-06	38.95	25595
Claw hammer	20-Jan-06	9.95	21225
Hicut chain saw, 16 in.	07-Feb-06	256.99	24288
1.25-in. metal screw, 25	01-Mar-06	6.99	21225
2.5-in. wd. screw, 50	24-Feb-06	8.45	21231
Steel matting, 4'x8'x1/6", .5" mesh	17-Jan-06	119.95	25595

Special Operators

- **BETWEEN**
- **IS NULL**
- **LIKE**
- **IN**
- **EXISTS**

Special Operators

- **BETWEEN** is used to define range limits

- Examples:
  - ```
    SELECT *
    FROM PRODUCT
    WHERE P_PRICE BETWEEN 50.00 AND 100.00;
    ```
  - ```
    SELECT *
    FROM PRODUCT
    WHERE P_PRICE >= 50.00 AND P_PRICE <= 100.00;
    ```

Special Operators

- **IS NULL** is used to check whether an attribute value is null.

- Examples:
  - ```
    SELECT P_CODE, P_DESCRIPT, V_CODE
    FROM PRODUCT
    WHERE V_CODE IS NULL;
    ```
  - ```
    SELECT P_CODE, P_DESCRIPT, P_INDATE
    FROM PRODUCT
    WHERE P_INDATE IS NOT NULL;
    ```

Special Operators

- **LIKE** is used to check for similar character strings.

- Examples:

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

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

Special Operators -LIKE: %

- SQL has two special *pattern matching* symbols:
  - % sequence of zero or more characters
  - _ (underscore): any single character
- **LIKE** 'Sm%' sequence of characters of any length starts with '*Sm*'.
- **LIKE** '%st' sequence of characters of any length ends with 'st'
- **LIKE** '%mi%' sequence of characters of any length **contains** 'mi'

- Examples:

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

Special Operators - LIKE: _

- SQL has two special *pattern matching* symbols:
  - % sequence of zero or more characters
  - _ (underscore): any single character
- find the vendor who has the last names with the following pattern 'Je_i'

```
SELECT * from Vendor
   where LastName LIKE 'Je_i';
```

- The pattern 'Je_i' matches any string that starts with 'Je', followed by one character, and then followed by 'i'
- e.g., Jeri or Jeni, but not Jenni.

Special Operators -LIKE: MIXED %, _

- find the vendor who has the last names with the following: start with the string 'Je' followed by two characters and then any number of characters.
- SELECT * from Vendor

  where LastName LIKE 'Je_%';

  - it will match any last name that starts with Je and has at least 3 characters.
  - e.g., Jeri or Jeni, but not Je nor Jari.

Special Operators

- **IN** is used to check whether an attribute value matches a value contained within a (sub)set of listed values.

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

- **EXISTS** is used to check whether an attribute has value.

```
DELETE FROM PRODUCT WHERE P_CODE EXISTS;
SELECT * FROM PRODUCT WHERE V_CODE EXISTS;
```