Data Definition Language (DDL)

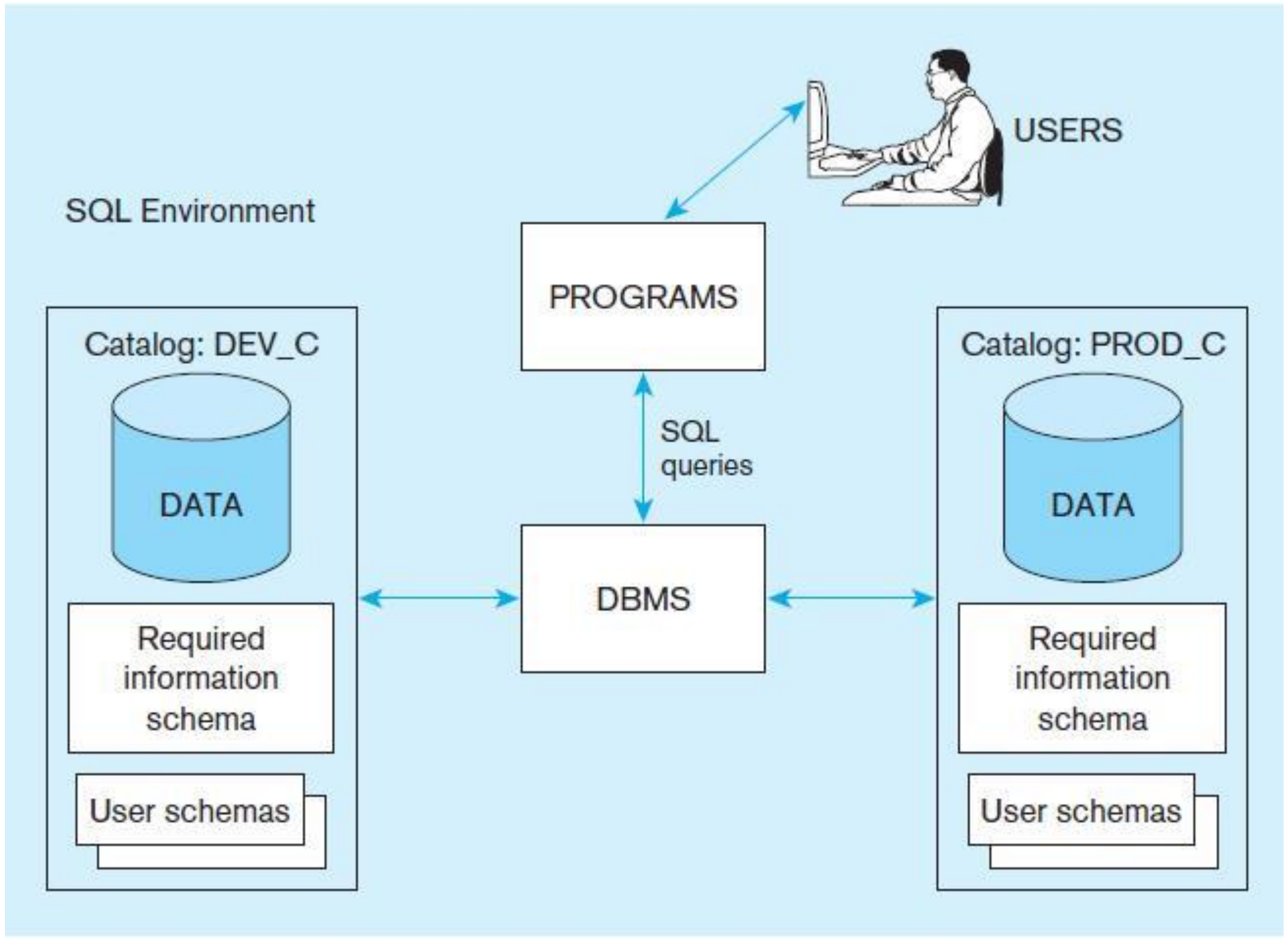
Database System Concepts, 7th Ed.

©Silberschatz, Korth and Sudarshan
See www.db-book.com for conditions on re-use

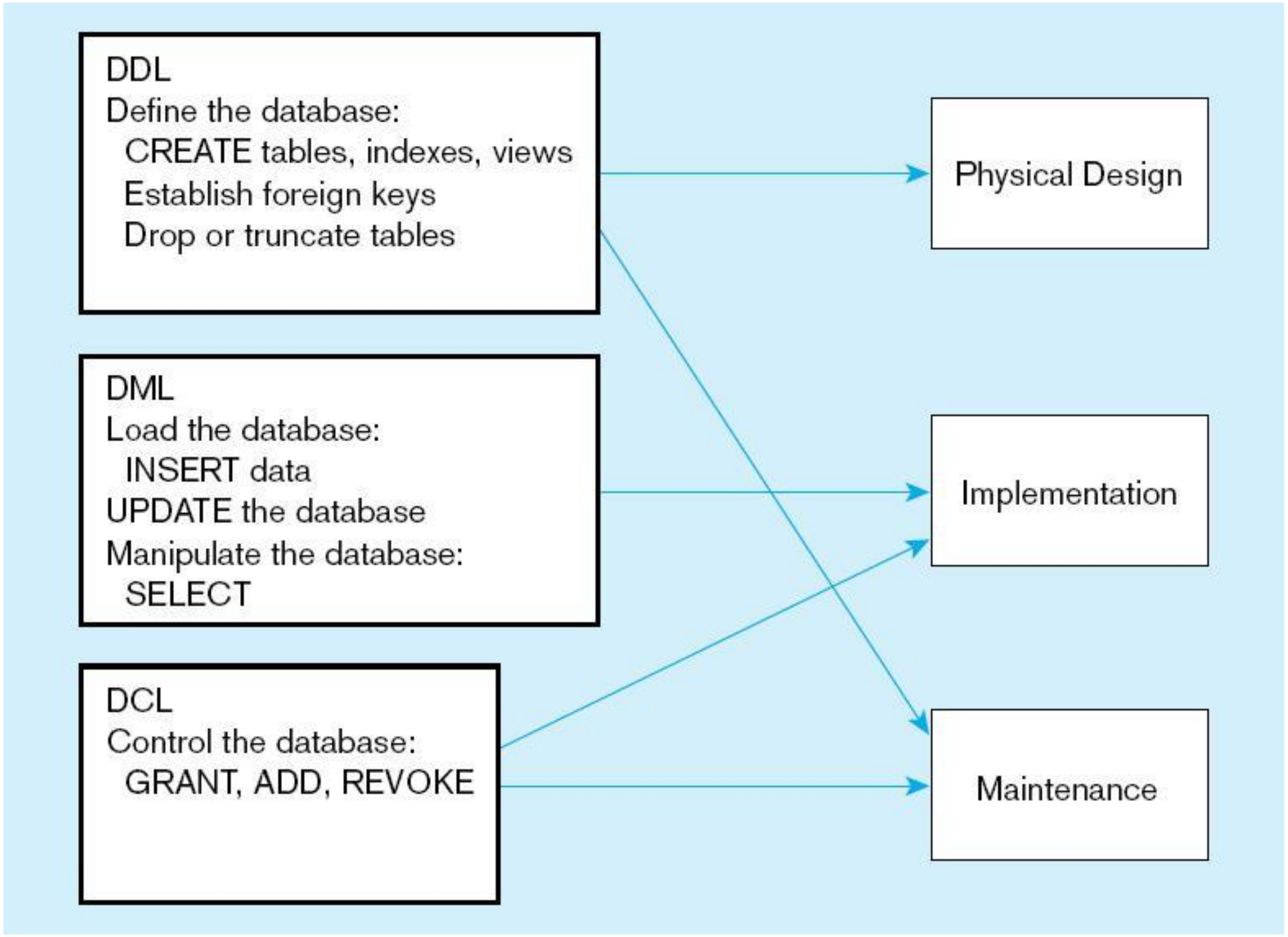
SQL Environment

- Catalog
  - A set of schemas that constitute the description of a database
- Schema
  - The structure that contains descriptions of objects created by a user (base tables, views, constraints)
- Data Definition Language (DDL)
  - Commands that define a database, including creating, altering, and dropping tables and establishing constraints
- Data Manipulation Language (DML)
  - Commands that maintain and query a database
- Data Control Language (DCL)
  - Commands that control a database, including administering privileges and committing data

A simplified schematic of a typical SQL environment



DDL (focus in this course), DML, DCL, and the database development process



Steps in Table Creation

1. Identify data types for attributes
2. Identify columns that can and cannot be null
3. Identify columns that must be unique (candidate keys)
4. Identify primary key–foreign key mates
5. Determine default values
6. Identify constraints on columns (domain specifications)
7. Create the table and associated indexes

SQL Data Types

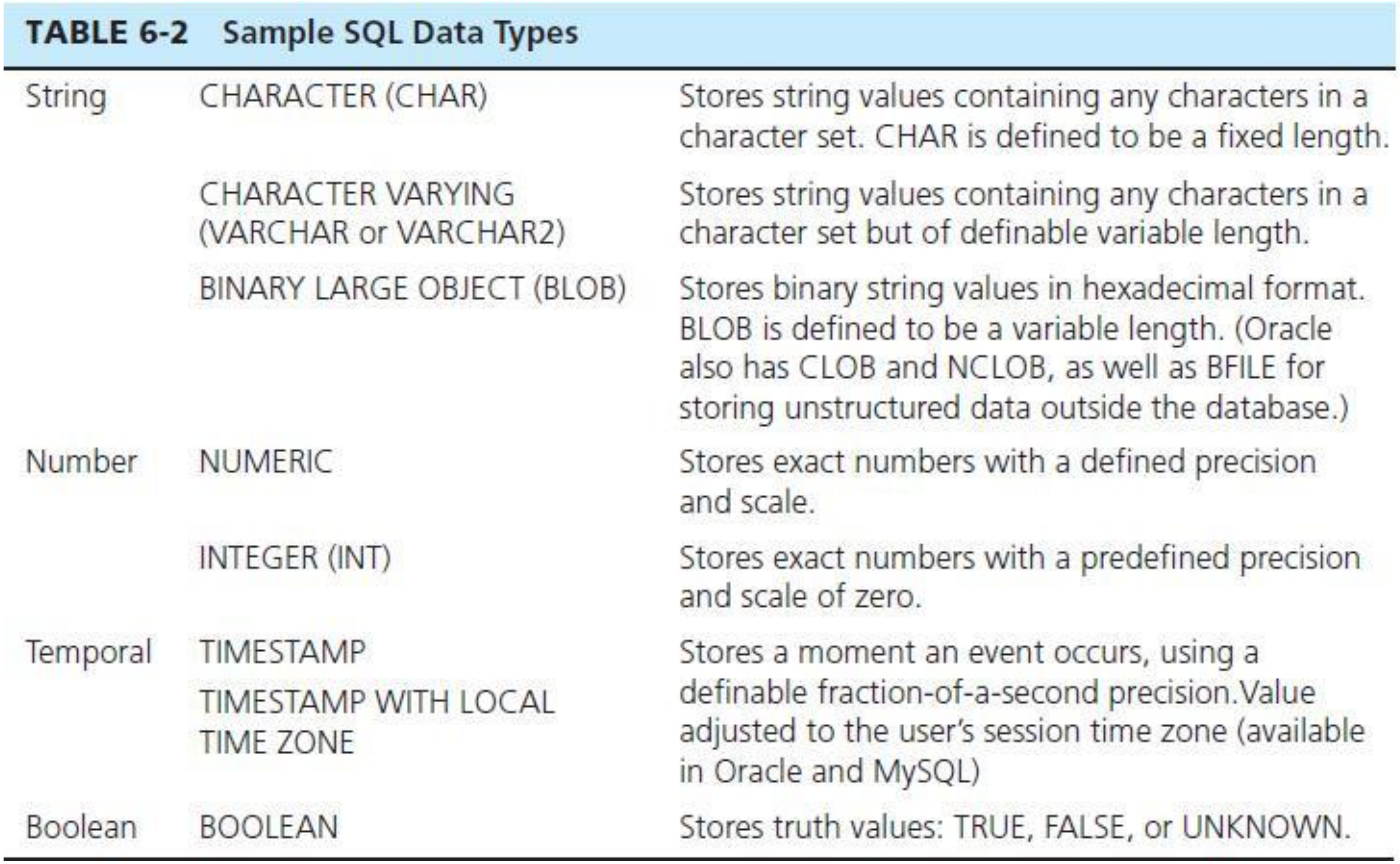


TABLE 6-2 Sample SQL Data Types

String	CHARACTER (CHAR)	Stores string values containing any characters in a character set. CHAR is defined to be a fixed length.
	CHARACTER VARYING (VARCHAR or VARCHAR2)	Stores string values containing any characters in a character set but of definable variable length.
	BINARY LARGE OBJECT (BLOB)	Stores binary string values in hexadecimal format. BLOB is defined to be a variable length. (Oracle also has CLOB and NCLOB, as well as BFILE for storing unstructured data outside the database.)
Number	NUMERIC	Stores exact numbers with a defined precision and scale.
	INTEGER (INT)	Stores exact numbers with a predefined precision and scale of zero.
Temporal	TIMESTAMP TIMESTAMP WITH LOCAL TIME ZONE	Stores a moment an event occurs, using a definable fraction-of-a-second precision. Value adjusted to the user's session time zone (available in Oracle and MySQL)
Boolean	BOOLEAN	Stores truth values: TRUE, FALSE, or UNKNOWN.

DDL

Introduction

- To understand the SQL Data Definition Language
  - Create
  - Insert
  - Delete
  - Drop
  - Truncate
  - Alter

DDL

Creating a Database

- To initialize a new database:
- **Syntax:**

  CREATE DATABASE database_name

- There are numerous arguments that go along with this command but are database specific
- Only some databases require database to be created and space to be allocated prior to creation of tables.
- Some databases provide graphical user interfaces to create databases and allocate space.
  - Access only allows database to be created using User Interface

DDL

Creating a Table

- **Syntax**

```
CREATE TABLE table_name
(Column_name      datatype[(size)],
 Column_name      datatype[(size)],
);
```

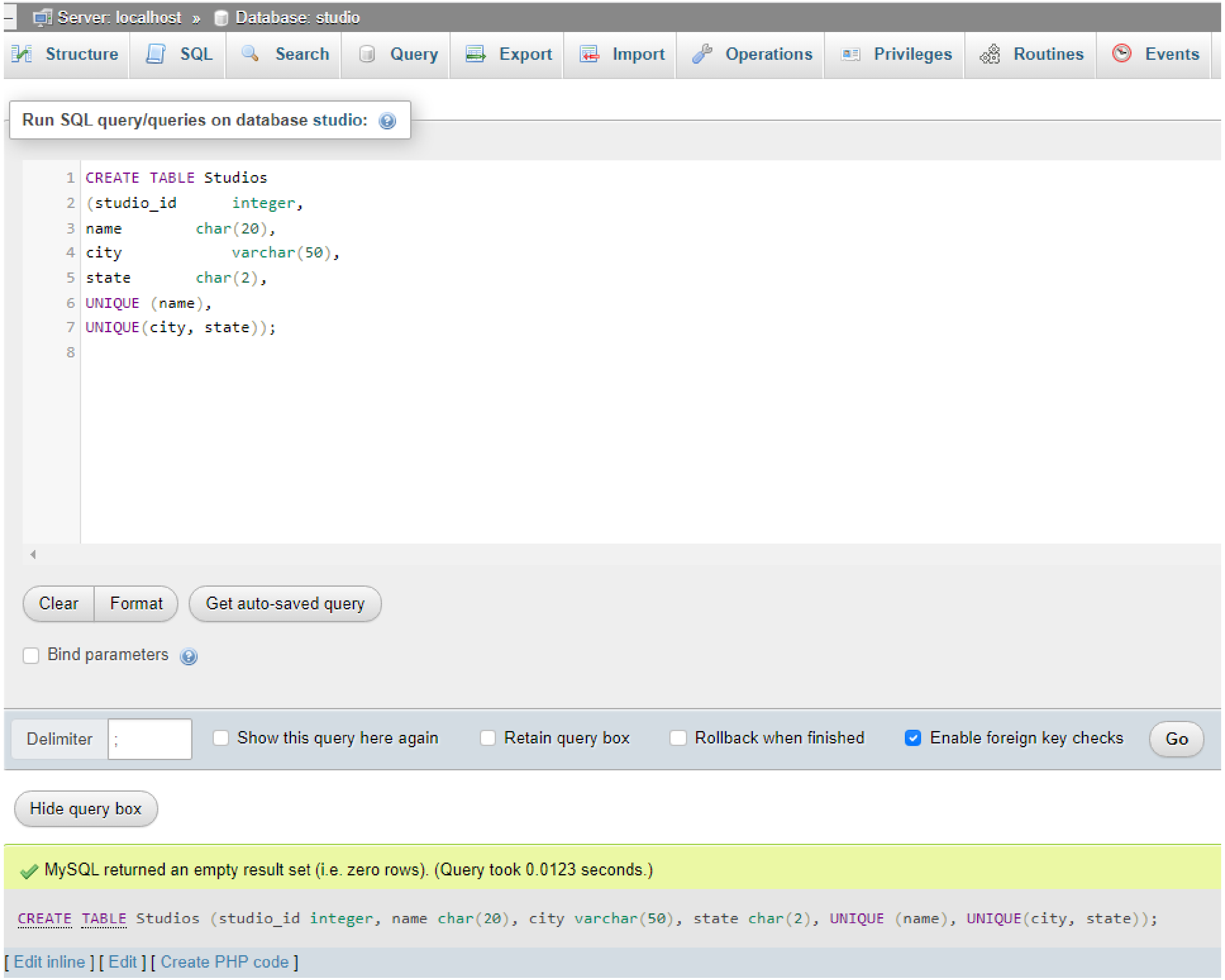
- **Example**

```
CREATE TABLE books
(ISBN             char(20),
Title             char(50),
AuthorID          Integer,
Price             float);
```

- Creates a books table with four columns

MySQL Interface – Using DDL

- Copy and Paste the previous DDL query into the SQL tab.



Run SQL query/queries on database studio:

```
CREATE TABLE Studios
(studio_id      integer,
name        char(20),
city            varchar(50),
state       char(2),
UNIQUE (name),
UNIQUE(city, state));
```

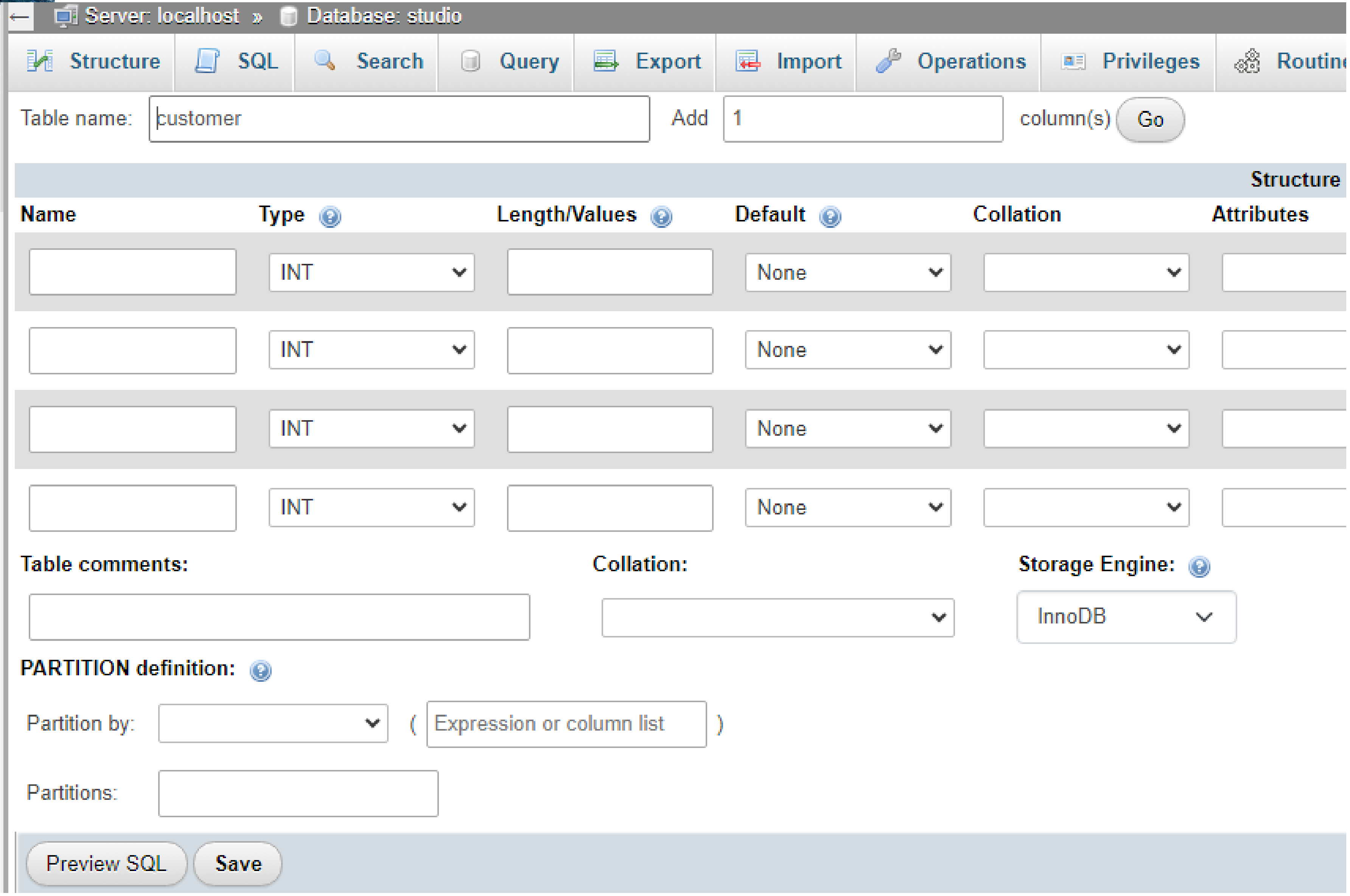
☐ Bind parameters

Delimiter ; ☐ Show this query here again ☐ Retain query box ☐ Rollback when finished ☑ Enable foreign key checks Go

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0123 seconds.)

```
CREATE TABLE Studios (studio_id integer, name char(20), city varchar(50), state char(2), UNIQUE (name), UNIQUE(city, state));
```

MySQL Interface – Using GUI



Server: localhost » Database: studio

Structure | SQL | Search | Query | Export | Import | Operations | Privileges | Routin

Table name: customer Add 1 column(s) Go

Structure

Name	Type	Length/Values	Default	Collation	Attributes
	INT		None		
	INT		None		
	INT		None		
	INT		None		

Table comments:

Collation:

Storage Engine: InnoDB

PARTITION definition:

Partition by: (Expression or column list)

Partitions:

Preview SQL | Save

DDL

Data Types

- Following broad categories of data types exist in most databases:
  - String Data
  - Numeric Data
  - Temporal Data
  - Large Objects

DDL

String Data

- **Fixed Length**:
- Occupies the same length of space in memory no matter how much data is stored in them.
- **Syntax:**

  char(n) where n is the length of the String

  e.g. name char(50)
- If the variable stored for name is ‘Sanjay’ the extra 43 fields are padded with blanks

DDL

String Data

- **Variable Length** string is specified with maximum length of characters possible in the string, however, the allocation is sized to the size of the data stored in memory.
- **Syntax:**

  Varchar(n) – n is the maximum length of data possible for the type
- There may be a restriction in the maximum length of the data that you can specify in the declaration which will vary according to the database.
- All character data has to be enclosed in single quotes during specification.

DDL

Numeric Data Types

- Store all the data related to purely numeric data.
- Some numeric data may also be stored as a character field e.g. zip codes
- **Common Numeric Types**:

Decimal	Floating point number
Float	Floating point number
Integer(size)	Integer of specified length
Money	A number which contains exactly two digits after the decimal point
Number	A standard number field that can hold a floating point data

Note: Different databases name their numeric fields differently and may not support all numeric types. They may also support additional numeric types.

DDL

Temporal Data Types

- **These represent the dates and time:**
- Three basic types are supported:
  - Dates
  - Times
  - Date-Time Combinations

DDL

Large Data Objects

- These are used for storing data objects like files and images:
- **There are two types**:
  - Character Large Objects (clobs)
  - Binary Large Objects (blobs)

BLOBs are used to store binary information, such as images, while CLOBs are used to store character information.

DDL

Specifying Keys- Introduction

- Unique keyword is used to specify candidate keys.
  - This ensures that duplicate rows are not created in the database.
- Both Primary keys and Candidate Keys can be specified in the database.
- Once a set of columns has been declared unique any data entered that duplicates the data in these columns is rejected.
- **Specifying a single column as unique**:
- Example

```
CREATE TABLE Studios
(studio_id          Number,
name                char(20),
city                varchar(50),
state               char(2),
UNIQUE (name));
```

- Here the name column has been declared as a candidate key and must be unique

DDL

Specifying Keys- Multiple Columns

- Specifying multiple columns as unique:
- **Example:**

```
CREATE TABLE Studios
(studio_id          integer,
name                char(20),
city                varchar(50),
state               char(2),
UNIQUE (name),
UNIQUE(city, state));
```

- Here both name & city/state combination are declared as unique and candidate keys

Unique Keys using Constraint Clause

```
CREATE TABLE studio (
  ...
  studio_id        Number,
  ...
  CONSTRAINT un_studio_id UNIQUE(studio_id))
```

Here studio_id must be unique. Alternatively, combinations of multiple columns can be made unique:

-
  ```
  CREATE TABLE studio (
  name               char(20),
  studio_id          Number,
  ...
  CONSTRAINT un_constraint_name UNIQUE(name,studio_id));
  ```

Here combination of name and studio_id must be unique

DDL

Specifying Keys- Primary Key

- Specifying multiple columns as unique:
- To specify the Primary Key the **Primary Key** clause is used
- **Example:**

```
 CREATE TABLE Studios
(studio_id          Number,
name                char(20),
city                varchar(50),
state               char(2),
PRIMARY KEY (studio_id),
UNIQUE (name),
UNIQUE(city, state)
);
```

DDL

Specifying Primary Key using Constraint Clause

- To specify the Primary Key constraint clause can also be used
- **Example:**

```
 CREATE TABLE Studios
(studio_id          Number,
name                char(20),
city                varchar(50),
state               char(2),
Constraint pk_studio_id PRIMARY KEY (studio_id),
UNIQUE (name),
UNIQUE(city, state)
);
```

DDL

Specifying Primary Key using Constraint Clause at Column Level

```
CREATE TABLE Studios
(studio_id          Number Constraint pk_studio_id PRIMARY KEY
        (studio_id),
name                char(20),
city                varchar(50),
state               char(2),
UNIQUE (name),
UNIQUE(city, state)
);
```

Creating a Primary Key for More than One Field

Creating a primary key with more than one field.

```
CREATE TABLE supplier
(
 supplier_id numeric(10) not null,
 supplier_name varchar2(50) not null,
 contact_name varchar2(50),
 CONSTRAINT supplier_pk PRIMARY KEY (supplier_id, supplier_name)
 );
```

DDL

Specifying Keys- Foreign Keys

- References clause is used to create a relationship between a set of columns in one table and a candidate key in the table that is being referenced.

- **Example:**

```
CREATE TABLE Movies
(movie_title        varchar(40),
studio_id           Number REFERENCES Studios(studio_id));
```

- Creates a relationship from the Movies table to the Studios table

DDL

Specifying Keys- Foreign Keys Using Constraint Clause

```
CREATE TABLE Movies
(movie_title   varchar(40),
studio_id      Number,
Constraint fk_studio_id FOREIGN KEY (studio_id)
      REFERENCES Studios(studio_id));
```

DDL

Constraints- Disallowing Null Values

Disallowing Null Values:

- Null values entered into a column means that the data in not known.
- These can cause problems in Querying the database.
- Specifying Primary Key automatically prevents null being entered in columns which specify the primary key

- Not Null clause is used in preventing null values from being entered in a column.

- **Example:**

```
CREATE TABLE Studios
( studio_id        number          NOT NULL        PRIMARY KEY,
  name             char(20)        NOT NULL,
  city             varchar(50)     NOT NULL,
  state            char(2)         NOT NULL);
```

- Null clause can be used to explicitly allow null values in a column also

DDL

Constraints- Value Constraints

Value Constraints:

- Allows value inserted in the column to be checked condition in the column constraint.

- Check clause is used to create a constraint in SQL

- **Example:**

```
CREATE TABLE Movies
(movie_titlevarchar(40)       PRIMARY KEY,
 studio_id                    Number,
 budget      Number    check     (budget > 50000)
) ;
```

- Table level constraints can also be defined using the Constraint keyword

- **Example:**

```
CREATE TABLE Movies
(movie_titlevarchar(40)                PRIMARY KEY,
 studio_id                    Number,
 budget      Number check (budget > 50000),
release_date                  Date,
 CONSTRAINT release_date_constraint Check (release_date between '01-Jan-1980' and '31-dec-
        1989));
```

- Such constraints can be activated and deactivated as required.

DDL

Constraints- Default Value

Default Value:

- A default value can be inserted in any column by using the Default keyword.

- **Example:**

```
CREATE TABLE Movies (
movie_title     varchar(40)     NOT NULL,
release_date    date            DEFAULT sysdate          NULL,
genre           varchar(20)     DEFAULT 'Comedy' Check genre In
        ('Comedy', 'Horror', 'Drama'));
```

- Table level constraints can also be defined using the Constraint keyword; CONSTRAINT release_date_constraint Check (release_date between '01-Jan-1980' and '31-dec-1989'))
- release_date defaults to the current date, however Null value is enabled in the column which will need to be added explicitly when data is added.
- **Note:** Any valid expression can be used while specifying constraints

W3C SQL Tutorial

- Please look at more examples regarding create table and constraints from

https://www.w3schools.com/sql/sql_create_db.asp

Create Table Examples in a Enterprise Domain

©Silberschatz, Korth and Sudarshan
See www.db-book.com for conditions on re-use

The following slides create tables for this enterprise data model

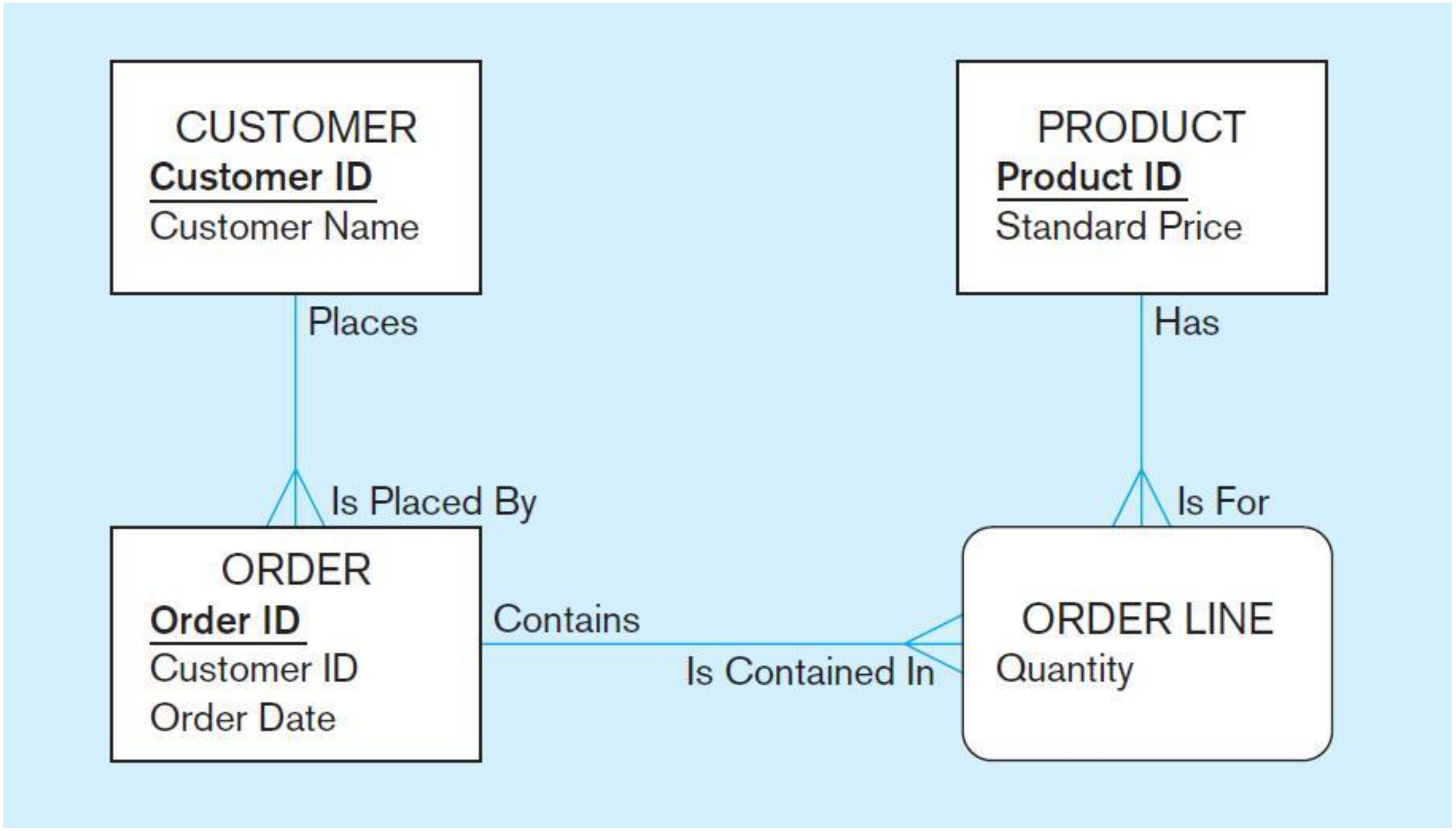
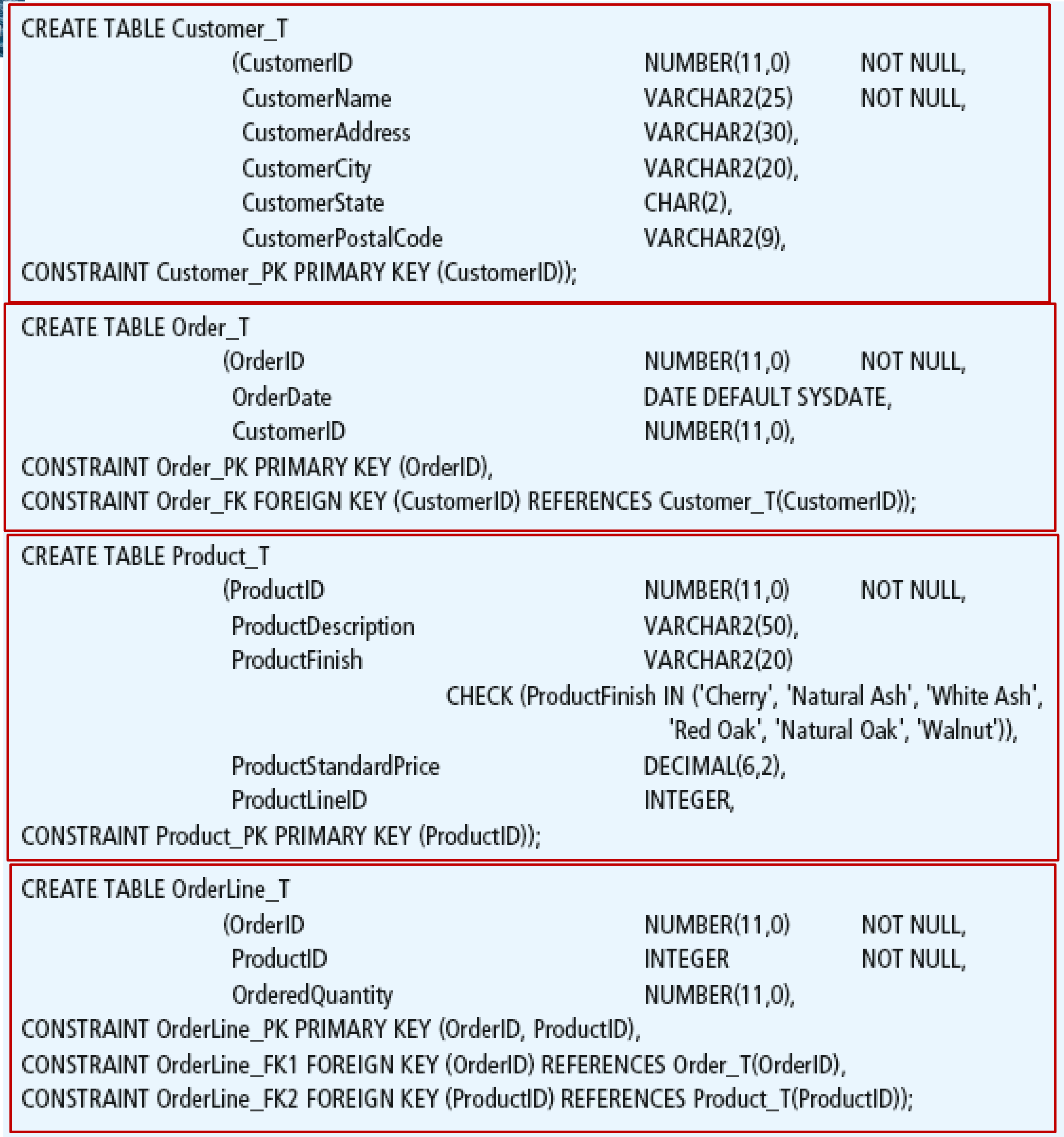


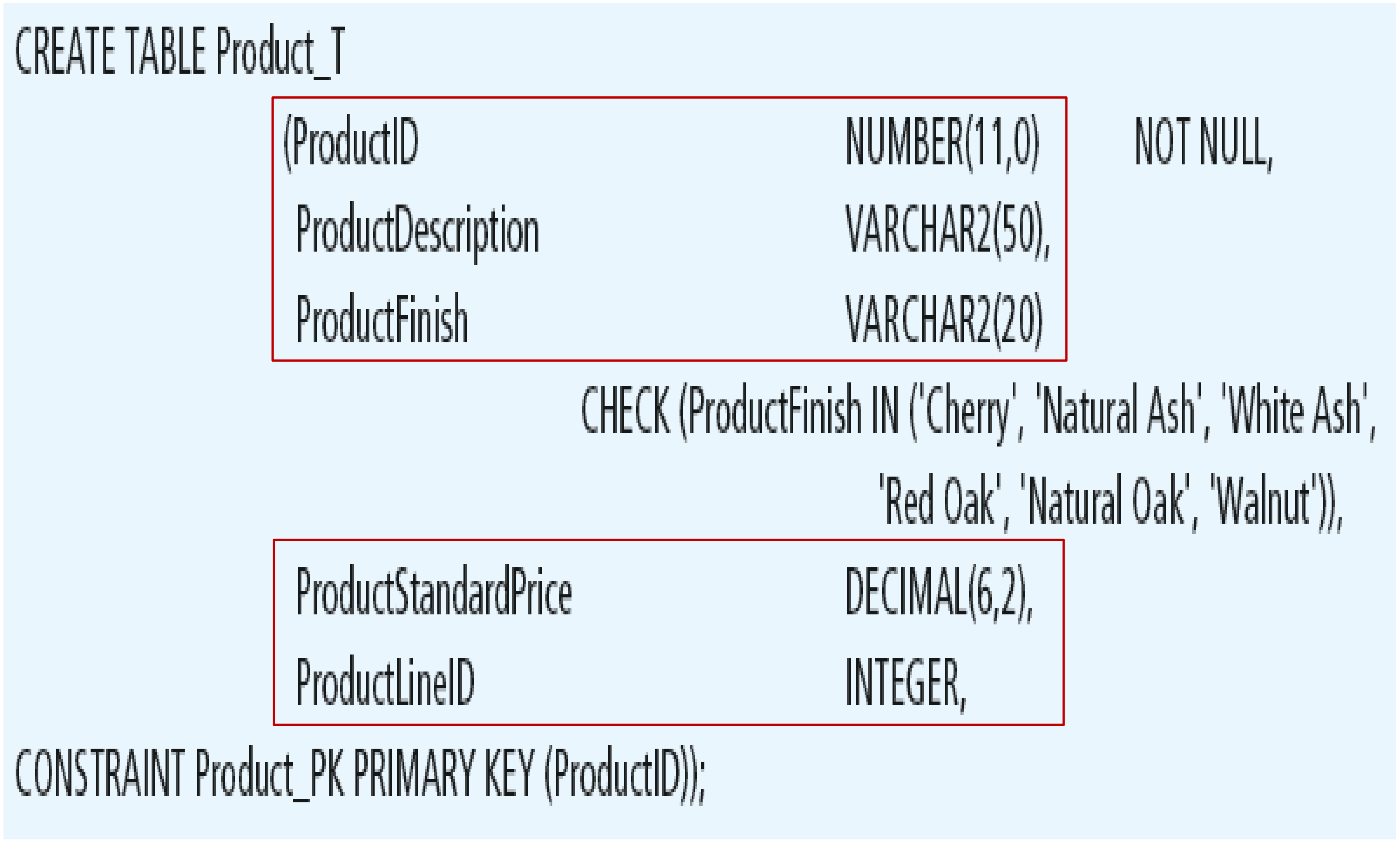
Figure 6-6 SQL database definition commands for Pine Valley Furniture Company (Oracle 11g)



```
CREATE TABLE Customer_T
                    (CustomerID                    NUMBER(11,0)      NOT NULL,
                     CustomerName                  VARCHAR2(25)      NOT NULL,
                     CustomerAddress               VARCHAR2(30),
                     CustomerCity                  VARCHAR2(20),
                     CustomerState                 CHAR(2),
                     CustomerPostalCode            VARCHAR2(9),
CONSTRAINT Customer_PK PRIMARY KEY (CustomerID));

CREATE TABLE Order_T
                   (OrderID                        NUMBER(11,0)      NOT NULL,
                    OrderDate                      DATE DEFAULT SYSDATE,
                    CustomerID                     NUMBER(11,0),
CONSTRAINT Order_PK PRIMARY KEY (OrderID),
CONSTRAINT Order_FK FOREIGN KEY (CustomerID) REFERENCES Customer_T(CustomerID));

CREATE TABLE Product_T
                   (ProductID                      NUMBER(11,0)      NOT NULL,
                    ProductDescription             VARCHAR2(50),
                    ProductFinish                  VARCHAR2(20)
                                       CHECK (ProductFinish IN ('Cherry', 'Natural Ash', 'White Ash',
                                                     'Red Oak', 'Natural Oak', 'Walnut')),
                    ProductStandardPrice           DECIMAL(6,2),
                    ProductLineID                  INTEGER,
CONSTRAINT Product_PK PRIMARY KEY (ProductID));

CREATE TABLE OrderLine_T
                   (OrderID                        NUMBER(11,0)      NOT NULL,
                    ProductID                      INTEGER           NOT NULL,
                    OrderedQuantity                NUMBER(11,0),
CONSTRAINT OrderLine_PK PRIMARY KEY (OrderID, ProductID),
CONSTRAINT OrderLine_FK1 FOREIGN KEY (OrderID) REFERENCES Order_T(OrderID),
CONSTRAINT OrderLine_FK2 FOREIGN KEY (ProductID) REFERENCES Product_T(ProductID));
```

Overall table definitions

Defining attributes and their data types



```
CREATE TABLE Product_T
            (ProductID                    NUMBER(11,0)    NOT NULL,
             ProductDescription           VARCHAR2(50),
             ProductFinish                VARCHAR2(20)
                          CHECK (ProductFinish IN ('Cherry', 'Natural Ash', 'White Ash',
                                      'Red Oak', 'Natural Oak', 'Walnut')),
             ProductStandardPrice         DECIMAL(6,2),
             ProductLineID                INTEGER,
CONSTRAINT Product_PK PRIMARY KEY (ProductID));
```

Copyright © 2014 Pearson Education, Inc.

Non-nullable specification

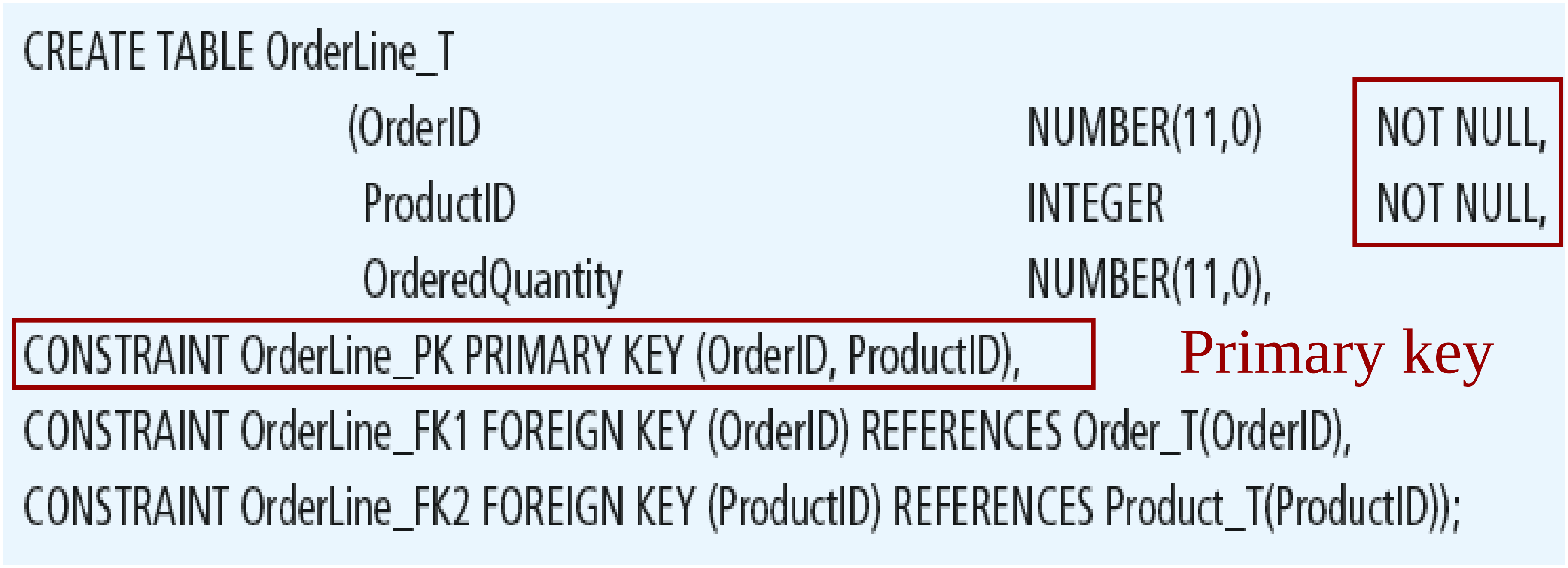
```
CREATE TABLE Product_T
            (ProductID                  NUMBER(11,0)    NOT NULL,
             ProductDescription         VARCHAR2(50),
             ProductFinish              VARCHAR2(20)
                     CHECK (ProductFinish IN ('Cherry', 'Natural Ash', 'White Ash',
                                'Red Oak', 'Natural Oak', 'Walnut')),
             ProductStandardPrice       DECIMAL(6,2),
             ProductLineID              INTEGER,
CONSTRAINT Product_PK PRIMARY KEY (ProductID));
```

Primary keys can never have NULL values

Identifying primary key

Copyright © 2014 Pearson Education, Inc.

Non-nullable specifications



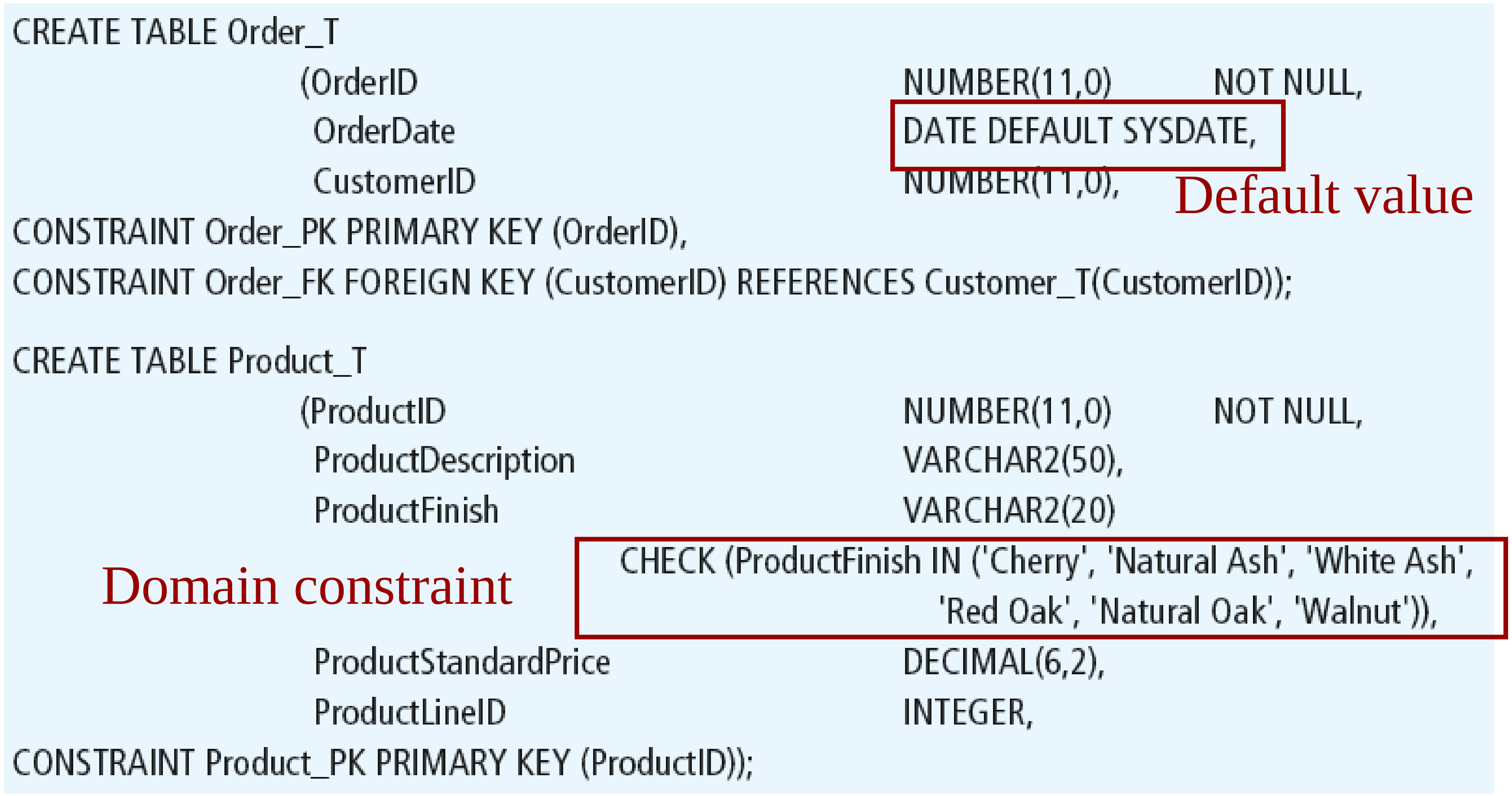
```
CREATE TABLE OrderLine_T
                    (OrderID                                    NUMBER(11,0)      NOT NULL,
                     ProductID                                  INTEGER           NOT NULL,
                     OrderedQuantity                            NUMBER(11,0),
CONSTRAINT OrderLine_PK PRIMARY KEY (OrderID, ProductID),
CONSTRAINT OrderLine_FK1 FOREIGN KEY (OrderID) REFERENCES Order_T(OrderID),
CONSTRAINT OrderLine_FK2 FOREIGN KEY (ProductID) REFERENCES Product_T(ProductID));
```

Primary key

Some primary keys are composite–
composed of multiple attributes

Copyright © 2014 Pearson Education, Inc.

Controlling the values in attributes



```
CREATE TABLE Order_T
                    (OrderID                                   NUMBER(11,0)         NOT NULL,
                     OrderDate                                 DATE DEFAULT SYSDATE,
                     CustomerID                                NUMBER(11,0),
CONSTRAINT Order_PK PRIMARY KEY (OrderID),
CONSTRAINT Order_FK FOREIGN KEY (CustomerID) REFERENCES Customer_T(CustomerID));

CREATE TABLE Product_T
                    (ProductID                                 NUMBER(11,0)         NOT NULL,
                     ProductDescription                        VARCHAR2(50),
                     ProductFinish                             VARCHAR2(20)
                                      CHECK (ProductFinish IN ('Cherry', 'Natural Ash', 'White Ash',
                                                      'Red Oak', 'Natural Oak', 'Walnut')),
                     ProductStandardPrice                      DECIMAL(6,2),
                     ProductLineID                             INTEGER,
CONSTRAINT Product_PK PRIMARY KEY (ProductID));
```

Default value

Domain constraint

Copyright © 2014 Pearson Education, Inc.