ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN
Vietnam - Korea University of Information and Communication Technology

Database Systems

Chapter 4: Structured Query Language (SQL)

Session 1: Introduction to SQL, DDL, DML

http://vku.udn.vn/

Outline

Introduction to SQL

❑Pronounced “S-Q-L” by some and “sequel” by others

❑Pronounced “S-Q-L” by some and “sequel” by others

❑**SQL is a Nonprocedural language**

❑The standard for relational database management systems (RDBMS).

History of SQL

- ❑1970–Edgar E. Codd develops relational database concept
- ❑1974-1979–System R with Sequel (later SQL) created at IBM Research Lab
- ❑1979–Oracle markets first relational DB with SQL
- ❑1986–ANSI (American National Standards Institue) SQL standards were first published
- ❑1989, 1992, 1999, 2003, 2006, 2008, 2011 and 2016–Major ANSI standard updates
- ❑Current–SQL is supported by most major database vendors

Purpose of SQL Standard

- ❑Specify syntax/semantics for data definition and manipulation
- ❑Define data structures
- ❑Enable portability
- ❑Specify minimal (level 1) and complete (level 2) standards
- ❑Allow for later growth/enhancement to standard

Benefits of a Standardized Relational Language

- ❑Reduced training costs
- ❑Productivity
- ❑Application portability
- ❑Application longevity
- ❑Reduced dependence on a single vendor
- ❑Cross-system communication

SQL Commands

- 3 types:

SQL commands

Data Definition Language(DDL)

Define the database:

- CREATE, ALTER, DROP TABLES, VIEWS, INDEXES
- ESTABLISHING CONSTRAINTS

Data Manipulation Language(DML)

Maintain and query the database:

- UPDATING, INSERTING, DELETING AND QUERYING DATA

Data Control Language(DCL)

Control the database:

- GRANT OR REVOKE PRIVILEGES TO ACCESS THE DATABASE
- COMMITING DATA

Data Definition Language(DDL)

- DDL statements are used to define a database
- Major CREATE statements:
  - CREATE DATABASE – create a new database
  - CREATE SCHEMA–defines a portion of the database owned by a particular user
  - CREATE TABLE–defines a table and its columns
  - CREATE VIEW–defines a logical table from one or more views

Creating a database

- Two tasks must be completed:
  - create the database structure
  - create the tables that will hold the end-user data
- First task
  - RDBMS creates the physical files that will hold the database
  - Tends to **differ** substantially **from one RDBMS to another**

Creating a database

- ❑A SQL Server database can be created, altered and dropped by one of two following methods:
  - Using the designer with SQL Server Management Studio (SSMS) or
  - Using a Query
- ❑After creating database, 2 files are generated:
  - .MDF file – Data file (contains actual data)
  - .LDF file _ Transaction Log file (used to recover the database)

Creating a new database

1. Using designer:
   - Right-click to Database menu, choose New Database

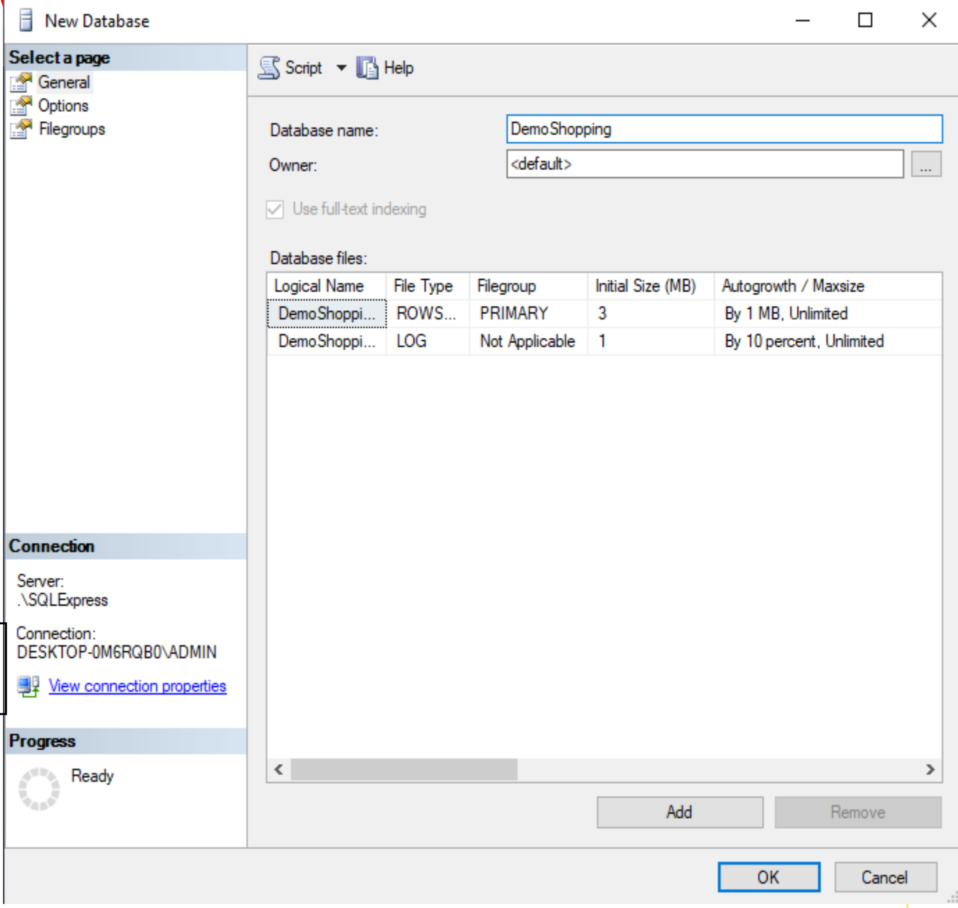
2. Using a Query: Ctrl +N
   To create a new query
   - Syntax:

```
CREATE DATABASE DatabaseName
```

   - Example:

```
CREATE DATABASE DemoShopping
```

   - Click Execute button to create the DemoShopping database.



New Database

Select a page: General, Options, Filegroups

Script ▾ Help

Database name: DemoShopping

Owner: <default>

☑ Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize
DemoShoppi...	ROWS...	PRIMARY	3	By 1 MB, Unlimited
DemoShoppi...	LOG	Not Applicable	1	By 10 percent, Unlimited

Connection

Server:
.\SQLExpress

Connection:
DESKTOP-0M6RQB0\ADMIN

View connection properties

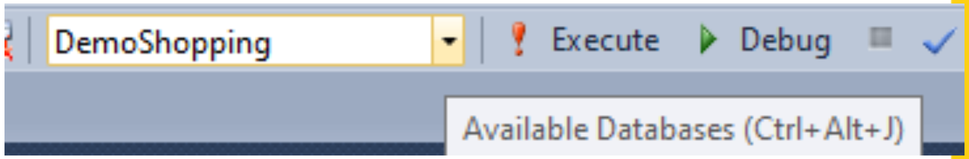
Progress

Ready

Creating a new database

❑Using the database

1. Click the combobox to choose the database



2. Use Query

- Syntax:

```
USE DatabaseName
```

- Example

```
USE DemoShopping
```

❑Right-click to the database, choose Properties, click Files to know database files

Altering/Deleting a Database

- **To alter an existing database:**
  - Syntax:

```
ALTER DATABASE OldDatabase MODIFY NAME = NewDatabase
```

  - Example:

```
ALTER DATABASE DemoShopping MODIFY NAME = DemoShopping2
```

- **To drop a database**
  - Syntax:

```
DROP DATABASE TheDatabase
```

  - Example:

```
DROP DATABASE DemoShopping2
```

  - Dropping a database will delete the LDF and MDF files.

Creating the Database Schema

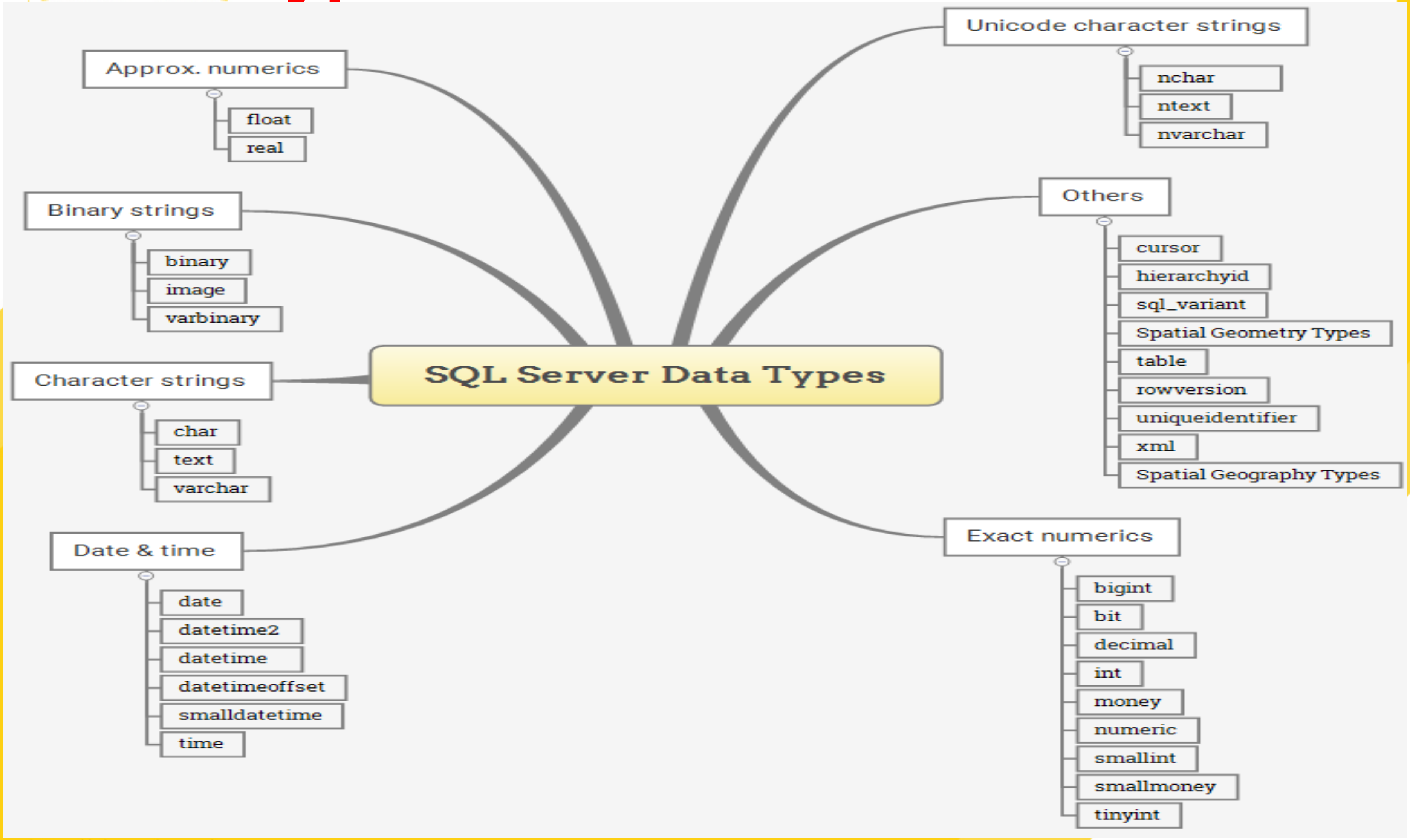
- Schema: logical database structure
  - Is a group of database objects- such as tables and indexes – that are related to each other.
- Authentication
  - Process through which the DBMS verifies that only registered users are able to access the database
  - Log on to the RDBMS using a user ID and a password created by the database administrator
- Syntax:

```
CREATE SCHEMA AUTHORIZATION <creator>
```

- Ex:

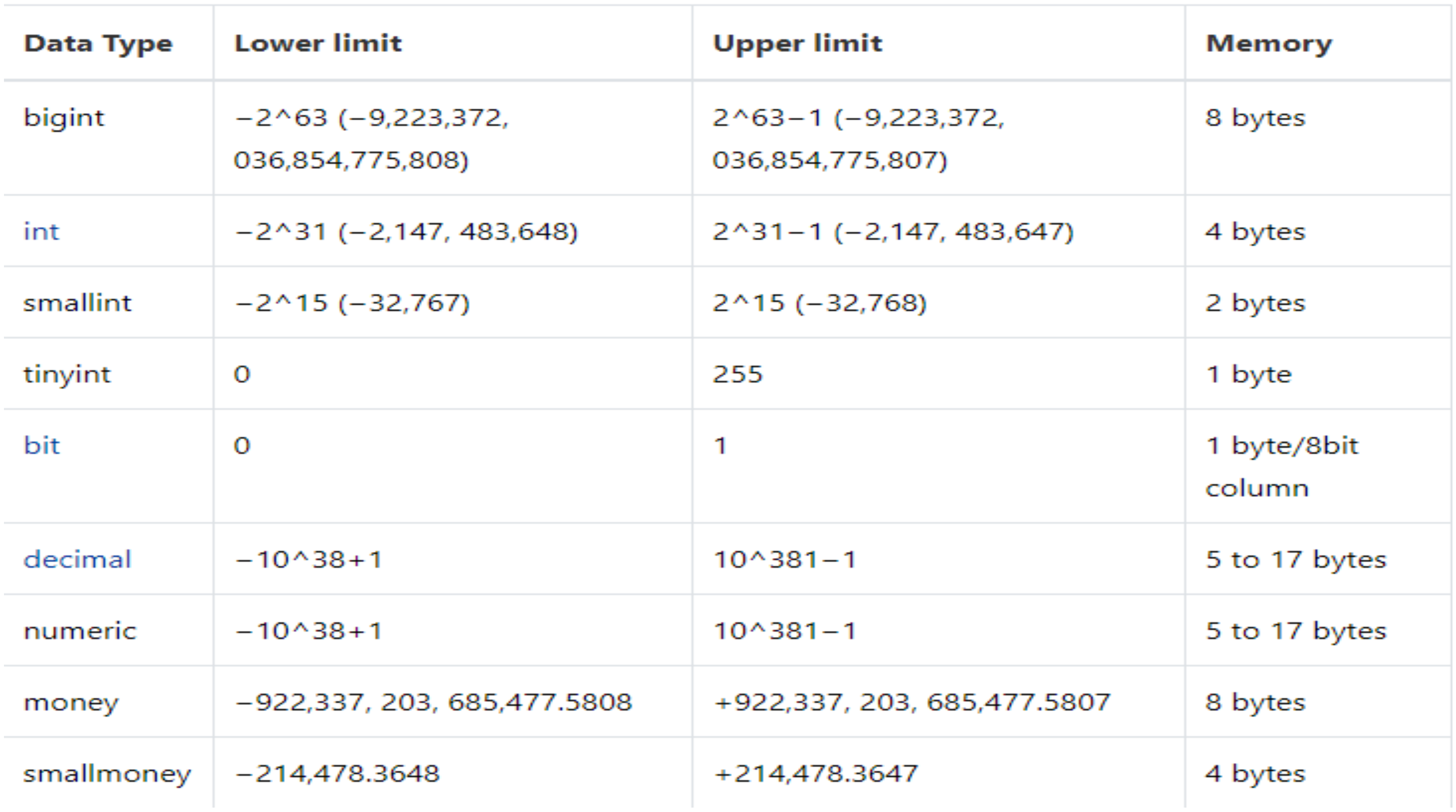
```
CREATE SCHEMA AUTHORIZATION vku
```

Some Common SQL Data Types



Some Common SQL Data Types

❑Exact numeric data types:

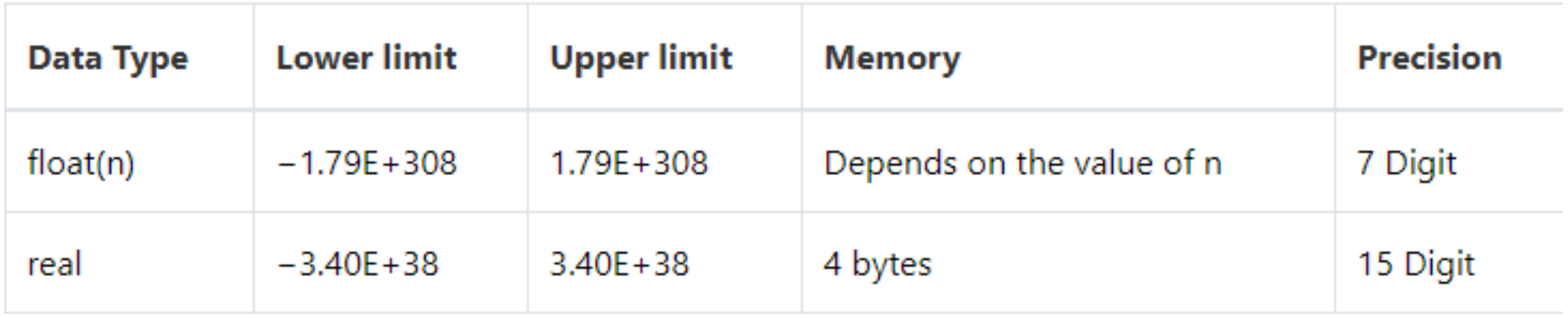


Data Type	Lower limit	Upper limit	Memory
bigint	−2^63 (−9,223,372, 036,854,775,808)	2^63−1 (−9,223,372, 036,854,775,807)	8 bytes
int	−2^31 (−2,147, 483,648)	2^31−1 (−2,147, 483,647)	4 bytes
smallint	−2^15 (−32,767)	2^15 (−32,768)	2 bytes
tinyint	0	255	1 byte
bit	0	1	1 byte/8bit column
decimal	−10^38+1	10^381−1	5 to 17 bytes
numeric	−10^38+1	10^381−1	5 to 17 bytes
money	−922,337, 203, 685,477.5808	+922,337, 203, 685,477.5807	8 bytes
smallmoney	−214,478.3648	+214,478.3647	4 bytes

Some Common SQL Data Types

❑Approximate numeric data types:

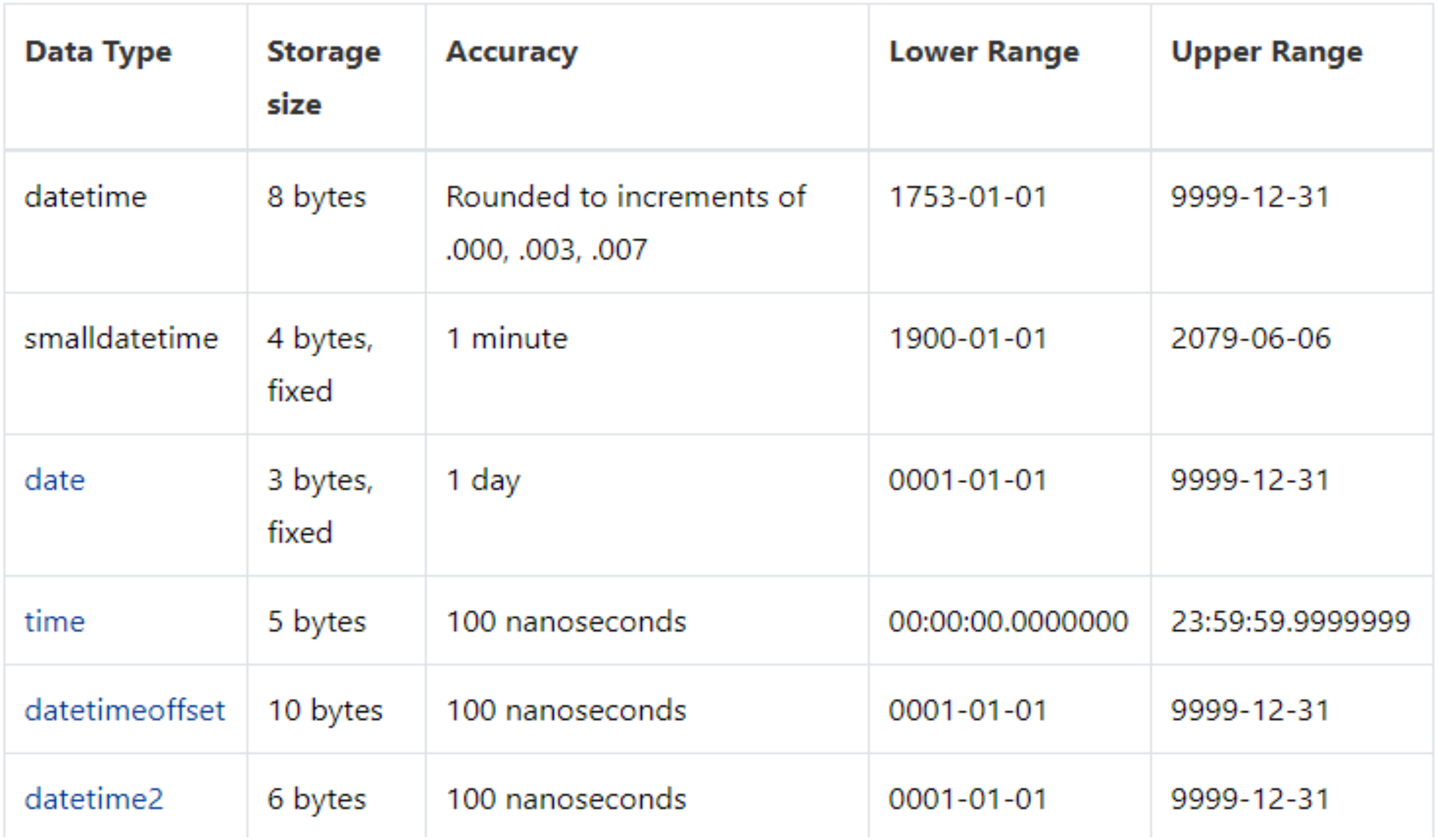
The approximate numeric data type stores floating point numeric data. They are often used in scientific calculations.



Data Type	Lower limit	Upper limit	Memory	Precision
float(n)	−1.79E+308	1.79E+308	Depends on the value of n	7 Digit
real	−3.40E+38	3.40E+38	4 bytes	15 Digit

Some Common SQL Data Types

❑Date & Time data types

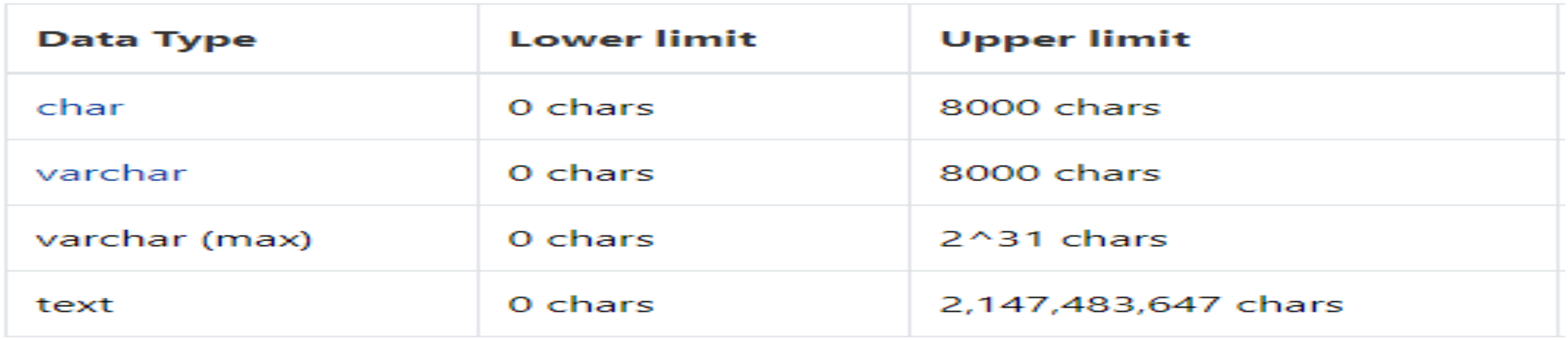


Data Type	Storage size	Accuracy	Lower Range	Upper Range
datetime	8 bytes	Rounded to increments of .000, .003, .007	1753-01-01	9999-12-31
smalldatetime	4 bytes, fixed	1 minute	1900-01-01	2079-06-06
date	3 bytes, fixed	1 day	0001-01-01	9999-12-31
time	5 bytes	100 nanoseconds	00:00:00.0000000	23:59:59.9999999
datetimeoffset	10 bytes	100 nanoseconds	0001-01-01	9999-12-31
datetime2	6 bytes	100 nanoseconds	0001-01-01	9999-12-31

Some Common SQL Data Types

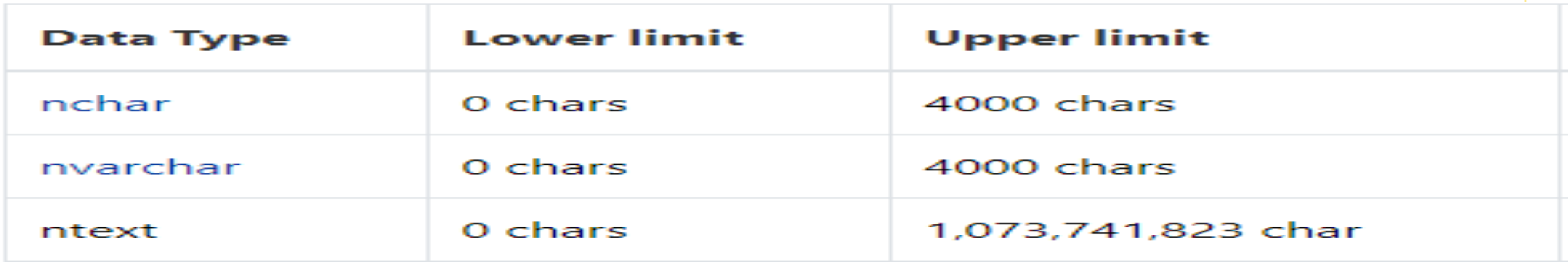
❑Character strings data types

Character strings data types allow you to store either fixed-length (char) or variable-length data (varchar).



Data Type	Lower limit	Upper limit
char	0 chars	8000 chars
varchar	0 chars	8000 chars
varchar (max)	0 chars	2^31 chars
text	0 chars	2,147,483,647 chars

❑Unicode character string data types store either fixed-length (nchar) or variable-length (nvarchar)

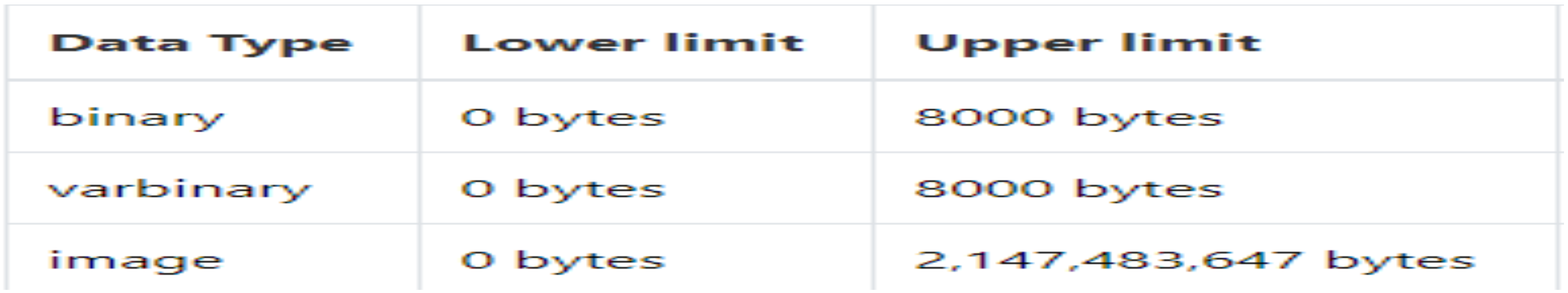


Data Type	Lower limit	Upper limit
nchar	0 chars	4000 chars
nvarchar	0 chars	4000 chars
ntext	0 chars	1,073,741,823 char

Some Common SQL Data Types

❑Binary string data types

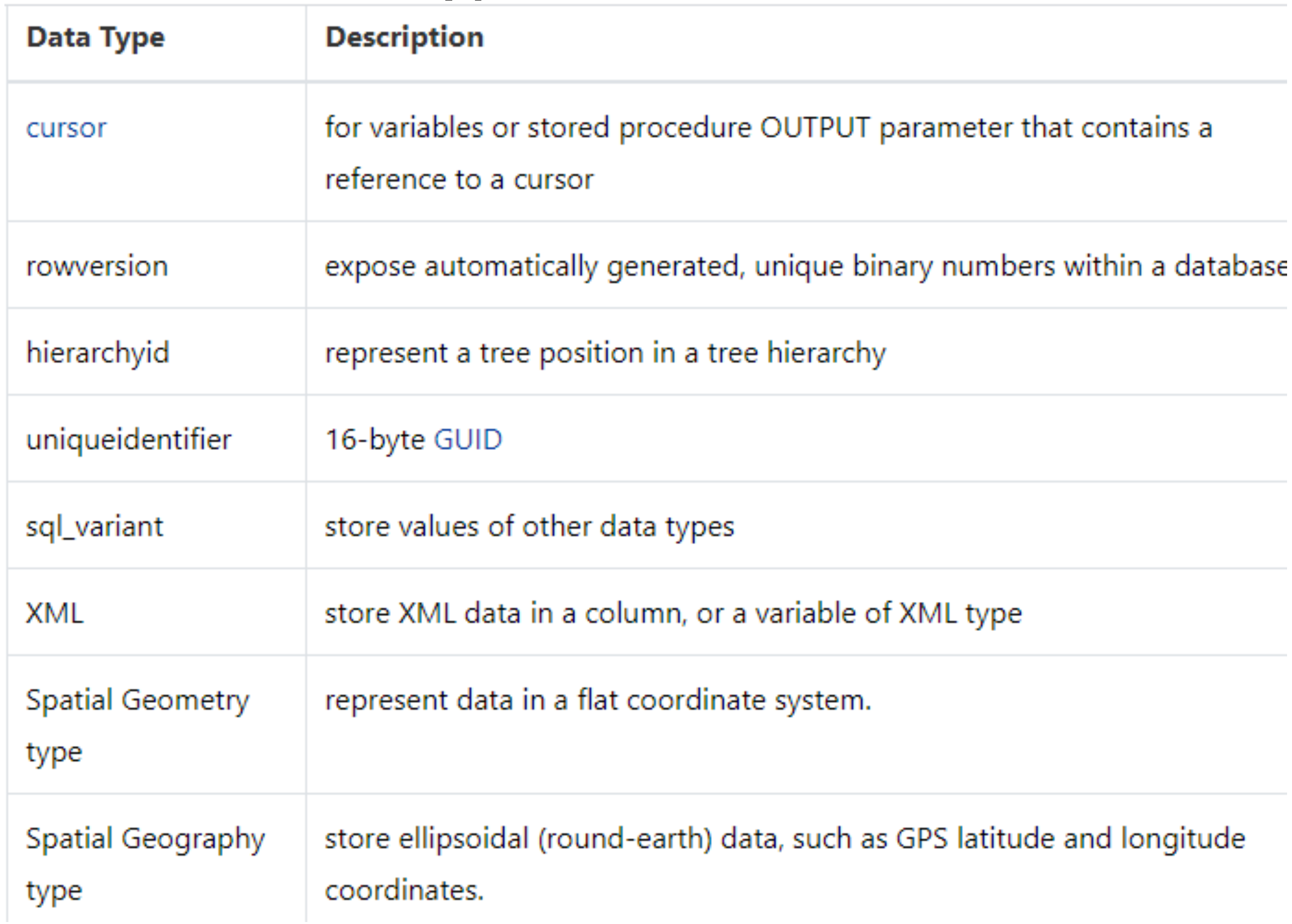
The binary data types stores fixed and variable length binary data.



Data Type	Lower limit	Upper limit
binary	0 bytes	8000 bytes
varbinary	0 bytes	8000 bytes
image	0 bytes	2,147,483,647 bytes

Some Common SQL Data Types

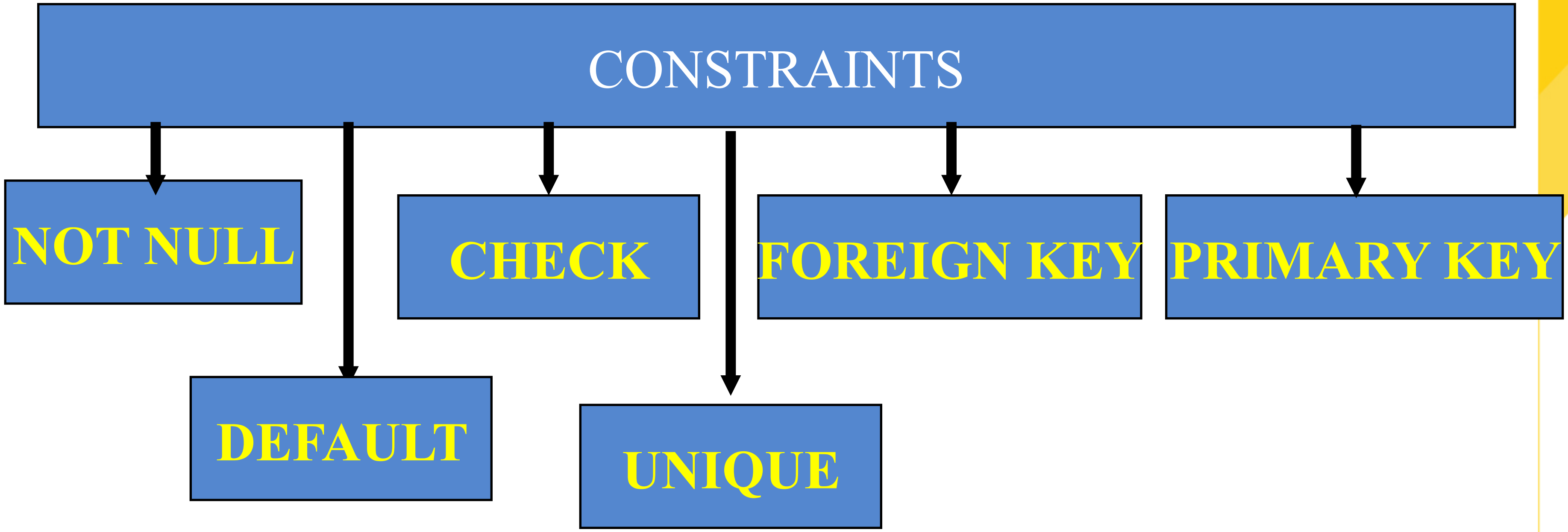
❑Other data types



Data Type	Description
cursor	for variables or stored procedure OUTPUT parameter that contains a reference to a cursor
rowversion	expose automatically generated, unique binary numbers within a database
hierarchyid	represent a tree position in a tree hierarchy
uniqueidentifier	16-byte GUID
sql_variant	store values of other data types
XML	store XML data in a column, or a variable of XML type
Spatial Geometry type	represent data in a flat coordinate system.
Spatial Geography type	store ellipsoidal (round-earth) data, such as GPS latitude and longitude coordinates.

SQL Constraints

- A constraint is a mechanism that may be used to limit the values entered into a column.



SQL Constraints

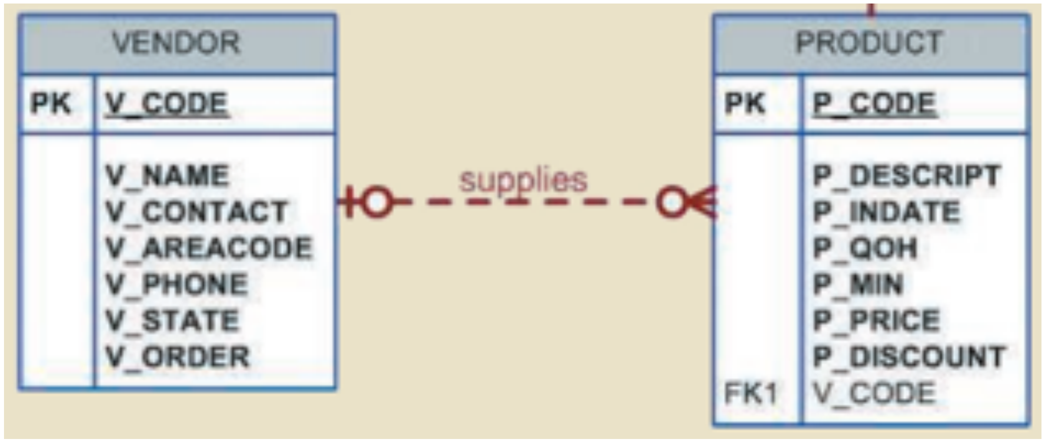
- Primary Key Constraints
  - Is a way to enforce **Entity Integrity**
  - Ensures that values in a primary key column are unique and not null.
  - A primary key can be one column or combination of columns
- Foreign key Constraints
  - Is a way to enforce **Referential integrity**
  - Ensures that if the foreign key contains a value, that value must refer to an existing value in the parent table.
  - The parent table in such a parent–child relationship should be created first so that the child table will reference an existing parent table when it is created

SQL constraints

- NOT NULL constraint
  - Ensures that a column does not accept nulls
- UNIQUE constraint
  - Ensures that all values in a column are unique
  - you can have many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table.
- DEFAULT constraint
  - Assigns a value to an attribute when a new row is added to a table
- CHECK constraint
  - is a kind of domain integrity
  - Validates data when an attribute value is entered

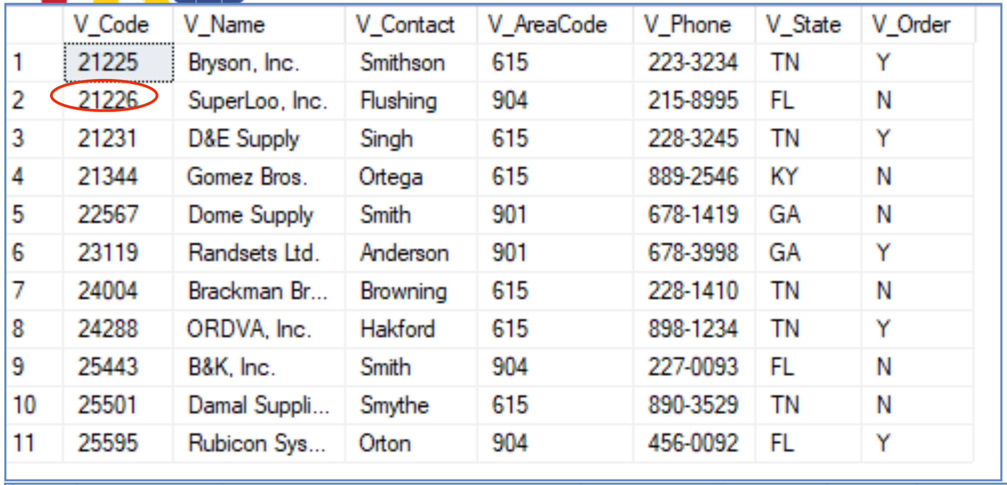
Creating a Table with foreign key constraint

❑Example:



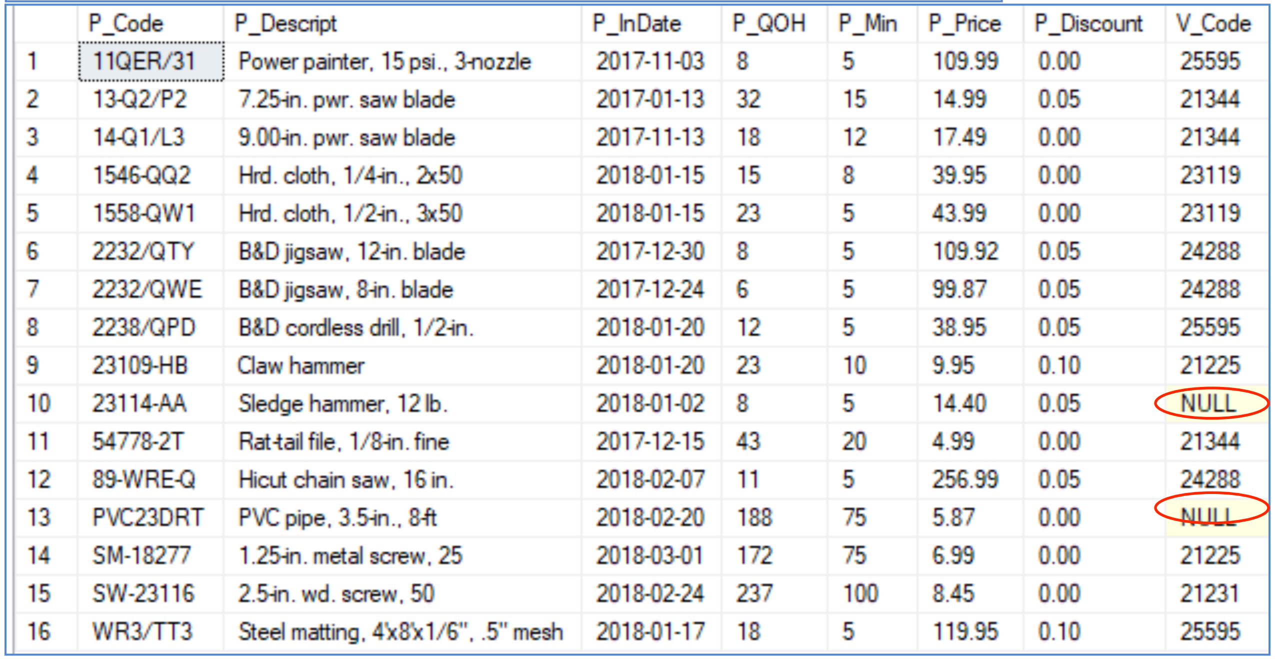
- Each product is supplied by <u>only a single vendor</u>.
- A vendor may supply <u>many products</u>
- PRODUCT is optional to VENDOR.
- Some vendors have never supplied a product (VENDOR is optional to PRODUCT)

The Vendor and Product tables



	V_Code	V_Name	V_Contact	V_AreaCode	V_Phone	V_State	V_Order
1	21225	Bryson, Inc.	Smithson	615	223-3234	TN	Y
2	21226	SuperLoo, Inc.	Flushing	904	215-8995	FL	N
3	21231	D&E Supply	Singh	615	228-3245	TN	Y
4	21344	Gomez Bros.	Ortega	615	889-2546	KY	N
5	22567	Dome Supply	Smith	901	678-1419	GA	N
6	23119	Randsets Ltd.	Anderson	901	678-3998	GA	Y
7	24004	Brackman Br...	Browning	615	228-1410	TN	N
8	24288	ORDVA, Inc.	Hakford	615	898-1234	TN	Y
9	25443	B&K, Inc.	Smith	904	227-0093	FL	N
10	25501	Damal Suppli...	Smythe	615	890-3529	TN	N
11	25595	Rubicon Sys...	Orton	904	456-0092	FL	Y

Vendor table

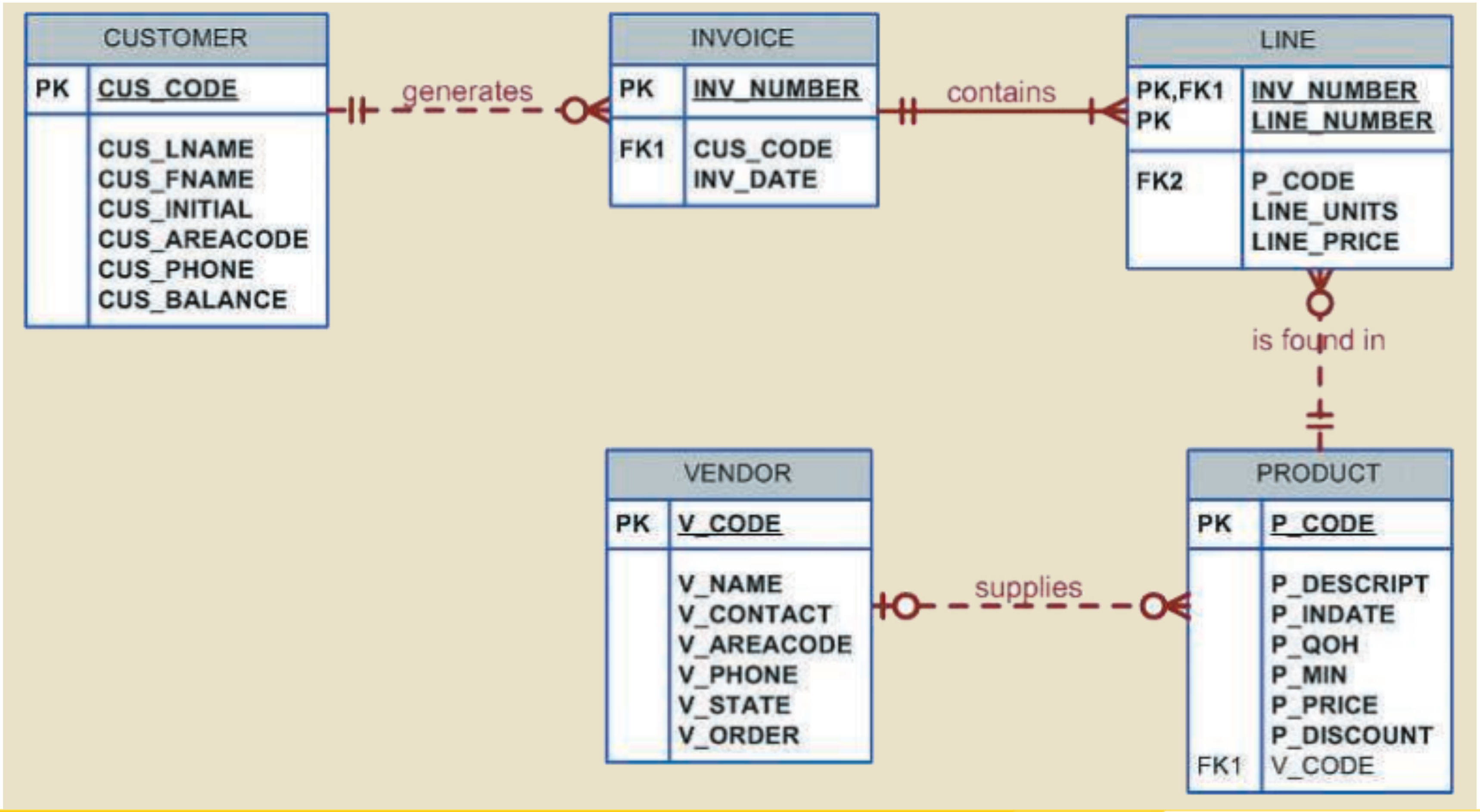


	P_Code	P_Descript	P_InDate	P_QOH	P_Min	P_Price	P_Discount	V_Code
1	11QER/31	Power painter, 15 psi., 3-nozzle	2017-11-03	8	5	109.99	0.00	25595
2	13-Q2/P2	7.25-in. pwr. saw blade	2017-01-13	32	15	14.99	0.05	21344
3	14-Q1/L3	9.00-in. pwr. saw blade	2017-11-13	18	12	17.49	0.00	21344
4	1546-QQ2	Hrd. cloth, 1/4-in., 2x50	2018-01-15	15	8	39.95	0.00	23119
5	1558-QW1	Hrd. cloth, 1/2-in., 3x50	2018-01-15	23	5	43.99	0.00	23119
6	2232/QTY	B&D jigsaw, 12-in. blade	2017-12-30	8	5	109.92	0.05	24288
7	2232/QWE	B&D jigsaw, 8-in. blade	2017-12-24	6	5	99.87	0.05	24288
8	2238/QPD	B&D cordless drill, 1/2-in.	2018-01-20	12	5	38.95	0.05	25595
9	23109-HB	Claw hammer	2018-01-20	23	10	9.95	0.10	21225
10	23114-AA	Sledge hammer, 12 lb.	2018-01-02	8	5	14.40	0.05	NULL
11	54778-2T	Rat-tail file, 1/8-in. fine	2017-12-15	43	20	4.99	0.00	21344
12	89-WRE-Q	Hicut chain saw, 16 in.	2018-02-07	11	5	256.99	0.05	24288
13	PVC23DRT	PVC pipe, 3.5-in., 8-ft	2018-02-20	188	75	5.87	0.00	NULL
14	SM-18277	1.25-in. metal screw, 25	2018-03-01	172	75	6.99	0.00	21225
15	SW-23116	2.5-in. wd. screw, 50	2018-02-24	237	100	8.45	0.00	21231
16	WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	2018-01-17	18	5	119.95	0.10	25595

Product table

The Database Model

❑Example

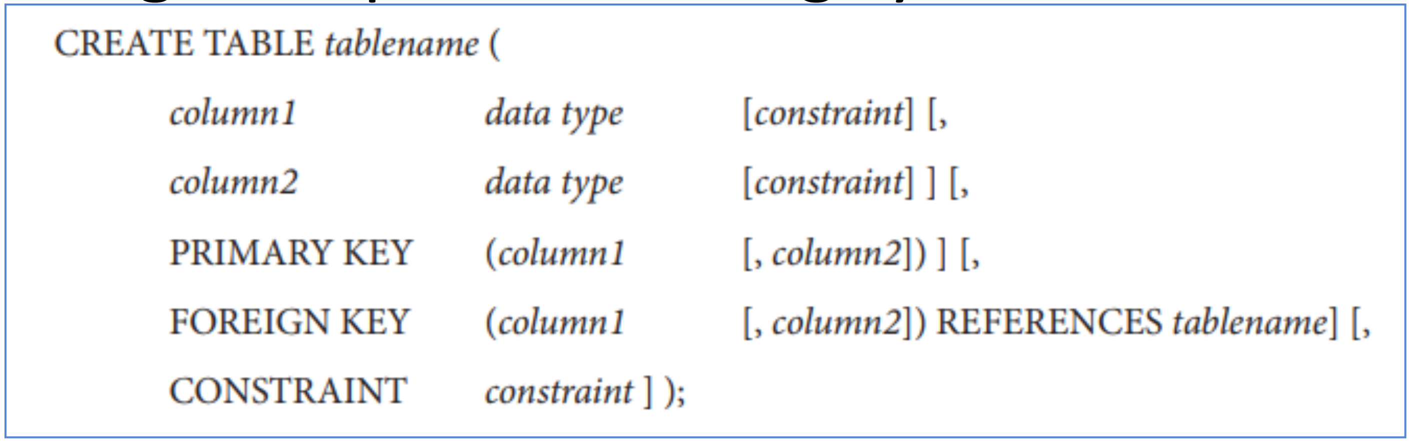


The Database Model

- The database model reflects the following business rules:
  - A customer may generate many invoices. Each invoice is generated by one customer.
  - An invoice contains one or more invoice lines. Each invoice line is associated with one invoice.
  - Each invoice line references one product. A product may be found in many invoice lines.
  - A vendor *may* supply many products. Some vendors do not yet supply products.
  - If a product is vendor-supplied, it is supplied by only a single vendor.
  - Some products are not supplied by a vendor.

Creating a table

- ❑Creating a simple table using syntax:



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

- ❑Data Types can be string, numeric, and date/time. Not all data types are supported by every relational database vendors.
- ❑To make the SQL code more readable:
  - one line per column definition,
  - SQL keywords should be upper-case letters
  - User-defined words should be lower-case letters

Creating a Table with primary key constraint

- Syntax 1:

```
CREATE TABLE table_name
(
  column1 datatype [ NULL | NOT NULL ] [ PRIMARY KEY ],
  column2 datatype [ NULL | NOT NULL ],
  ...
);
```

- Syntax 2: you can named for constraint

```
CREATE TABLE table_name
(
  column1 datatype [ NULL | NOT NULL ],
  column2 datatype [ NULL | NOT NULL ],
  ...
  CONSTRAINT constraint_name PRIMARY KEY (column1, column2, ... column_n)
);
```

Creating a table with primary key constraint

- ❑Example:

```
CREATE TABLE Vendor (
        V_Code INT PRIMARY KEY,
        V_Name VARCHAR(35) NOT NULL,
        V_Contact VARCHAR(15) 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
)
```

- ❑Or

```
CREATE TABLE Vendor (
        V_Code INT  NOT NULL,
        V_Name VARCHAR(35) NOT NULL,
        V_Contact VARCHAR(15) 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,
        CONSTRAINT PK_Vendor PRIMARY KEY (V_Code)
)
```

Primary keys can never have NULL values

Creating a Table with foreign key constraint

❑Syntax

```
CREATE TABLE child_table
(
  column1 datatype [ NULL | NOT NULL ],
  column2 datatype [ NULL | NOT NULL ],
  ...

  CONSTRAINT fk_name
    FOREIGN KEY (child_col1, child_col2, ... child_col_n)
    REFERENCES parent_table (parent_col1, parent_col2, ... parent_col_n)
    [ ON DELETE { NO ACTION | CASCADE | SET NULL | SET DEFAULT } ]
    [ ON UPDATE { NO ACTION | CASCADE | SET NULL | SET DEFAULT } ]
);
```

Creating a Table with foreign key constraint

❑Example

Primary key of parent table

```
CREATE TABLE Vendor (
        V_Code INT NOT NULL,
        …
        V_Order CHAR(1) NOT NULL,
        CONSTRAINT PK_Vendor PRIMARY KEY(V_CODE)
)
```

```
CREATE TABLE Product (
        P_Code VARCHAR(10) NOT NULL,
        P_Descript VARCHAR(35) NOT NULL,
        P_InDate DATE NOT NULL,
        P_QOH INT NOT NULL,
        P_Min INT NOT NULL,
        P_Price DECIMAL(8,2) NOT NULL,
        P_Discount DECIMAL(8,2) NOT NULL,
        V_Code INT NULL,
        CONSTRAINT PK_Product PRIMARY KEY (P_Code),
        CONSTRAINT FK_Product_Vendor_Vcode
          FOREIGN KEY (V_Code) REFERENCES Vendor)
```

Foreign key of child_table

Foreign key constraint with Delete and Update rules

DELETE AND UPDATE rules in SQL Server foreign key can be use with the following options:

❑**NO ACTION** (the default)
  - Error message would be generated, and no action is performed

❑**CASCADE**
  - if the parent record is deleted/updated, associated records in child table are also deleted/updated.

❑**SET NULL**
  - if the parent record is deleted/updated, associated records in child table are set to null
  - Foreign key column should allow NULL values

❑**SET DEFAULT**
  - if the parent record is deleted/updated, associated records in child table are set to default value specified in column definition.
  - Also default value should be present in primary key column.

Foreign key constraint with Delete and Update rules

- Example

```
CREATE TABLE Product (
        P_Code VARCHAR(10) NOT NULL,
        P_Descript VARCHAR(35) NOT NULL,
        P_InDate DATE NOT NULL,
        P_QOH INT NOT NULL,
        P_Min INT NOT NULL,
        P_Price DECIMAL(8,2) NOT NULL,
        P_Discount DECIMAL(8,2) NOT NULL,
        V_Code INT NULL,
        CONSTRAINT PK_Product PRIMARY KEY (P_Code),
        CONSTRAINT FK_Product_Vendor_Vcode
                FOREIGN KEY (V_Code) REFERENCES Vendor
                ON UPDATE CASCADE
  )
```

  - ON UPDATE CASCADE specification ensures that if you make a change in any V_CODE in VENDOR table that will result in that value changing in the PRODUCT table to match.

Creating a table with DEFAULT, CHECK, UNIQUE constraint

❑Example: CUSTOMER table

```
CREATE TABLE Customer (
    CUS_Code INT PRIMARY KEY,
    CUS_Lname NVARCHAR(30) NOT NULL,
    CUS_Fname NVARCHAR(30) 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 DECIMAL DEFAULT 0.00,
    CONSTRAINT UQ_CUS_LName_FName
               UNIQUE (CUS_Lname, CUS_Fname)
)
```

Creating a table with constraints

- Invoice table
  - the DEFAULT constraint assigns a default date to a new invoice
  - the CHECK constraint validates that the invoice date is greater than January 1, 2016

```
CREATE TABLE Invoice (
    INV_Number  INT PRIMARY KEY,
    CUS_CodeINT NOT NULL,
    INV_Date  DATETIME DEFAULT CURRENT_TIMESTAMP NOT NULL,
    CONSTRAINT FK_Inv_Cus_CusCode FOREIGN KEY(CUS_Code)
        REFERENCES Customer,
    CONSTRAINT CK_INVDate CHECK (INV_Date > '2016-01-01')
)
```

Creating a table with constraints

- Invoice table has
  - a composite primary key (INV_NUMBER, LINE_NUMBER)
  - a UNIQUE constraint
    in INV_NUMBER and P_CODE to ensure that the same product is not ordered twice in the same invoice.

```
CREATE TABLE Line (
    INV_Number INTEGER NOT NULL,
    LINE_Number NUMERIC(2,0) NOT NULL,
    P_Code VARCHAR(10) NOT NULL,
    LINE_Units DECIMAL(9,2) DEFAULT 0.00 NOT NULL,
    LINE_Price DECIMAL(9,2) DEFAULT 0.00 NOT NULL,
    PRIMARY KEY (INV_Number,LINE_Number),
    FOREIGN KEY (INV_Number) REFERENCES Invoice ON DELETE CASCADE,
    FOREIGN KEY (P_Code) REFERENCES Product(P_Code),
    CONSTRAINT UQ_Line UNIQUE(INV_Number, P_Code))
```

Some primary keys are composite– composed of multiple attributes

Altering a table

- Use ALTER TABLE statement is used to
  - To add a column to an existing table:

```
ALTER TABLE table_name
ADD column_name datatype
```

  - To change the data type of a column in a table

```
ALTER TABLE table_name
ALTER COLUMN column_name datatype
```

  - To drop a column

```
ALTER TABLE table_name
DROP COLUMN column_name
```

Altering a table

- Use ALTER TABLE statement is also used to
  - To add a constraint to an existing table:

```
ALTER TABLE table_name
ADD CONSTRAINT constraint_name constraint_type syntax
```

  - To remove a constraint

```
ALTER TABLE table_name
DROP CONSTRAINT constraint_name;
```

- Show all constraints in a table

```
exec sp_helpconstraint 'table_name'
```

Altering a table with examples

- To add column for a table
  - Add a column named CUS_Address to the Customer table

```
ALTER TABLE dbo.Customer
ADD CUS_Address NVARCHAR(50) DEFAULT N'Đà Nẵng'
```

- To add constraints for a table
  - Add a primary key for Vendor table if you forget to create it. Note: you only create a primary key on column(s) that are already defined as NOT NULL

```
ALTER TABLE Vendor
ADD CONSTRAINT PK_Vendor PRIMARY KEY(V_Code)
```

  - Add a foreign key for Product table

```
ALTER TABLE Product
ADD CONSTRAINT FK_Product_Vendor_Vcode
              FOREIGN KEY (V_Code) REFERENCES Vendor
              ON UPDATE CASCADE
```

Removing a table

- DROP TABLE statement removes the specified table from a database
- Syntax

```
DROP TABLE Table_name [RESTRIC|CASCADE]
```

  - Default option is RESTRICT, **the table will not be dropped** if there are any dependent objects, such as views or constraints, that currently reference the table.
  - If CASCADE is specified, **all dependent objects will also be dropped** as the table is dropped.

Removing a table

- Example:

```
DROP TABLE Table_name [RESTRIC|CASCADE]
```

- Many RDBMSs allows users to **retain the table's structure** but **remove all of the data** that have been entered in the table with its **TRUNCATE TABLE** command
  - Syntax:

```
TRUNCATE TABLE Table_name
```

  - Example

    TRUNCATE TABLE Customer

Auto-increment

- Auto-increment allows a **unique number** to be generated automatically when a new record is inserted into a table.
- Often this is the primary key field that we would like to be created automatically every time a new record is inserted.

Auto-increment

- To create an identity column for a table:

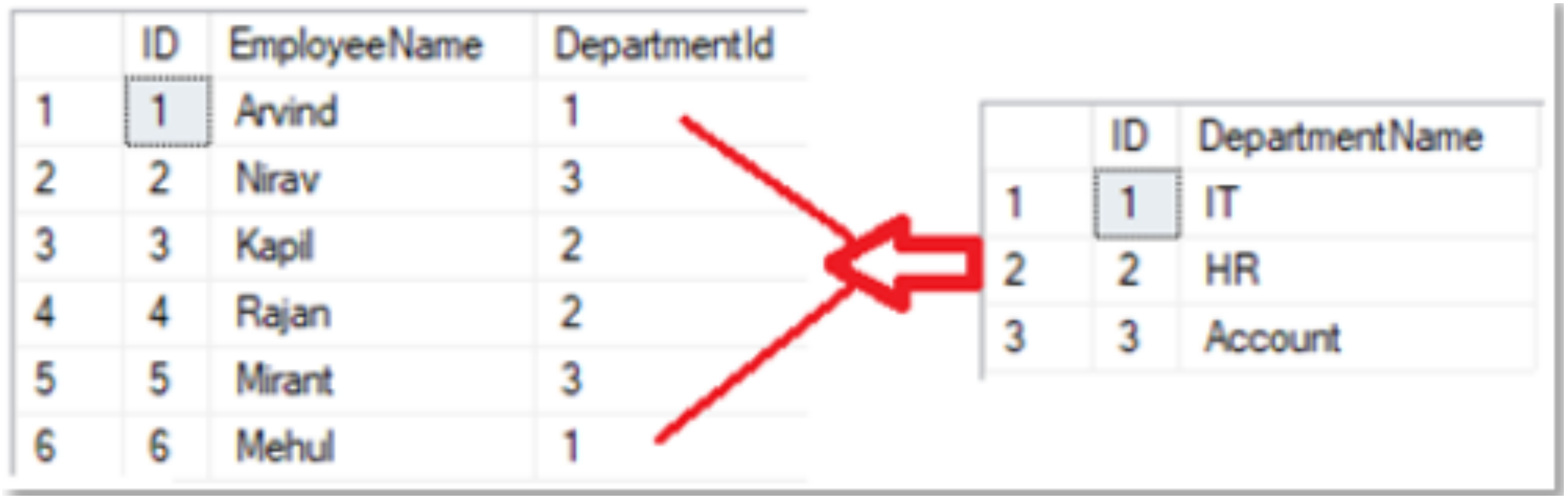
```
IDENTITY[(seed,increment)]
```

  - **seed** is the value of the first row loaded into the table.
  - **increment** is the incremental value added to the identity value of the previous row.
- The default value of seed and increment is 1.
- Example:

```
CREATE TABLE Employee
(
Emp_ID int PRIMARY KEY IDENTIY(1,1),
LastName nvarchar(255) NOT NULL,
FirstName nvarchar(255),
Address nvarchar(255),
)
```

Exercises

- Ex1: Create the following tables (Employee, Department)
  - Use Auto-increment
  - apply 4 rules: No action, Cascade, set null, set default.



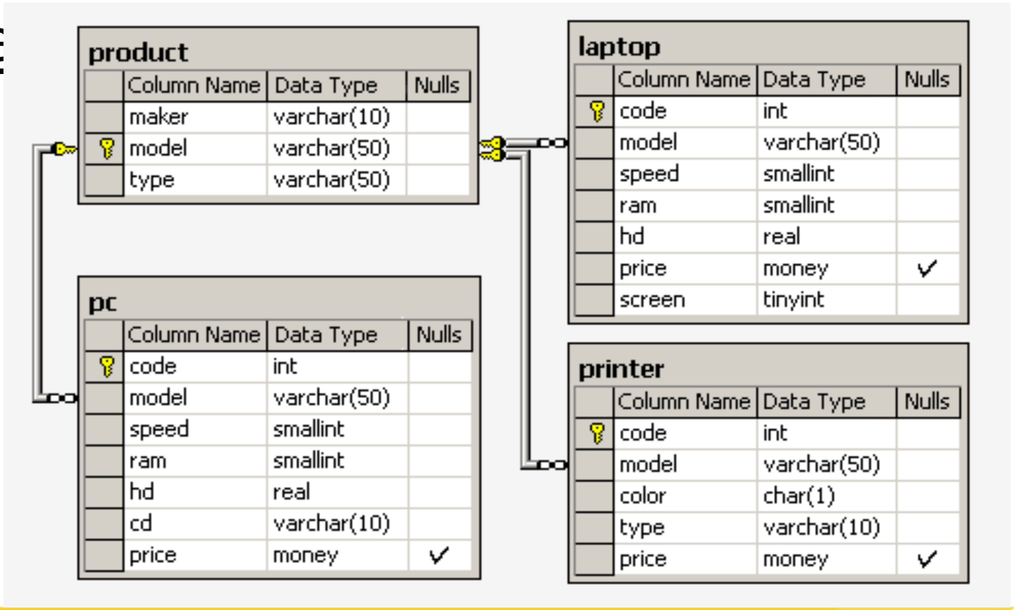
	ID	EmployeeName	DepartmentId
1	1	Arvind	1
2	2	Nirav	3
3	3	Kapil	2
4	4	Rajan	2
5	5	Mirant	3
6	6	Mehul	1

	ID	DepartmentName
1	1	IT
2	2	HR
3	3	Account

Exercises

Ex2: Create a database scheme that consists of four tables:

- Product(maker, <u>model</u>, type)
- PC(<u>code</u>, *model*, speed, ram, hd, cd, price)
- Laptop(<u>code</u>, *model*, speed, ram, hd, screen, price)
- Printer(<u>code</u>



product

	Column Name	Data Type	Nulls
	maker	varchar(10)	
🔑	model	varchar(50)	
	type	varchar(50)	

laptop

	Column Name	Data Type	Nulls
🔑	code	int	
	model	varchar(50)	
	speed	smallint	
	ram	smallint	
	hd	real	
	price	money	✓
	screen	tinyint	

pc

	Column Name	Data Type	Nulls
🔑	code	int	
	model	varchar(50)	
	speed	smallint	
	ram	smallint	
	hd	real	
	cd	varchar(10)	
	price	money	✓

printer

	Column Name	Data Type	Nulls
🔑	code	int	
	model	varchar(50)	
	color	char(1)	
	type	varchar(10)	
	price	money	✓

Exercises

- Ex3: Create a database named BookDB
  - Use the SQL statements to create the tables: Books (BookID, BookTitle, CopyRight, Year), Authors (AuthorID, AuthorFName, AuthorMName, AuthorLName, DateOfBirth, Gender, Address) and AuthorBook (BookID, AuthorID)
  - Set the constraints for the tables

INSERT Statement

- INSERT command is used to enter data into a table.

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

  - Example:

```
INSERT INTO Vendor
(V_Code,V_Name,V_Contact,V_AreaCode,V_Phone,V_State,V_Order)
VALUES(21225,'Bryson, Inc.','Smithson','615','223-3234','TN','Y')
```

- Note for INSERT statement
  - The row contents are entered between parentheses
  - Character (string) and date values must be entered between apostrophes (’).
  - Numerical entries are *not* enclosed in apostrophes.
  - Attribute entries are separated by commas.
  - A value is required for each column in the table.

INSERT Statement

- ❑ You do not need to specify the column (s) name if you are adding values for all the columns of the table. However, **make sure the order of the values is in the same order as the columns in the table.**

```
INSERT INTO TABLE_NAME VALUES (value1,value2,value3,...valueN);
```

```
INSERT INTO Vendor VALUES(21225,'Bryson,
Inc.','Smithson','615','223-3234','TN','Y')
```

- ❑ Inserting Rows with **Null Attributes** with NULL keyword when all the attribute values must be specified

```
INSERT INTO Product
VALUES ('BRT-345','Titanium drill bit','18-Oct-15', 75, 10, 4.50,
0.06, NULL)
```

INSERT Statement

- Inserting Rows with **Optional Attributes**
  - Rather than declaring each attribute as NULL in the INSERT command, you can **indicate just the attributes that have required values.**
  - Example: assume that the only required attributes for the PRODUCT table are P_Code and P_Descript

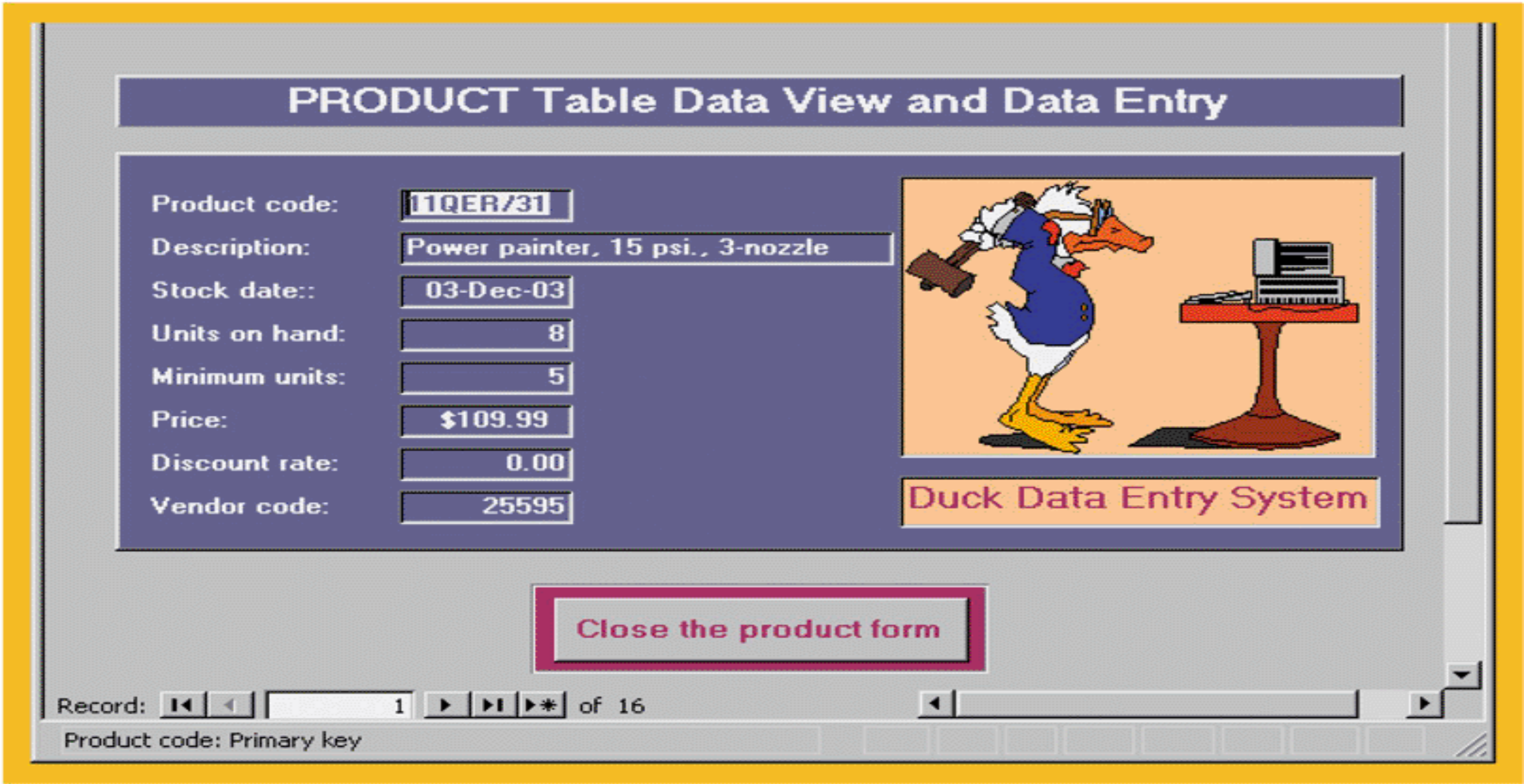
```
INSERT INTO Product(P_Code, P_Descript)
VALUES('BRT-345','Titanium drill bit')
```

- You can insert more than one record at a time.
  - Example

```
INSERT INTO Vendor VALUES
(21226,'SuperLoo, Inc.','Flushing','904','215-8995','FL','N'),
(21231,'D&E Supply'     ,'Singh'    ,'615','228-3245','TN','Y'),
(21344,'Gomez Bros.'    ,'Ortega'   ,'615','889-2546','KY','N')
```

INSERT Statement

- **End-user applications** are best created with utilities to create a form-based data view and entry screen



PRODUCT Table Data View and Data Entry

Product code:	11QER/31
Description:	Power painter, 15 psi., 3-nozzle
Stock date::	03-Dec-03
Units on hand:	8
Minimum units:	5
Price:	$109.99
Discount rate:	0.00
Vendor code:	25595

Product code: Primary key

INSERT Statement

- SQL Server **automatically uses the following value for the column** that is available in the table but does *not appear in the column list* of the INSERT statement:
  - The next incremental value if the column has an IDENTITY property.
  - The default value if the column has a default value specified.
  - The NULL if the column is nullable
  - The calculated value if the column is a computed column.
  - The current timestamp value if the data type of the column is a timestamp data type

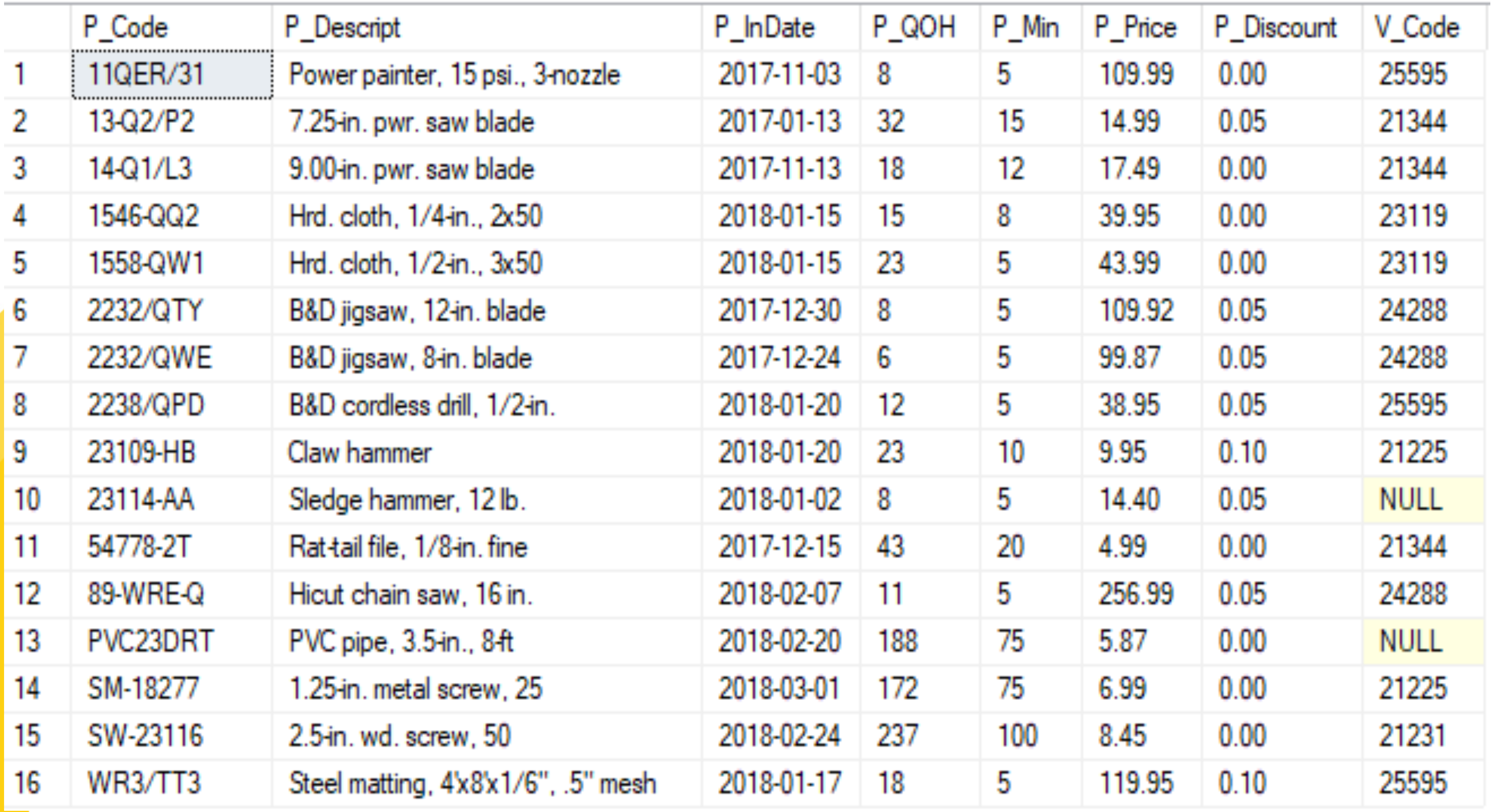
INSERT Statement

- Any changes made to the table contents **are not physically saved** on disk until
  - Database is closed
  - Program is closed
  - **COMMIT** command is used
- Saving table changes by

  COMMIT;

  - Will permanently save any changes (such as rows added, attributes modified, and rows deleted) made to any table in the database

The content of Product table



	P_Code	P_Descript	P_InDate	P_QOH	P_Min	P_Price	P_Discount	V_Code
1	11QER/31	Power painter, 15 psi., 3-nozzle	2017-11-03	8	5	109.99	0.00	25595
2	13-Q2/P2	7.25-in. pwr. saw blade	2017-01-13	32	15	14.99	0.05	21344
3	14-Q1/L3	9.00-in. pwr. saw blade	2017-11-13	18	12	17.49	0.00	21344
4	1546-QQ2	Hrd. cloth, 1/4-in., 2x50	2018-01-15	15	8	39.95	0.00	23119
5	1558-QW1	Hrd. cloth, 1/2-in., 3x50	2018-01-15	23	5	43.99	0.00	23119
6	2232/QTY	B&D jigsaw, 12-in. blade	2017-12-30	8	5	109.92	0.05	24288
7	2232/QWE	B&D jigsaw, 8-in. blade	2017-12-24	6	5	99.87	0.05	24288
8	2238/QPD	B&D cordless drill, 1/2-in.	2018-01-20	12	5	38.95	0.05	25595
9	23109-HB	Claw hammer	2018-01-20	23	10	9.95	0.10	21225
10	23114-AA	Sledge hammer, 12 lb.	2018-01-02	8	5	14.40	0.05	NULL
11	54778-2T	Rat-tail file, 1/8-in. fine	2017-12-15	43	20	4.99	0.00	21344
12	89-WRE-Q	Hicut chain saw, 16 in.	2018-02-07	11	5	256.99	0.05	24288
13	PVC23DRT	PVC pipe, 3.5-in., 8-ft	2018-02-20	188	75	5.87	0.00	NULL
14	SM-18277	1.25-in. metal screw, 25	2018-03-01	172	75	6.99	0.00	21225
15	SW-23116	2.5-in. wd. screw, 50	2018-02-24	237	100	8.45	0.00	21231
16	WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	2018-01-17	18	5	119.95	0.10	25595

Update statement

- The UPDATE statement is used to modify the existing records in a table.

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

  - Value1, value2,…can be an expression.
  - If more than one attribute is to be updated in the row, separate the corrections with commas.
  - The WHERE clause is optional, that specifies which record(s) that should be updated.
  - If you omit the WHERE clause, all records in the table will be updated!

Update statement

- ❑Example

```
UPDATE Product
SET P_InDate = '01-18-2016'
WHERE P_Code = '13-Q2/P2'
```

```
UPDATE Product
SET P_InDate = '01-18-2016', P_Price = 17.99, P_Min = 10
WHERE P_CODE = '13-Q2/P2'
```

- ❑Use can use the command to list contents of table

```
SELECT * FROM Product
```

UPDATE statement

- Restoring Table Contents by ROLLBACK statement
  - Used restore the database to its previous condition
  - Only applicable if COMMIT command has not been used to permanently store the changes in the database
- Syntax

```
ROLLBACK;
```

- Use BEGIN TRANSACTION before DML commands before using ROLLBACK command.
- COMMIT and ROLLBACK only work with **data manipulation commands** that are used to add, modify, or delete table rows

DELETE statement

- The DELETE statement is used to delete existing records in a table.

```
DELETE FROM table_name WHERE condition;
```

  - The WHERE clause is optional, that specifies which record(s) should be deleted.
  - If you omit the WHERE clause, all records in the table will be deleted!

DELETE statement

- Example

```sql
DELETE FROM Product
WHERE P_Code = 'BRT-345'
```

```sql
DELETE FROM Product
WHERE P_Min = 5
```

DELETE statement

- TRUNCATE vs DELETE
  - TRUNCATE is a DDL whereas DELETE is a DML
  - You can use WHERE clause(conditions) with DELETE but you can't use WHERE clause with TRUNCATE
  - You can't rollback data in TRUNCATE but in DELETE you can rollback data
  - TRUNCATE is faster than DELETE.

Exercises

❑Complete the previous exercise with DML commands.

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN
Vietnam - Korea University of Information and Communication Technology

Thank You !

http://vku.udn.vn/