QM-7093-01 ENTERPRISE DATA SYSTEMS
CASE STUDY (CS-7) – NOAH L. SCHRICK - 1492657

Instructor: Dr. Ismail Abdulrashid,

Instructions:

You can find the Review Questions in the textbook at the end of Chapter 7: Questions A-T of the Morgan Importing Project

- Answer questions A through T of the Morgan Importing Project found on page 416.
- Create and set up the database with the given information.
- Do not include the result table unless specifically directed to.
- Include at least one line of white space between answers.
- Submit thru Harvey drop box
- Deliverable: You are expected to submit
    - A single SQL script file (.sql) prepared and saved in SQL Server Management Studio that includes your SQL statements that answer each of the questions in order.
    - This word file that you copied all of your SQL script (no result tables) from your SQL file.
- Each query should start with a comment line that looks like the following (last character corresponding to question number):
    - /* *** CS7-MI.A *** */
    - This line should follow the SQL statement that is your answer to the particular question (e.g. B, C, D, ...)
- You should include at least one line of white space between your answer SQL statements
- Do not include the result table unless it is said so!
- Check Harvey for the due date!

Your answer should look like this:

```
/* Your Names-Group Name */
/* *** CS1-2.17 *** */
SELECT      SKU, SKU_Description
FROM        INVENTORY;

/* *** CS1-2.18 *** */
SELECT      SKU, SKU_Description
FROM        INVENTORY;

/* *** CS1-2.19 *** */
SELECT      SKU, SKU_Description
FROM        INVENTORY;
```

Please write your solution below:

/* Noah L. Schrick */

/* *** CS7-MI.A *** */

STORE does not need a surrogate key, but it may be useful to incorporate one. Using a surrogate key will mitigate a few issues, such as instances where multiple stores of the same name may be entered into the database, or mitigating consistency issues such as including “LLC” or “Inc” with the store name vs not.

When using the surrogate key in STORE, the following changes will be needed:

Table:

PURCHASE_ITEM(**PurchaseItemID**, *StoreID, PurchasingAgentID,* PurchaseDate, ItemDesription, Category, PriceUSD)

Referential Integrity Constraints:

StoreID in PURCHASE_ITEM must exist in StoreID in STORE

StoreID Start at 1000 Increment by 50

/* *** CS7-MI.B *** */

EMPLOYEE

Column	NULL Contraints
EmployeeID	NOT NULL
LastName	NULL
FirstNAME	NULL
Department	NULL
Position	NULL
Supervisor	NOT NULL
OfficePhone	NULL
OfficeFax	NULL
EmailAddress	NULL

STORE

Column	NULL Contraints
StoreID	NOT NULL
StoreName	NULL
City	NULL
Country	NULL
Phone	NULL
Fax	NULL

EmailAddress	NULL
Contact	NULL

PURCHASE_ITEM

Column	NULL Contraints
PurchaseItemID	NOT NULL
StoreID	NOT NULL
PurchasingAgentID	NOT NULL
PurchaseDate	NULL
ItemDescription	NOT NULL
Category	NULL
PriceUSD	NOT NULL

SHIPPER

Column	NULL Contraints
ShipperID	NOT NULL
ShipperName	NULL
Phone	NULL
Fax	NULL
EmailAddress	NULL
Contact	NULL

SHIPMENT

Column	NULL Contraints
ShipmentID	NOT NULL
ShipperID	NOT NULL
PurchasingAgentID	NOT NULL
ShipperInvoiceNumber	NOT NULL
Origin	NULL
Destination	NOT NULL
ScheduledDepartureDate	NULL
ActualDepartureDate	NULL
EstimatedArrivalDate	NULL

SHIPMENT_ITEM

Column	NULL Contraints
ShipmentID	NOT NULL
ShipmentItemID	NOT NULL
PurchaseItemID	NOT NULL
InsuredValue	NULL

SHIPMENT_RECEIPT

Column	NULL Contraints
ReceiptNumber	NOT NULL
ShipmentID	NOT NULL
PurchaseItemID	NOT NULL
ReceivingAgentID	NOT NULL
ReceiptDate	NULL
ReceiptTime	NULL
ReceiptQuantity	NULL
IsReceivedUndamaged	NULL
DamageNotes	NULL

/* *** CS7 – MI.C *** */

As the tables stand, SHIPMENT.ShipperInvoiceNumber is an alternate key.

The NULL constraints in Question B were designed around the existence of surrogate keys. As a result, alternate keys are not

employed. However, alternate keys can be used in certain tables assuming the null constraints were altered.

For example, ItemDescription in PURCHASE_ITEM could be an alternate.

If names could be assumed unique, then STORE.StoreName and SHIPPER.ShipperName could be alternate keys.

/* *** CS7 – MI.D *** */

PURCHASE_ITEM *Purchased From* STORE

Min Cardinality: Optional-to-Mandatory

Max Cardinality: Many-to-Many

PURCHASE_ITEM *Purchased By* EMPLOYEE

Min Cardinality: Optional-to-Mandatory

Max Cardinality: Many-to-One

SHIPMENT *Was Shipped By* SHIPPER

Min Cardinality: Optional-to-Mandatory

Max Cardinality: Many-to-One

SHIPMENT *Was Arranged By* EMPLOYEE

Min Cardinality: Optional-to-Mandatory

Max Cardinality: Many-to-One

SHIPMENT_ITEM *Is In* SHIPMENT

Min Cardinality: Mandatory-to-Optional

Max Cardinality: Many-to-One

SHIPMENT_ITEM *Is Item* PURCHASE_ITEM

Min Cardinality: Optional-to-Mandatory

Max Cardinality: One-to-One

SHIPMENT_RECEIPT *Is From* SHIPMENT

Min Cardinality: Optional-to-Mandatory

Max Cardinality: Many-to-One

SHIPMENT_RECEIPT *Includes Item* PURCHASE_ITEM

Min Cardinality: Optional-to-Mandatory

Max Cardinality: One-to-Many

SHIPMENT_RECEIPT *Obtained From* EMPLOYEE

Min Cardinality: Optional-to-Mandatory

Max Cardinality: Many-to-One

/* *** CS7 – MI.E *** */

STORE (Parent) is Required	Action on STORE (Parent)	Action on PURCHASE_ITEM (Child)
Insert	None.	Get a parent.
Modify key or foreign key	Prohibit. Surrogate key is used as primary key.	Prohibit. Surrogate key is used as primary key.
Delete	Prohibit if PurchaseItemID exists with specified Store. Allow if no PurchaeItemIDs exist with specified Store.	None.

EMPLOYEE (Parent) is Required	Action on EMPLOYEE (Parent)	Action on PURCHASE_ITEM (Child)
Insert	None.	Get a parent.

Modify key or foreign key	Prohibit. Surrogate key is used as primary key.	Prohibit. Surrogate key is used as primary key.
Delete	Prohibit if PurchaseItemID exists with specified Employee. Allow if no PurchaeItemIDs exist with specified Employee.	None.

SHIPPER (Parent) is Required	**Action on SHIPPER(Parent)**	**Action on SHIPMENT (Child)**
Insert	None.	Get a parent.
Modify key or foreign key	Prohibit. Surrogate key is used as primary key.	Prohibit. Surrogate key is used as primary key.
Delete	Prohibit if ShipmentID exists with specified Shipper. Allow if no ShipmentIDs exist with specified Shipper.	None.

EMPLOYEE (Parent) is Required	**Action on EMPLOYEE (Parent)**	**Action on SHIPMENT (Child)**
Insert	None.	Get a parent.
Modify key or foreign key	Prohibit. Surrogate key is used as primary key.	Prohibit. Surrogate key is used as primary key.
Delete	Prohibit if ShipmentID exists with specified Employee. Allow if no ShipmentIDs exist with specified Employee.	None.

SHIPMENT (Parent) is Required	Action on SHIPMENT (Parent)	Action on SHIPMENT_ITEM (Child)
Insert	None.	Get a parent.
Modify key or foreign key	Prohibit. Surrogate key is used as primary key.	Prohibit. Surrogate key is used as primary key.
Delete	Prohibit if ShipmentItemID exists with specified Shipment. Allow if no ShipmentItemIDs exist with specified Shipment.	None.
STORE (Parent) is Required	**Action on STORE (Parent)**	**Action on PURCHASE_ITEM (Child)**
Insert	None.	Get a parent.
Modify key or foreign key	Prohibit. Surrogate key is used as primary key.	Prohibit. Surrogate key is used as primary key.
Delete	Prohibit if PurchaseItemID exists with specified Store. Allow if no PurchaeItemIDs exist with specified Store.	None.

PURCHASE_ITEM (Parent) is Required	Action on PURCHASE_ITEM (Parent)	Action on SHIPMENT (Child)
Insert	None.	Get a parent.
Modify key or foreign key	Prohibit. Surrogate key is used as primary key.	Prohibit. Surrogate key is used as primary key.

Delete	Prohibit if ShipmentID exists with specified PurchaseItemID. Allow if no ShipmentIDs exist with specified PurchaseItemID.	None.

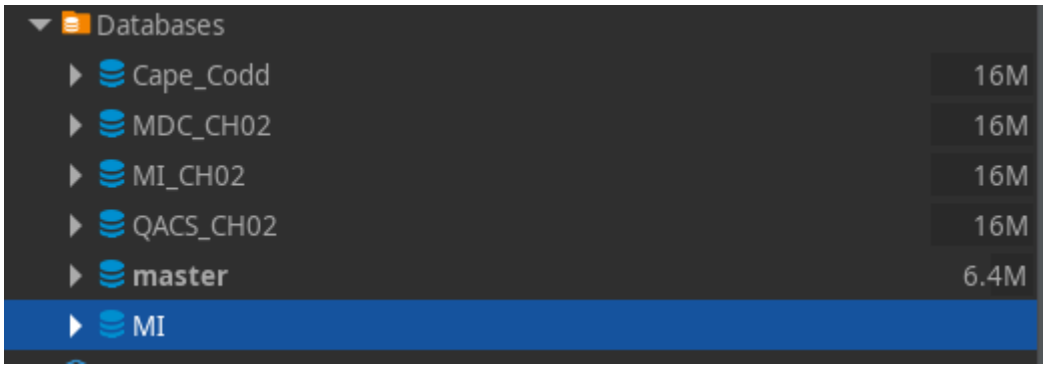
SHIPMENT (Parent) is Required	**Action on SHIPMENT(Parent)**	**Action on SHIPMENT_RECEIPT (Child)**
Insert	None.	Get a parent.
Modify key or foreign key	Prohibit. Surrogate key is used as primary key.	Prohibit. Surrogate key is used as primary key.
Delete	Prohibit if ShipmentReceiptID exists with specified Shipment. Allow if no ShipmentReceiptID exist with specified Shipment.	None.

PURCHASE_ITEM (Parent) is Required	**Action on PURCHASE_ITEM(Parent)**	**Action on SHIPMENT_RECEIPT (Child)**
Insert	None.	Get a parent.
Modify key or foreign key	Prohibit. Surrogate key is used as primary key.	Prohibit. Surrogate key is used as primary key.
Delete	Prohibit if ShipmentReceiptID exists with specified PurchaseItem.	None.

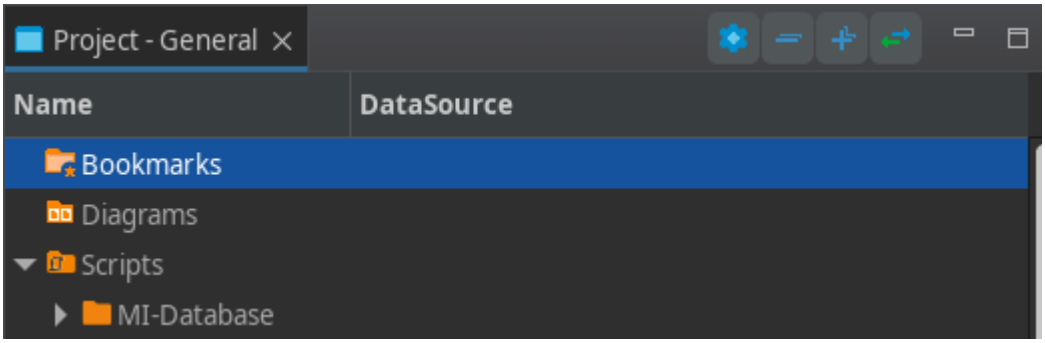
	Allow if no ShipmentReceiptID exist with specified PurchaseItem.	

EMPLOYEE (Parent) is Required	Action on EMPLOYEE (Parent)	Action on SHIPMENT_RECEIPT (Child)
Insert	None.	Get a parent.
Modify key or foreign key	Prohibit. Surrogate key is used as primary key.	Prohibit. Surrogate key is used as primary key.
Delete	Prohibit if ShipmentReceiptID exists with specified Employee. Allow if no ShipmentReceiptID exist with specified Employee.	None.

/* *** CS7 – MI.F *** */



/* *** CS7 – MI.G *** */




```
USE MI;

CREATE TABLE EMPLOYEE(
        EmployeeID INT IDENTITY(101,1),
        LastName CHAR(25),
        FirstName CHAR(25),
        Department CHAR(25),
        Position CHAR(25),
        Supervisor INT NOT NULL,
        OfficePhone VARCHAR(32),
        OfficeFax VARCHAR(32),
        EmailAddress Char(50),
        PRIMARY KEY(EmployeeID)
);

CREATE TABLE STORE(
        StoreID INT IDENTITY(1000,50),
        StoreName CHAR(25),
        City CHAR(25),
```

```
	Country CHAR(25),
	CONSTRAINT chk_country CHECK (Country IN ('Hong Kong', ‘India’,  'Japan', ‘Peru’, ‘Philippines’, ‘Singapore’, ‘United States’)),
	Phone VARCHAR(32),
	Fax VARCHAR(32),
	EmailAddress Char(50),
	Contact CHAR(25),
	PRIMARY KEY(StoreID)
);

CREATE TABLE PURCHASE_ITEM(
	PurchaseItemID INT IDENTITY(500,5),
	StoreID INT FOREIGN KEY REFERENCES STORE(StoreID),
	PurchasingAgentID INT FOREIGN KEY REFERENCES EMPLOYEE(EmployeeID),
	PurchaseDate DATE,
	ItemDescription CHAR(50),
	Category CHAR(25),
	PriceUSD Char(25),
	PRIMARY KEY(PurchaseItemID)
);

CREATE TABLE SHIPPER(
	ShipperID INT IDENTITY(1,1),
	ShipperName CHAR(25),
	Phone VARCHAR(32),
	Fax VARCHAR(32),
	EmailAddress CHAR(50),
```

```
	Contact CHAR(25),
	PRIMARY KEY(ShipperID)
);

CREATE TABLE SHIPMENT(
	ShipmentID INT IDENTITY(100,1),
	ShipperID INT FOREIGN KEY REFERENCES SHIPPER(ShipperID),
	PurchasingAgentID INT FOREIGN KEY REFERENCES EMPLOYEE(EmployeeID),
	ShipperInvoiceNumber INT,
	Origin CHAR(25),
	Destination CHAR(25),
	ScheduledDepartureDate DATE,
	ActualDepartureDate DATE,
	EstimatedArrivalDate DATE,
	PRIMARY KEY(ShipmentID)
);

CREATE TABLE SHIPMENT_ITEM(
	ShipmentID INT FOREIGN KEY REFERENCES SHIPMENT(ShipmentID),
	ShipmentItemID INT,
	PurchaseItemID INT FOREIGN KEY REFERENCES PURCHASE_ITEM(PurchaseItemID),
	InsuredValue INT,
	PRIMARY KEY(ShipmentID, ShipmentItemID)
);

CREATE TABLE SHIPMENT_RECEIPT(
	ReceiptNumber INT IDENTITY(200001,1),
```

```
        ShipmentID INT FOREIGN KEY REFERENCES SHIPMENT(ShipmentID),
        PurchaseItemID INT FOREIGN KEY REFERENCES PURCHASE_ITEM(PurchaseItemID),
        ReceivingAgentID INT FOREIGN KEY REFERENCES EMPLOYEE(EmployeeID),
        ReceiptDate DATE,
        ReceiptTime TIME,
        ReceiptQuantity INT,
        isReceivedUndamaged BIT,
        DamageNotes CHAR(50),
        PRIMARY KEY(ReceiptNumber)
);

/* *** CS7-I *** */
/*UPDATE SHIPMENT SET InsuredValue = PURCHASE_ITEM.PriceUSD



/* *** CS7-J *** */
USE MI;

/* EMPLOYEE */
DBCC CHECKIDENT ('EMPLOYEE', RESEED, 100);
INSERT INTO EMPLOYEE ( LastName, FirstName, Department, Position, Supervisor,
OfficePhone, OfficeFax, EmailAddress)
VALUES ('Morgan', 'James', 'Executive', 'CEO', '101', '310-208-1401', '310-208-1499',
'James.Morgan@morganimporting.com');

INSERT INTO EMPLOYEE ( LastName, FirstName, Department, Position, Supervisor,
OfficePhone, OfficeFax, EmailAddress)
```

```
VALUES ('Morgan', 'Jessica', 'Executive', 'CFO', '101', '310-208-1402', '310-208-1499',
'Jessica.Morgan@morganimporting.com');

INSERT INTO EMPLOYEE ( LastName, FirstName, Department, Position, Supervisor,
OfficePhone, OfficeFax, EmailAddress)

VALUES ('Williams', 'David', 'Purchasing', 'Purchasing Manager', '101', '310-208-1434', '310-
208-1498', 'David.Williams@morganimporting.com');

INSERT INTO EMPLOYEE ( LastName, FirstName, Department, Position, Supervisor,
OfficePhone, OfficeFax, EmailAddress)

VALUES ('Gilbertson', 'Teri', 'Purchasing', 'Purchasing Agent', '103', '310-208-1435', '310-
208-1498', 'Teri.Gilbertson@morganimporting.com');

INSERT INTO EMPLOYEE ( LastName, FirstName, Department, Position, Supervisor,
OfficePhone, OfficeFax, EmailAddress)

VALUES ('Wright', 'James', 'Receiving', 'Receiving Supervisor', '101', '310-208-1456', '310-
208-1497', 'James.Wright@morganimporting.com');

INSERT INTO EMPLOYEE ( LastName, FirstName, Department, Position, Supervisor,
OfficePhone, OfficeFax, EmailAddress)

VALUES ('Douglas', 'Tom', 'Receiving', 'Receiving Agent', '105', '310-208-1457', '310-208-
1497', 'Tom.Douglas@morganimporting.com');

/* STORE */

DBCC CHECKIDENT ('STORE', RESEED, 950);

INSERT INTO STORE(StoreName, City, Country, Phone, Fax, EmailAddress, Contact)

VALUES ('Eastern Sales', 'Singapore', 'Singapore', '65-543-1233', '65-543-1239',
'Sales@EasternSales.com.sg', 'Jeremy');

INSERT INTO STORE(StoreName, City, Country, Phone, Fax, EmailAddress, Contact)
```

VALUES ('Eastern Treasures', 'Manila', 'Philippines', '63-2-654-2344', '63-2-654-2349', 'Sales@EasternTreasures.com.ph', 'Gracielle');

INSERT INTO STORE(StoreName, City, Country, Phone, Fax, EmailAddress, Contact)

VALUES ('Jade Antiques', 'Singapore', 'Singapore', '65-543-3455', '65-543-3459', 'Sales@JadeAntiques.com.sg', 'Swee Lai');

INSERT INTO STORE(StoreName, City, Country, Phone, Fax, EmailAddress, Contact)

VALUES ('Andes Treasures', 'Lima', 'Peru', '51-14-765-4566', '51-14-765-4569', 'Sales@AndesTreasures.com.pe', 'Juan Carlos');

INSERT INTO STORE(StoreName, City, Country, Phone, Fax, EmailAddress, Contact)

VALUES ('Eastern Sales', 'Hong Kong', 'Peoples Republic of China', '852-876-5677', '852-876-5679', 'Sales@EasternSales.com.hk', 'Sam');

INSERT INTO STORE(StoreName, City, Country, Phone, Fax, EmailAddress, Contact)

VALUES ('Eastern Treasures', 'New Delhi', 'India', '91-11-987-6788', '91-11-987-6789', 'Sales@EasternTreasures.com.in', 'Deepinder');

INSERT INTO STORE(StoreName, City, Country, Phone, Fax, EmailAddress, Contact)

VALUES ('European Imports', 'New York City', 'United States', '800-432-8766', '800-432-8769', 'Sales@EuropeanImports.com.sg', 'Marcello');

/* PURCHASE_ITEM */

DBCC CHECKIDENT ('PURCHASE_ITEM', RESEED, 495);

INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate, ItemDescription, Category, PriceUSD)

VALUES ('1050', '101', '12/10/2017', 'Antique Large Bureaus', 'Furniture', '13415');

INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate, ItemDescription, Category, PriceUSD)

VALUES('1050', '102', '12/12/2017', 'Porcelain Lamps', 'Lamps', '13300');

INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate, ItemDescription, Category, PriceUSD)

VALUES('1200', '104', '12/15/2017', 'Gold Rim Design China', 'Tableware', '38500');

INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate, ItemDescription, Category, PriceUSD)

VALUES('1200', '104', '12/16/2017', 'Gold Rim Design Serving Dishes', 'Tableware', '3200');

INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate, ItemDescription, Category, PriceUSD)

VALUES('1050', '102', '4/7/2018', 'QE Dining Set', 'Furniture', '14300');

INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate, ItemDescription, Category, PriceUSD)

VALUES('1100', '103', '5/18/2018', 'Misc Linen', 'Linen', '88545');

INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate, ItemDescription, Category, PriceUSD)

VALUES('1000', '103', '5/19/2018', 'Large Masks', 'Decorations', '22135');

INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate, ItemDescription, Category, PriceUSD)

VALUES('1100', '104', '5/20/2018', 'Willow Design China', 'Tableware', '147575');

```
INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate,
ItemDescription, Category, PriceUSD)

VALUES('1100', '104', '5/20/2018', 'Willow Design Serving Dishes', 'Tableware', '12040');


INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate,
ItemDescription, Category, PriceUSD)

VALUES('1150', '102', '6/14/2018', 'Woven Goods', 'Decorations', '1200');


INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate,
ItemDescription, Category, PriceUSD)

VALUES('1150', '101', '6/16/2018', 'Antique Leather Chairs', 'Furniture', '5375');


INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate,
ItemDescription, Category, PriceUSD)

VALUES('1100', '104', '7/15/2018', 'Willow Design Serving Dishes', 'Tableware', '4500');


INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate,
ItemDescription, Category, PriceUSD)

VALUES('1000', '103', '7/17/2018', 'Large Bureau', 'Furniture', '9500');


INSERT INTO PURCHASE_ITEM(StoreID, PurchasingAgentID, PurchaseDate,
ItemDescription, Category, PriceUSD)

VALUES('1100', '104', '7/20/2018', 'Brass Lamps', 'Lamps', '1200');



/* SHIPPER */


DBCC CHECKIDENT ('SHIPPER', RESEED, 0);

INSERT INTO SHIPPER(ShipperName, Phone, Fax, EmailAddress, Contact)
```

```
VALUES('ABC Trans-Oceanic', '800-234-5656', '800-234-5659',
'Sales@ABCTransOceanic.com', 'Jonathan');

INSERT INTO SHIPPER(ShipperName, Phone, Fax, EmailAddress, Contact)

VALUES('International', '800-123-8898', '800-123-8899', 'Sales@International.com',
'Marylin');

INSERT INTO SHIPPER(ShipperName, Phone, Fax, EmailAddress, Contact)

VALUES('Worldwide', '800-123-4567', '800-123-4569', 'Sales@worldwide.com', 'Jose');

/* SHIPMENT */

DBCC CHECKIDENT ('SHIPMENT', RESEED, 99);

INSERT INTO SHIPMENT(ShipperID, PurchasingAgentID, ShipperInvoiceNumber, Origin,
Destination, ScheduledDepartureDate, ActualDepartureDate, EstimatedArrivalDate)

VALUES('1', '103', '2017651', 'Manila', 'Los Angeles', '10-Dec-17', '10-Dec-17', '15-Mar-18');

INSERT INTO SHIPMENT(ShipperID, PurchasingAgentID, ShipperInvoiceNumber, Origin,
Destination, ScheduledDepartureDate, ActualDepartureDate, EstimatedArrivalDate)

VALUES('1', '104', '2018012', 'Hong Kong', 'Seattle', '10-Jan-18', '12-Jan-18', '20-Mar-18');

INSERT INTO SHIPMENT(ShipperID, PurchasingAgentID, ShipperInvoiceNumber, Origin,
Destination, ScheduledDepartureDate, ActualDepartureDate, EstimatedArrivalDate)

VALUES('1', '103', '49100300', 'Manila', 'Los Angeles', '05-May-18', '05-May-18', '17-Jun-18');

INSERT INTO SHIPMENT(ShipperID, PurchasingAgentID, ShipperInvoiceNumber, Origin,
Destination, ScheduledDepartureDate, ActualDepartureDate, EstimatedArrivalDate)

VALUES('1', '104', '399400', 'Singapore', 'Portland', '02-Jun-18', '04-Jun-18', '17-Jul-18');
```

```
INSERT INTO SHIPMENT(ShipperID, PurchasingAgentID, ShipperInvoiceNumber, Origin,
Destination, ScheduledDepartureDate, ActualDepartureDate, EstimatedArrivalDate)
VALUES('1', '103', '84899440', 'Lima', 'Los Angeles', '10-Jul-18', '10-Jul-18', '28-Jul-18');

INSERT INTO SHIPMENT(ShipperID, PurchasingAgentID, ShipperInvoiceNumber, Origin,
Destination, ScheduledDepartureDate, ActualDepartureDate, EstimatedArrivalDate)
VALUES('1', '104', '488955', 'Singapore', 'Portland', '05-Aug-18', '09-Aug-18', '11-Sep-18');

/* SHIPMENT_ITEM */
INSERT INTO SHIPMENT_ITEM
VALUES('100', '1', '500', '15000');

INSERT INTO SHIPMENT_ITEM
VALUES('100', '2', '505', '15000');

INSERT INTO SHIPMENT_ITEM
VALUES('101', '1', '510', '40000');

INSERT INTO SHIPMENT_ITEM
VALUES('101', '2', '515', '3500');

INSERT INTO SHIPMENT_ITEM
VALUES('102', '1', '520', '15000');

INSERT INTO SHIPMENT_ITEM
VALUES('103', '1', '525', '90000');
```

```
INSERT INTO SHIPMENT_ITEM
VALUES('103', '2', '530', '25000');

INSERT INTO SHIPMENT_ITEM
VALUES('103', '3', '535', '150000');

INSERT INTO SHIPMENT_ITEM
VALUES('103', '4', '540', '12500');

INSERT INTO SHIPMENT_ITEM
VALUES('104', '1', '545', '12500')

INSERT INTO SHIPMENT_ITEM
VALUES('104', '2', '550', '5500');

INSERT INTO SHIPMENT_ITEM
VALUES('105', '1', '555', '4500');

INSERT INTO SHIPMENT_ITEM
VALUES('105', '2', '560', '10000');

INSERT INTO SHIPMENT_ITEM
VALUES('105', '3', '565', '1500');

/* SHIPMENT_RECEIPT */
INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID,
ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)
```

VALUES('100', '500', '105', '17-Mar-18', '10:00 AM', '3', '1', NULL);

INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID, ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('100', '505', '105', '17-Mar-18', '10:00 AM', '50', '1', NULL);

INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID, ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('101', '510', '105', '23-Mar-18', '3:30 PM', '100', '1', NULL);

INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID, ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('101', '515', '105', '23-Mar-18', '3:30 PM', '10', '1', NULL);

INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID, ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('102', '520', '106', '19-Jun-18', '10:15 AM', '1', '0', 'One leg on one chair broken');

INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID, ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('103', '525', '106', '20-Jul-18', '2:20 AM', '1000', '1', NULL);

INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID, ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('103', '530', '106', '20-Jul-18', '2:20 AM', '100', '1', NULL);

INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID, ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('103', '535', '106', '20-Jul-18', '2:20 AM', '100', '1', NULL);

```
INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID,
ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('103', '540', '106', '20-Jul-18', '2:20 AM', '10', '1', NULL);


INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID,
ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('104', '545', '105', '29-Jul-18', '9:00 PM', '100', '1', NULL);


INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID,
ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('104', '550', '105', '29-Jul-18', '9:00 PM', '5', '1', NULL);


INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID,
ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('105', '555', '106', '14-Sep-18', '2:45 PM', '4', '1', NULL);


INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID,
ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('105', '560', '106', '14-Sep-18', '2:45 PM', '1', '1', NULL);


INSERT INTO SHIPMENT_RECEIPT(ShipmentID, PurchaseItemID, ReceivingAgentID,
ReceiptDate, ReceiptTime, ReceiptQuantity, isReceivedUndamaged, DamageNotes)

VALUES('105', '565', '106', '14-Sep-18', '2:45 PM', '10', '0', 'Base of one lamp scratched');
```

```
/* *** CS7 – MI.K *** */

UPDATE STORE

SET

        City = 'NYC'

WHERE

        City = 'New York City'

;


/* *** CS7 – MI.L *** */

INSERT INTO SHIPMENT(ShipperID, PurchasingAgentID, ShipperInvoiceNumber, Origin,
Destination, ScheduledDepartureDate, ActualDepartureDate, EstimatedArrivalDate)

VALUES('1', '104', '999999', 'DELETE', 'DELETE', '02-Jun-18', '04-Jun-18', '17-Jul-18');


INSERT INTO SHIPMENT_ITEM

VALUES('106', '99', '500', '99');


DELETE FROM SHIPMENT_Item WHERE ShipmentID = '106';

DELETE FROM SHIPMENT WHERE ShipmentID = '106';


Needs two deletes since a cascade delete rule is not set.


/* *** CS7-MI.M *** */

CREATE VIEW EmployeeSupervisorView AS

        SELECT

                E1.LastName AS EmployeeLastName,

                E1.FirstName AS EmployeeFirstName,

                E1.Position AS EmployeePosition,
```

```
        E2.LastName AS SupervisorLastName,
        E2.firstName AS SupervisorFirstName
            FROM EMPLOYEE E1
                INNER JOIN EMPLOYEE E2 on E1.Supervisor = E2.EmployeeId
;

SELECT * FROM EmployeeSupervisorView;



/* *** CS7 - MI.N *** */
CREATE VIEW PurchaseSummaryView AS
    SELECT PurchaseItemID, PurchaseDate, ItemDescription, PriceUSD
        FROM PURCHASE_ITEM;

SELECT * FROM PurchaseSummaryView;



/* *** CS7 - MI.O *** */
CREATE FUNCTION StoreContactAndPhone(@StoreContact CHAR(25), @ContactPhone
VARCHAR(32))
    RETURNS VARCHAR(59)
        AS
            BEGIN
                RETURN (SELECT RTRIM(@StoreContact) + ':' + SPACE(1) +
@ContactPhone)
            END
```

```
/* *** CS7 - MI.P *** */

CREATE VIEW StorePurchaseHistoryView AS

	SELECT STORE.StoreName, STORE.Phone, STORE.Contact, I.PurchaseItemID,
I.PurchaseDate, I.ItemDescription, I.PriceUSD

		FROM STORE, PURCHASE_ITEM I

			WHERE STORE.StoreID = I.StoreID

;


SELECT * FROM StorePurchaseHistoryView;


/* *** CS7 - MI.Q *** */

CREATE VIEW StoreContactPurchaseHistoryView AS

	SELECT STORE.StoreName, dbo.StoreContactAndPhone(STORE.Contact,
STORE.Phone) AS ContactAndPhone, I.PurchaseItemID, I.PurchaseDate, I.ItemDescription,
I.PriceUSD

		FROM STORE, PURCHASE_ITEM I

			WHERE STORE.StoreID = I.StoreID

;


SELECT * FROM StoreContactPurchaseHistoryView;


/* *** CS7 - MI.R *** */

CREATE VIEW StoreHistoryView AS

	SELECT StoreName, SUM(CAST(PriceUSD AS INT)) AS StorePurchaseHistory

		FROM dbo.StorePurchaseHistoryView

			GROUP BY StoreName

;
```

```
SELECT * FROM StoreHistoryView;


/* *** CS7 - MI.S *** */

CREATE VIEW MajorSources AS

        SELECT StoreName

                FROM StoreHistoryView

                        WHERE StorePurchaseHistory > 100000

;


SELECT * FROM MajorSources;


/* *** CS7 – MI.T ***  */
```

Triggers can be used to enforce minimum cardinality by examining the NULL status of data. On Inserts, triggers can be set to ensure that the “Mandatory” component of the relationship is not null. Likewise, updates and deletes can include triggers to ensure that the Mandatory components are not being removed.